Online Stability Validation Using Sector Analysis

Joseph Porter Graham Hemingway Nicholas Kottenstette

Gabor Karsai Janos Sztipanovits

Institute for Software Integrated Systems
Vanderbilt University
Nashville, TN 37203 USA
{jporter,heminggs,nkottens,gabor,sztipaj}@isis.vanderbilt.edu

ABSTRACT

Our previous work has explored the use of compositional stabilization techniques for embedded flight control software[9] based on passivity properties of controller components and systems. Zames [21] presented a compositional behavior-bounding technique for evaluating stability of nonlinear systems based on real intervals representing cones (sectors) that bound possible component behaviors. Many innovations in control theory have developed from his insights. We present a novel use of his sector bound theory to validate the stability of embedded control implementations online. The sector analysis can be implemented as a computationally efficient check of stability for different parts of a control design. The advantage of the online application of this technique is that it takes into account software platform effects that impact stability, such as time delays, quantization, and data integrity.

We present a brief overview of the sector concept, our compatible control design approach, application of the technique to model-based embedded control software design, an example of its use to find design defects, and insights that may be drawn from our investigation so far. In the present work we only consider software (discrete-time) control of nonlinear continuous-time systems without switching.

Categories and Subject Descriptors

C.3 [**Special-Purpose and Application-Based Systems**]: Real-time and embedded systems; C.4 [**Performance of Systems**]: Measurement Techniques; D.2.4 [**Software/Program Verification**]: Validation

General Terms

Verification

Keywords

Runtime validation, Passive control

1. INTRODUCTION

High confidence embedded systems development features multi-faceted design activities complemented by model and data analysis to ensure the correct operation of the final product. Such analysis is multi-faceted due to the heterogeneity of domains involved in the problems (e.g. cyber-physical systems), leading to high costs and extended schedules for development efforts. Analyses suffer from difficulties of scale due to state and parameter space explosion problems. Further, interactions between design domains (such as the physical continous-time and computational discrete-events) are poorly understood, often requiring analyses to be overly conservative. Recently the research community for embedded systems has explored compositional techniques for design and analysis, where system correctness properties may be inferred from component correctness properties and more abstract interaction models defined on the interconnection structure of the design. Compositional methods strive to dramatically reduce state explosion problems by assuring correctness during model construction, while avoiding overly conservative approximations.

ESMoL is a modeling language built to experiment with encoding compositional techniques for high-confidence systems into a single modeling environment. Prior published work on ESMoL has dealt with modeling, code generation, and deployment for distributed control software implemented using our realization of the time-triggered model of computation [18, 13, 12]. We rely on the time-triggered paradigm to reduce jitter in sampling and actuating control tasks, to preserve synchronous design properties (such as deadlock freedom), and to detect timing faults[7]. The modeling tools also include the ability to generate platform-specific simulation models using the TrueTime toolkit for Simulink [5, 11]. The execution environment addresses only part of the problem. High-confidence designs require assurances of correct software behavior, including guarantees for stability, schedulability and deadlock freedom. The ESMoL-centered research effort seeks to address such problems using compositional design paradigms, for example using passive control techniques to ensure closed-loop stability for the controlled system (see Section 2 for an overview).

Beyond the multi-faceted challenges described above, one significant challenge lies in assessing the correct behavior of control system software once all of the components have been integrated and deployed to the hardware platform. For example, control designers use a very clean notion of timing behavior that is based on simple uniform sampling intervals, and which hard real-time implementations try to emulate. In reality, moving a control design from models to deployment is much more difficult in the case of distributed controllers. Calculation uncertainties due to limited data precision and timing uncertainties due to network delays introduce errors into the control system behavior. Commonly, control designers try to include enough 'slack' in the design to avoid such platform effects. We would like to be able to assess whether a deployed software control component is still stable as designed, or whether platform effects have changed the system behavior in a way that violates our assumptions.

To meet this challenge we propose an online stability validation technique. The specific requirements and challenges are as follows:

1. It must be efficient, both in space and in computation.
2. When operating, it must be sensitive enough to detect instability for various platform effects. In essence it should provide an abstract metric for stability in the presence of different possible destabilizing conditions.
3. The analyzer should operate on sample traces collected either from simulation or from the running system. We would like to be able to provide meaningful comparisons of results between simulation and the actual deployed system to aid system troubleshooting.

Our solution is based on sector theory, proposed originally by Zames[21] and extended by Willems[20] to deal with nonlinear elements in a control design. Sectors provide two real-valued parameters which represent bounds on the possible input/output behaviors of a control loop. Kottenstette [9] presented a sector analysis block for validating a control design in Simulink. We propose to use the same structure to verify the deployed control software online. Note that our assessment of item 3 above is still only preliminary. This particular technique provides several advantages beyond our original requirements:

1. For a given component, the sector measures behavior simultaneously over multiple inputs and outputs, so only one sector analyzer is required per control loop.
2. Because of our passive abstraction of the system design (described below), using multiple sector analyzers allowed us to quickly isolate problem components in the deployed design.

Section 2 gives a brief overview of the our control design approach, relevant sector analysis concepts, and describes our solution. Section 3 introduces describes our example and models in more detail. Section 4 discusses our experimental setup, and gives a first example of a problem uncovered by the online sector analysis. Section 5 describes our preliminary findings on the relationships between various system parameters and the sector values. Finally we visit related work (Section 6), and future work and conclusions (Section 7).

2. BACKGROUND

2.1 Passivity

Passivity is a property of physical systems, defined as a constraint that the output energy is bounded by the input energy so far as well as any stored energy. In control system design, passive components are themselves stable, and interconnected passive systems are compositional with respect to stability (subject to a few constraints on the interconnections). Frequently controllers are designed to be passive in order to gain this property. Digital controller designs can also be passive, and are insensitive to many kinds of errors introduced when the digital design is deployed to a computer platform. Platform behavior effects such as data and parameter quantization [4] and timing delays due to processor scheduling and network communications [3] [8] have little effect on the stability of a passive control design.

Passivity guarantees bounded-input/bounded-output (BIBO) stability, and with a few additional constraints can also ensure asymptotic stability, which is necessary for tracking. De la Sen [10] develops the concept of passivity formally using the following discussion. Let $u(t)$, $y(t)$, $D(t)$, and $S(t)$ be real-valued time-domain functions representing respectively, the input signals, output signals, dissipated energy, and stored energy. Then the power balance for a system is given by

$$u(t)y(t) = \dot{S}(t) + \dot{D}(t) \tag{1}$$

and the energy balance is given as

$$\langle u, y \rangle_t = S(t) + D(t) - S(0) - D(0). \tag{2}$$

The dot superscript denotes the time-derivative, and the inner product is the L_2 inner product of the two signals. Then using this notation, the component is passive if $\dot{D}(t) \geq 0$. Then $D(t) \geq D(0)$, $u(t)y(t) \geq \dot{S}(t)$, and

$$\langle u, y \rangle_t \geq S(t) - S(0). \tag{3}$$

Intuitively, the output can not exceed the sum of input energy and any remaining stored energy. As far as interconnections, parallel and feedback interconnections of control components maintain passivity, but serial interconnections may not.

2.2 Towards a Sufficient Stability Condition

Most approaches to nonlinear control system design rely on continuous time assumptions. When we consider discrete time implementation in software subject to network delays and finite-precision quantization effects, linear approximations and high sample rates tend to be used to obtain tractable analysis and realizable execution. In practice we have found that compositional techniques based on passivity have allowed us to construct reasonably low data rate digital controllers for nonlinear systems without resorting to conservative linear approximations.

Passive control techniques have proven successful for many cases of nonlinear continuous time controllers, but nonlinear discrete time control poses several challenges. Unfortunately many control structures are not passive in discrete time. If we can approximate our controlled system as a cascade of passive systems then we can apply a systematic control design strategy, for which stability can be validated online. After presenting the relevant theory we will illustrate our control design approach using a simple example.

2.3 Nonlinear Sector Analysis

Digital control for nonlinear physical systems with fast dynamics (such as a quadrotor helicopter) use a zero-order hold to convert control values produced at discrete time instants into step functions held over a continuous interval of time. For certain inputs and state trajectories, the hold process can introduce small amounts of new energy into the environment, violating passivity. The sector bounds analysis proposed by Zames [21] can be used to assess the amount of "active" (energy-producing) behavior which we can expect from a design under nominal operating conditions.

Zames critical insight was that many causal nonlinear systems' dynamic input-output relationships can be confined to being either inside or outside a conic region. Systems whose input-output rela-

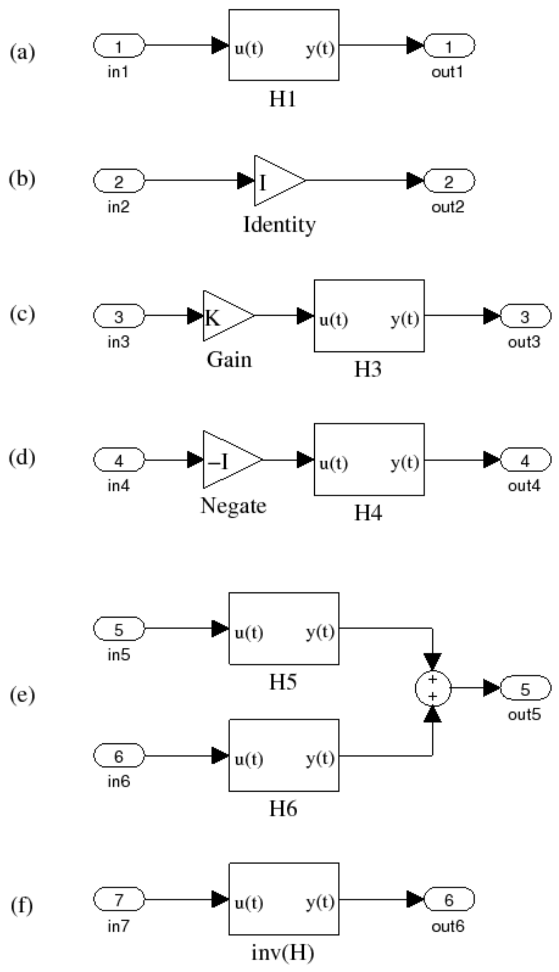

Figure 1: Block diagram interconnection examples for conic system composition rules.

tionships can be confined inside a conic region are known as interior conic systems. Equivalently these interior conic systems can be described as residing *inside the sector* $[a, b]$ in which a and b are real coefficients[21]. If there exist a real coefficients a and b such that Eq. 4 is satisfied then the system is an interior conic system inside the sector $[a, b]$ conversely if the inequality of Eq. 4 is reversed the system is exterior conic and outside the sector $[a, b]$. Table 1 describes the quantities used in Eq. 4. For the special case when $|a| < b < \infty$, b is also the gain of the system (i.e. $\|y_T\|_2 \le b\|u_T\|_2$). For linear time invariant (LTI) single input single output (SISO) systems the term a is the most negative real part of its corresponding Nyquist plot, it therefore is an approximate measure of the phase shift of a stable system. A passive system is equivalent to an interior conic system which is inside the sector $[0, \infty]$ therefore a passive LTI SISO system has no more than +/-90 degrees of phase shift in which all real parts of its corresponding Nyquist plot are *positive real.*

$$\|y_T\|_2^2 - (a + b)\langle y, u\rangle_T + ab\|u_T\|_2^2 \le 0 \qquad (4)$$

A conic system can also be modeled as a functional relation between the possible input and output signal spaces. This corresponds intuitively to a causal block diagram where the function specified

Quantity	Description
$u(t)$	Input signal
$y(t)$	Output signal
$\|y_T\|_2^2$	Energy produced by the component so far (output) in a time interval of length T.
$\|u_T\|_2^2$	Energy received by the component so far (input) in a time interval of length T.
$\langle y, u\rangle_T$	Correlation between the input and output sample values in a time interval of length T. This is a measure of dissipation.
a	Real-valued lower bound for the sector.
b	Real-valued upper bound for the sector.

Table 1: Quantities for the sector formula.

in the block relates the inputs to the outputs, as in Fig. 1 (a). See Zames for a complete formal functional description of sector analysis. Given conic relations H, H_1 with H in $[a, b]$ and H_1 in $[a_1, b_1]$ ($b, b_1 > 0$), and given a constant $k \ge 0$, we have the following sector composition rules from [21]:

1. I is in $[1, 1]$ (Fig. 1 (b))
2. kH is in $[ka, kb]$ (Fig. 1 (c))
3. $-H$ is in $[-b, -a]$ (Fig. 1 (d))
4. sum rule $H + H_1$ is in $[a + a_1, b + b_1]$ (Fig. 1 (e))
5. inverse rule(s) (Fig. 1 (f))
   (a) $a > 0 \rightarrow H^{-1}$ is in $[\frac{1}{b}, \frac{1}{a}]$.
   (b) $a < 0 \rightarrow H^{-1}$ is outside $[\frac{1}{a}, \frac{1}{b}]$.
   (c) $a = 0 \rightarrow (H^{-1} - (\frac{1}{b}I)$ is positive.

For rule 5 an inverse system model must be well-defined (i.e. exist), as in the case of invertible linear system models.

The sector composition rules illustrate the compositional nature of sector analysis. Zames also gives conditions under which feedback-interconnected conic systems exhibit stability [21, Theorems 2a,2b]. We will not describe all of the details here, but simply relate the sufficient condition described in Kottenstette to establish stability of feedback control loops where the included systems are conic [9, Corollary 2]:

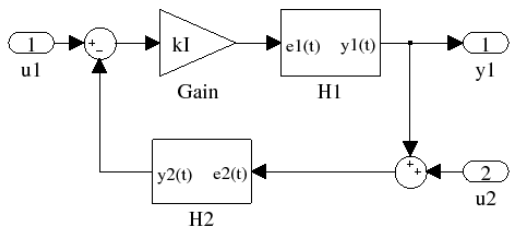

Figure 2: Block diagram interconnection example for feedback structure.

Assume that the combined dynamic system $H : [u_1, u_2] \rightarrow [y_1, y_2]$ depicted in Fig. 2 consists of two dynamic systems $H_1 : u_1 \rightarrow y_1$ and $H_2 : u_2 \rightarrow y_2$ which are respectively inside the sector $[a_1, b_1]$ and strictly inside the sector $[0, 1+\epsilon]$, for all $\epsilon > 0$. Then H is bounded (L_2^m stable for the continuous time case or l_2^m stable for the discrete time case) if:

$$-\frac{1}{\max\{|a|, b\}} < k < -\frac{1}{a_1}, \text{ if } a_1 < 0$$

$$-\frac{1}{b} < k < \infty, \text{ otherwise.}$$

The extra output ports (numbered 2 and 3 in Fig. 2) in the diagram will be covered later. They represent points at which we attach signal monitors to perform online assessment of the sector bounds.

2.4 Digital Control Design Example

Our design example controls a continuous-time system whose model represents a simplified version of a quadrotor UAV. Fig. 4 depicts the basic component architecture for the control design. Our example excludes the nonlinear rotational dynamics of the actual quadrotor for simplicity, but retains the difficult stability characteristics. Fig. 3 shows a Simulink model depicting the simplified dynamics. For the fully detailed quadrotor model and a complete discussion of the control design philosophy, see [9]. The example model controls the stack of four integrators (and motor lag model) using two nested PD control loops, including a nonlinear model element for actuator saturation in the outer loop. The Plant block contains the integrator models. The two control loops (inner and outer, as shown) are implemented on separate processors, and the execution of the components is controlled by a simple time-triggered virtual machine that releases tasks and messages at pre-calculated time instants.

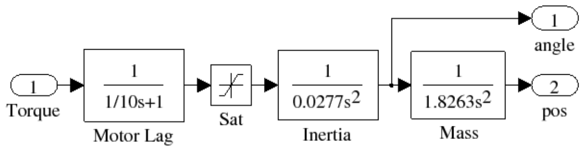

Figure 3: Simplified quadrotor plant dynamics. The rotational dynamics have been removed to facilitate easier study of the behavior. The full model includes all of the dynamics.

If the plant dynamics were passive, we would have considerable freedom in setting gains and choosing control structures. The combination of fast dynamics and new energy introduction into the environment by the zero-order hold outputs means that each of the control loops must mitigate small amounts of "active" behavior. The sector bound a quantifies the energy-generating behavior of each control loop, but we still lose the flexibility of evaluating the design associatively while retaining the delay-insensitivity effects conferred by passive structures. In our quadrotor system, we expect the bound a to be small and negative and choose the gains appropriately.

This particular design must be evaluated from the innermost loop to the outermost loop in order to make sense of the gain constraints. Fig. 5 shows the nested loop structure of the design. The actual design and implementation are complicated by the physical architecture of the digital realization:

1. Sensors acquire digital attitude and position information, so velocities must be estimated.

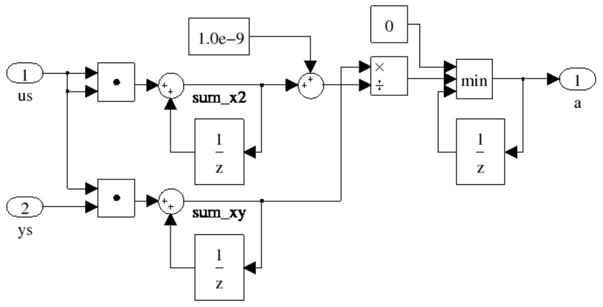

Figure 6: Simulink implementation of the online sector analysis algorithm.

2. The controller components and velocity estimators are deployed to different processors in the digital implementation. The two inner loop controllers, motor lead compensator, and angular velocity estimator are deployed to an onboard controller with rich I/O capabilities. The outer loop controllers and velocity estimator are deployed to a faster processor which also receives reference trajectory data. Data messages are exchanged between the two processors using a time-triggered protocol.

3. Motor thrust commands are issued periodically using a zero-order hold. As discussed previously, this introduces additional energy back into the environment.

2.5 Online Sector Analysis

Fig. 6 shows the structure of an online sector calculator in Simulink. We refer to its implementation as the (possibly misnamed) online sector search or the sector analyzer – it is included with deployed digital controller code to assess stability by monitoring the estimated sector values. The C source was generated using Real-Time Workshop. The block input x corresponds to the input variables to be monitored, and y to the output variables to be monitored. The sector analyzer operates by calculating the accumulated input energy and dissipation terms for each sample, computing their ratio (the sector value so far), and maintaining the minimum. The calculated sector bound value decreases monotonically, and we rely on the calculated bound achieving a steady-state value (settling to a constant in the limit) for a particular input. The minimimum over all realizable inputs is the sector bound for the system. The "search" concept refers to the idea that we explore these bounds over a range of inputs and controller parameters to find the worst-case bound to use during operation. In online operation the sector calculation is not a "search" as much as a validation of behavior with respect to our analytic and simulated sector values. The name "sector search" has stuck, though.

Empirical verification of the sector bounds does not provide a necessary condition to verify stability for the implementation, but it is sufficient. If sectors are estimated conservatively, then measurements which exceed the bounds are significant. This indicates that the selected control gains may not have been adequate to stabilize the system with the platform delay and quantization effects included in the closed-loop controller. In our case an input which drives the maximum output slew rate should yield the largest active sector bound value.

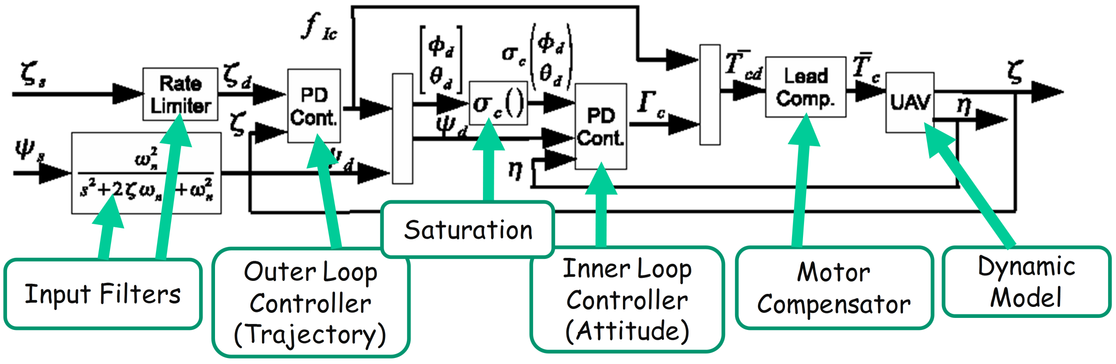

Figure 4: Basic architecture for the quadrotor control problem[9].

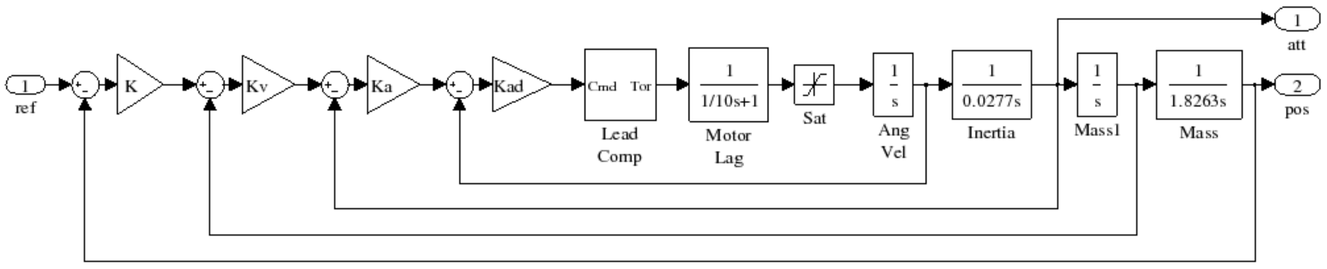

Figure 5: Conceptual nested loop structure of the controller.

2.6 Limitations

The current implementation of the online sector analysis has a few limitations:

- The sector calculation implementation is an approximation, the error of which may be aggravated by our ϵ term which is used to prevent divide-by-zero (in Fig. 6, for example, ϵ is $1.0e-9$). In this form the estimated sector bounds could be smaller than the actual bound. In that case the online search could fail to detect small trajectory deviations that might indicate instability. In practice the difference is small or negligible, and the sector analyzer is sensitive enough to capture platform effects.

- The sector calculation in its current form requires high precision. We use double-precision floating point, as even single-precision calculations introduced instability in the sector calculation. This may be a significant obstacle for deployments to small embedded controllers that lack floating-point hardware. For the experiments on actual hardware we were required to export much of the sampled data out of the control processors in order to perform the sector search with sufficient precision, despite the availability of sufficient runtime to do the calculations on the controllers.

Both of these deficiencies could be remedied by a more careful design of the search implementation. In particular, a good fixed-point implementation would be very useful for deployment to microprocessors that lack floating point capabilities. Another approach which avoids introducing computational overhead is to use external signal measurement devices and perform the sector analysis externally using another computer. This could be done using a digital scope for electric signals, a JTAG debugger to get digitally stored values, or a logic probe.

3. SOLUTION OVERVIEW AND EXAMPLE

As discussed previously, the sector analysis technique is part of a larger research effort to create design tools for high-confidence distributed control system designs. As such, each element and technique in our tool suite performs a particular function within a larger model-driven design process. We will briefly cover the design process to highlight potential uses of the online sector analysis technique, and illustrate our model-based approach with an example.

1. Control designers work from requirements and (if possible) a passive decomposition of the design to create controllers.

2. Simulink simulations with included sector analysis blocks provide preliminary evidence that the design satisfies the requirements.

3. Software modelers import the Simulink design into the Generic Modeling Environment (GME) [6], creating an instance of a model in the ESMoL language.

4. In ESMoL software designers specify software component interfaces and functions for the imported Simulink functions, create hardware platform models, map instances of software control components to the platform, and specify dependencies and timing information.

5. A time-triggered schedule is calculated based on the model structure and timing parameters, and release times are written back into the model for component instances and timed data messages [12].

6. With the schedule in the model, control designers can now synthesize a platform-included Simulink model from the ESMoL model using the TrueTime toolkit[5]. The sector analysis technique can be applied to the platform-specific simulation.

7. Control designers and software modelers iterate over the previous steps to bring the platform-specific simulated design into conformity with the requirements.

8. Software testers synthesize controller software from the models and deploy it to a hardware-in-the-loop simulation for more detailed integration and analysis. The sector analysis technique applies at this stage as well. The model-based analysis and generation approach makes rapid redesign and re-assessment straightforward when needed.

9. The deployed software and hardware are tested in the actual environment, again using sector analysis if appropriate. Often the final integration of the controller into the physical environment makes assessment much more costly, as needed data may be difficult or impossible to get (hence the focus on simulation). If our platform-specific and hardware-in-the-loop simulations are sufficiently detailed, we should be able to avoid major surprises in the final deployment.

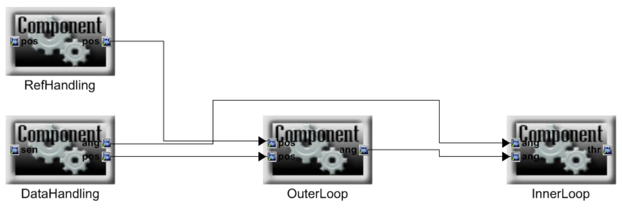

Figure 7: Deployment: The ESMoL logical architecture model specifies data dependencies between software component instances. The ports on the blocks represent data messaging interfaces into and out of the component.

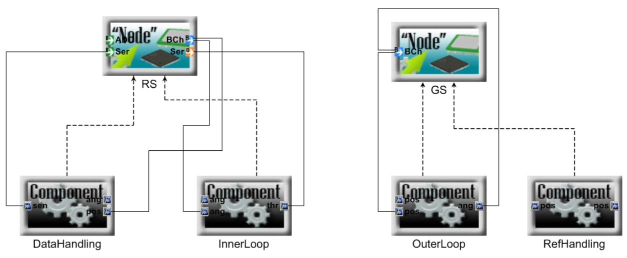

Figure 8: Deployment: The platform mapping model specifies the hardware components that will support the transfer of data to and from each processing node, the software components that will send and receive the data, and the message instances in which the data will be carried.

Figs. 7 and 8 show different parts of the GME design model for the quad integrator controller. The model-integrated computing approach uses integrated sublanguages to represent models in the various aspects of the design[6]. Fig. 7 portrays logical data dependencies between software component instances, independent of their distribution over different processors. The arrows between the components represent the transfer of message data. Fig. 8 shows the deployment model – the mapping of software components to processing nodes (dashed lines from the components to the processors), and data messages to communication ports (solid lines between component message interface ports and processor peripheral ports). RS (Robostix) is an 8-bit ATMega128 CPU which runs the DataHandling and InnerLoop software components. GS (Gumstix) is an Intel PXA ARM-based CPU which runs the RefHandling and OuterLoop components. An execution model (not shown) allows the designer to attach timing parameter blocks to components and messages, such as execution period and worst-case execution time. The quad integrator controller runs all of the blocks at a frequency of 50Hz. The execution model also indicates which components and messages will be scheduled independently, and which will be grouped into a task. Our example model assigns each component to its own task. All tasks are executed with a static schedule on a time-triggered bus. The static message schedule can introduce additional time delays into the controller, so delay effects are a concern.

The sector blocks are attached around each controller, so the input and output are from the point of view of the control element. The output of the controller (input to the rest of the system) is connected to the sector analyzer input port. The signal controlled by the controller (before the error term is formed) is part of the input to the controller, but from our point of view it is the output of the system, so it connects to the sector analyzer output port. Fig. 9 displays the connection of the sector search around the position control gain for our example. K_x is the proportional gain for the outer loop PD controller, and K_v is the derivative gain. Currently the sector blocks are inserted by hand, as we have not yet completed the modeling extensions for online validation and input scenarios.

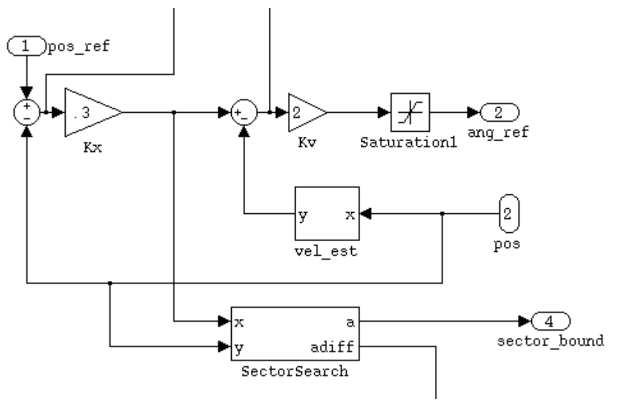

Figure 9: Illustration of the sector analysis block connected around the position controller in the quad integrator test model. The lines leading off of the figure correspond to other data taps that are not relevant to illustrate the concept.

4. TROUBLESHOOTING A HIL CONTROL SYSTEM WITH SECTORS

Our test platform is a hardware-in-the-loop simulation environment that allows us to execute the plant model, and to interface the controller using actual serial links so that it appears to be a genuine plant with respect to the hardware interface, software interface, and timing delays. Our plant simulator uses the Mathworks' *xPC Target* tool to run the plant dynamics[17]. *xPC Target* provides library blocks for standard I/O devices. We transfer data to and from the controller using a serial link, just as in the actual quadrotor.

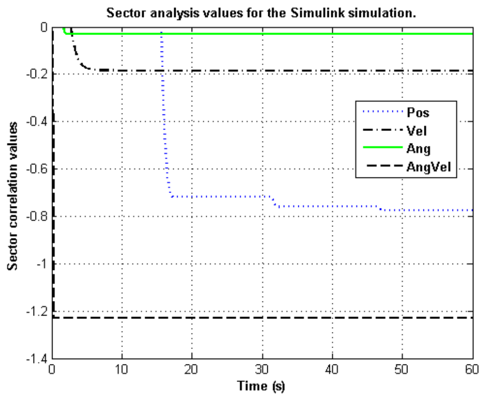

Figure 10: Sector search values for the simulated quad integrator example.

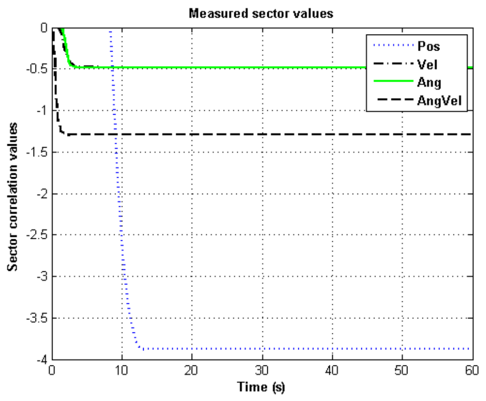

Figure 11: Sector search values for execution on the control platform (including schedule effects).

For this test we selected a square wave reference input near the highest frequency admissible by the controller. Platform effects caused a significant deviation from our ideal sector estimates and bounds, as illustrated by the sector bound changes in Table 2. For each digital control signal the table records the following (by column):

1. Original Bound: the sector bound based on the original gain value ($-\frac{1}{k}$).
2. Simulated Sector: the sector value recorded in simulation.
3. Measured Sector: the initial sector value measured on the platform.
4. Delta: the sector difference between the measured and simulated values.
5. New Bound: the sector bound based on the newly adjusted gains.
6. New Sector: the sector value measured on the platform with the new gains.

Although the initial platform gains satisfied the sector stability conditions analytically and in simulation (comparing the Bound column tothe Simulated column in the table), the overall system response when deployed to the target platform resulted in significant position overshoot. The measured sector value for position measured the farthest from the predicted value, and exceeded the gain bound for stability ($-1/k$), though no evidence of instability was visible in the plot of the output trajectory. As all of the gains moved right up to the edge of their bounds when deployed, we adjusted all of the gains by $\frac{1}{2}$. Note that changing the gains changes the acceptable sector bound as well as the actual sector bounds themselves as shown in the table. After adjusting the gains all of the sector values fell within the bounds.

On closer inspection we discovered that the most significant platform effect was a non-ideal position gain condition for signals with frequencies too close to the sampling rate. Figs. 12 and 13 show a comparison of the ideal frequency response of the outer loop controller block with an empirically measured frequency response for the same controller block deployed on the target hardware. The remedy was to add a simple input filter to cut off frequencies too close to the sampling rate. This effectively slows down the possible commands that can be issued to the system. The sector analysis blocks helped identify the position control component as the element whose behavior was farthest from predicted when deployed to the platform.

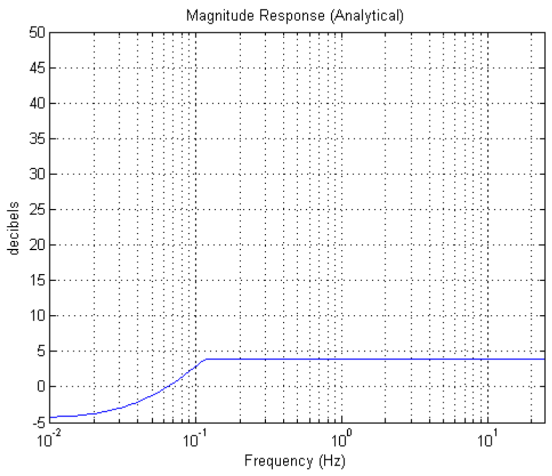

Figure 12: Magnitude frequency response (analytically predicted) for the quad integrator model.

5. PLATFORM EFFECTS AND SECTOR VALUES

5.1 Data Quantization

In order to explore the sensitivity of the online sector analysis technique with respect to data quantization effects, we simulated the possible degradation of the sector bound measurement as the data precision increased. All Simulink simulation runs in these experiments were sampled at 50 Hz. The model under test adds an input filter to remove frequencies right near the Nyquist frequency (to avoid the difficulties encountered earlier). We simulated the fixed-point quantization effects by adding an appropriate noise component to the system inputs (zero-mean Gaussian noise with variance $\frac{2^{-2bits}}{12}$. For the chosen input signal 100 seconds was sufficient to

Signal	Original Bound	Simulated Sector	Measured Sector	Delta	New Bound	New Sector
Angular Velocity	-1.333	-1.2292	-1.2963	-0.0671	-2.667	-1.4568
Angle	-0.5	-0.0295	-0.4831	-0.4536	-1.0	-0.0068
Velocity	-0.5	-0.1856	-0.4830	-0.2974	-1.0	-0.9324
Position	-3.333	-.7757	-3.8811	-3.1054	-6.667	-1.6081

Table 2: Sector value comparisons for simulation and execution on the actual platform.

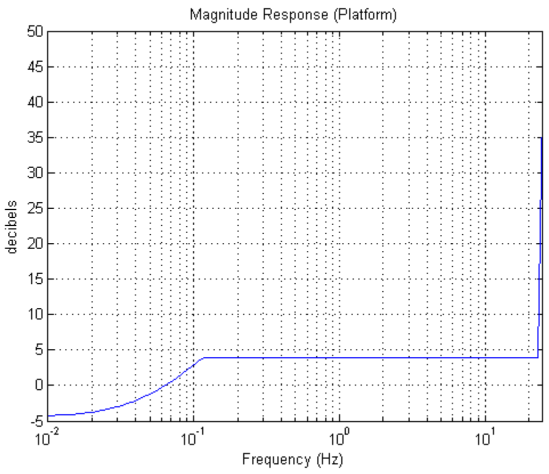

Figure 13: Magnitude frequency response (measured) for the quad integrator model. Note the spike at the right-hand side of the plot. This is a nonlinear gain anomaly due to the effects of the saturation block, and which appears only for signals with frequencies right near the Nyquist sampling rate.

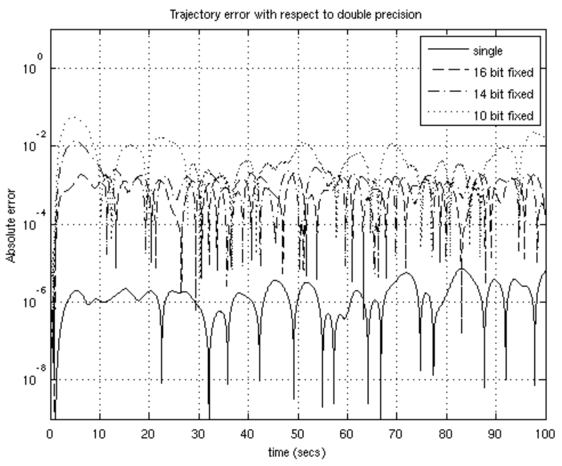

Figure 14: Output tracking error over time with respect to data quantization differences. The reference input is a square wave.

get the sector measurement to settle. A more rigorous investigation is certainly needed, but our initial goal was to obtain a rough idea of the sensitivity of the sector to quantization without deploying and debugging all of the different configurations.

Quant Level	Ang Velocity	Angle	Velocity	Position
Gain bound $a > -\frac{1}{k}$	-4.0000	-1.0951	-1.0000	-18.0505
double	-3.0116	-0.9237	-0.7252	-1.0625
single	-3.0127	-0.9237	-0.7267	-1.0625
fix16	-3.0127	-2.1673	-0.7266	-12.0967
fix14	-3.0126	-2.1834	-0.7258	-26.7733
fix10	-7.5519	-2.1845	-0.5887	-29.1194
Gain bound	-4.0000	-1.0951	-1.0000	-18.0505

Table 3: Sector value under different quantization levels.

Table 3 presents sector values for the various simulated quantization levels. Different control loops degrade at different points under the current input. The idea was to choose an input with a wide range of frequency content, and overdrive the system slightly. Our design did not perform well with fixed-point quantization, suggesting that an operational loss of data precision could present a risk of destabilizing our control system. For comparison, Fig. 14 displays the relative tracking error for each of the quantization levels. The sector bounds did not vary quite as smoothly with respect to quantization as did the position output. It appears that the sector analyzer is sufficiently sensitive to explore these effects.

5.2 Scheduling and Delay Effects

To further explore the sensitivity of the design and to validate the search technique, we simulated a wide range of synchronous (integer) delay values for the links between the components. This captures effects both of actual network delay as well as delays introduced by our calculated static schedule [12]. We collected data over the input and parameter ranges specified in Table 4 Even our relatively small and simple design required 472,392 data sets to assess the behavior of the sector bounds with respect to the input and parameter design spaces. We opted for an exhaustive simulation set, as the required compute power and storage space were available. Still, the simulations took approximately three days to run. Each simulation was driven by a square wave input with the specified amplitude and frequency. Note that the input amplitudes correspond to slightly unrealistic input levels, as overdriving the control system seemed to help explore the full range of the sector values.

Surprisingly, no immediate pattern emerged from the data analysis. Our hope was that the change in total delay ($D_T = D_{PH} + D_{HO} + D_{OI} + D_{IP}$) through the system would correlate well with changes in the sector values measured for any given set point of the other parameters. Slicing the data separately by sample rate, by amplitude, and by frequency showed roughly the same range of sector values for all runs. Interestingly, the total delay also did not seem to change the pattern.

Our preliminary observation is that (in general) system behavior with respect to the sector bounds depends more on the distribution of delays through the system than on any other parameter. Unfortunately, our early investigation into the delay parameter dependence shows some unusual patterns. Fig. 15 illustrates this concept for a single set point (Input Freq = .033 Hz, Amplitude = 0.5, and Sample rate = 100 Hz). Searching for delay patterns in the generated

Quantity	Simulation Values
Delay from Plant to Data Handler D_{PH}	0–8 (ticks)
Delay from Data Handler to Outer Loop D_{HO}	0–8 (ticks)
Delay from Outer Loop to Inner Loop D_{OI}	0–8 (ticks)
Delay from Inner Loop to Plant D_{IP}	0–8 (ticks)
Reference input frequency	.20, .10, .033 (Hz)
Sample rate	25, 35, 50, 75, 90, 100 (Hz)
Reference input amplitude	0.5, 1, 2, 4 (m)

Table 4: Parameter test ranges.

data, the outlying points around $a = -0.9$ roughly correspond to the condition $D_{PH} > 0 \wedge D_{HO} = 0 \wedge D_{OI} = 0 \wedge D_{IP} > 0$, where the delay is split between the links before the controller and after the controller (between the inner loop and the plant). The ordering and scheduling of the data flow and execution for the inner and outer loops in this case are fully sequential, as they incur no sample ticks between their invocations and transfers. A more careful investigation is necessary in order to be able to characterize the space of possible delay values and discern meaningful patterns.

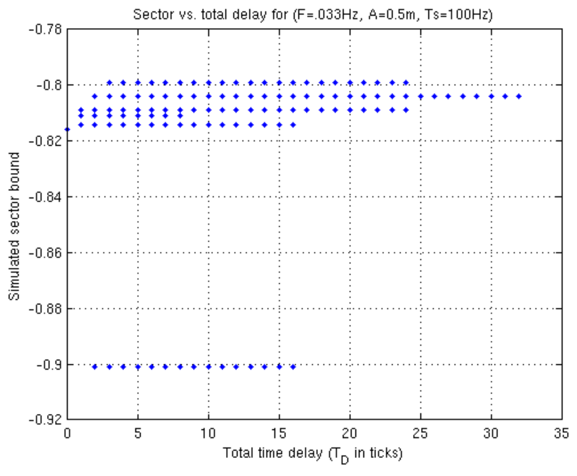

Figure 15: The sector variations seem to depend more on the distribution of delays in the individual links rather than on the delay values themselves.

Our conclusion from this preliminary investigation is that our controller is reasonably insensitive to delays, and therefore the sector analysis might not detect delay anomalies in the system, depending on where in the communication chain they occur. Repeating the analysis with a known delay-sensitive control structure would provide a good comparison, as well as trying a wider range of parameter and input conditions.

6. RELATED WORK

The present technique has much in common with techniques in online verification. Online verification is not a new concept in computational domains. See for example [2] and [19] for examples which integrate formal logical models and executing code to detect errors.

Sammapun, Lee, and Sokolsky[14] address lightweight online verification for checking timeliness and reliability in real-time systems. RT-MaC uses a stochastic approach which yields confidence intervals for specified properties. This is probably the closest in practice to our technique. See also recent work from Zuliani, Platzer, and Clarke [22] which focuses on statistically checking properties on simulated trajectories in Simulink models, and which should be just as applicable to traces collected from a running control system.

From the control community self-triggered control[1] is a framework which could also be used for online stability guarantees. Control actions are timed based on specified criteria on measured or estimated states rather than periodic sampling. This suggests interesting extensions to our technique to accomodate non-uniform sampling. Skaf and Boyd present techniques for calculating and verifying quantization levels offline using constraint problems representing design requirements, system behavior, and objective functions for assessing performance[15]. [16] describes a technique for calculating a set from which controller coefficients can be selected without compromising performance. A better understanding of the sector relationships to platform and design parameters may lead us to be able to benefit from this technique.

7. FUTURE WORK AND CONCLUSIONS

7.1 Future Work

For the future, we would like to address a few things to advance the usefulness of this technique:

- Carefully evaluate numerical calculation effects for the sector analyzer implementation. Determine whether a fixed-point implementation is meaningful.
- Empirically study the sensitivity of the sector bounds to data and parameter quantization on actual hardware and using a wider range of test models.
- Compare sector results between TrueTime simulations and the actual platform.
- Revisit the effects of link delays on the sector values, comparing with models which are more sensitive to delay effects.
- Develop the relevant theory and assess the use of the sector analysis for switched and hybrid systems.

7.2 Conclusions

We conclude that online sector analysis is a potentially useful technique for validation of passive control systems software. It seems able to detect quantization and sampling effects which could violate design assumptions in passive digital control systems. Further investigation is required to better characterize its capability with respect to network delays.

8. ACKNOWLEDGEMENTS

This work is sponsored in part by the National Science Foundation (grant/contract number NSF-CCF-0820088) and by the Air Force Office of Scientific Research, USAF (grant/contract number FA9550-06-0312). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government.

9. REFERENCES

[1] A. Anta and P. Tabuada. To sample or not to sample: Self-triggered control for nonlinear systems. *IEEE Trans. on Aut. Control*, PP(99):1 –1, 2010.

[2] H. Barringer, A. Goldberg, K. Havelund, and K. Sen. Rule-based runtime verification. In *Verification, Model Checking, and Abstract Interpretation*, volume LNCS 2937, pages 277–306. Springer Berlin / Heidelberg, 2004.

[3] N. Chopra, P. Berestesky, and M. Spong. Bilateral teleoperation over unreliable communication networks. *IEEE Trans. on Control Systems Tech.*, 16(2):304–313, Mar 2008.

[4] A. Fettweis. Wave digital filters: theory and practice. *Proc. of the IEEE*, 74(2):270 – 327, 1986.

[5] G. Hemingway, J. Porter, N. Kottenstette, H. Nine, C. vanBuskirk, G. Karsai, and J. Sztipanovits. Automated Synthesis of Time-Triggered Architecture-based TrueTime Models for Platform Effects Simulation and Analysis. In *RSP '10: 21st IEEE Intl. Symp. on Rapid Systems Prototyping*, Jun 2010.

[6] G. Karsai, J. Sztipanovits, A. Ledeczi, and T. Bapty. Model-integrated development of embedded software. *Proc. of the IEEE*, 91(1):145–164, January 2003.

[7] H. Kopetz and G. Bauer. The Time-Triggered Architecture. *Proc. of the IEEE*, 91(1):112–126, Jan 2003.

[8] N. Kottenstette and P. J. Antsaklis. Stable digital control networks for continuous passive plants subject to delays and data dropouts. In *CDC '07: Proc. of the 46th IEEE Conf. on Decision and Control*, pages 4433 – 4440, 2007.

[9] N. Kottenstette and J. Porter. Digital passive attitude and altitude control schemes for quadrotor aircraft. In *ICCA '09: 7th IEEE Intl. Conf. on Control and Automation*, Christchurch, New Zealand, 2009.

[10] M. D. la Sen. Some conceptual links between dynamic physical systems and operator theory issues concerning energy balances and stability. *Informatica*, 16(3):395–406, 2005.

[11] M. Ohlin, D. Henriksson, and A. Cervin. *TrueTime 1.5 Ref. Manual*. Dept. of Automatic Control, Lund Univ., Sweden, Jan 2007. http://www.control.lth.se/truetime/.

[12] J. Porter, G. Karsai, and J. Sztipanovits. Towards a time-triggered schedule calculation tool to support model-based embedded software design. In *EMSOFT '09: Proc. of ACM Intl. Conf. on Embedded Software*, Grenoble, France, Oct 2009.

[13] J. Porter, G. Karsai, P. Volgyesi, H. Nine, P. Humke, G. Hemingway, R. Thibodeaux, and J. Sztipanovits. Towards model-based integration of tools and techniques for embedded control system design, verification, and implementation. In *ACES-MB '09: Workshops and Symp. at MoDELS 2008, LNCS 5421*, Toulouse, France, 2009. Springer.

[14] U. Sammapun, I. Lee, and O. Sokolsky. RT-MaC: runtime monitoring and checking of quantitative and probabilistic properties. In *11th IEEE Intl. Conf. on Emb. and Real-Time Comp. Systems and App.*, pages 147 – 153, Aug 2005.

[15] J. Skaf and S. Boyd. Filter design with low complexity coefficients. *IEEE Trans. on Signal Proc.*, 56(7):3162–3169, Jul 2008.

[16] J. Skaf and S. Boyd. Controller Coefficient Truncation Using Lyapunov Performance Certificate. *Intl. Journal of Robust and Nonlinear Control*, Mar 2010.

[17] The MathWorks, Inc. Simulink/Stateflow Tools. http://www.mathworks.com.

[18] R. Thibodeaux. The specification and implementation of a model of computation. Master's thesis, Vanderbilt Univ., May 2008.

[19] W. Visser, K. Havelund, G. Brat, S. Park, and F. Lerda. Model checking programs. *Automated Software Engineering Journal*, 10(2), April 2003.

[20] J. C. Willems. *The Analysis of Feedback Systems. Research Monograph No. 62*. MIT Press, Cambridge, MA, 1971.

[21] G. Zames. On the input-output stability of time-varying nonlinear feedback systems part one: Conditions derived using concepts of loop gain, conicity, and positivity. *IEEE Trans. on Aut. Control*, 11(2):228–238, Apr 1966.

[22] P. Zuliani, A. Platzer, and E. M. Clarke. Bayesian statistical model checking with application to stateflow/simulink verification. In *HSCC 2010: 13th Intl. Conf. on Hybrid Systems: Computation and Control*, pages 12–16, Stockholm, Sweden, Apr 2010.