From Tracking Continuous Mode Hypotheses to Diagnosing Technical Systems

Bernhard Rinner
Institute for Technical Informatics
Graz University of Technology, AUSTRIA
b.rinner@computer.org

Abstract

As most technical systems such as plants, automobiles and robots are getting more and more complex, the need for automatic monitoring and diagnosis of such systems is steadily increasing. Technical systems provide many challenges for a monitoring and diagnosis systems, whereas the most important ones are: First, technical systems are typically dynamic, i.e., they change their state while the monitoring and diagnosis system is processing its input. Second, the supervised system is in most cases not completely known, and an exact model of that system may not be specified. Third, the observations can only provide an incomplete view on the supervised system due to discrete sampling, limited observability and noise.

In this paper, we present our model-based approach to monitoring and diagnosing technical systems. We address the challenges by (i) modeling the supervised system as a hybrid system and (ii) tracking continuous mode hypotheses. We have used this approach to implement a self-calibrating monitoring system which starts with a coarse description of the supervised system and exploits the observation to refine the behavior prediction and the underlying model. We discuss important issues to extend self-calibrating monitoring to diagnosing technical systems.

keywords: semi-quantitative reasoning; hybrid systems; tracker; self-calibrating monitoring; diagnosis

Introduction

As most technical systems such as plants, automobiles and robots are getting more and more complex, the need for automatic monitoring and diagnosis is steadily increasing. The primary objective of monitoring and diagnosis is to detect abnormal behaviors as soon as possible to avoid possible shutdown or damage, to propose hypotheses for detected abnormal behaviors, and to isolate possible faulty components.

This paper reports on recent results from an ongoing research project about monitoring and diagnosing technical systems. The goal of this project is to develop a model-based monitoring and diagnosis system (MDS) for online operation. This MDS is targeted for technical systems, such as intelligent robots, automotive applications and production processes. Common to all these systems is the tight connection between the (supervised) physical object and the monitoring and diagnosis system. Data from the supervised system is read via sensors, and (control) actions are issued from the monitoring and diagnosis system to the supervised system via actuators.

Technical systems provide a vast variety of challenges for a monitoring and diagnosis system. These challenges are mainly influenced by the properties of the technical system and the expected functionality of the monitoring and diagnosis system. In the following, we briefly summarize the main challenges.

Dynamic Systems Technical systems are typically dynamic, i.e., they change their internal state while the monitoring and diagnosis systems is processing its input. Thus, the monitoring and diagnosis system must be able to reason with the temporal evolution of the technical system, and the underlying model(s) must be able to express and predict dynamic behaviors.

Incomplete Knowledge Traditional monitoring approaches typically use a single precise model of the supervised system. However, even if the system is behaving properly, precise parameter values and functional relationships are often not known. More importantly, monitoring and diagnosis systems are designed to detect unexpected events or faults, after which knowledge of the system is by definition incomplete. The modeling framework for technical systems must therefore be able to express incomplete knowledge.

Uncertain Observations The accurate state of technical systems can not be observed through measurements due to the following reasons. First, measurements from the technical system may be corrupted by faulty sensors or noise. Second, limited observability of the supervised system as well as discrete sampling reduce further the capability to observe the accurate state of technical systems.

We address these challenges by (i) modeling the technical system as a hybrid system and (ii) by monitoring continuous mode hypotheses using trackers. In the remainder of this paper we present our approach in detail

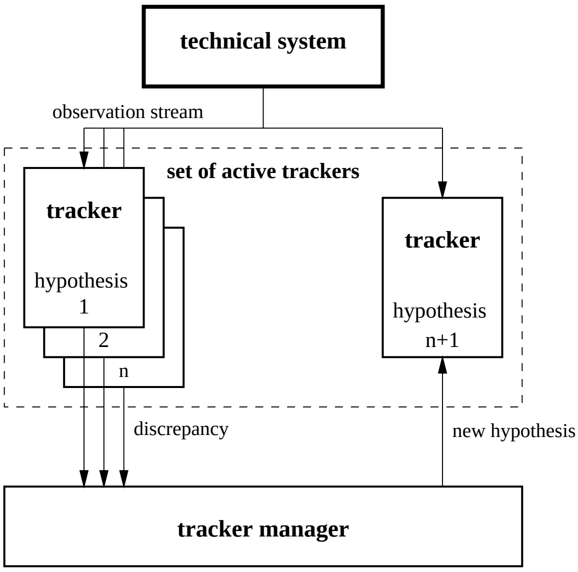

Figure 1: Monitoring and diagnosis architecture based on trackers.

and demonstrate its application to *self-calibrating monitoring*. Self-calibrating monitoring starts with a coarse description of the technical system and uses the observation stream to refine the behavior prediction and to reduce the imprecision in the underlying model. A brief discussion on related work and an outlook for further work concludes this paper.

Our Tracker-Based Approach to Monitoring and Diagnosis

Overview

Our general architecture for monitoring and diagnosis is centered around *trackers*. This architecture originates from the MIMIC approach (Dvorak & Kuipers 1991) and is an extension from our previous work (Rinner & Kuipers 1999b)(Rinner & Kuipers 1999a).

An overview of the general architecture is depicted in Figure 1. A set of (active) trackers monitors the state of the supervised system. A tracker monitors a particular hypothesis by comparing the observation from the technical system with the predicted behavior of the hypothesis. When a discrepancy between observation and prediction is detected, the discrepancy is signaled to the *tracker manager* and the tracker is dropped from the set of active trackers. The tracker manager then proposes new hypotheses, i.e., it generates new models which may better capture the current behavior of the supervised system. For each new hypothesis, a new tracker is initiated and added to the set of active trackers. Thus, our monitoring and diagnosis architecture allows to track multiple hypotheses in parallel.

Currently, our approach has been implemented using semi-quantitative reasoning techniques. These techniques excels in expressing and reasoning with incomplete knowledge. The main parts of our model-based monitoring and diagnosis architecture are (i) the modeling and simulation component, (ii) the trackers, and (iii) the tracker manager. We describe the main parts in the following:

Modeling and Simulation

We model and simulate technical systems as *hybrid systems* (Iwasaki *et al.* 1995)(Nishida & Doshita 1987). Hybrid systems exhibit a sequence of piecewise continuous behaviors interleaved with discontinuous changes (Branicky 1995). A continuous segment of the system's behavior is referred to as mode of operation and a discontinuous change is referred to as transition between modes.

The two main reasons for using hybrid systems in our approach are (i) model simplification and (ii) discrete control of physical systems. Many complex systems exhibit fast nonlinear behaviors that are irrelevant for many applications. In such cases, fast nonlinear system behaviors can be abstracted to discrete transitions resulting in a hybrid system model of the complex physical system (Mosterman & Biswas 1997). Our MDS is also targeted for monitoring and diagnosing technical systems with embedded discrete supervisory controllers. Supervisory controllers impose multiple continuous behavior segments that are best modeled as a hybrid system.

We assume that there are three possible causes for discontinuous changes in the model of a technical system. First, the autonomous operation of the technical system moves from one operating mode to the next. Second, the operator/controller takes a known action. In this case, we know the effect of the action, i.e., the new mode, but not a priori the condition or time when the action is issued. Finally, an unexpected and externally caused event such as a failure takes place. In this case, we do not know the new mode and, therefore, no pre-defined model is available as in the other cases.

The SQSIM framework (Kay 1998)(Kuipers 1994) is used to model and simulate technical systems. A continuous mode is imprecisely modeled by semi-quantitative differential equations (SQDEs). Imprecision in the SQDE is represented by numerical intervals bounding possible values of unknown parameters, and by static envelopes – functions bounding the possible graphs of unknown monotonic functions. The SQSIM simulator generates the behavior prediction of the mode. There are three levels of abstracted properties of the trajectories, corresponding to the level of detail derived by the components of SQSIM: *qualitative* (QSIM), *event* (Q2), and *dynamic envelope* (NSIM) descriptions (Figure 2). The qualitative description is defined by a sequence of symbols ($\downarrow$, $\ominus$ and $\uparrow$) representing the derivative's sign (qdir) of the trajectory at time points and intervals between time points. The event description specifies intervals bounding the trajectory at particular time points, i.e., magnitude and time ranges.

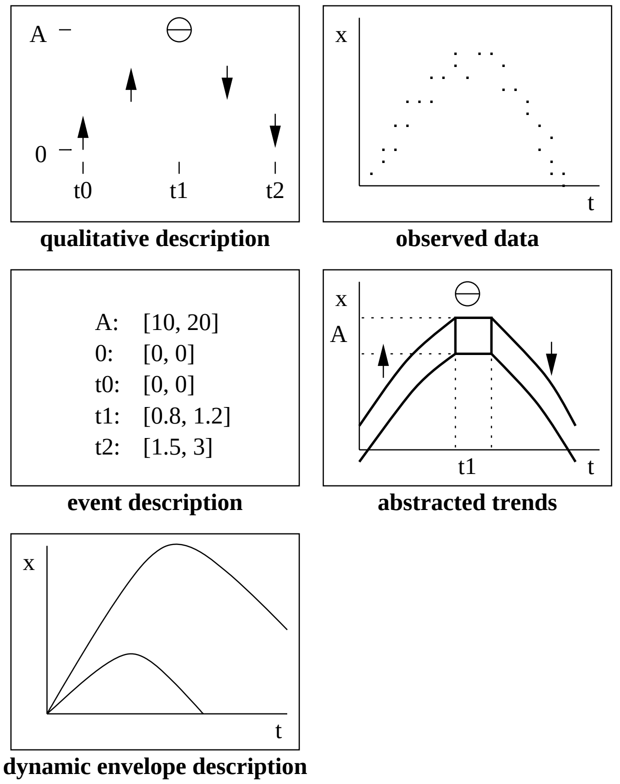

Figure 2: SQSIM describes the mode trajectory (SQ prediction) at the qualitative, event and dynamic envelope level. The observed data is abstracted at the same levels (SQ trend).

The dynamic envelope description bounds the trajectory by a lower and an upper envelope. A *trend* represents the abstracted properties of the observed data (Figure 2), i.e., symbols representing the qdir, bounding intervals on extrema and bounding envelopes for monotonic segments.

Transition functions specify the discontinuous changes from one continuous mode to another. Transition functions may be either triggered by an autonomous operation of the system, i.e., when the validity region of a mode is exceeded or a known operator/controller action.

Tracker

The objective of the tracker component is to track a single hypothesis about the current state of the technical system. The tracker is initialized with the behavior prediction of the hypothesis. It consumes an observation stream from the technical system and produces information about the matching between observation and behavior prediction.

The observation stream from the technical system may be preprocessed, i.e., filtered and abstracted, at the instrumentation interface. Matching the observation with the prediction is achieved by intersecting the

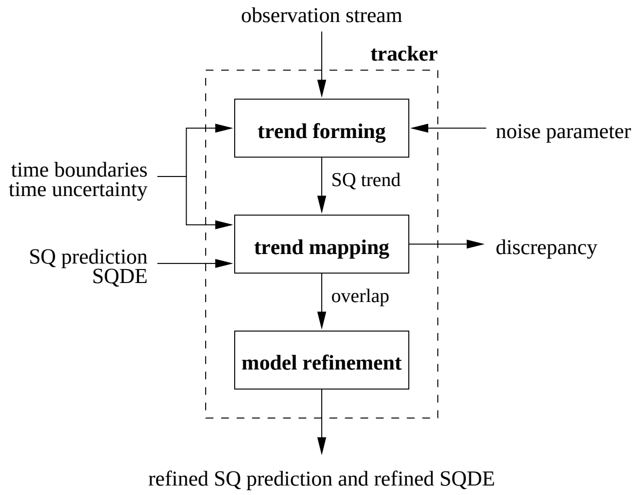

Figure 3: The tracker architecture based on SQUID.

observation and the prediction. If the intersection is empty, a discrepancy between the prediction and the observation has been detected. Otherwise, the observation supports the prediction derived from the model of the hypothesis. The intersection between observation and prediction is transferred to the tracker manager.

Since we model the technical system as a hybrid system, the tracker monitors a single *mode* hypothesis. A particular hypothesis is, therefore, a sequence of mode hypotheses $[H_1(t_0, t_1); H_2(t_1, t_2); \ldots H_n(t_{n-1}, t_n)]$ with mode changes at $t_1 \ldots t_{n-1}$.

The tracker is currently realized by SQUID (Kay, Rinner, & Kuipers 2000), a method for semi-quantitative system identification. SQUID compares the semi-quantitative trajectory descriptions generated by SQSIM (the SQ prediction) and the corresponding properties of the observation (the SQ trend). If the observation provides sufficient new information, it refines the underlying SQDE model. If the overlap between SQ prediction and SQ trend is empty, a discrepancy has been detected. Figure 3 presents the architecture of a tracker based on SQUID. It consists of three steps:

Trend forming generates an SQ trend describing each variable in the observation stream by breaking the samples into monotonic segments. The segments are determined by computing the slope of a linear least-squares fit to the data within a sliding window over the samples. Dynamic envelope descriptions for monotonic segments are generated by MSQUID, a neural network-based estimator for monotonic functions (Kay & Ungar 1993; 1999), out to any given confidence bound. Each segment representing an extremum is described by the segment's time interval and the minimum and maximum sample values over that interval.

The goal is to detect the qualitative dynamics of the underlying signal in the noisy observation. In the current implementation it is assumed that Gaussian noise of fixed mean and variance is superimposed on the "pure" signal. For each observed variable, parameters specify bounds on mean and variance for noise.

Trend mapping compares the SQ trend derived from the observations with the SQ prediction by stepping through both sequences. If an inconsistency is detected between the trend and the prediction, the current hypothesis is refuted, so the mapping process and the current tracker are aborted.

Mapping at the qualitative level generates a correspondence between the symbols $\uparrow$, $\ominus$ and $\downarrow$ in the prediction and the trend. A valid correspondence may fail to be one-to-one because (i) the samples in the observation stream may end before some of the qualitative changes in the prediction take place; (ii) the prediction terminates with a mode change before the end of the current trend; or (iii) the prediction may include small qdir changes which are not detectable in a noisy observation stream.

Mapping at the event level ensures consistency of corresponding behavior events in the trend and the prediction, in the sense that their time and magnitude bounds overlap.

Mapping at the dynamic envelope level ensures consistency by intersecting the dynamic envelopes for corresponding monotonic segments of the trend and the prediction.

Model refinement takes place when trend mapping decreases the bounds on some variables in the SQDE. Parameter imprecision is refined by using interval arithmetic to derive bounds on independent variables from dependent ones. SQUID guarantees that portions of the model space are ruled out only when they are inconsistent with the observations (Kay, Rinner, & Kuipers 2000).

By using SQUID a tracker clearly distinguishes between the cases where (i) an observation is consistent with the model but provides no new information; (ii) the observation provides new information further reducing the current model space; and (iii) the observation provides new information that reduces the current model space to the empty set, refuting the hypothesis.

Tracker Manager

The tracker manager coordinates all trackers during the monitoring process. Thus, its main tasks are (i) to detect mode changes and faults, (ii) to initiate new trackers at mode changes (and drop old ones), and (iii) to discriminate among competing hypotheses.

Detection of Mode Changes and Faults Analyzing the data delivered by the trackers is important for the detection of mode changes and the detection of faults. Monitoring and diagnosis with tracking continuous mode hypotheses relies on the early detection of mode changes. If a mode change is missed, fault detection is delayed, and in the worst case the fault may never be detected at all. In case of a missed mode change, the (initially) correct hypothesis becomes wrong and the corresponding tracker matches the observed data of the new mode to the prediction of the old mode. The wrong hypothesis is only detected when the overlap between prediction and observation becomes empty.

Mode changes may be detected in different ways, i.e., (i) by checking for abrupt changes in signals,[1] (ii) by checking for exceeding an a priori known validity regions of modes, and (iii) by exploiting auxiliary signals from controllers indicating control actions. All techniques are applied in our self-calibrating monitoring system in order to avoid missing a mode change.

Faults may be detected at a mode change or during tracking a mode hypothesis. A detected mode change is caused by a fault when the mode changes is not due to an autonomous operation or a known controller action. Thus, if a mode change has been detected, the tracker manager compares the detected mode change with the possible mode changes of this mode. If it is a unknown change a fault is signaled. During tracking the intersection between the SQ trend and the SQ prediction may become empty indicating an inconsistency between prediction and observation, i.e., detecting a fault. In order to detect faults as early as possible the evolution of the intersection is also monitored. A decreasing intersection is a strong indication for an (incipient) fault.

Instantiation of new Trackers In case of a mode change, the tracker for the old mode must be dropped from the list of active trackers and a tracker for the new mode must be instantiated and added to the list of active trackers. Self-calibrating monitoring makes an important assumption about the mode changes: they are either caused by the autonomous operation of the technical system or by a known operator action. Thus, the models of all modes are known, and no model generation is required by the tracker manager.

Our self-calibrating monitoring system refines the underlying SQDE model during tracking. Although the mode model for the new mode is known a priori, the achieved refinements can only be incorporated into the mode model at the tracker instantiation. Thus, model refinements such as refined variable bounds or bounding envelopes may be inherited from one model to the next across a mode transition. In the current implementation, the user specifies the variables and functional relations whose refinements may be inherited.

Discriminating Competing Hypotheses This tracker-based architecture allows to track a number of hypotheses in parallel. The difficulty is that in realistic situations the number of possible hypotheses may be intractable. So we must focus on the most plausible fault hypotheses, but without sacrificing coverage. We

[1] Mode changes are often manifested in abrupt changes of signals.

address the problem of a large number of concurrent fault hypotheses by exploiting the likelihood of trackers and by using heuristics to discriminate among competing trackers.

Conclusion

This paper has presented self-calibrating monitoring – a monitoring and diagnosis approach based on tracking continuous mode hypotheses. Self-calibrating monitoring is an important step toward completely diagnosing technical systems. It facilitates the treatment of incomplete and imprecise knowledge of the supervised system as well as uncertain and noisy observations, all of which is important in technical environments.

In this approach, imprecision is expressed by *guaranteed* intervals for the prediction and the observation.[2] These "hard" bounds are essential for genuinely refuting hypotheses during the monitoring and diagnosis process. These "hard" bounds may, however, get large and should be complemented by "soft" bounds derived by statistical methods such as parameter estimation.

Closely related work to this research has been done by Biswas et al. (Mosterman, Zhao, & Biswas 1998)(Manders, Mosterman, & Biswas 1999). In their framework for model-based diagnosis the physical system is modeled by a temporal causal graph. Qualitative candidate models are derived from this representation and parameter estimation techniques are applied to fit the candidate models to the observation (Manders *et al.* 2000). This quantitative/qualitative approach to diagnosis has been recently extended to hybrid systems (Narasimhan *et al.* 2000).

TrenDx (Haimowitz & Kohane 1993) is a monitoring system which uses a semi-quantitative representation of a behavior and attempts to fit data to this behavior representation. Since TrenDx uses pre-defined behavior templates no refinement can be performed. Loiez and Taillibert (Loiez & Taillibert 1997) use piecewise polynomial functions, so-called temporal band sequences, to bound the observation stream. The behavior of components in analog circuits is modeled by sums of temporal functions including derivatives of any order. This approach is only able to detect discrepancies but not to predict the behavior of the system. McIlraith et al. (McIlraith *et al.* 1999) present an approach for diagnosing hybrid systems where all mode transitions, i.e., the history of executed actions, are known. Candidate generation and model estimation are based on the model-based diagnosis framework of (Mosterman, Zhao, & Biswas 1998) and a tracker-based framework is adopted to refine multiple candidates.

Further work of this project includes the modeling and simulation component, the tracker manager, and the application of the MDS. We are currently investigating to extend our semi-quantitative modeling approach with HCC (Gupta *et al.* 1995), a hybrid modeling environment based on concurrent constraint programming languages. An important issue is how HCC can be exploited for automatic model building. In order to extend self-calibrating monitoring to diagnosis, the tracker manager must be able to automatically select relevant model fragments to express initially weak knowledge about a fault to a mode model. Finally, we will demonstrate our monitoring and diagnosis system on a complex technical system.

[2]The observed data stream is bounded by the function estimator MSQUID out to any confidence bound.

Acknowledgment

This work has taken place at the Institute for Technical Informatics, Graz University of Technology, Austria and at the Department of Computer Sciences, University of Texas at Austin. This research has been supported by the Austrian Science Fund under grants J1429-MAT and P14233-INF.

References

Basseville, M., and Nikiforov, I. 1993. *Detection of abrupt changes: theory and applications*. Prentice-Hall.

Branicky, M. 1995. *Studies in Hybrid Systems: Modeling, Analysis, and Control*. Ph.D. Dissertation, Department od Electrical Engineering and Computer Science, Massachusetts Institute of Technology.

Dvorak, D., and Kuipers, B. 1991. Process Monitoring and Diagnosis: A Model-Based Approach. *IEEE Expert* 5(3):67–74.

Gupta, V.; Jagadeesan, R.; Saraswat, V.; and Bobrow, D. 1995. Programming in hybrid constraint languages. In Antsaklis, P.; Kohn, W.; Nerode, A.; and Sastry, S., eds., *Hybrid Systems II*, volume 999 of *Lecture Notes in Computer Science*. Springer Verlag.

Haimowitz, I. J., and Kohane, I. S. 1993. Automated Trend Detection with Alternate Temporal Hypotheses. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, 146–151.

Iwasaki, Y.; Farquhar, A.; Saraswat, V.; Bobrow, D.; and Gupta, V. 1995. Modeling Time in Hybrid Systems: How Fast Is "Instantaneous"? In *Proceedings of the Ninth International Workshop on Qualitative Reasoning*.

Kay, H., and Ungar, L. H. 1993. Deriving Monotonic Function Envelopes from Observations. In *Working Papers from the Seventh International Workshop on Qualitative Reasoning about Physical Systems (QR-93)*, 117–123.

Kay, H., and Ungar, L. 1999. Estimating Monotonic Functions and Their Bounds using MSQUID. Technical Report TR AI99-280, University of Texas at Austin.

Kay, H.; Rinner, B.; and Kuipers, B. 2000. Semi-Quantitative System Identification. *Artificial Intelligence* 119(1-2):103–140.

Kay, H. 1998. SQSIM: A Simulator for Imprecise ODE Models. *Computers and Chemical Engineering* 23(1):27–46.

Kuipers, B. 1994. *Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge.* Artificial Intelligence. MIT Press.

Loiez, E., and Taillibert, P. 1997. Polynomial Temporal Band Sequences for Analog Diagnosis. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, 474–479. Nagoya, Japan: Morgan Kaufmann.

Manders, E.-J.; Narasimhan, S.; Biswas, G.; and Mosterman, P. J. 2000. A Combined Qualitative/Quantitative Approach for Fault Isolation in Continuous Dynamic Systems. In Edelmayer, A., ed., *Proceedings of the 4th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS'2000)*, 512–517.

Manders, E.; Mosterman, P. J.; and Biswas, G. 1999. Signal to Symbol Transformation Techniques for Robust Diagnosis in TRANSCEND. In *Proceedings of the Tenth Workshop on Principles of Diagnosis (DX99)*, 155–165.

McIlraith, S.; Biswas, G.; Clancy, D.; and Gupta, V. 1999. Towards Diagnosing Hybrid Systems. In *Working Notes of the AAAI 1999 Spring Symposium Series on Hybrid Systems and AI: Modeling Analysis and Control of Discrete + Continuous Systems*, 128–135. Stanford, CA: AAAI.

Mosterman, P. J., and Biswas, G. 1997. Formal Specifications for Hybrid Dynamical Systems. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, 568–573.

Mosterman, P. J.; Zhao, F.; and Biswas, G. 1998. An Ontology for Transitions in Physical Dynamic Systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, 219–224.

Narasimhan, S.; Zhao, F.; Biswas, G.; and Hung, E. 2000. Fault Isolation in Hybrid Systems Combining Model Based Diagnosis and Signal Processing. In Edelmayer, A., ed., *Proceedings of the 4th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS'2000)*, 1074–1079.

Nishida, T., and Doshita, S. 1987. Reasoning about Discontinuous Change. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, 643–648. Seattle, Washington: AAAI.

Rinner, B., and Kuipers, B. 1999a. Monitoring Piecewise Continuous Behaviors by Refining Semi-Quantitative Trackers. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, 1080–1086. Stockholm, Sweden: Morgan Kaufmann.

Rinner, B., and Kuipers, B. 1999b. Monitoring Piecewise Continuous Behaviors by Refining Trackers and Their Models. In *Hybrid Systems and AI: Modeling, Analysis and Control of Discrete + Continuous Systems*, AAAI Technical Report SS-99-05, 164–169.

Wang, Y. 1995. Jump and sharp cusp detection via wavelets. *Biometrika* 82(2):385–397.

Most papers of the author are available at `www.iti.tu-graz.ac.at/en/people/rinner`.