Oscillation Analysis in Behavior-Based Robot Architectures

Lisa Wilhelm[1] and Martin Proetzsch[1] and Karsten Berns[1]

Robotics Research Lab, University of Kaiserslautern, Gottlieb-Daimler-Str., 67663 Kaiserslautern

Abstract. This paper presents a method for detecting oscillations in behavior-based robot control networks. Two aspects are considered. On the one hand the detection of oscillations inside single behavior modules is based on analyzing the signal in the frequency domain using the Fast Fourier Transformation (FFT). On the other hand tracing oscillations through the behavior network helps to evaluate its propagation and to find its root cause. Results of the oscillation analysis are presented using appropriate visualization techniques. The suitability of the proposed approach is shown by an indoor application.

1 Introduction

In the domain of robot control architectures the behavior-based approach is a widely accepted method for building up complex robotic systems. However, one often mentioned criticism is that this methodology is unprincipled and hard to evaluate [1]. Although several implementations [2] have validated the suitability of the approach, the problem of analyzing large scale networks of distributed components with heavy interaction persists.

One aspect with a possibly bad influence on the performance of robotic systems is oscillations occurring inside the control structure. The knowledge about this information may help in diagnosing design errors with negative consequences on the robustness, performance, and reliability of the system. For analyzing these effects, this paper presents techniques for detecting oscillations inside behavior-based networks. The oscillation analysis problem splits up into two aspects:

Oscillation detection: Oscillations have to be detected in single behaviors and must be evaluated concerning the severity using suitable quality criteria.

Oscillation tracing: In order to find the root cause of an oscillation, the path it takes through the behavior-based control network has to be determined.

These problems are also known in mechanical and electrical engineering, where oscillations in control units impair continuous work routines in plants. A lot of research is done to optimize these processes leading to many techniques for detecting and diagnosing oscillations in plants [3]. For software systems, however, it seems that oscillation detection is not sufficiently discussed, although it is a very interesting part of the analysis of robot control systems and can be used to generate quality data for assessing system properties. The oscillation detection algorithm presented here was developed on the basis of the techniques used by electrical engineers and adapted to the behavior-based robot control system iB2C (integrated Behavior-Based Control) [4].

The development of complex robotic systems requires frameworks and tools in order to allow an effective realization in respect to given goals [5]. Three topics have to be addressed: The specification of system properties, the execution of functionality, and the validation of system characteristics. The work at hand deals with the latter aspect in the sense that desired or unwanted oscillations are tracked and visualized.

In this field, several approaches deal with formal verification of systems. Typical examples are proving logical aspects and temporal characteristics of spacecraft applications [6] and real-time systems [7]. However, the complexity of these systems is limited. Furthermore, on-line analysis methods are required to detect effects appearing through the interaction with a dynamic environment. Other approaches (e. g. [8]) analyze systems by determining architectural characteristics under certain hypotheses. However, only quite simple subsystems are analyzed or the possible set of validated properties is limited.

Further analysis techniques involve data visualization tools [9,10]. These approaches have in common that situations have to be evaluated by the developer by looking through the given data, diagrams, or charts. In the field of electrical engineering several oscillation detection approaches are used for optimizing automation processes [3,11]. On account of the applications the techniques are developed for off-line analysis of logged data and therefore have to be adapted accordingly to be applicable for on-line analysis of robot control systems.

2 Development Framework

While the chosen approach for oscillation analysis can be applied to applications other than the one presented the following prerequisites have to be fulfilled:

1. Data acquisition has to be provided with a fixed cycle time due to the FFT.
2. In order to trace oscillations and find the root cause a modular structure with uniform connection characteristics has to be given.

A framework complying with these demands is the Modular Controller Architecture 2 (MCA2) [12,13] with the iB2C extension made at the Robotics Research Lab at the University of Kaiserslautern (MCA2-KL[1]).

MCA2 is constructed as a hierarchical control structure composed of components which are connected by edges via standardized interfaces. A *module* is the basic unit of the MCA2 framework providing a special functionality. Furthermore, *groups* can be used to include several modules and therefore add a further level of hierarchical abstraction. Execution of included modules or groups is performed using so-called *parts* which control the internal data flow. The iB2C architecture provides a behavior-based extension to the basic MCA2 network [4].

[1] `http://rrlib.cs.uni-kl.de/`

Based on the iB2C principles, large scale behavior-based networks for different kinds of robots have been built up. In order to keep track of the system implementations several tools are used, e. g. a graph as presented in section 3.2. Here, the network is represented as nodes connected by edges transferring data between the behaviors.

3 Oscillation Analysis

The procedure of analyzing oscillations in large scale behavior-based networks is divided into two parts [14]: the detection of oscillations in the signal data of single behaviors (see Sect. 3.1) and the propagation of oscillations through the behavior network (see Sect. 3.2).

3.1 Oscillation Detection Procedure

The oscillation detection algorithm is required to find oscillations in a data signal during run time. The method described here is a further development of [11] and analyzes the frequency spectrum of the supervised signal using the Fast Fourier Transformation (FFT). Thornhill's method is extended to allow real-time oscillation detection and the calculation of a threshold for the classification of detected oscillations is adapted to the special needs of the behavior system. The algorithm can be used with any data. In our approach the activity data of the behaviors is used. In order to apply the FFT, the sampled signal data (see Fig. 1 (a)) is stored in a ring buffer. The sampling rate f_s determines the highest detectable frequency $f_{\max}$ and the resolution of the frequency spectrum $f_{\min}$ depending on the length of the ring buffer $l = 2^n$: $f_{\max} = (f_{\min} \cdot l)/2 = f_s/2$.

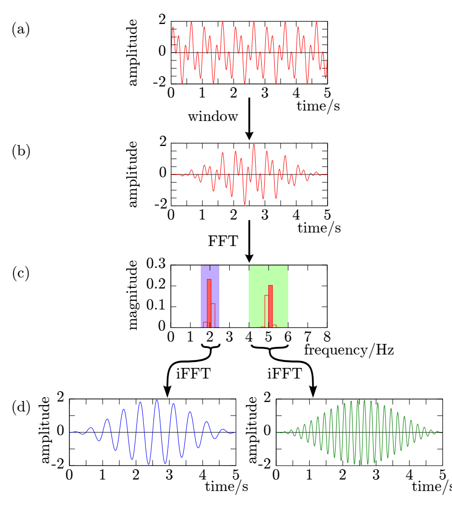

Fig. 1. Three steps of the oscillation detection algorithm

As a first step, the Hann-window is applied to the original signal to avoid leakage effects (see Fig. 1 (b)). Then the signal is transformed to the frequency domain (see Fig. 1 (c)) using the FFTW C subroutine library[2]. The peaks in the power spectrum indicate potential oscillations which need to be further classified in order to neglect false diagnoses, e. g. due to noise. For that purpose two properties—*regularity* and *power*—are introduced. Both are based on a filtered spectrum which contains only a certain interval around the related peak (see Fig. 1 (c)). The *power* is calculated by summing up the magnitudes of the filtered spectrum. The *regularity* represents the deviation of the distances between zero-crossings of the signal in time domain. See the two signals in Fig. 1 (d), where the two parts of the frequency spectrum are transformed back to time domain. The most powerful and regular oscillations should be further analyzed.

Thresholds for both properties are user defined. Here the method of [11] is used to indicate a regular oscillation, assuming that the standard deviation of the period is less than one third of the mean value: $r = \overline{T_p}/(3 \cdot \sigma_{T_P})$, with values of $r > 1$ indicating a regular oscillation. The determination of a lower border of the power is derived on the basis of the following consideration. A signal with n oscillations of the same power but different frequencies will have a spectrum, where the power p_i is approximately uniformly distributed between all peaks with $p_i \approx 1/n$. Therefore, in our approach an oscillation is said to be relevant if $p_i > \sum_{i=0}^{n-1} p_i/(n+1)$. Although the proposed procedure makes use of computationally efficient techniques like the FFT, continuously analyzing all signals in a complex system can have negative effects on the system's cycle time. To avoid this, only a limited number of modules continuously performs the presented calculations, while the other modules only store relevant data in order to execute the oscillation analysis on demand.

3.2 Oscillation Tracing

In order to facilitate the tracing of oscillations, the behavior network is represented using the Boost Graph Library (BGL)[3]. This way, the tracking of signals becomes easy and a graphical output of the structure with overlaid oscillation information can be automatically generated using Graphviz[4].

The tracing is done recursively starting from the behavior module that first detects an oscillation (see Alg. 1). The algorithm stops either when a module has no oscillating neighbors or when a module has no other in/out edges. This module is then likely to be the root-cause of the oscillation. In order to reduce the computational complexity, only selected modules continuously execute the oscillation detection algorithm, while all modules buffer the signal data. In the case of an oscillating module, all modules which are connected via activity edges perform the algorithm and report back if they also detect oscillations. In iB2C, activity deals as a measure for the intended impact of a behavior. The results containing the frequency, regularity, and power are temporarily stored.

[2] `http://www.fftw.org/`
[3] `http://www.boost.org`
[4] `http://www.graphviz.org`

Algorithm 1 Tracing of oscillations

```
1:  start data buffering in every module
2:  start oscillation detection in specified module
3:  if module detects one or more oscillations then
4:      save result temporarily
5:      for all in/out activity edges do
6:          if target module has not been checked before then
7:              execute line 3 for target module
8:          end if
9:      end for
10: end if
```

As an example for the visualization of the oscillation path, Fig. 2 shows the Graphviz representation of the control network of one of our autonomous robots with overlaid information about detected oscillations. Here, the paths of oscillations are depicted by overlaying color information over the behavior network representation. Furthermore, information about frequency (f), power (p), and regularity (r) are given.

4 Application and Results

The oscillation analysis and tracing algorithm has been applied to some of the robot control systems in our lab to show the different application possibilities mentioned above. Here, the algorithm is used to find modules responsible for jiggling of the indoor robot ARTOS (Autonomous Robot for Transport and Service) during obstacle avoidance maneuvers. ARTOS is a service robot developed in the Robotic Research Lab for usage in the BelAmI project[5] [13].

The control network of ARTOS consists of over 60 interconnected modules. Fig. 2 shows the group `Behaviors` with all of its nested groups which are responsible for the control of the robot's behavior. The large group `Safety behaviors` includes components responsible for processing sensor data and for generating motion commands. This data is converted to wheel velocities by the lowest group `Basic Differential Drive behaviors`.

It is known from former experiments that ARTOS visibly oscillates in certain situations, for example when a small table leg disrupts its path. Therefore, the oscillation tracing method has been used for tracking down involved behaviors. As start modules the four behaviors responsible for forward, backward, turn left, and turn right motion are selected, since these act as an interface for all motion commands.

The results of the oscillation tracing algorithm point out that the oscillation begins in different sensor processing modules. Fig. 2 shows one outcome of the oscillation analysis. Here the path of an oscillation starts in the modules processing the data of the sensors responsible for keeping a certain distance

[5] `http://www.belami-project.de`

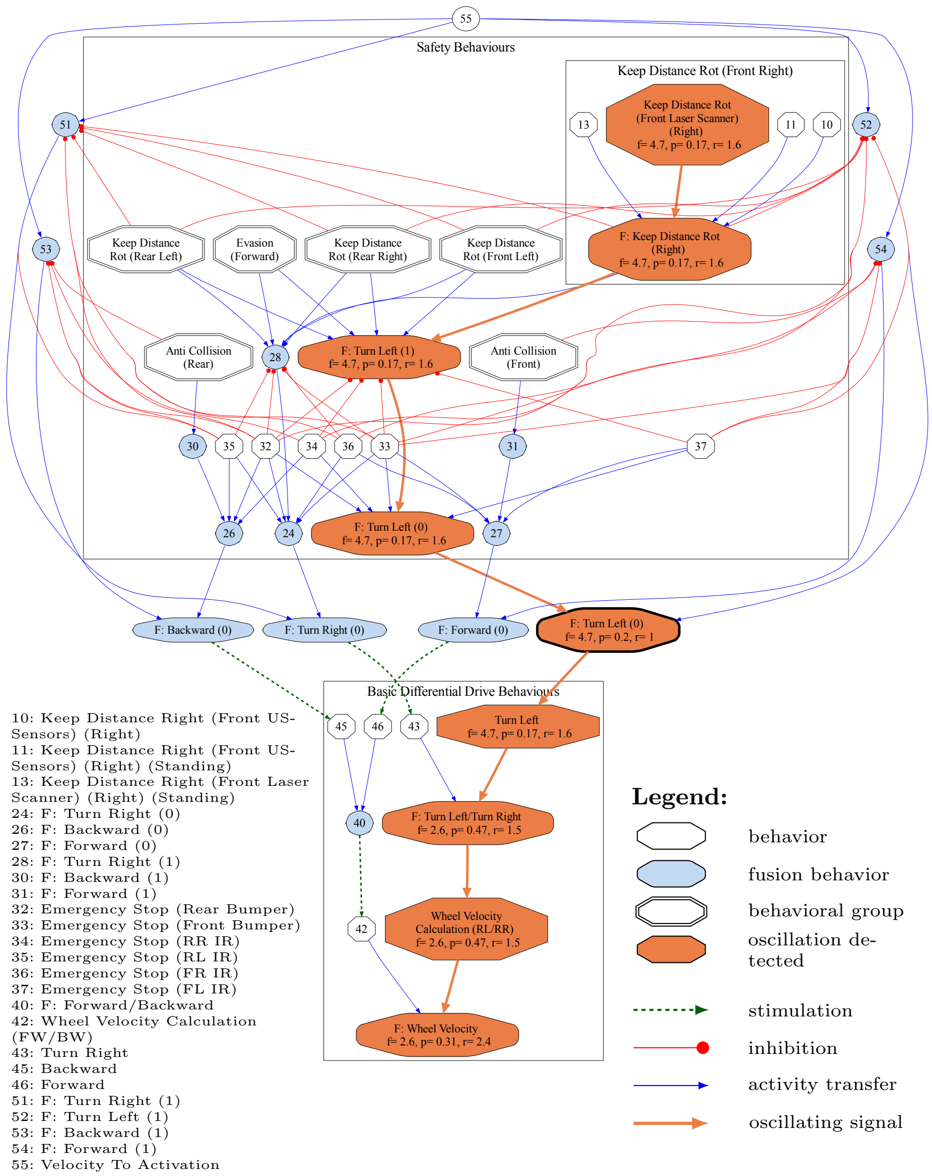

Fig. 2. Boost graph representation of the control network of the mobile robot ARTOS visualized with Graphviz. The course of the oscillation through the network is depicted by marked behaviors and edges, allowing a rapid evaluation of the oscillation tracing result. The module which detected the oscillation first is framed with a thick border.

to obstacles by rotating away (`Keep Distance Rot (Front Laser Scanner) (Right)`). The oscillation is further tracked down to the module which is responsible for the control of the actuators. The same oscillation (same frequency) could be found in the behavior responsible for driving in the opposite direction. The results of the algorithms confirmed the apparently existing oscillation, identified the corresponding frequencies and characteristics and detected the involved behaviors, i. e. the course of the oscillation through the network.

For a further evaluation of the detected oscillation, Fig. 3 (top) shows the activity signal of the `Turn Left` behavior. The diagram in the middle depicts the frequencies and time intervals of detected oscillations. As can be seen, several different oscillations were found in the signal during the jiggling motion. This test was done with a sampling rate of 100 Hz and a window length of 1024 samples. Fig. 3 (bottom) shows one of the corresponding frequency spectra and the peak which was classified as oscillation.

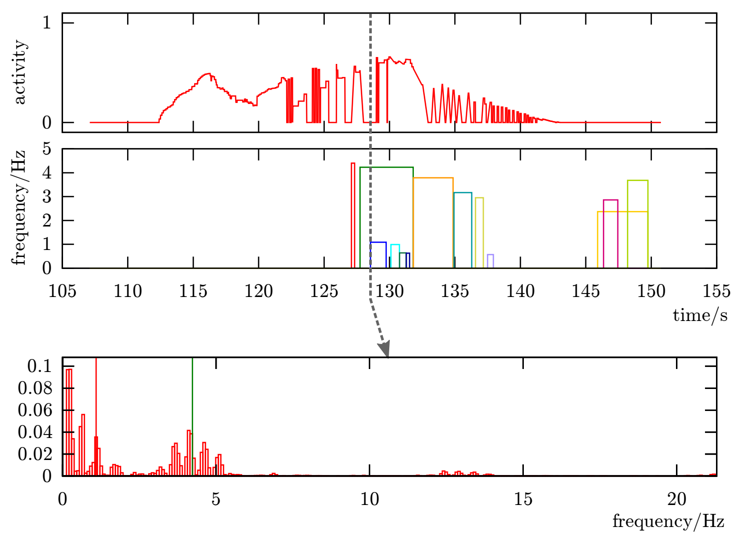

Fig. 3. Activity signal (top) and detected oscillations (middle) of the module `Turn Left` of the ARTOS-project. Every single detected frequency is shown with a different line type. A peak corresponds to the previous 1024 data values. Bottom: Frequency spectrum of the activity signal for the oscillations detected after about 128 s.

5 Conclusion and Future Work

The algorithms presented in this paper introduce methods for on-line oscillation detection and tracing inside behavior-based control networks. It is shown how this procedure deals with the detection of possible malfunctions inside the control structure. This way, the evaluation results can be used as quality data for larger scale system assessments. Furthermore, several visualization techniques facilitate the evaluation of the results.

Future work includes automatic methods for adapting the system behavior in the case of unwanted oscillations in order to improve the system performance.

Acknowledgments

This work was partly funded by the German Federal Ministry of Education and Research (BMBF) in the context of the project ViERforES (No.: 01 IM08003 E).

References

1. R. A. Brooks, "Intelligence without reason," Massachusetts Institute of Technology (MIT), Artificial Intelligence Laboratory," A.I. Memo No. 1293, 1991.
2. R. Arkin, *Behavior-Based Robotics.* MIT Press, 1998.
3. P. Odgaard and K. Trangbaek, "Comparison of methods for oscillation detection - case study on a coal-fired power plant," in *Proceedings of IFAC Symposium on Power Plants and Power Systems Control 2006*, 2006.
4. M. Proetzsch, T. Luksch, and K. Berns, "Development of complex robotic systems using the behavior-based control architecture iB2C," *Robotics and Autonomous Systems*, vol. 58, no. 1, 2010.
5. E. Coste-Maniere and R. Simmons, "Architecture, the backbone of robotic systems," in *Robotics and Automation, 2000. Proceedings. ICRA 2000. IEEE International Conference on*, vol. 1, 2000.
6. M. Lowry, K. Havelund, and J. Penix, "Verification and validation of AI systems that control deep-space spacecraft," in *Proceedings Tenth International Symposium on Methodologies for Intelligent Systems.* Springer, 1997.
7. K. Kapellos, D. Simon, M. Jourdan, and B. Espiau, "Task level specification and formal verification of robotics control systems: state of the art and case study," in *Int. Journal of Systems Science*, ser. 11, vol. 35, 1999, pp. 1227–1245.
8. L. Steels, "Mathematical analysis of behavior systems," *From Perception to Action Conference, IEEE Computer Society Press*, 1994.
9. R. Simmons and G. Whelan, "Visualization tools for validating software of autonomous spacecraft," in *In Proc. of the Fourth International Symposium on Artificial Intelligence, Robotics, and Automation for Space (iSAIRAS)*, 1997.
10. Z. Wang, J. Tsai, and C. Zhang, "Fault-toleration by duplication and debugging for distribution real-time systems," *Tamkang Journal of Science and Engineering*, vol. 3, no. 3, pp. 187–194, 2000.
11. N. Thornhill, B. Huang, and H. Zhang, "Detection of multiple oscillations in control loops," *Journal of Process Control*, vol. 13, pp. 91–100, 2003.
12. K. U. Scholl, J. Albiez, and G. Gassmann, "MCA – an expandable modular controller architecture," in *3rd Real-Time Linux Workshop*, Milano, Italy, 2001.
13. J. Koch, M. Anastasopoulos, and K. Berns, "Using the modular controller architecture (MCA) in ambient assisted living," in *3rd IET International Conference on Intelligent Environments (IE07)*, Ulm, Germany, September 24-25 2007, pp. 411–415.
14. L. Wilhelm, "Oscillation detection in behaviour-based robot architectures," Diploma Thesis, University of Kaiserslautern, November 2008, unpublished.