Compositionaly Analyzing a Proportional-Integral Controller Family

Ashish Tiwari

Abstract—We define always eventually region stability and then formulate the absolute always eventually region stability problem as the problem of finding a class of plants that a generic proportional-integral (PI) controller can always eventually stabilize. We use real quantifier elimination methods to solve the absolute always eventually region stability problem. The class of plants found in our solution includes nonlinear and switched plant models. Our analysis reveals that if a PI controller satisfies two assumptions of the form:
(a) the proportional gain $K_p$ and the integral gain $K_i$ satisfy an inequality, and
(b) the integral of the error term in the controller is saturated, then such a PI controller can be used to establish always eventually region stability for any plant that suitably responds to the control input. Such a result is useful for compositionally verifying a system consisting of a plant and a controller.

I. INTRODUCTION

Any complex cyber-physical system will necessarily contain one or more controllers. One of the most commonly used controller is a proportional-integral (PI) or a proportional-integral-derivative (PID) controller. In this paper, we use formal verification techniques to analyze a generic PI controller controlling a generic plant as shown in Figure 1.

Our investigation is motivated by two reasons. First, there is recent push in exploring the idea of building complex cyber-physical systems compositionally. The hope is that there would be a library of components from which one could pick components and connect them to build complex systems. The ability to analyze systems compositionally is crucial for making progress on this vision. Since controllers will necessarily be part of any such library, we need verification techniques that can analyze individual components, such as a controller. The second reason comes from the need to build scalable techniques for formally verifying complex systems. One way of achieving scalability is to perform verification compositionally. Hence, we need approaches for analyzing each component separately in such a way that the analysis results can be used later to verify the composed system.

The current practice in verification of complex cyber-physical systems is based on performing extensive (numerical) simulation and testing. However, simulation-based methods are incomplete — how do we know that the system has been tested enough? In the past few years, there has been an extensive push for extending formal verification approaches to also verify physical and cyber-physical systems. Broadly speaking, these techniques can be classified as follows:

1) reach-set methods that compute the set of all reachable states of the system, either exactly [15], or approximately [6], [29], [14], [9], [12], [10]
2) abstraction-based methods that first abstract the system and then analyze the abstraction [28], [1], [7], [2], [24]
3) certificate-based methods that directly search for certificates of correctness (such as inductive invariants and Lyapunov functions) of systems [25], [20], [22], [21], [11], [1], [26], [3], [19]

While all these techniques have had some success, the certificate-based methods are turning out to be particularly effective in proving deep properties of complex systems. Certificate-based methods work by fixing a template for the “certificate of correctness”, and casting the verification problem as a problem of finding an appropriate instantiation of the template. This search is accomplished using numeric or symbolic solvers that reason about arithmetic constraints in the theory of reals. For example, for verifying stability, the user can provide a template for the Lyapunov function (say a quadratic Lyapunov function) and the solver can find a concrete quadratic function that proves stability of the given system. Most commonly used solvers include ones for sums-of-squares (SOS) programming [21] and real quantifier elimination. Numeric approaches, such as SOS programming, have two limitations: first, they may give incorrect answers, and second, they have their limitations when solving Boolean combination of constraints. We use symbolic solvers in the form of real quantifier elimination in this paper.

Motivated by the need to analyze systems compositionally, the question we answer in this paper is the following: what is the class of plant models that can be stabilized by a given PI controller. This problem is related to the so-called absolute stability problem [30], [17], but there are important differences that we will discuss later. We focus on a system consisting of a PI controller driving a plant to a given setpoint. We wish to prove that the error — difference between the setpoint and the observed state — always eventually falls below a given constant. We formalize this notion as always eventually region stability. The problem of finding the class
of plant models can be formulated as the **absolute always eventually region stability problem**. Our approach for finding the class of plant models is based on using templates to describe the dynamics of all the possible plants, and then deriving conditions on the parameters in the plant model. Specifically, we find that PI controllers can guarantee always eventually region stability for nonlinear and switched plant models.

The outline of the paper is as follows. Section II describes the main technical result. Section III illustrates the main result using some concrete examples. The tools used for performing the real quantifier elimination are described in Section IV. Comparison to work on absolute stability is presented in Section V.

II. PI CONTROLLER: COMPOSITIONAL REASONING

A proportional-integral, or PI (respectively, proportional-integral-derivative or PID), controller is a generic controller that can be instantiated to give a feedback controller by fixing two (respectively, three) parameters. It is widely used in the design of industrial control systems.

Our goal is to formally verify a PI controller. A PI controller cannot be verified in isolation, and we need a model of the plant (process) that the PI controller is controlling to formally state and prove its safety and/or stability. Here, we assume that we do not have access to the plant model, and hence the goal is to verify the PI controller under suitable generic assumptions about the plant.

Consider a plant (process) that needs to be driven to some desired state. Suppose the state of the plant is given by the values of \( n \) real-valued variables. The state of the plant is, hence, a point in \( \mathbb{R}^n \). Assume that we need to drive the plant to a state such that a given variable, say \( Y \), takes a particular value, say \( y_{sp} \in \mathbb{R} \) (sp refers to the desired setpoint). If \( y \in \mathbb{R} \) is the (estimate of the) current value of the variable \( Y \) in the plant, then the error, \( err \), is given by

\[
err = y - y_{sp} \tag{1}
\]

Let \( intern \) denote the integral of this error, but saturated to stay within some range, say \([-1, 1]\); that is,

\[
\frac{d intern}{dt} = \begin{cases} 
err & \text{if } intern^2 = 1 \\
err \cdot intern & \text{if } intern^2 < 0 \\
0 & \text{otherwise}
\end{cases} \tag{2}
\]

The symbol \( \land \) denotes conjunction (and). Note that the above dynamics for \( intern \) guarantees that it stays in the range \([-1, 1]\). Saturation blocks that restrict a signal to a specific range, such as the one modeled above, are commonly used in engineering designs. A PI controller computes the control input for the plant as a weighted sum of the \( err \) and \( intern \).

\[
u = K_p \cdot err + K_i \cdot intern \tag{3}
\]

where the constants \( K_p \) and \( K_i \) are chosen by the control designer. Let us say we are given \( K_p = 500 \) and \( K_i = 10 \). We will later analyze the controller for symbolic \( K_p \) and \( K_i \).

The description of the PI controller is contained in Equation 1, Equation 2, and Equation 3. However, to enable any analysis, we need to relate the control input \( u \) to the plant and the output \( y \) of the plant. In the spirit of **assume-guarantee** reasoning, we assume a very generic plant whose dynamics, when projected onto the variable \( Y \), can be abstracted to the following **nondeterministic** differential equation

\[
\frac{dy}{dt} = \beta - \alpha \cdot u \\
\alpha \in [a, b] \\
\beta \in [a_1, b_1] \tag{4}
\]

where \( \alpha \) and \( \beta \) can vary nondeterministically in the interval \([a, b]\) and \([a_1, b_1]\) respectively. In other words, \( \frac{da}{dt} \) and \( \frac{db}{dt} \) are both unrestricted and can be anything as long as \( \alpha, \beta \) remain in the specified range.

Note that the output \( y \) of the plant will, in general, depend on the internal state \( (\vec{x}) \) of the plant. Hence, \( \alpha \) and \( \beta \) are actually functions \( \alpha(\vec{x}) \) and \( \beta(\vec{x}) \) of the internal state \( \vec{x} \) of the plant. Since we do not assume anything about the dynamics of \( \vec{x} \), we abstract the functions \( \alpha(\vec{x}) \) and \( \beta(\vec{x}) \) by variables \( \alpha \) and \( \beta \) and impose no restrictions on the dynamics for \( \alpha \) and \( \beta \).

The closed loop control system is now described by the union of Equation (1), Equation (2), Equation (3), and Equation (4).

For such a closed loop system, we are interested here in a version of stability we call **always eventually region stability**.

**Definition 1 (always eventually region stability):** A system is **always eventually region stable (a.e.r.s)** with respect to a region \( R \subseteq \mathbb{R}^n \) if, for any trajectory \( s(\vec{x}_0, t) \) starting from any initial state \( \vec{x}_0 \in \mathbb{R}^n \) of the system, it is the case that

\[
\forall t \geq 0 : \forall \epsilon > 0 : \exists t' \geq t : s(\vec{x}_0, t') \in R^\epsilon
\]

where \( R^\epsilon \) is the set of points that are at most \( \epsilon \) distance away from \( R \).

When the initial state \( \vec{x}_0 \) of the system can be arbitrary, then the above condition is equivalent to

\[
\forall \vec{x}_0 \in \mathbb{R}^n : \forall \epsilon > 0 : \exists t' \geq 0 : s(\vec{x}_0, t') \in R^\epsilon
\]

This is similar to global asymptotic stability, except that (a) we consider a region \( R \) here and not a point, and (b) we do not require that the trajectory remain inside \( R^\epsilon \) at all times after \( t' \). Such a weak notion of region stability is useful for two reasons: first, it naturally allows us to say that the error \( e \) always eventually falls below a given constant, say 1, and second, it helps in getting simpler criteria for the absolute always eventually region stability problem formalized below.

**Definition 2 (Absolute a.e.r.s problem):** Given an open system, such as the one specified in Equation (1), Equation (2), and Equation (3), the problem is to derive conditions involving only \( a, b, a_1, b_1 \) that will guarantee that the system is always eventually region stable with respect to a given region, say \( err \in [-1, 1] \), for every plant that satisfies Equation (4).
Solving the always eventually region stability problem

We use the certificate-based verification approach to solve the absolute a.e.r.s problem. The certificate here is a Lyapunov-like function. Specifically, we establish the existence of a common Lyapunov function that works for all possible plants that satisfy the (unknown) plant assumptions. We fix a form of the Lyapunov function, say, $V = c \cdot \text{err}^2 + d \cdot \text{interr}^2$ (5)

We have to simultaneously find the assumptions on the plant – namely, values $a, b, a_1$ and $b_1$ – and the certificate for the closed-loop system – namely, values $c, d$ – such that a.e.r.s stability holds. We find these values by solving the exists-forall formula $\phi_0$ in Equation 6 below. Before we present the formula, let us fix some definitions. Note that we have two modes in the closed-loop system based on whether $\text{interr}$ is saturated or not.

1) In $\text{Mode}_1$, $\text{interr}$ is not saturated. In this case, the dynamics are given by

$$d\text{interr}/dt := \text{err}$$
$$d\text{err}/dt := \beta - \alpha * (K_p * \text{err} + K_i * \text{interr})$$

with the mode invariant

$\text{Inv}_1 := (\text{interr}^2 \leq 1 \land (\text{interr}^2 < 1 \lor \text{err} \cdot \text{interr} < 0) \land a \leq a_1 \land b_1 \geq b)$

2) In $\text{Mode}_2$, $\text{interr}$ is saturated. In this case, the dynamics are given by

$$d\text{interr}/dt := 0$$
$$d\text{err}/dt := \beta - \alpha * (K_p * \text{err} + K_i * \text{interr})$$

with the mode invariant

$\text{Inv}_2 := (\text{interr}^2 \geq 1 \land \text{err} \cdot \text{interr} \geq 0 \land a \leq a_1 \land b_1 \geq b)$

Note that the mode invariants are determined by the saturation conditions of $\text{interr}$ (see Equation (2)).

Let $\frac{dV}{dt}_{\text{Mode}_i}$ denote the Lie derivative of $V$ in $\text{Mode}_i$. It is obtained by symbolically differentiating the expression for $V$ with respect to time in the two modes. Specifically, $\frac{dV}{dt}$ in $\text{Mode}_2$ can be obtained by replacing $d\text{err}/dt$ and $d\text{interr}/dt$ in the following expression by their respective definitions in $\text{Mode}_i$:

$$\frac{dV}{dt} := 2 \cdot c \cdot \text{err} \cdot \frac{d\text{err}}{dt} + 2 \cdot d \cdot \text{interr} \cdot \frac{d\text{interr}}{dt}$$

Now, we are ready to write the formula that encodes the absolute a.e.r.s problem. The formula below states that there exist some values for $a, b, a_1, b_1, c, d$ (parameters of the nondeterministic plant and the parametric Lyapunov function) such that $V$ is a Lyapunov function that proves a.e.r.s stability of the system.

$$\phi_0 := \exists a, b, a_1, c, d : \phi_1$$
$$\phi_1 := \forall \text{err, interr}, \alpha, \beta : \phi_2$$
$$\phi_2 := b > a \land b_1 > a_1 \land c \geq 0 \land d \geq 0 \land \phi_3$$
$$\phi_3 := (\text{err}^2 \geq 1 \land \text{Inv}_1 \Rightarrow \left| \frac{dV}{dt} \right|_{\text{Mode}_1} < 0) \land (\text{err}^2 \geq 1 \land \text{Inv}_2 \Rightarrow \left| \frac{dV}{dt} \right|_{\text{Mode}_2} < 0)$$

Here $\Rightarrow$ denotes logical implication. Since we are interested in finding the plant parameters, $a, b, a_1, b_1$, we will find the constraints on $a, b, a_1, b_1$ such that if some concrete values for $a, b, a_1, b_1$ satisfy those constraints, then those values will make the Formula $\exists c, d : \phi_1$ true. This is precisely what quantifier elimination achieves.

The correctness of our approach relies on the following proposition.

**Proposition 1:** Let $\psi_1(a, b, a_1, b_1)$ be a formula containing only the variables $a, b, a_1, b_1$ that is obtained as a result of eliminating quantifiers from Formula $\exists c, d : \phi_1$, where $\phi_1$ is defined in Equation 6; that is,

$$\psi_1(a, b, a_1, b_1) \leftrightarrow \exists c, d : \phi_1$$

Then, $\psi_1$ is a solution for the absolute a.e.r.s problem.

**Proof:** (Sketch) Suppose $a, b, a_1, b_1$ are concrete values that satisfy $\psi_1$. We do a proof by contradiction, and assume a.e.r.s stability does not hold for this choice of $a, b, a_1, b_1$. Then, starting from some initial state and for some plant that satisfies Equation 4, the system trajectory never reaches $\text{err}^2 \leq 1$. This means, $\text{err}^2 > 1$ always. By $\phi_3$, $\frac{dV}{dt} < 0$ always, and hence $\text{err}$ gets arbitrarily close to 0, which contradicts that $\text{err}^2 > 1$ always.

**Analysis Results**

We use symbolic tools that perform real quantifier elimination to eliminate quantifiers from Formula $\phi_1$ shown above in Equation 6. Details of the specific tools and techniques used can be found in Section IV. Here we present the results obtained by analyzing Formula $\phi_1$ above.

We eliminate all the $\forall$ quantified variables from Formula $\phi_1$. The result is a formula containing only the remaining variables $a, b, a_1, b_1, c, d$ that is equivalent to Formula $\phi_1$. There are automatic tools that perform quantifier elimination. The following formula, slightly rewritten to fit margins, is returned by the tool.

$$\psi_1 := a_1 \leq b_1 \land 0 < a \leq b \land ((10 \cdot c \cdot a \geq d \land \{c \cdot a_1, c \cdot b_1\} \leq \{490 \cdot c \cdot a + d\}) \lor (10 \cdot c \cdot a < d \land \{c \cdot a_1, c \cdot b_1\} \leq \{510 \cdot c \cdot a - d\}))$$

Here $|x|$ denotes the absolute value of $x$ and $\{a, b\} \leq x$ denotes $a \leq x \land b \leq x$. Formula $\psi_1$ gives conditions on the parameters in the plant model, along with conditions on the coefficients $c, d$ of the Lyapunov function. If we existentially quantify $c, d$ in the above formula, and ask the quantifier
elimination tool to eliminate \( c, d \) from \( \exists c, d : c \geq 0 \land d \geq 0 \land \psi_1 \), then the tool returns the formula \( \psi_0 \) shown below.

\[
\psi_0 := b_1 - 500 \ast a \leq 0 \land a_1 + 500 \ast a \geq 0
\]

Formula \( \psi_0 \) is now a solution of the absolute a.e.r.s problem.

**Theorem 1**: If there is a constant \( a > 0 \) such that the dynamics of a plant can be shown to satisfy the following assumptions,

\[
\frac{dy}{dt} = \beta - a \ast u \\
\alpha \in [a, +\infty), a > 0 \\
\beta \in [-500 \ast a, 500 \ast a]
\]

then, when this plant is composed with a PI controller

\[
u = 500 \ast \text{err} + 10 \ast \text{interr}
\]

\[
\frac{d\text{interr}}{dt} = \text{err}, \text{interr} \text{ saturates to } [-1, 1]
\]

the resulting system is always eventually region stable with respect to the region err\(^2 \leq 1\).

Note that the plant can be nonlinear, and can even have switched dynamics, and still satisfy the conditions of Theorem 1. This is because \( \alpha, \beta \) can change *dynamically* – they need not even be continuous – as long as they satisfy the conditions in the theorem above. Thus, Theorem 1 is applicable to a very large class of plants.

We note here that if we remove the saturation module from the PI controller, then we are unable to prove a.e.r.s stability for such a class of nondeterministic plant models. (that is, where \( \frac{d\alpha}{dt} \) and \( \frac{d\beta}{dt} \) are unrestricted as long as \( \alpha, \beta \) remain inside the given bounds). Without the saturation module, we can, of course, prove stability for many fixed plant models (that is, where \( \frac{d\alpha}{dt} = \frac{d\beta}{dt} = 0 \), but assuming \( \frac{d\alpha}{dt} = \frac{d\beta}{dt} = 0 \) significantly narrows the class of plants and hence it is not very useful for performing compositional reasoning on systems.

**Generic PI Controller**

We can now generalize from a particular value of \( K_p \) and \( K_i \) and instead use symbolic values for these gains. We can now ask the question for what values of \( K_p \) and \( K_i \) does the same solution of the absolute a.e.r.s problem work?

We solve this problem by formulating it as a quantifier elimination problem. In fact, we can find the desired conditions on \( K_p \) and \( K_i \) by eliminating the quantified variables from the following Formula \( \phi_4 \):

\[
\phi_4 := \exists a, c, d : a > 0 \land c \geq 0 \land d \geq 0 \land \phi_5
\]

\[
\phi_5 := \forall \text{err, interr}, \alpha, \beta : \phi_3
\]

where \( \phi_3 \) is as defined in Equation 6 except for two differences: first, Inv\(_1\) and Inv\(_2\) are now given by

\[
\text{Inv}_1 := \text{Inv}_0 \land \text{interr}^2 \leq 1 \land \\
(\text{interr}^2 = 1 \Rightarrow \text{err} \ast \text{interr} < 0)
\]

\[
\text{Inv}_2 := \text{Inv}_0 \land \text{interr}^2 = 1 \land \text{err} \ast \text{interr} > 0
\]

\[
\text{Inv}_0 := (a < \alpha \land -500 \ast a < \beta \land \beta < 500 \ast a)
\]

That is, we are considering only the class of plants for which the concrete PI controller worked. As a consequence, the variables \( a_1, b_1, b \) were present in Formula \( \phi_1 \) in Equation 6, but they are no longer present in Formula \( \phi_4 \) in Equation 8.

The second difference between Formula \( \phi_1 \) in Equation 6 and Formula \( \phi_4 \) in Equation 8 is that \( K_p \) and \( K_i \) were replaced by their fixed values (500 and 10 respectively) in the former, whereas variables \( K_p \) and \( K_i \) are still present in the latter.

We use real quantifier elimination to first eliminate the universal quantifiers from Formula \( \phi_1 \) in Equation 6. We get the following equivalent formula \( \psi_5 \) in Equation 8.

\[
\psi_5 := K_p \geq K_i \land c \ast a \ast (K_p - 500) \geq |c \ast a \ast K_i - d|
\]

Now we eliminate the existential variables from Formula \( \phi_4 \) and get the following Formula \( \psi_4 \) as the answer.

\[
\psi_4 := K_p \geq 500 \land K_p \geq K_i \land K_p + K_i \geq 500
\]

Thus, our solution for the absolute a.e.r.s problem works for any PI controller that satisfies this condition. Note that our concrete choice \( K_p = 500 \) and \( K_i = 10 \), that we had used to obtain results of the previous section, satisfies this condition.

**Theorem 2**: Suppose \( K_p, K_i \) satisfy Formula \( \psi_4 \), and suppose \( a > 0 \), \( b = +\infty \), \( a_1 = -500 \ast a \) and \( b_1 = 500 \ast a \) in Equation 4. Then, the feedback control system defined by Equation 1, Equation 2, Equation 3, and Equation 4 always eventually reaches a state where err\(^2 \leq 1\).

It is folklore in control theory that PI controllers can stabilize “almost any plant”. The above result is one possible formulation of this fact, which we have derived using advanced quantifier elimination techniques.

**III. Experiments**

We present simulation plots in this section to illustrate the results above.

First consider a PI controller that has fixed parameters, namely, \( K_p = 500 \) and \( K_i = 10 \). We consider three different plant models. The first plant model switches between two different dynamics every 0.1 time units. We only give the value of the parameters \( \alpha \) and \( \beta \) of the plant in Equation 4 here. Between time \( t = 0 \) to \( t = 2.5 \) units, the plant switches between the following modes.

\[
\text{Mode}_1 : \alpha = 10^{-5}, \beta = 5 \ast 10^{-3}
\]

\[
\text{Mode}_2 : \alpha = 5 \ast 10^{-3}, \beta = -2.5
\]

We are assuming \( \frac{d\alpha}{dt} = \frac{d\beta}{dt} = 0 \) inside each mode. Between time \( t = 2.5 \) to \( t = 5 \) units, the plant switches between the following modes.

\[
\text{Mode}_3 : \alpha = 10^{-5}, \beta = -5 \ast 10^{-3}
\]

\[
\text{Mode}_4 : \alpha = 5 \ast 10^{-3}, \beta = 2.5
\]

Note that the plant parameters satisfy the condition \( \psi_0 \), and hence Theorem 1 tells us that the error will always eventually approach the range \([-1, 1]\) for this case. Figure 2(a) presents
Now, since \( a > 0 \), we have that \( A \) is Hurwitz, \((A,B)\) is controllable, and \((C,A)\) is observable (if \( K_i \neq 0 \)). We can

IV. QUANTIFIER ELIMINATION TOOLS

Quantifier elimination refers to the problem of finding a quantifier-free formula that is equivalent to a quantified formula. For example, \( \exists x : y = x^2 \) is equivalent to the quantifier-free formula \( y \geq 0 \) (in the theory of reals). In the theory of reals, for any given quantified formula, a quantifier-free equivalent formula always exists. In fact, it can even be computed algorithmically.

Real quantifier elimination is computationally hard. In our experiments, we used a new approach [23] that performs quantifier elimination by combining (1) virtual substitution method of redlog [31], [16], [32], (2) the simplifier slfq [4], and (3) the procedure qepcad [8], [5]. In all our quantifier elimination runs, the time taken by virtual substitution was negligible. In the first quantifier elimination exercise where Formula \( \psi_1 \) was generated, the simplifier slfq took 61.1 seconds of system time and made 8669 calls to qepcad. The step of eliminating \( c, d \) from \( \psi_1 \) was performed using a single call to qepcad, which took 100 milliseconds of system time. In the next quantifier elimination exercise where we generated \( \psi_5 \), virtual substitution took negligible (at most 1 minute of real time) time, whereas slfq needed 205.87 seconds of system time and made 7779 calls to qepcad. Finally, the quantifier elimination problem for generating \( \psi_4 \) was performed by a single call to qepcad and it again took negligible time. (All computations described here have been carried out on Intel Xeon E5630 2.53GHz single-core processor (x86_64 arch) with 4G RAM running Ubuntu Linux 2.6.32-26.) The scripts containing quantifier elimination problems generated in the experiments are publicly available [27].

V. ABSOLUTE STABILITY

Our results are analogous to the results on absolute stability and the Lur’e problem [30], [17]. In fact, in retrospect, now that we know the result of Theorem 1 and Theorem 2, we can see if we can prove global (exponential) stability of the feedback system using the known results. The known results are not applicable if \( \beta \neq 0 \). Hence, let us fix \( \beta = 0 \).

To compare, we first use our technique to analyze the case when \( \beta = 0 \). We applied our method to find the condition on \( a \) such that the feedback system with \( b = \infty \) is always eventually region stable with respect to the region \( \text{err} = 0 \). Using a template for a general quadratic Lyapunov function, and for the case when \( K_p = 500, K_i = 10 \), our tools produced \( a \geq 1/25000 \).

Now, let us use the passivity theorem. We can express our feedback system in the form required for passivity theorem as follows:

\[
\begin{align*}
\frac{d\vec{x}}{dt} &= A\vec{x} + B\vec{u} \\
y &= C\vec{x} + D\vec{u} \\
u &= -\Phi(y)
\end{align*}
\]

In our case, we have \( \vec{x} \) is a 2-d vector of err and interr, and

\[
\begin{align*}
A &= \begin{bmatrix} -a \ast K_p & -a \ast K_i \\ 1 & 0 \end{bmatrix} \\
B &= \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\
C &= \begin{bmatrix} K_p \\ K_i \end{bmatrix} \\
D &= 0
\end{align*}
\]

The function \( \Phi \) is a nonlinear function belonging to the sector \([0, \infty)\). Note that the lower bound \( a \) was moved into the \( A \) matrix. (This is an equivalent way of capturing our condition in Equation 4 where \( b = \infty \).)

Now, since \( a > 0 \), we have that \( A \) is Hurwitz, \((A,B)\) is controllable, and \((C,A)\) is observable (if \( K_i \neq 0 \)). We can
apply the passivity theorem (circle criterion) and conclude
that the feedback system is globally exponentially stable if
the transfer matrix $H(s) = C(sI - A)^{-1}B + D$ satisfies
the condition $\inf_{\omega \in \mathbb{R}} \Re(H(j\omega)) > 0$. However, this latter
condition is false in general, but it holds when $a > K_D^2 \geq K_I$,
which evaluates to $a \geq 1/25000$ for the case when $K_p = 500, K_i = 10$. Thus, we get the same result using passivity
theorem as we did using our quantifier elimination procedure.
However, our technique is an algorithmic procedure – it can
be applied to any system – and it can handle systems on
which passivity theorem may not be applicable (for example,
when $\beta \neq 0$ or $A$ is not Hurwitz). Also note that our results
guarante a.e.r.s stability and not global exponential stability.
For instance, even when $0 < a < 1/25000$, we can prove
a.e.r.s stability, whereas passivity theorem does not give any
useful information when $a < 1/25000$.

We briefly mention other related work in the literature.
Quantifier elimination has been applied to solve nonlinear
control system design [13] before. However, most of this
early work focused only on continuous dynamical systems
(and not switched systems), and moreover, only on simple
properties (and not safety and stability). The classical way
of proving stability by finding Lyapunov functions is an
instance of certificate-based verification, where the search for
Lyapunov function of a particular form is reduced to solving
of an $\exists \phi$ formula, where $\phi$ is an atomic fact. Numerical
techniques in the form of semidefinite programming exist for
solving such sum-of-squares problems [18], [3]. The work on
barrier certificates [20] moves this overall approach from
stability to safety.

VI. CONCLUSIONS
We defined the notion of always eventually region stability
and formulated the absolute always eventually region sta-
BILITY PROBLEM. We then used the certificate-based approach
and reduced the absolute a.e.r.s problem to a first-order
(formulated) formulas over the reals. We then used real quan-
tifier elimination methods to eliminate quantifiers from the
formula and get a solution for the absolute a.e.r.s problem.
An important observation we make here is that the study of
absolute stability and absolute always eventually region
stability is important for analyzing systems compositionally.
Several avenues exist for future work. While we have framed
and solved the absolute a.e.r.s problem for a one-dimen-
sional target region, we believe the results generalize
to higher dimensions and this is left for future work. Our
result also suggests that it can be used to create abstractions
of open systems and this needs to be explored further too.

REFERENCES
systems via predicate abstraction. In HSCC, volume 2289 of LNCS.
robot motion planning and control in polygonal environments. IEEE
inequalities in system and control theory. SIAM, 1994. Volume 15
of Studies in Applied Mathematics.
[5] C. W. Brown. QEPCAD B: a program for computing with semi-
 algebraic sets using CADs. ACM SIGSAM Bulletin, 37(4):97–108,
2003.
to flow pipes for dynamic systems. In 37th IEEE Conference on
Decision and Control, 1998.
[7] E. M. Clarke, A. Fehnker, Z. Han, B. H. Krogh, O. Stursberg, and
M. Theobald. Verification of hybrid systems based on counterexample-
guided abstraction refinement. In 9th Intl. TACAS, volume 2619 of
[8] G. E. Collins. Quantifier elimination for real closed fields by cylin-
drical algebraic decomposition – preliminary report. ACM SIGSAM
[10] G. Frehse. PHAVer: Algorithmic verification of hybrid systems past
hytech. In HSCC, volume 3414 of LNCS, pages 258–273. Springer,
2005.
hybrid systems. In Proc. 20th CAV, volume 5123 of LNCS, pages
checker for hybrid systems. Software Tools for Technology Transfer,
[13] M. Jirstrand. Nonlinear control system design by quantifier elimina-
reachability analysis. Dynamics of Continuous, Discrete and Impulsive
computations for families of linear vector fields. J. Symbolic Compu-
barrier certificates. In Proc. 7th HSCC, volume 2993 of LNCS, pages
caltech.edu/sostools.
for hybrid systems. In HSCC, volume 2993 of LNCS, pages 539–554.
[23] T. Sturm and A. Tiwari. Verification and synthesis using real
linear systems. IEEE Trans. on Automatic Control, 51(12):1862–1877,
2006.
techniques for the verification of hybrid systems. Proc. of the IEEE,
91(7), 2003.
[31] V. Weispfenning. The complexity of linear problems in fields. J. of
[32] V. Weispfenning. Quantifier elimination for real algebra—the
quadratic case and beyond. Applicable Algebra in Engineering