Stability Analysis of Simplex Architecture Controlled Inverted Pendulum

Taylor Johnson
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801
Email: johnso99@illinois.edu

***Abstract*—Switched controllers are being used frequently for control of complex systems. However, they introduce new challenges for verification of stability, of which a great deal of work in the hybrid systems domain has been formalizing recently. This work is a stability case study of the classical inverted pendulum, in this case controlled using the Simplex architecture of [1], combining to form a complete system as in [2]. The main result shown uses small-gain theorems to prove the stability of the system with regards to measurement delays.**

I. Introduction and Preliminaries

The physical system in Figure 1 consists of a DC-motor driven cart and a pendulum attached to the cart, with the control goal of keeping the angle θ of the pendulum at $0°$ measured from the vertical. There is an additional control goal of moving the cart to a set point x_s along the x-axis.

This system is described by the nonlinear form

$$\dot{x} = f(x, u) \tag{1}$$

However, we work with a linearized model of the form

$$\dot{x} = Ax + Bu \tag{2}$$

The linearization is justified after the system description. There are four state variables, the cart position x, cart velocity $\dot{x}$, pendulum angle θ, and pendulum velocity $\dot{\theta}$ (we will now denote X as the state vector and x as the position, seen together in Equation 3). As the system has been implemented, it is subject to physical constraints. The range of x is between $[-0.7, 0.7]$ meters (and the set point $[x_s]$ range is between $[-0.5, 0.5]$ meters), $\dot{x}$ is between $[-1.0, 1.0]$ meters/second, θ is between $[-30, 30]°$, and $\dot{\theta}$ is unconstrained. Note that in the real implementation, to account for errors due to linearization and estimation (seen later), the constraints are made more conservative, so $x - x_s$ is between $[-0.2, 0.2]$ meters and θ is between $[-15, 15]°$.

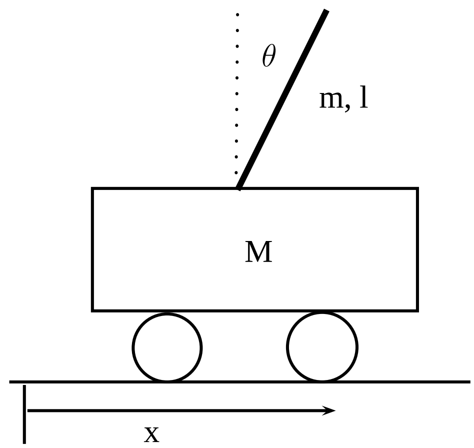

Fig. 1. Inverted Pendulum System

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x - x_s \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} \tag{3}$$

The system is stabilized by linear state feedback of the form $\dot{X} = (A + BK)\,X$. The control input, u is the armature voltage of the DC-motor (V_a) and is constrained between $[-4.96, 4.96]$ volts. Thus, due to this control constraint, it is necessary to look at the system in the form of $\dot{X} = AX + Bu$ at some points of the later analysis.

The primary linearizations to note are that we ignore static friction (with respect to the cart wheels and ground, and with respect to the pendulum arm and joint) and take the armature inductance

($L_a = 18$ millihenries) to be 0 henry hence reducing the order of the system by making the armature current state variable I_a a function of V_a. Without this simplification, two control states would be necessary. The linearization is justified since the control objective is to stabilize the system in a neighborhood of the vertical equilibrium, defined in this coordinate system as $\theta = 0°$.

For the remainder of the paper, we work towards proving stability of the system, first in the classical Lyapunov sense, and then using newer small-gain theorem results to also discuss stability in the delayed system as in [3] and [4]. Recent results using small-gain theorems allow us to prove stability of systems with delay by analyzing the stability of a simpler system without delay. The idea is to treat the error introduced by the delay as a disturbance input with bounded-gain, allowing us to talk about input-to-state stability. Using this, we can find the maximum delay the system can tolerate. Despite the linearizations of the system model, the small-gain techniques apply to nonlinear systems and can also be used to analyze stability in the more general system model.

II. System and Controller Models

Figure 2 shows a high-level view of the entire control system. We now discuss each block individually. We are using hybrid input-output automata to describe the blocks, and we assume the reader is familiar with such notation, but if not, [5] or [6] can serve as references to the notation.

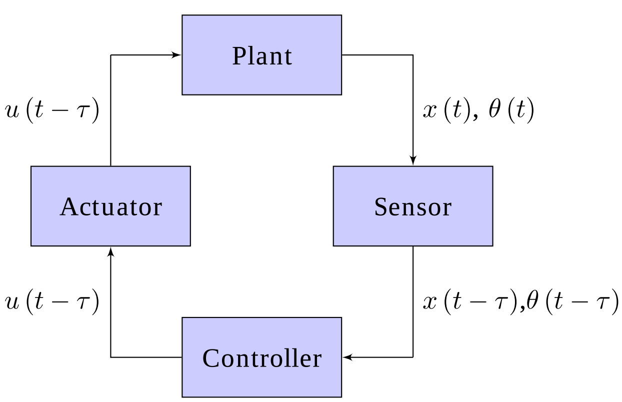

Fig. 2. System Model

A. Plant Model

The plant model of the inverted pendulum is described in Figure 3, where the two input parameters to the automaton plant, $A : \texttt{Real}^{4\times4}$ and $B : \texttt{Real}^{4\times1}$, are defined as

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -a_{22} & -a_{23} & a_{24} \\ 0 & 0 & 0 & 1 \\ 0 & a_{42} & a_{43} & -a_{44} \end{bmatrix} \tag{4}$$

and

$$B = \begin{bmatrix} 0 \\ b_2 \\ 0 \\ -b_4 \end{bmatrix} \tag{5}$$

where $a_{22} = \frac{4\bar{B}}{D_l}$, $a_{23} = \frac{3mg}{D_l}$, $a_{24} = \frac{6B_\theta}{lD_l}$, $a_{42} = \frac{6\bar{B}}{lD_l}$, $a_{43} = \frac{6\bar{M}g}{lD_l}$, $a_{44} = \frac{12\bar{M}B_\theta}{ml^2D_l}$, $b_2 = \frac{4B_l}{D_l}$, and $b_4 = \frac{6B_1}{lD_l}$, for $D_l = 4\bar{M} - 3m$, $\bar{B} = \frac{K_gB_m}{r^2} + \frac{K_g^2K_iK_b}{r^2R_a}$, $B_l = \frac{K_gK_i}{rR_a}$, $\bar{M} = \frac{m+M+(K_gJ_m)}{r^2}$, and where g is gravity, R_a is the armature resistance, r is the driving wheel radius, J_m is the motor rotor inertia, B_m is the motor's coefficient of viscous friction, B_θ is the pendulum joint's coefficient of viscous friction, K_i is the motor torque constant, K_b is the motor back-e.m.f. constant, K_g is the gear ratio, M is the cart mass, m is the pendulum mass, and l is the pendulum length. After applying real values, the A and B matrices used are

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -10.95 & -2.75 & 0.0043 \\ 0 & 0 & 0 & 1 \\ 0 & 24.92 & 28.58 & -0.044 \end{bmatrix} \tag{6}$$

and

$$B = \begin{bmatrix} 0 \\ 1.94 \\ 0 \\ -4.44 \end{bmatrix} \tag{7}$$

B. Sensor Model

The sensor model can be seen in Figure 4, and is an analog-to-digital converter (ADC). This is where the delay is introduced to the system in the form of measurement delay, primarily due to filtering.

automaton Plant(A : Real$^{4\times4}$, B : Real$^{4\times1}$)
variables
  input u : Real;
  output θ : Real; x : Real;
  internal θ_h : Real := 0; $\dot{\theta}_h$: Real := 0;
    x_h : Real := 0; $\dot{x}_h$: Real := 0;

trajectories
  trajdef plantDynamics
    evolve $d(x_h) = \dot{x}_h$;
      $d(\dot{x}_h) = -a_{22}\dot{x}_h - a_{23}\theta_h + a_{24}\dot{\theta}_h + b_2 u$;
      $d(\theta_h) = \dot{\theta}_h$;
      $d(\dot{\theta}_h) = a_{42}\dot{x}_h + a_{43}\theta_h - a_{34}\dot{\theta}_h - b_4 u$;
      $\theta = \theta_h$; $x = x_h$;

Fig. 3. Linearized Plant Model

Between the sensor and the controller there exists a low-pass filter which will introduce a delay between one and two sampling periods ($T_s = 20$ milliseconds) long. There is also nondeterministic delay introduced due to the ADC and controller not being perfectly synchronized. The ADC is utilizing a sample-and-hold, which must be finished before the controller reads the value, but the time difference between when the sample-and-hold finishes and the controller reads the value is unknown, but is assumed to be certainly less than one sampling period (T_s) long. The filter delay will be decreased by model-based state projection described later.

automaton Sensor(T_s : Real) **where** $T_s > 0$
signature
  output sample(θ', x' : Real)

variables
  input θ : Real; x : Real;
  internal θ_s : Real; x_s : Real;
    now_s : Real := 0;
    $next\_sample$: AugmentedReal := T_s;
    let $time\_left$:= $next\_sample - now_s$

transitions
  output sample(θ', x')
    pre $now_s = next\_sample$
      $\wedge\, \theta' = \theta_s$
      $\wedge\, x' = x_s$;
    eff $next\_sample := next\_sample + T_s$;

trajectories
  trajdef periodicSample
    stop when $now_s = next\_sample$
    evolve $d(now_s) = 1$; $\theta_s = \theta$; $x_s = x$;

Fig. 4. Sensor

C. Actuator Model

The actuator model serves little purpose at the present time, but is included for completeness as seen in Figure 5, and may be used in the future to model the quantization error and delay introduced by this block. The actuator is a digital-to-analog converter (DAC), followed by a higher power voltage source to drive the DC motor. As a note, the delay introduced by the actuator is in general quite small, such as the switching time of toggling a power MOSFET, usually on the order of a few microseconds. In the case of our control cycle period of 20 milliseconds, this is negligible.

automaton Actuator(T_a : Real) **where** $T_a > 0$
  signature
    input controllerOutput(u' : Real)

  variables
    output u : Real;
    internal u_a : Discrete Real := 0;
      $ready_a$: Bool := $false$;
      now_a : Real := 0;

  transitions
    input switchingOutput(u')
      eff $u_a = u'$;
        $ready_a := true$;

  trajectories
    trajdef hold
      evolve $d(now_a) = 1$; $u = u_a$;

Fig. 5. Actuator

D. Controller Model

The controller is implemented following the Simplex architecture of [1]. The Simplex architecture is built on the concept of analytic redundancy of [7], in this case, that several controllers implement the same control objective with different performance. There are three controllers in this architecture, a safety controller that can stabilize the system from the largest set of initial conditions but with poor performance, a baseline controller that has better performance than the safety controller but cannot stabilize from such a wide set of initial conditions, and an experimental controller that has the best performance but the smallest set of stabilizable initial conditions. The usefulness of such an architecture could be in upgrades of real-time systems that cannot afford downtime even for maintenance, such as critical infrastructure. In our case, the architecture is used to differentiate between the different sets of recoverable initial conditions and performance. Each controller uses linear state feedback for stabilization, with the only difference being higher gains in the baseline and experimental controllers to stabilize faster, with the downside that they cannot recover from certain initial conditions that the safety controller can. Thus, the system is described as

$$\dot{X} = (A + BK_\sigma) X \tag{8}$$

where K_σ is one of the safety, baseline, or experimental controller gains and the solution to this is

$$X = e^{(A+BK_\sigma)t} X_0 \tag{9}$$

where X_0 is an initial condition within the stabilizable region.

Note that, as only θ and x are measurable, $\dot{\theta}$ and $\dot{x}$ are constructed by the first-order approximations $\dot{\theta}(t) = \frac{[\theta(t)-\theta(t-mT_s)]}{mT_s}$ and $\dot{x}(t) = \frac{[x(t)-x(t-mT_s)]}{mT_s}$, where m is an integer greater than one (chosen as 2 by experimentation). In the safety, baseline, and experimental controller automata, this first-order approximation is accomplished by storing a buffer of previous sampled values. It is more logical to do this computation here than in the sensor automaton as this calculation is done in the controllers of the implemented system.

1) Discretization: As the system is implemented in embedded software, the system model needs to be discretized. In continuous-time the system solution is of Equation 9, while the discrete solution is of the form

$$X(t_0) = FX(t_0 - T_s) + Gu(t_0 - T_s) \tag{10}$$

where

$$F = e^{AT_s} \tag{11}$$

and

$$G = \int_0^{T_s} e^{A\tau} d\tau \tag{12}$$

To get back to linear state feedback form, take the control as

$$u(t_0) = KX(t_0 - T_s) \tag{13}$$

yielding the full form

$$X(t_0) = FX(t_0 - T_s) + GBKX(t_0 - 2T_s) \tag{14}$$

This allows easy state projection of the form in Equation 15, where n is the number of sampling periods to project forward.

$$X(t_0 + nT_s) = \tag{15}$$
$$FX(t_0 + (n-1)T_s) + GBKX(t_0 - (n-2)T_s)$$

This projection is done to reduce the error introduced from the various delays in the system, primarily the digital implementation delay (that we are measuring the present state at t_0 and controlling the next state at $t_0 + T_s$) and the aforementioned filter delay.

2) Feasible and Stabilizable Regions: The feasible region is the region defined by the aforementioned state constraints, so looking at only the X and $\dot{X}$ constraints would produce a rectangle. While $\dot{\theta}$ is unconstrained, since V_a is in fact constrained, a maximal value of $\dot{\theta}$ is generated. The stabilizable region is the region within which a given controller can stabilize the system. This region is determined by solving a linear matrix inequality (LMI) (see, e.g. [8]) of the determinant maximization form proposed by Vandenberghe in [9]. This paper utilizes the work of [10] and [11] to solve the LMI and generate the stabilizable region for each controller. The stabilizable region is inversely related to the magnitude of the gains, so larger gains produce a smaller stabilizable region. Thus, as the safety controller's stabilizable region corresponds to the largest stabilizable region, all other controllers will have stabilizable regions that are subsets of the safety controller's region. This makes intuitive sense, as larger gains will create larger oscillations in the solution, potentially causing the system to go outside of even the feasible region.

The stabilizable regions for x and $\dot{x}$ of each controller are seen in Figure 6 and the regions for θ and $\dot{\theta}$ of each controller are in Figure 7. The switching controller described later utilizes these regions to determine which gains to use. Solving the LMI problem gives a matrix P_σ for each controller, where checking $X^T P_\sigma X < 1$ shows which stabilizable region (for $\sigma \in [sc, bc, ec]$) the states are within currently.

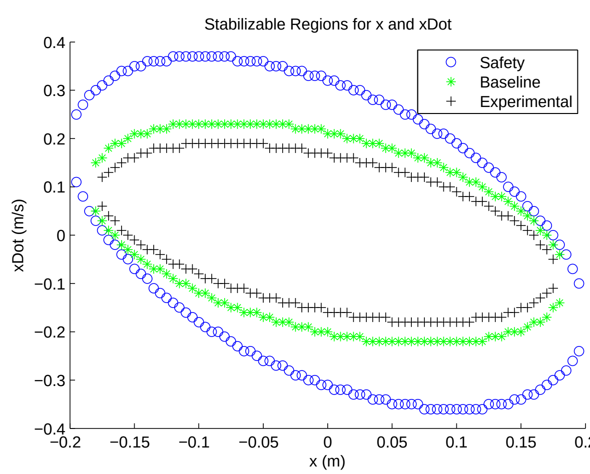

Fig. 6. Stabilizable Region for x and $\dot{x}$

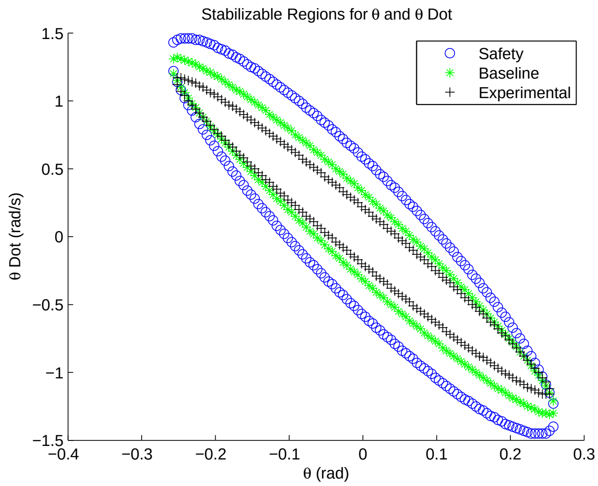

Fig. 7. Stabilizable Region for θ and $\dot{\theta}$

automaton SigmaController(K_σ : $\mathtt{Real}^{4\times 1}$, T_s : $\mathtt{Real}$, m : $\mathtt{Int}$)
signature
input sample(θ', x' : $\mathtt{Real}$)
output sigmaOutput(u'_σ : $\mathtt{Real}$)

variables
internal θ_σ : $\mathtt{Real} := 0$; $\dot{\theta}_\sigma$: $\mathtt{Real} := 0$;
x_σ : $\mathtt{Real} := 0$; $\dot{x}_\sigma$: $\mathtt{Real} := 0$;
u_σ : $\mathtt{Real} := 0$; rt : $\mathtt{Real} := 0$;
$next\_cycle$: $\mathtt{AugmentedReal} := T_s$;
buffer: $\mathtt{Seq}[prevTheta : \mathtt{Real}, prevX : \mathtt{Real}] := \{\}$;
let $time\_left := next\_cycle - rt$;
let $length := length(buffer)$

transitions
input sample(θ', x')
eff $buffer := buffer \vdash [\theta_\sigma, x_\sigma]$
$\theta_\sigma := \theta'$; $x_\sigma := x'$;

output sigmaOutput(u'_σ)
pre $rt = next\_cycle \wedge u'_\sigma = u_\sigma$
eff $next\_cycle := next\_cycle + T_s$;
$\dot{\theta}_\sigma := [\theta_\sigma - head(tail(buffer)).prevTheta]/(mT_s)$;
$\dot{x}_\sigma := [x_\sigma - head(tail(buffer)).prevX]/(mT_s)$;
$u'_\sigma := K_{\sigma 1} * x_\sigma + K_{\sigma 2} * \dot{x}_\sigma + K_{\sigma 3} * \theta_\sigma + K_{\sigma 4} * \dot{\theta}_\sigma$;
$buffer := tail(buffer)$;

trajectories
trajdef periodicSample
stop when $rt = next\_cycle$
evolve $d(rt) = 1$;

Fig. 8. Safety Controller

3) Safety, Baseline, and Experimental Controllers: The safety, baseline, and experimental controllers are of the exact same form as the sigma controller in Figure 8, except to replace $\sigma \in [sc, bc, ec]$, noting that they use different gain matrices, K_{sc} for the safety controller, K_{bc} for the baseline controller, and K_{ec} for the experimental controller. We take the gains as follows to maximize the stability region for the safety controller, and to improve the performance for the baseline and experimental controllers. In this form of control, increasing the gains will cause faster convergence with larger oscillations. These increased oscillations for the baseline and experimental controllers could take the system outside of the largest stabilizable region, hence, their stabilizable regions must be subsets of the largest stabilizable region, each decreased by a factor to ensure that the largest oscillations do not take the system outside of the largest stabilizable region.

$$K_{sc} = \begin{bmatrix} 6.0 \\ 20.0 \\ 60.0 \\ 16.0 \end{bmatrix} \tag{16}$$

$$K_{bc} = \begin{bmatrix} 8.0 \\ 32.0 \\ 120.0 \\ 12.0 \end{bmatrix} \tag{17}$$

$$K_{ec} = \begin{bmatrix} 10.0 \\ 36.0 \\ 140.0 \\ 14.0 \end{bmatrix} \tag{18}$$

4) Switching Controller: The switching controller determines which of the analytically redundant controllers to use for a given control cycle, based primarily on which stabilizable region the current and future states (determined by model-based state projection) of the system are within. If the state measurements show the system to be within only the safety controller's stabilizable region, then the safety controller gains will be used. Similarly, if the state is in the experimental stabilizable region, and will remain there for the next control cycle, the switching controller will utilize the experimental gains. The specific controller to be used must also be ready. The switching controller automaton can be seen in Figure 9. After the switching controller receives the desired control value from one of the analytically redundant controllers, a corresponding ready flag is set. Next, when the deadline is reached for the controller to output the new value to the motor, at T_s, the choice of controller is made. While in general there is a matrix inequality to solve here, showing that $X^T P X < 1$, it suffices to check that the control is within its constraints, as this model is linear, for an unconstrained control, it could stabilize the system from any initial condition in any amount of time. Checking the matrix inequality could also be done easily (as one just has to check the norms of each state variable summed together are less than one), but this method removes unnecessary clutter from the automaton model.

---

automaton SwitchingController$(u_{min}, u_{max}, T_s : \mathtt{Real})$
type
Mode $=$ Enumeration [*sc*, *bc*, *ec*]

signature
input safetyOutput$(u'_{sc} : \mathtt{Real})$
input baselineOutput$(u'_{bc} : \mathtt{Real})$
input expOutput$(u'_{ec} : \mathtt{Real})$
output switchingOutput$(u' : \mathtt{Real})$

variables
internal $u_{sw} : \mathtt{Real} := 0$; $rt : Real := 0$;
$next\_cycle : \mathtt{AugmentedReal} := T_s$;
let $time\_left := next\_cycle - rt$;
$u_{sc} : \mathtt{Real}$; $u_{bc} : \mathtt{Real}$; $u_{ec} : \mathtt{Real}$;
$ready_{sc} : \mathtt{Bool} := false$; $ready_{bc} : \mathtt{Bool} := false$;
$ready_{ec} : \mathtt{Bool} := false$; $mode : Mode := sc$;

transitions
input safetyOutput(u'_{sc})
eff $u_{sc} := u'_{sc}$;
$ready_{sc} := true$;

input baselineOutput(u'_{bc})
eff $u_{bc} := u'_{bc}$;
$ready_{bc} := true$;

input expOutput(u'_{ec})
eff $u_{ec} := u'_{ec}$;
$ready_{ec} := true$;

output switchingOutput(u')
pre $now_s = next\_cycle \wedge (ready_{sc} \vee ready_{bc} \vee ready_{ec})$
eff $next\_cycle := next\_cycle + T_s$;
if $ready_{ec} \wedge (u_{ec} \in [u_{min}, u_{max}])$
$u_{sw} := u_{ec}$;
else if $ready_{bc} \wedge (u_{bc} \in [u_{min}, u_{max}])$
$u_{sw} := u_{bc}$;
else
$u_{sw} := u_{bc}$;
$ready_{sc} := false$;
$ready_{bc} := false$;
$ready_{ec} := false$;

trajectories
trajdef periodicControl
stop when $rt = next\_cycle$
evolve $d(rt) = 1$;

Fig. 9. Switching Controller

III. Stability Analysis

The stability analysis follows from classical Lyapunov stability (see, e.g. [12] or [13]), and extends to the small-gain stability of systems with disturbances, such as in [3].

A. Lyapunov Analysis

From classical Lyapunov stability, we know that for linear time-invariant systems, if there exists some positive definite P that solves the Lyapunov equation in Equation 19 for some positive definite Q

$$PA + A^T P + Q = 0 \tag{19}$$

then the system is asymptotically stable and a function V defined by Equation 20 is a Lyapunov function.

$$V = X^T P X \tag{20}$$

For our system, we have the following P matrices

$$P_{sc} = \begin{bmatrix} 1.8030 & 2.3010 & 0.8447 & 0.3839 \\ 2.3010 & 10.0825 & 2.8171 & 1.4036 \\ 0.8447 & 2.8171 & 1.0109 & 0.4782 \\ 0.3839 & 1.4036 & 0.4782 & 0.2606 \end{bmatrix} \tag{21}$$

$$P_{bc} = \begin{bmatrix} 2.6836 & 2.7166 & 1.1849 & 0.5533 \\ 2.7166 & 10.0974 & 3.0287 & 1.4297 \\ 1.1849 & 3.0287 & 1.1695 & 0.5365 \\ 0.5533 & 1.4297 & 0.5365 & 0.2645 \end{bmatrix} \tag{22}$$

$$P_{ec} = \begin{bmatrix} 1.6987 & 1.8877 & 0.7664 & 0.3461 \\ 1.8877 & 8.8812 & 2.5537 & 1.1747 \\ 0.7664 & 2.5537 & 0.9601 & 0.4318 \\ 0.3461 & 1.1747 & 0.4318 & 0.2050 \end{bmatrix} \tag{23}$$

and then the corresponding Lyapunov functions $V_{sc} = X^T P_{sc} X$, $V_{bc} = X^T P_{bc} X$, and $V_{ec} = X^T P_{ec} X$. Figures 10, 11, and 12 show the Lyapunov functions along the system trajectory from initial conditions of $X_0 = \begin{bmatrix} -0.170 & 0.130 & -0.258 & 1.170 \end{bmatrix}^T$.

B. Small-Gain Theorems

Small-gain theorems allow us to bound the error introduced by the delay to talk about a system without delay, and instead with a disturbance injection. Writing the delayed control as

$$u(t) = KX(t - \tau) = KX(t) + \theta(t) \tag{24}$$

where $\theta(t) = KX(t - \tau) - KX(t)$ is the error from delay.

Now writing the system model again with this delay error as a disturbance input we have

$$\dot{X}(t) = (A + BK) X(t) + B\theta(t) \tag{25}$$

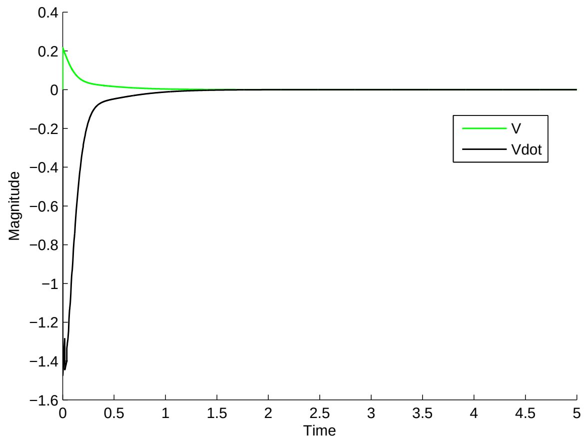

Fig. 10. Lyapunov Function for Safety Controller

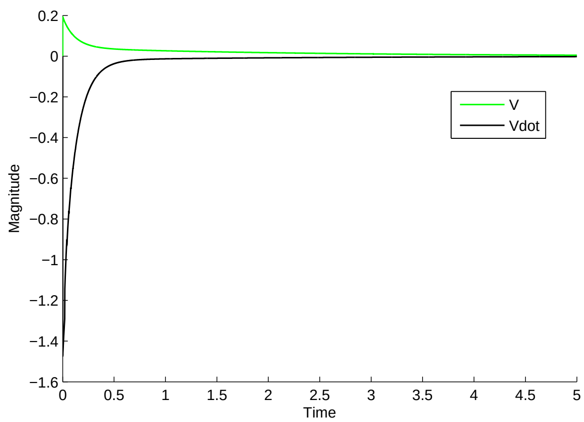

Fig. 11. Lyapunov Function for Baseline Controller

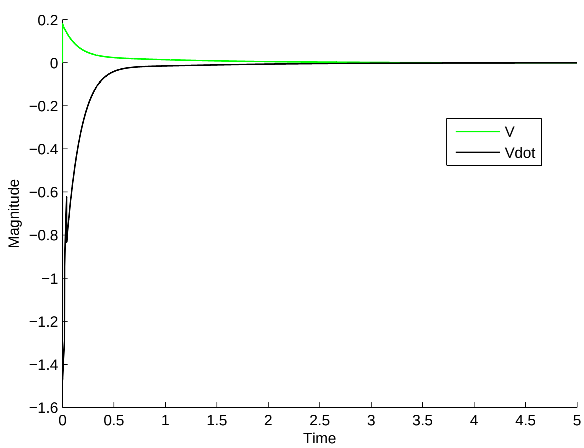

Fig. 12. Lyapunov Function for Experimental Controller

The Lyapunov function time-derivative changes in the form of

$$\dot{V} = -X^T QX + X^T PB\theta \tag{26}$$

where P and Q are the same functions defined above in Equation 19.

We then apply the conservative bound

$$\dot{V} \leq -X^T QX + |X| \cdot |\theta| \cdot ||PB|| \tag{27}$$

which can be computed as

$$\dot{V} \leq -\lambda_{min}\left(Q\right)|X|^2 + |X| \cdot |\theta| \cdot ||PB|| \tag{28}$$

where $\lambda_{min}\left(Q\right)$ is the smallest eigenvalue of Q.

$$\dot{V} = -|x|\left(\lambda_{min}\left(Q\right)|x| - ||PB|| \cdot |\theta|\right) \tag{29}$$

$$|x| > \frac{||PB||}{\lambda_{min}\left(Q\right)} \cdot |\theta| \tag{30}$$

Define $\rho\left(r\right) = cr$, and take

$$c = \frac{||PB||}{\lambda_{min}\left(Q\right)} \tag{31}$$

Writing θ now as

$$\theta = -\int_{t-\tau}^{t} K\left(AX\left(s\right) + BKX\left(s-\tau\right)\right) ds \tag{32}$$

we can now bound θ as

$$|\theta| \leq \tau\left(||KA|| + ||KBK||\right) \cdot ||X||_{[t-2\tau,t]} \tag{33}$$

Defining a constant d as this bound

$$d = \left(||KA|| + ||KBK||\right) \tag{34}$$

The small-gain theorem states then, assume there exists some τ such that

$$\tau cd < 1 \tag{35}$$

then we can define the maximum tolerable delay as

$$\tau < \frac{1}{cd} \tag{36}$$

Following this definition for τ, we compute τ_σ for each of the controllers, yielding $\tau_{sc} = 2.03$, $\tau_{bc} = 0.995$, and $\tau_{ec} = 0.684$ milliseconds of delay is tolerable for each of the respective controllers to still ensure stability. Note that all of these tolerable delays are less than one control cycle period of $T_s = 20$ milliseconds. This result used a rather conservative bound in Equation 28, which looking at the work of [14] could perhaps improve. This result has not included state projection of the type defined in Equation 15, which is described next.

IV. Results

Our analysis of the inverted pendulum system using the small-gain theorem indicates that the model-based state projection used in the controller is necessary for stability. The maximum delay the 'pure' system (without state projection) can tolerate is less than the sampling period. Tolerable delay is inversely related to gain, that is, it is directly related to the size of the stabilizable region. Given that the size of the stabilizable region is directly related to magnitude of the stability margin. This makes intuitive sense, and shows that because the safety controller has the largest stabilizable region, it can thus tolerate the largest disturbances, in this case, a disturbance caused by a delay.

To remove the delay caused by filtering and digital implementation, one must employ model-based state projection. If all measurements were perfect, this projection would remove all effects of the measurement errors. Of course however they are not, especially when considering the approximations for $\dot{x}$ and $\dot{\theta}$ which could potentially have large error. There are some obvious solutions from classical control theory to solve these problems. One could easily use an observer and reform the system to remove much of this linearization error, so instead of approximating the derivative, it is in effect actually taken. Similarly, more advanced estimators could be employed, of note is the Kalman filter used to remove the effects of measurement noise. If then considering the nonlinear case, one could utilize the extended Kalman filter, as the $\sin(\theta)$ terms are differentiable. Assuming optimal filtering of this form, with state projection, the system can tolerate $nT_s + \tau_\sigma$ delay, where n is the number of sampling cycle periods to project forward.

A large amount of work for this analysis was spent looking at system trajectories, so the Figures 13 and 14 show the system stabilizing from an initial condition of $X_0 = \begin{bmatrix} -0.170 & 0.130 & -0.258 & 1.170 \end{bmatrix}^T$, which lies on the bounds of the experimental controller's stabilizable region, and utilizing switching significantly faster than what would most likely be seen in the real implementation (at every T_s). Finally, Figure 15 shows the Lyapunov function and its derivative along this trajectory.

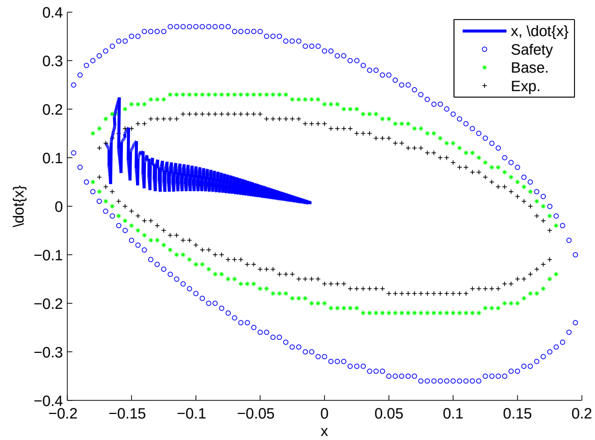

Fig. 13. Trajectory of x, $\dot{x}$ Switching Every T_s

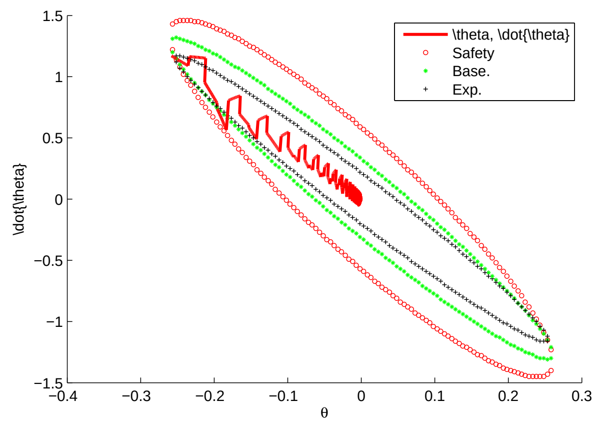

Fig. 14. Trajectory θ, $\dot{\theta}$ Switching Every T_s

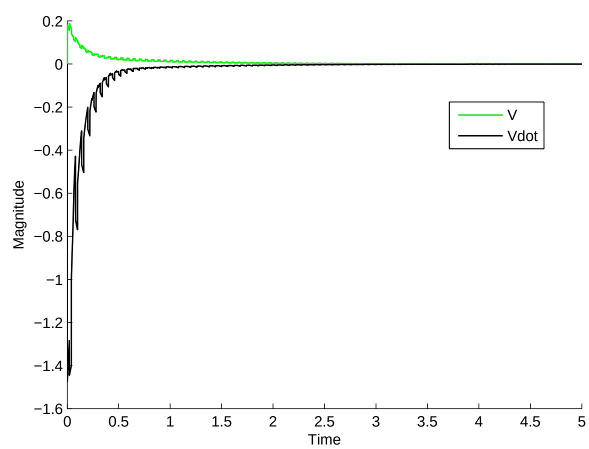

Fig. 15. Lyapunov Function for Switching Every T_s

V. Future Work

This work began with the desire to prove stability of the Simplex architecture switching controller in this case study. However, we have not quite yet proved this, although progress has been made. We theorize that the tolerable delay of the entire system is the minimum tolerable delay across all controllers, so that $\tau_\sigma = \tau_{ec}$.

We have begun looking at the small-gain result in the system with minor nonlinearities that can be linearized, such as the $sin(\theta)$ term which for small θ can be linearized to θ. Some derivation in this direction is as follows. Consider the nonlinear model,

$$\dot{X} = f(X, u) \tag{37}$$

We can then describe the equations of motion for the plant as

$$(m+M)\ddot{x} + \frac{1}{2}ml\cos(\theta)\ddot{\theta} - \frac{1}{2}ml\sin(\theta)\dot{\theta}^2 = F - f_c \tag{38}$$

$$\frac{1}{2}ml\theta\ddot{x} + \frac{1}{3}ml^2\ddot{\theta} - \frac{1}{2}mgl\sin(\theta) = -f_p \tag{39}$$

Skimming over the details of the motor model, we can achieve the entire system model including motor dynamics

$$\ddot{x} = \frac{1}{D}\left[\frac{1}{3}ml^2\left(B_lV_a - f_c - C_1\right) + \frac{1}{2}ml\cos\theta\left(f_p + C_2\right)\right] \tag{40}$$

$$\ddot{\theta} = \frac{1}{D}\left[-\frac{1}{2}ml\cos\theta\left(B_lV_a - f_c - C_1\right) - \bar{M}\left(f_p + C_2\right)\right] \tag{41}$$

where $D = \frac{1}{3}\bar{M}ml^2 - \frac{1}{4}m^2l^2\cos^2(\theta)$, $\bar{M} = \frac{m+M+(K_gJ_m)}{r^2}$, $C_1 = \bar{B}\dot{x} - \frac{1}{2}ml\sin(\theta)\dot{\theta}^2$, $C_2 = -\frac{1}{2}mgl\sin(\theta)$, $\bar{B} = \frac{K_gB_m}{r^2} + \frac{K_g^2K_iK_b}{r^2R_a}$, and $B_l = \frac{K_gK_i}{rR_a}$. The immediate next step is to derive a Lyapunov function for this system, and when considering the earlier form of $V = X^TPX$, one could interchange the bounds on the θ states with $a\cos(\theta)$ where a is some constant (since now we are considering $\sin(\theta)$). After checking that this P is positive definite and satisfies $A^TP + PA = -Q$, we could then proceed with our analysis as before.

With either the nonlinear or linearized models, there seem to be several areas to investigate in terms of common Lyapunov functions, multiple Lyapunov functions, and dwell-time, all seen in [15]. So another step is an investigation of the average-dwell time (ADT) of the switching between the safety, baseline, and experimental controllers to see if ADT relates to the small-gain delay and stability regions. Given the large length of time between switches, $T_s = 20$ms, it may be possible to find a dwell-time less than this to prove overall stability.

As far as long-term future work, for the verification of the model to most closely match the real system, analyzing both the measurement and actuation errors from quantization and delay (from analog-to-digital conversion and digital-to-analog conversion, respectively) would provide a complete answer of the disturbances introduced by these non-ideal blocks. There is also another avenue potentially to follow in the area of software verification through the co-stability concept.

VI. Conclusion

This case study of the inverted pendulum with a controller following the Simplex architecture displays that recent results in small-gain theorems can be useful in practical systems. Since the maximum tolerable delay found for stability was less than the control period length, it was necessary to project the state forward to the next control cycle to guarantee stability. Thus, this work has shown that the system is still stable even with small delays introduced by measurement, so a stronger stability result was found for this system.

Acknowledgment

The author would like to thank Sayan Mitra for helpful commentary and feedback throughout the work, Qixin Wang for guidance in working with the real inverted pendulum system, Lui Sha for revealing details not seen in his earlier work, and Daniel Liberzon for providing a clear description of small-gain theorems.

References

[1] D. Seto, B. Krogh, L. Sha, and A. Chutinan, "The simplex architecture for safe on-line control system upgrades," in *Proc. American Control Conference*, Philadelphia, PA, Jun. 1998, pp. 3504–3508.

[2] D. Seto and L. Sha, "A case study on analytical analysis of the inverted pendulum real-time control system," Carnegie Mellon Univ., Pittsburgh, PA, CMU/SEI Tech. Rep. 99-TR-023, Nov. 1999.

[3] D. Liberzon, "Quantization, time delays, and nonlinear stabilization," *IEEE Trans. Autom. Control*, vol. 51, no. 7, pp. 1190–1195, Jul. 2006.

[4] R. Sanfelice and A. Teel, "On hybrid controllers that induce input-to-state stability with respect to measurement noise," in *Proc. 44th IEEE Conf. on Decision and Control*, Seville, Spain, Dec. 2005, pp. 4891–4896.

[5] N. Lynch, R. Segala, and F. Vaandrager, "Hybrid i/o automata," *Inf. Comput.*, vol. 185, no. 1, pp. 105–157, 2003.
[6] S. Mitra, "A verification framework for hybrid systems," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA 02139, Sep. 2007. [Online]. Available: http://users.crhc.uiuc.edu/mitras/research/thesis.pdf
[7] M. Bodson, J. Lehoczky, R. Rajkumar, L. Sha, and J. Stephan, *Analytic redundancy for software fault-tolerance in hard real-time systems*. Kluwer Academic Publishers, 1994.
[8] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, ser. Studies in Applied Mathematics. Philadelphia, PA: SIAM, Jun. 1994, vol. 15.
[9] L. Vandenberghe, S. Boyd, and S.-P. Wu, "Determinant maximization with linear matrix inequality constraints," *SIAM J. Matrix Anal. Appl.*, vol. 19, no. 2, pp. 499–533, 1998.
[10] J. Lfberg, "Yalmip : A toolbox for modeling and optimization in MATLAB," in *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. [Online]. Available: http://control.ee.ethz.ch/~joloef/yalmip.php
[11] K. C. Toh, M. J. Todd, and R. H. Tutuncu, "Sdpt3 a matlab software package for semidefinite programming," *Optimization Methods and Software*, vol. 11, pp. 545–581, 1999.
[12] C. Chen, *Linear System Theory and Design*, 3rd ed. New York, NY: Oxford University Press, 1999.
[13] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2002.
[14] E. Fridman, M. Dambrine, and N. Yeganefar, "Input to state stability of systems with time-delay: A matrix inequalities approach," *Automatica*, vol. 44, no. 9, pp. 2364–2369, 2008.
[15] D. Liberzon, *Switching in Systems and Control*. Boston, MA: Birkhuser, 2003.