HAL
open science

Ride-through for Autonomous Vehicles

Aaron Kane, Philip Koopman

To cite this version:

Aaron Kane, Philip Koopman. Ride-through for Autonomous Vehicles. SAFECOMP 2013 - Workshop CARS (2nd Workshop on Critical Automotive applications : Robustness & Safety) of the 32nd International Conference on Computer Safety, Reliability and Security, Sep 2013, Toulouse, France. pp.NA. hal-00848195

HAL Id: hal-00848195

https://hal.science/hal-00848195v1

Submitted on 25 Jul 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Ride-through for Autonomous Vehicles*

Aaron Kane and Philip Koopman

Carnegie Mellon University, Pittsburgh, PA
akane@cmu.edu, koopman@cmu.edu

Abstract. Safety critical systems often have shutdown mechanisms to bring the system to a safe state in the event of a malfunction. We examine the use of ride-through, a technique to reduce the frequency of safety shutdowns by allowing small transient violations of safety rules. An illustrative example of enforcing a speed limit for an autonomous vehicle shows that using a rate-limited ride-through bound permits a tighter safety limit on speed than a fixed threshold without creating false alarm shutdowns. Adding state machines to select specific safety bounds based on vehicle state accommodates expected control system transients. Testing these principles on an autonomous utility vehicle resulted in improved detection of speed limit violations and shorter shutdown stopping distances without needing to increase the false alarm shutdown rate.

1 Introduction

Automobiles, like many other safety-critical systems, are incorporating complex software-based features at an increasing pace. Runtime monitoring [2] is an attractive way to help ensure safe system operation because traditional design techniques (such as model checking) don't scale well enough to the complexity of these systems. One way to use these monitors is allow them to trigger a safety shutdown (or at least shut down a potentially faulty component) when they see a safety policy violation.

A common way to reduce safety shutdown frequency in industrial control systems is by using the concept of "ride-through," in which a minor transient violation of system safety properties is ignored so long as it is not too severe [4]. However, there has been little academic discussion of this topic (especially outside the power domain [3]), and no critical evaluation of how and when such approaches are useful.

Ride-through can be used in autonomous ground vehicles to provide increased accuracy of a runtime monitor, both to avoid triggering on false-positives as well as to reduce the detection time for some types of faults.

This paper uses an illustrative example of enforcing a vehicle speed limit to discuss ride-through, showing that using a sliding rate-limited ride-through bound permits a tighter safety limit on speed than a fixed threshold, without increasing false alarm shutdowns. We also show that the usual approach of a

* This research was funded in part by General Motors through the GM-Carnegie Mellon Vehicular Information Technology Collaborative Research Lab

non-rate-limited bound is no better than a fixed threshold. Adding state machines to select safety bounds can accommodate expected control system transients. Feasibility tests on an autonomous vehicle resulted in improved detection of speed safety threshold violations and shorter shutdown stopping distances without needing to increase the false alarm shutdown rate.

2 Safety Limits

Most safety-critical embedded systems run periodic control loops, affecting the values of some system properties. Such systems typically have a known safe operating envelope, which can be described by the values of its system properties. We have found that simple property bounds (such as speed limits) are extremely useful and have been sufficient to support safety cases [1] in full-scale systems we have built. The value of a property at which the system is no longer safe is its *safety limit* (e.g., a safety limit on the engine of 7500rpm might be set because the engine cannot operate above that speed without violent failure, or a top vehicle speed might be set based on the maximum speed rating of tires).

Safety-critical systems often have both a normal control system and a simple and reliable emergency backup system which has independent actuators whose express purpose is to bring a faulty system to a safe state as quickly as possible. We call the emergency control system the *hard limit* system. At the hard limit the system is about to become unsafe and must immediately be returned to a safe state, using a *hard shutdown* to avoid exceeding the safety limit. A hard shutdown for autonomous ground vehicles may include opening an electric vehicle safety relay under high current load (which causes significant wear to the relay). Classically, hard shutdowns might leave an environmental mess, cause equipment damage, or at the very least necessitate a service outage to effect system recovery, making false alarm activations highly undesirable.

In production operation, hard shutdowns are typically avoided at all practical cost due to the expense and disruption of system shutdown and recovery. For that reason, many embedded control systems also use a *soft limit*. The soft limit serves as a warning for the control system to attempt to transition the system to a safe state via a *soft stop*, which is an orderly shutdown that avoids the worst of the consequences of an emergency control system activation. For example, a soft stop might be a command to vehicle autonomy software to stop the vehicle without activating the vehicle safety relay. This might work in many cases, but cannot be counted upon if the autonomy software itself is what has failed.

3 Ride-through Bounds

A baseline soft limit system has a fixed soft limit threshold located some distance before the hard limit threshold is reached. Whenever the soft limit threshold is crossed, the system performs a soft shutdown.

A very simple ride-through mechanism adds a maximum time and maximum magnitude to a soft limit violation. This allows the system to violate the soft limit for a certain amount of time as long as the property stays under a given magnitude for the entire duration. This type of bound can be visualized by

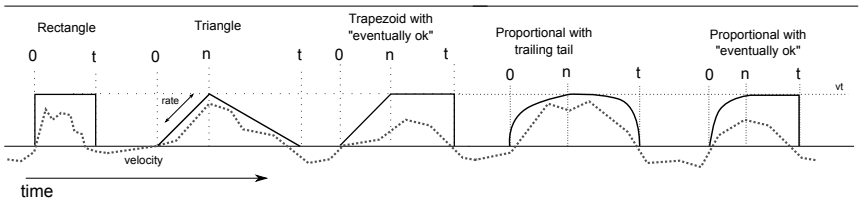

Fig. 1. Visualization of rectangle, rate-limited, and proportional bounds

enclosing the curve of the property in a rectangle above the soft limit (see Figure 1). As long as the value stays within the rectangle, the system can continue to ride-through the violation. This is probably the most common ride-through strategy in current use, and it is common for the maximum magnitude to simply be the hard limit (i.e., wait a short period after the soft limit is violated before actually activating the soft shutdown.)

For systems with unbounded dynamics, rectangular bounds are functionally a higher static limit and thus no more effective than using an equivalent higher soft limit. This is because the soft recovery threshold must be low enough that a worst case (rate which is vertical) violation can be recovered before triggering an emergency stop. But this is where the static threshold is usually located, even without ride-through. *In other words, using a rectangle doesn't actually help at all if you are designing a safety-critical system for the worst case!* A rectangular bound (a higher static limit with a maximum duration) is only useful for systems in which the property is unlikely to jump immediately and surpass the hard limit.

We can do better than a rectangular bound by limiting the rate of change of the violating property. Rate-limiting bounds (possibly including a bounded recovery) are more flexible and potentially more effective than static bounds. Figure 1 includes visualizations of a few rate-limited bounds.

Most systems have significant inertia in the form of physical mass, thermal mass, pressure storage vessels, moments of inertia, and the like that make them unlikely to exceed a certain rate of change unless a catastrophic failure has taken place. It may be very reasonable for a designer to assume that if a worst case violation doesn't happen at the outset, it is unlikely to happen in the near future. This permits playing the odds in terms of letting a system exceed its soft limit to give it a chance to recover on its own so long as the excursion past that limit is not dramatic. This can reduce system shutdowns for mild soft limit violations at the slightly increased risk of hard limit violations if the system suddenly exhibits a rapid change toward the hard limit threshold. (Playing the odds this way does not compromise safety since since the hard limit is ultimately what guarantees system safety.)

This idea can be enhanced by setting different soft limit bounds based on the operating state of the system (for example, relaxed ride-through limits when accelerating compared to steady state cruise if speed transients are more likely to be benign when accelerating).

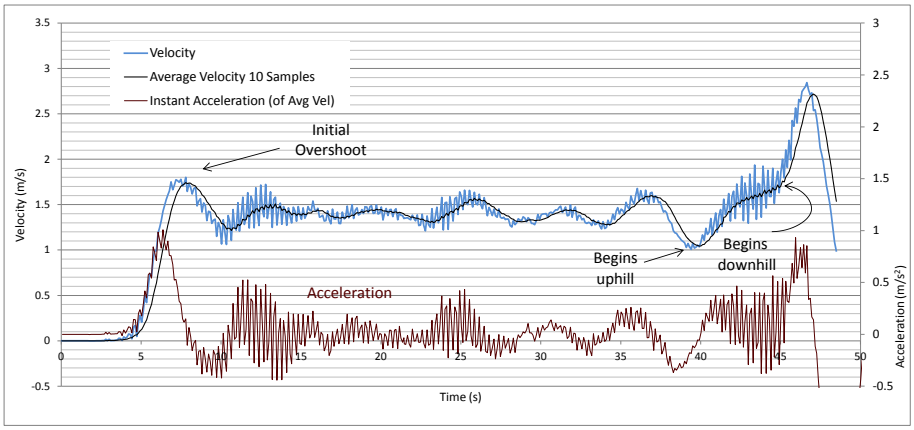

Fig. 2. Autonomous vehicle velocity and acceleration during test run

Modal	Rule	Time
no	0.4 Rate, Soft at 1.61	7.377
no	0.5 Rate, Soft at 1.61	44.611
no	0.6 Rate, Soft at 1.61	45.835
no	Static 1.75	45.223
yes	0.4 Rate, Soft at 1.54	42.673
yes	0.5 Rate, Soft at 1.54	44.611

(a) Shutdown trigger times for replayed ride-through rules

Description	Stopping Dist
Static 1.75	9.667m
Static 1.60	9.186m
0.5 Rate, Soft at 1.54	9.215m
0.4 Rate, Soft at 1.54	8.293m
0.35 Rate, Soft at 1.54	8.996m

(b) Stopping distance after triggered stop on vehicle

Table 1. Data from autonomous vehicle testing

4 Applying Ride-Through

We have collected data from an autonomous utility vehicle in which we induced an environmental fault to illustrate different ride-through techniques. We use a two-passenger, 1500lb electric utility vehicle outfitted with a simple autonomy package that has been used internally for hundreds of autonomy tests at a robotic research facility over the past five years.

Figure 2 shows a graph of the velocity of our autonomous vehicle commanded across a flat field and over a hill at $1.4m/s$ with our prototype monitor just passively listening and logging data. Because this vehicle is a relatively lightweight electric vehicle, it has little inertia and the velocity oscillates rapidly. We use a 10 sample running average velocity (calculated on the monitor) to provide a smoothed value to monitor (which we refer to as the velocity).

We replayed a logged trial of the vehicle on the monitor with a set of different ride-through rules. The limit values were picked based on the system dynamics seen in the traces, using rates near the worst case rates seen and soft limits barely above the usual value peaks. The time at which a simulated shutdown was triggered for each rule is shown in Table 1a. The **Modal** column shows whether the rule included the `cstate` mode-based rules to skip the overshoot

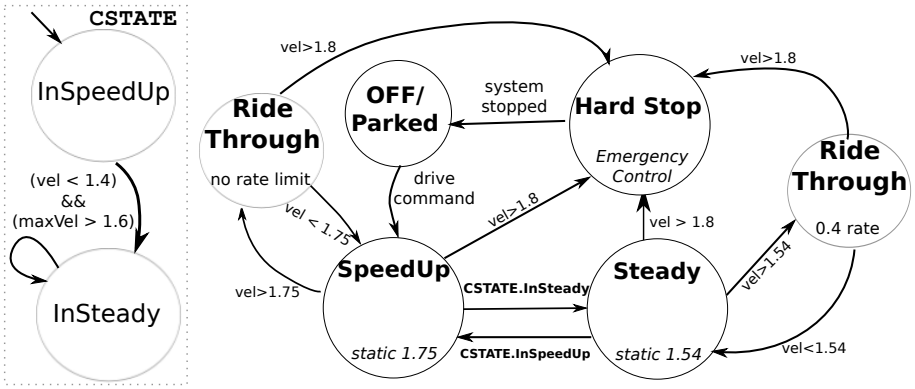

Fig. 3. State machine for mode `cstate` and ride-through strategy utilizing it

(See Section 4.1). The $1.75m/s$ static rule (acceleration ignored) is the lowest static rate that doesn't trigger on the initial speed limit overshoot. The $0.5m/s$ rate is the best performer on this trial (triggering earliest during the actual violation), showing that a rate-limiting bound can outperform the best static bound. (Note that the trigger time of $7.377s$ is a false positive, triggering on the initial vehicle speedup rather than during the true over-speed caused by the hill.)

4.1 Mode Based Limits

We can see that the vehicle widely overshoots its commanded speed when it begins accelerating on flat ground. A high static limit would be needed for the monitor to disregard this situation, but a high static limit that would allow the initial overshoot would permit significant violations of the speed limit that might be more problematic during other times as well.

Surprisingly, even a rate-limited bound doesn't necessarily protect us from issues such as this overshoot. As seen in Figure 2, the acceleration behaves just as erratically as the velocity does. Mode based limits can be used to monitor systems more effectively even when they have startup transients and the like.

For our example case, we can use a mode to select different safety bounds during the initial overshoot and for normal operation afterwords. This allows us to use a lower soft limit that would trip a false positive without the mode switch. Figure 3 shows a state machine depicting the `cstate` mode used during the experiment and the overall ride-through strategy that utilizes the mode. This rule uses a higher static limit until the velocity drops under $1.4m/s$ after peaking above $1.6m/s$. The depicted strategy has a hard limit of $1.8m/s$ with a static soft limit at $1.75m/s$ when the system is initially accelerating and a static $1.54m/s$ soft limit once the system has reached a steady state. The system attempts to ride-through speed limit violations with bounded rates of $0.4m/s^2$ under steady-state and no rate limit during the initial acceleration. As seen in Table 1, the specific static soft limit and ride-through rates can be changed to create different operating envelopes.

Having different safety rules during different system states is not unreasonable. For example, an autonomous vehicle might be able to move in a parking lot with a certain speed (parking lot mode), but probably should not be moving at all if its doors are open while in that parking lot (passenger entry/exit mode).

4.2 Combined Safety Rules

Ride-through and state based rules are independent yet complementary solutions – combining ride-through with state based rules allows tailoring the rate bounds based on the operating state of the system. As might be expected, these combined rules can lead to further improvement.

Table 1b shows the stopping distance for the vehicle after a soft stop (commanded $0m/s$ speed) was triggered by the monitor using the combined safety rules. We can see that for this particular system a rate limit of $0.4m/s^2$ and a soft threshold of $1.54m/s$ produced the minimum stopping distance, which is better than could be attained by either method alone.

5 Conclusion

Ride-through allows a runtime monitor to tolerate transient violations of the safety threshold of certain safety properties if the small violation can be considered within a safe operating envelope. This can be a useful addition to a safety system on an autonomous vehicle in order to reduce the chance of false alarm shutdowns while managing the risk of false negatives that result in hard shutdowns. The obvious simple ride-through bound (a rectangle) is in many cases no better than a fixed static threshold because it is vulnerable to false negatives for high-slope violations. Using both using state-based rules and/or rate limited bounds can provide a more effective ride-through strategy that tolerates mild soft limit violations while triggering a soft limit shutdown for more dramatic soft limit violations. Experiments on an autonomous electric vehicle show that state-based modes and rate limited bounds can provide a shorter stopping distance for a soft limit violation while avoiding false alarm shutdowns as well as undesirable hard limit shutdowns. This provides a proof-of-concept that these ride-through approaches can help improve the effectiveness of vehicle runtime monitoring systems.

References

1. Bishop, P., Bloomfield, R.: A methodology for safety case development. In: SAFETY-CRITICAL SYSTEMS SYMPOSIUM, BIRMINGHAM, UK, FEB 1998. Springer-Verlag, ISBN 3-540-76189-6 (1998),
2. Leucker, M., Schallhart, C.: A brief account of runtime verification. Journal of Logic and Algebraic Programming 78(5), 293 – 303 (2009), the 1st Workshop on Formal Languages and Analysis of Contract-Oriented Software (FLACOS'07)
3. Nelson, R.: Fault ride-through trip curves. In: Power and Energy Society General Meeting, 2012 IEEE. pp. 1–4 (2012)
4. Nelson, R.J., Ma, H., Goldenbaum, N.M.: Fault ride-through capabilities of siemens full-converter wind turbines. In: Power and Energy Society General Meeting, 2011 IEEE. pp. 1–5 (2011)