Organic Computing in Off-highway Machines

Micaela Wünsche
Institute AIFB
Karlsruhe Institute of Technology
Karlsruhe, Germany
micaela.wuensche@kit.edu

Sanaz Mostaghim
Institute AIFB
Karlsruhe Institute of Technology
Karlsruhe, Germany
sanaz.mostaghim@kit.edu

Hartmut Schmeck
Institute AIFB
Karlsruhe Institute of Technology
Karlsruhe, Germany
hartmut.schmeck@kit.edu

Timo Kautzmann
Institute MOBIMA
Karlsruhe Institute of Technology
Karlsruhe, Germany
timo.kautzmann@kit.edu

Marcus Geimer
Institute MOBIMA
Karlsruhe Institute of Technology
Karlsruhe, Germany
marcus.geimer@kit.edu

ABSTRACT

Machine management systems in off-highway machines such as tractors or wheel loaders are designed for efficient operation and reduced fuel consumption in some predefined scenarios for which the machine has been developed. In this paper, we outline how concepts from Organic Computing may be used to realize a self-organizing, reliable, adaptive, and robust machine management system that is capable of adjusting to new situations. We propose an architecture for a machine management system based on the generic Observer/Controller architecture and focus on the structure of the Observer. Furthermore, we study the feasibility of working cycle detection by the Observer and analyze real and synthetic data from an off-highway machine. To extract features by which different working cycles of an off-highway machine can be distinguished, we employ principal component analysis.

Categories and Subject Descriptors

J.7 [**Computers in other Systems**]: *Command and control*; C.3 [**Special-Purpose and Application-Based Systems**]: *Signal processing systems*; I.5.4 [**Pattern Recognition**]: Applications—*Signal processing*; J.2 [**Physical Sciences and Engineering**]: *Engineering*

General Terms

Design, Reliability

SOAR'10, June 7, 2010, Washington, DC, USA.

Keywords

Organic Computing, Observer/Controller architecture, Machine Management, Off-highway Machines, Principal Component Analysis

1. INTRODUCTION

Fuel consumption is one of the major issues in vehicle technology, and particularly in mobile machinery. For this reason, several machine management systems have been developed [2, 6, 5, 19], by which designers intend to reduce fuel consumption, e. g., providing strategies and parameter settings that ensure efficient coordination of the machine's components in different operating modes. In this paper, we study the family of off-highway machines such as tractors, wheel loaders, etc. Usually, these vehicles are designed to work in specific pre-optimized modes. However, if they are used in situations for which they have not been specifically designed, fuel consumption increases dramatically. Other off-highway machines like tractors are intended for multipurpose use, making it especially difficult to optimize them for all eventualities.

The main idea of this paper is to present a new approach to machine management systems that enables an off-highway machine to adapt to new situations and optimize according to given objectives. Minimization of fuel consumption in a tractor is used as an example. For achieving a self-organizing and adaptive machine management system we present an architecture adopted from Organic Computing [3]. This generic Observer/Controller (O/C) architecture [15] has been developed in order to control complex self-organizing systems and improve their robustness and reliability. It intends to keep them manageable, and to provide an interface for external users to influence the system.

The basic structure of this architecture is similar to an adaptive control loop [18, 1]. However, in adaptive control, parameters of the controller of a single system are adjusted in order to enable this one controller to work efficiently in a larger variety of environmental conditions. The three most common approaches to adaptive control are Model Reference Adaptive Control (MRAC), Self Tuning Regulators (STR) and gain scheduling. In MRAC, basically the system is extended to contain the original controller, and an outer con-

trol loop is added in which an adaption mechanism controls the inner controller's parameters. Feedback of this outer control loop is the difference (error) between the output of the actual controlled process and the output of a model of it. This difference is to be adjusted to zero. STR estimates unknown system parameters of the controlled process and adjusts controller parameters accordingly, while gain scheduling implements a static mapping from certain a priori identified process variables to controller parameters.
In Organic Computing, however, not single systems or processes are considered, but collectives of many subsystems, each individually controlled, working together in order to achieve a common goal. The task of the O/C architecture in this is to monitor the whole system from a hierarchically higher point of view, and to ensure that the collaboration of all individually controlled subsystems works effectively. This might, in some situations, even mean decreasing the effectiveness of single subsystems in order to improve performance of the whole system.
Another difference to adaptive control theory is that the O/C architecture explicitly allows for learning. Although also the adaption mechanisms in adaptive control loops can be implemented as self learning, for instance using artificial neural networks [21], the learning mechanism in the O/C architecture is an integral part of the controller. It contains online as well as offline learning mechanisms (see Section 2.2 or [13]) that enable the architecture to adjust even to situations that have been previously unknown or unforeseen.
In this paper, the generic Observer/Controller architecture [15] is adapted to an off-highway machine, thus introducing a hierarchically higher level of control than in traditional machine management systems. In these, optimization usually takes place on a lower level, maximizing the efficiency of the individual components. The Observer/Controller architecture is explicitly intended to regard the system as a whole, optimizing the coordination of the machine's components, and realizing a self-organizing, reliable, adaptive, and robust machine management system.
In this paper, we outline a specific design of the Observer/Controller architecture, in which we consider the off-highway machine and its peripheral machinery components as system under observation and control. Furthermore, we focus on the Observer and analyze the behavior of the off-highway machine during different working cycles. The Observer, as proposed in the generic architecture, is responsible for monitoring and analyzing the data measured in the machine. However, this data is highly noisy as measurements take place during operation. Besides filtering of the datastreams, the major task is to detect the current working situation, driving sequences and cycles from the incoming data. We describe several working cycles in which off-highway machines are used and focus on the workscope of a tractor. The aim is to recognize the specific working cycle the machine is executing and to employ the right parameters via the Controller to the system. These parameter settings depend on the current working cycle and the machine itself. As a first step, we intend to extract features from the datastream that will enable the Observer to recognize the working cycles and to distinguish the different types. We use Principal Component Analysis (PCA) [10, 7] and examine feature extraction in several datastreams of different dimensionalities.
The structure of the paper is as follows. In Section 2, we shortly describe approaches of current machine management systems and then introduce the generic Observer/Controller architecture from Organic Computing. In Section 3, a tractor is defined as part of this architecture. In Section 4, we give a closer look at the structure of the Observer, and in Section 5 at one of its components, the data analyzer. Its purpose is described, and methods to fulfill this purpose are determined. After a description of the experiments that were performed and their results, the paper is completed by a conclusion and an outlook on future work.

2. RELATED WORK

This section gives a brief outline of the existing machine management systems and the concept of the generic Observer/Controller architecture in Organic Computing.

2.1 Present Machine Management Systems

Today, existing machine management systems are designed to optimize single components of the machine following static functions.
Systems optimizing the engine- and gearing-management are the current state of the art. They adjust engine power to keep efficiency at a maximum [19] by reducing shaft speed while driving at low load. Other systems regulate the shaft speed of the engine with respect to the pivoting angle and pressure of the hydraulic pump in order to achieve optimizations like low fuel consumption [5].
Currently, a combination of different optimization approaches is not in effect as each one considers the associated subsystem only, while essential parameters that have to be set pertain to several subsystems or are dependent of each other, which might result in a conflict as soon as different strategies are combined.

2.2 Observer/Controller Architecture

The generic Observer/Controller (O/C) architecture [15] has been proposed to overcome design complexity of technical systems by leaving a considerable degree of freedom for their structure and behavior and by bestowing upon them some "organic" characteristics allowing them to learn and adapt with respect to dynamically changing environments. The architecture is closely related to the MAPE architecture from Autonomic Computing [8], where the data from so called autonomic elements is monitored and analyzed and then an interaction is planned and executed. While Autonomic Computing focusses on large scale enterprise server systems and on data centre management, the O/C architecture adresses technical systems in general and provides both online- and offline-learning methods and an interface for an external user to set global objectives for the control strategy. Thus, it is suitable to control complex systems, make them adaptive to their environment, and at the same time keep them controllable.
The O/C architecture contains three major components: *System under Observation and Control* (SuOC), *Observer*, and *Controller*. The SuOC is the actual technical system that is to be controlled. The Observer is designed to monitor and analyze the SuOC in order to characterize the current system status, predict future states and identify possible emergent behavior that needs to be either interrupted or enforced. To some extent emergent behavior in self-organizing systems can be detected by quantitatively measuring statistical or entropy values of the analyzed data [4]. The task of the Controller is to influence the SuOC such that it exhibits

the desired behavior as specified by the user. Any undesired (emergent) behavior should be disrupted as quickly and efficiently as possible. To enable the Controller to adjust to a changing environment and continually improve its performance, it is equipped with learning capabilities, both online and offline. In online learning, the Controller continually evaluates the impact of its actions in certain situations. In offline learning, it uses an integrated simulation model of the SuOC and a planning component (like an evolutionary algorithm) to test new actions in known situations (more details in [13]). An interesting aspect of the O/C architecture is that the external user (or, the next level entity) can change its preferences on the fly and the system has to be able to adapt its parameters accordingly. To support this, there is a feedback from Controller to Observer by which the Controller can change the Observer's *model of observation*, meaning it can select the parameters of the SuOC that are to be analyzed and influence the algorithms that process the data in the Observer.

In its classical form, the O/C architecture can be employed in a centralized approach, where the SuOC is being observed and controlled by one centralized Observer and Controller. For distributed systems, consisting of complex and largely independent entities, a hierarchically structured approach is designed where each entity has one local O/C architecture possibly supplemented by a global (centralized) Observer and Controller.

The O/C architecture has been applied to several practical applications. In [11, 20], it has been successfully used in traffic control where the SuOC consists of several traffic lights. Usually, in a traffic system undesired effects such as traffic jams or long waiting times at red traffic lights occur. This could be improved considerably using the O/C architecture. In another approach, O/C is applied to avoid the undesired bunching effect that occurs when a large number of elevators work independently of each other in a building [12]. The term bunching effect refers to a synchronized movement of the elevators, that practically converts them into a single big elevator.

In the following, we describe how the O/C architecture can be customized for the requirements of an off-highway machine.

3. TRACTOR AS SYSTEM UNDER OBSERVATION AND CONTROL

Based on the above described O/C architecture, we propose an architecture for observing and controlling off-highway machines. In contrast to other SuOCs to which the generic O/C architecture has already been successfully applied [14, 11, 20], an off-highway machine does not consist of several separate entities. Instead, there are the components of a single machine whose interactions are closely interrelated. In off-highway machines, those components are the traction drive, the power take-off, the hydraulic system, as well as varying auxiliary components. These units are mechanically or fluidically connected or they communicate and interact, e.g., via bus-systems. The input to the SuOC consists of a series of input variables such as fuel, driver interactions and environmental conditions like changing subsoil or varying peripheral components connected to the machine. The main output of the system which can be measured is its fuel consumption. Sensors in the SuOC measure input and output of the system, as well as different parameters of its components, such as torque of rear and front axles, pressures at different locations within the tilt and lift cylinders, deflections of tilt, lift, and steering cylinders, pressures of hydraulic and steerings pumps, shaft speed, engine torque, left and right drive torque, engine speed, or vehicle speed. The possibilities for the controller to interact with the system consist in the setting of reference values for conventional controllers within the machine, or communication with the driver by providing suggestions how to increase efficiency.

Within the tractor, the data collected by the various sensors is transferred to the *Tractor Electronic Control Unit* (TECU), which is the customary central controller, containing substantial know-how of the manufacturer (see Figure 1). To measure system parameters, the signals already present

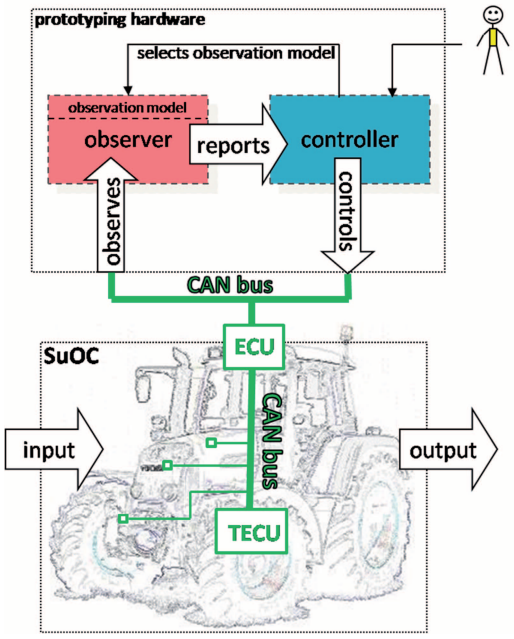


Figure 1: Observer/Controller architecture as a management layer of an off-highway machine

in the system will be routed to an external *electronic control unit* (ECU) that serves as an interface between the tractor and the prototyping hardware which contains the algorithms of the O/C architecture. Further sensors can also be applied throughout the vehicle with their signals being passed on directly to the prototyping hardware.

In an off-highway machine, Observer and Controller of the O/C architecture form a management layer that is hierarchically superior to the already existing TECU (see Figure 1). This TECU stays in charge of the low level control of the vehicle, for which it was intended. This ensures that the tractor as the System under Observation and Control is completely capable of performing its functions independent of the overlying control structures. These are intended to monitor the SuOC as a whole and to interact only if a potential for increasing efficiency is detected. For instance, such interaction could be a reconfiguration of the traction drive or of the hydraulic system.

4. OBSERVER IN OCOM

Figure 2 shows the internal structure of the Observer, adapted to the requirements of an off-highway machine. Just

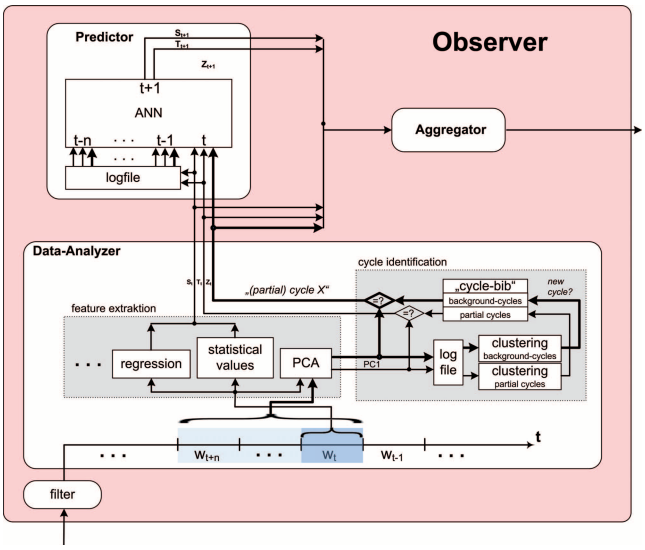


Figure 2: Internal structure of the Observer

as in the generic O/C architecture, it receives the sensor measurements as raw data from the SuOC. After obtaining the data via CAN-Bus and saving it in a log file, the next step is to filter out noise and outliers.
Subsequently, the incoming data is divided into windows w_i and much larger background windows bw_j that are processed individually by the data analyzer module of the Observer. The appropriate size for such windows, as well as the reason for using a parallel approach with large background windows, will be discussed in Section 5. In the feature extraction part of the data analyzer, several values are computed that serve as input for the cycle identification part, as well as for the predictor module. The data analyzer is the major focus of this paper. Its purpose and structure will be described in more detail in Section 5. Basically, the outputs of the data analyzer module represent the current status of the System under Observation and Control at any given time t, computed from the measurement values in windows w_i and bw_j.
The predictor module is intended to predict the system status at time $t+1$. To do this, it takes as input the outputs of the data analyzer for the past n windows w_i and the current background window bw_j. Further research will investigate the most appropriate value for n in order to achieve a satisfying prediction. As for the prediction algorithm, this module will be implemented as an artificial neural network, to enable it to adjust its prediction to the specific operational conditions of every individual machine. By continuous online adaption, the predictor will learn which system state is the most likely to occur next.
All the information that has been gathered in the Observer during feature extraction, cycle identification, and prediction, is collected by the aggregator module of the Observer. From there, it is passed on to the Controller, where it will serve as a basis for all decisions made by the architecture.

In the following section, we provide a closer look at the data analyzer module within the Observer.

5. DATA ANALYZER

Off-highway machines usually perform specific working cycles, which highly influence the efficiency of the machine. In the following, we describe some typical working cycles:

- Y-cycle: In this working cycle, the off-highway machine is basically used to move a load from one location to another, shovel by shovel. The vehicle moves forward in one direction to take up a load, then retreats, and again moves forward into another direction towards the target location to unload, thus following a Y-shaped trajectory.
- Ploughing: Turning over the upper layer of soil to bring up fresh nutrients and burying the remains of previous crops.
- Grubbering: Loosening of soil, without turning it over (as a plough does).
- Harrowing: Breaking up clods of soil after ploughing or grubbering.
- Mowing: Cutting crops.
- Transport: This is the most unspecific working cycle. The machine is used to drive from one location to another, to get to a working site, or to transport goods over a longer distance.

Since those are only some of the many possible working cycles, the machines are usually produced for general purposes and their parameter settings are optimized to accommodate a large variety of possible applications. However, if such sequences could be detected autonomously in an intelligent way, the machine management system could make use of the machinery parts of the system more efficiently and control the system according to the specific situation. Thus, an important task of the Observer is to detect the current working cycle while the machine is operating in order to enable the Controller to choose the right parameters that ensure an efficient use of system components. Appropriate parameter selection depends on the detected working cycle and the specific machine. In the OCOM architecture, cycle identification is implemented in the data analyzer within the Observer.
The first step to identify a working cycle or to recognize any pattern within the data is to extract appropriate features that are characteristic for different cycles.

5.1 Feature Extraction

The data stream from the sensors enters the Observer continually, without interruption. There is no apparent information about where a working cycle starts or where it ends. Moreover, the length of a cycle can vary considerably. Therefore, it is important to identify features which vary from one type of working cycle to another, but stay the same for two cycles of the same type. At the same time, they have to be indifferent to the possible variations that can occur within several executions of the same working cycle. A method is needed that does not exploit the absolute values of the incoming data, but the correlations between

its different dimensions. In other words, we want to focus on the dependencies between the data coming from different sensors.
A method by which this can be achieved is principal component analysis which is briefly explained below.
Principal Component Analysis (PCA) involves a mathematical procedure that transforms a number of possibly correlated variables into a smaller number of uncorrelated variables called principal components. The principal components are ordered by significance, each one containing information about the dimensions, in which the data primarily extends: each principal component is a vector that points into the direction in which the data points are spread the most. Figure 3 gives an impression of this, it illustrates an example of m data points in two-dimensional space. Transferred to an off-highway machine, this could be interpreted as two sensors x_1 and x_2 each measuring a system parameter. The longer vector EV_1 is the most significant principal component that has been computed. It points along the direction where the data points are scattered the most. The other vector EV_2 is next in significance. If the data had more then two dimensions, more principal components would have been computed, respectively.

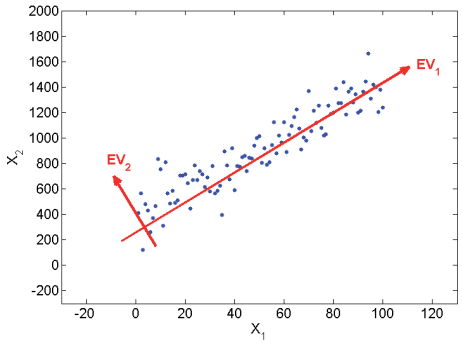


Figure 3: EV_1 and EV_2 illustrate the eigenvectors computed from the covariance matrix of the underlying data.

As known from statistics, two characteristic parameters for the analysis of measurement series are their variance σ^2 and standard deviation σ of a given data set:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{m}(X_i - \overline{X})^2}{(m-1)}},$$

where X_i $(i = 1, \ldots, m)$ are the values in one dimension of the measurement series and $\overline{X}$ is their mean value.
Furthermore, to analyze possible correlations between different dimensions their covariances are computed, i.e. for any two dimensions d_i and d_j one computes

$$cov(d_i, d_j) = \frac{\sum_{i=1}^{m}(X_i - \overline{X})(Y_i - \overline{Y})}{(m-1)},$$

where X_i and Y_i $(i = 1, \ldots, m)$ denote the values in dimensions d_i and d_j of the measurement series and $\overline{X}$ and $\overline{Y}$ are their mean values.
As is well-known, a positive value indicates that the values in both dimensions increase together, while a negative value shows that the two dimensions are negatively correlated. Between dimensions that are independent of each other, the covariance value would be zero.
Thus, for n-dimensional data PCA computes an n-by-n-matrix C where each entry c_{ij} is the covariance between the i-th and the j-th dimension of the data. As $cov(d_i, d_j) = cov(d_j, d_i)$, this is a symmetric matrix.
To determine the main directions along which the data points are distributed, PCA computes the eigenvalues and eigenvectors of C of the data. The resulting eigenvectors constitute a new basis for the data, while the corresponding eigenvalues are related to the variance of the data along these axes. The vector that belongs to the largest eigenvalue is the most significant.

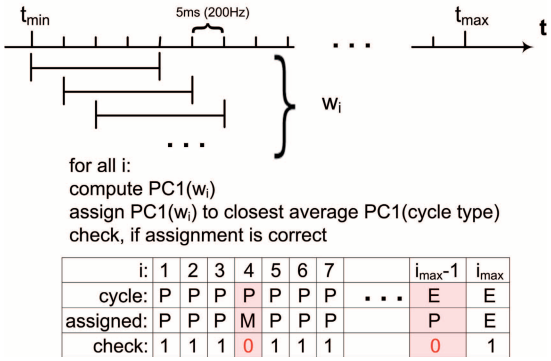
5.2 Experiments

In this section, the experiments that have been conducted are briefly described. The next Section 5.3 presents the results.
The purpose of the experiments is to examine the potential of the data analyzer within the Observer to identify different working cycles using PCA. The cycles to be analyzed are Y-cycle, ploughing, mowing, grubbering and harrowing. Two different datasets are used: The first has been measured at a wheel loader and contains the complete information of 17 sensors for 82 executions of a Y-cycle over approximately 45 minutes. The other one consist of artificially created data of a tractor, describing a potential working day, filled with four different cycle types typical for a tractor: ploughing, mowing, grubbering and harrowing (P, M, G, and H, respectively). It contains 20 consecutive executions of each cycle type. All of them were measured with the same six sensors. For both datasets the measurements have been sampled at a frequency of 200 Hz. The measured parameters in the first dataset (Y-cycle data) contain torque of rear and front axles, pressures and deflections of different cylinders, pressures of different pumps, shaft speed, tractive power, and vehicle speed. The parameters in the other dataset (tractor working day) consist of vehicle speed, tractive power, rear and front torque, speed of rotation motor as well as volume flow rate of the steering cylinder.
Results from the analysis of the first data set can not be compared directly to results of the second one, because they apply to a different kind of machine and the sensors used to collect the data do not fully correspond. Therefore, the first dataset will only be used to determine whether the extracted features are consistent throughout several instances of Y-cycles.
The aim in this first step is to determine whether PCA is suitable to distinguish between different types of working cycles, while different instances of the same cycle should result in identical features.
For this purpose, all datasets are decomposed into smaller subsets each containing the data of an individual instance of a working cycle. PCA is performed on each subset by computing the covariance matrix C_{subset} and identifying its respective eigenvalues and eigenvectors. The most important eigenvector of each subset (its most significant principal component $PC1(subset)$) is used as the extracted feature.
To examine if the first principal component of one cycle is characteristic for its cycle type, all $PC1$ of cycles of the same type are compared with each other to check whether they are alike.

After that it is analyzed whether different cycle types can be held apart. For this, the $PC1$ of all types must be different, so subsequently, for each cycle type the average $PC1$ is compared to that of all other types, to observe if they can be distinguished.

Although we are aware of the fact that, in practice, it will hardly be possible to isolate the data of exactly one working cycle in order to perform PCA on it, positive results to these experiments would show that PCA is, in principle, capable of distinguishing between different types of working cycles.

The next step is to examine whether these results can be used to identify working cycles also online, during operation and without having to "cut out" a single cycle from beginning to end. To do this, a sliding window is dragged over the measurement data of the second data set, the tractor working day with four different cycle types. For each window w_i the first principal component $PC1(w_i)$ is computed and then compared to the average principal components of the four cycle types ploughing, mowing, grubbering and harrowing. It is assigned to the one that is the closest. Figure 4 illustrates this. To determine how the length of the window



i:	1	2	3	4	5	6	7		i_{max}-1	i_{max}
cycle:	P	P	P	P	P	P	P	. . .	E	E
assigned:	P	P	P	M	P	P	P		P	E
check:	1	1	1	0	1	1	1		0	1

Figure 4: Assignment of $PC1(w_i)$ to average $PC1$ of cycle types

w_i influences the result of this assignment, it is repeated for several window lengths.

5.3 Results

In this section, Y, P, M, G and H indicate Y-cycle, ploughing, mowing, grubbering and harrowing, respectively. Table 1 shows the average angle between the first principal component $PC1$ of each subset and that of all other subsets of the same kind of working cycle (for 82 Y-cycles, this are 3321 comparisons, for the 20 executions of the other cycle types 190 comparisons). Median and standard deviation σ are also shown. As comparison is done only within one cycle type, results of data set one (Y-cycle measurements) are included. As expected, all the principal components point into the same direction and the average angle between them is close to zero with very low standard deviation. The only exception is mowing (M). This is due to the fact that approximately half of the vectors for this cycle type point into directly opposite directions. The values of median and standard deviation indicate that these are the only two major occurring directions. As we are only interested in the dimensions along which the data is distributed the farthest, it can be ignored in which direction along these dimensions the principal component vector is pointing.

Table 1: Average angle, median and standard deviation σ between the first principal component of all subsets of Y-cycle (Y), ploughing (P), mowing (M), grubbering (G) and harrowing (H) (compared within one type of working cycle)

	avg	*median*	σ
Y	0.2572°	0.2151°	0.1520°
P	0.0371°	0.0324°	0.0188°
M	90.9370°	179.9206°	90.1988°
G	0.0399°	0.0391°	0.0138°
H	0.0228°	0.0197°	0.0128°

These results show that PCA is able to characterize one type of working cycle by means of the first principal component of the high dimensional measurement data.

Now the average principal components of different working cycles are compared with each other. Table 2 shows the angle between the average first principal components of two different kinds of working cycles, each computed over the 20 executions of each cycle type in data set one. In this comparison of different kinds, only the tractor data from dataset two is considered, with its four different cycle types (P, M, G, H). All computed angles between different cycles are by far larger than the standard deviation of angles within one type of cycle (see Table 1).

Table 2: Angle between the average first principal components of two different kinds of working cycles

	P	M	G	H
P	0°	87.0468°	52.8273°	22.2571°
M	87.0468°	0°	90.8125°	85.5550°
G	52.8273°	90.8125°	0°	74.8631°
H	22.2571°	85.5550°	74.8631°	0°

This indicates that, in principle, PCA can distinguishing between different types of working cycles.

The next step is to examine whether these results can be used to identify working cycles also online, during operation. To do this, a sliding window is dragged over the measurement data of data set two (working day of a tractor), and for each window w_i the first principal component $PC1(w_i)$ is computed. This is then compared to the average principal components of the four cycle types ploughing, mowing, grubbering and harrowing, and assigned to the one that is closest (cf. Figure 4). Finally, the percentage of correctly assigned windows for each window size is determined. The computation has been repeated for several different window sizes, from 1000 samples to 30000 samples per window in steps of 1000 (the length of the full cycles is 30000 in case of ploughing, grubbering and harrowing, and 28000 in case of mowing). Figure 5 shows the results of this computation. For each window size, the y-axis displays the percentage of windows over the whole day that have been assigned correctly to the average $PC1$ of the cycle type that has actually just been executed. It is evident, that the hit-rate rises up to 100% as the window size approaches the cycle length. What can also be seen, is that satisfying results will only be achieved using very large windows. For instance, a hit-rate of 95% requires a window size of 25000 samples, which is, at a sample rate of 200 Hz a time span of 125 sec-

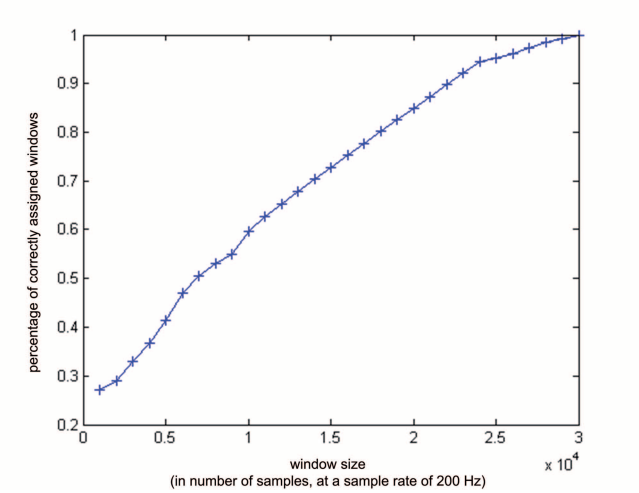


Figure 5: Hit-rate of assignment of windows to cycle types

onds. For the Observer, that would mean that situation parameters can only be computed every two minutes, which is too long. Therefore, the data analyzer within the Observer has been designed with two kinds of windows, that are processed in parallel (cf. Figure 2). Situation parameters that are derived of a small window w_i at time t enter the prediction module directly, and the principal components $PC1(w_i)$ and $PC1(bw_j)$ of a larger background window go into a hierarchical cycle identification: bw_j will provide information about the background cycle, while w_i determines the currently executed partial cycle within it. Both results go into the predictor, as additional information in order to improve prediction quality.

6. CONCLUSION AND FUTURE WORK

In this paper, we proposed an architecture for a machine management system of an off-highway machine that is based on the generic Observer/Controller (O/C) architecture. It exceeds traditional systems in two major points: On the one hand, while existing systems optimize the machine specifically towards efficient performance in its intended tasks, our architecture contains machine learning mechanisms and is thus capable of adjusting the machine to changing environments and situations. On the other hand, optimizations in off-highway machines are often done at a low level of abstraction, maximizing the efficiency of single components. The architecture described here introduces a hierarchically superior management layer that regards the machine and the interactions of its components as a whole.

After describing the architecture of our intended O/C based machine management system, we took a closer look at the Observer in this architecture and specifically the data analyzer module within it. We explained how it can be designed to fulfill its major task of working cycle recognition, and outlined our corresponding experiments: we tested the ability of principal component analysis to extract features from the measurement data that are characteristic for each type of working cycle, while at the same time they are indifferent to the variations that might occur from one execution of that working cycle to the next. The proposed method was adjusted to online cycle recognition during operation, by applying it to a sliding window over the incoming data stream. Future work will include an analysis of a suitable prediction algorithm for the corresponding module within the Observer and, subsequently, the appropriate design of the O/C Controller to the requirements of an off-highway machine.

6.0.1 Acknowledgment

This work has been supported by the German research foundation (DFG), which supports the project OCOM - Organic Computing in Off-highway Machines.

7. REFERENCES

[1] K. J. Astrom. Theory and Applications of Adaptive Control - A Survey. *Automatica*, 19(5):471–486, 1983.

[2] S. Boettinger and A. Stoll. Informations- und Regelsysteme an Maehdreschern und Feldhaeckslern. *Landtechnik 60*, (2/2005):86–87, 2005.

[3] J. Branke, M. Mnif, C. Mueller-Schloer, H. Prothmann, U. Richter, F. Rochner, and H. Schmeck. Organic computing - addressing complexity by controlled self-organization. In *Post-Conference Proceedings of the 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA 2006)*, pages 185–191. IEEE, November 2006.

[4] E. Cakar, M. Mnif, C. Mueller-Schloer, U. Richter, and H. Schmeck. Towards a quantitative notion of self-organisation. In *Proceedings of the 2007 IEEE Congress on Evolutionary Computation (CEC 2007)*, pages 4222–4229, September 2007.

[5] J. Forche. Management hydraulischer Antriebe in mobilen Arbeitsmaschinen. *Landtechnik*, pages 239–244, 2003.

[6] J. Forche. *Antriebsstrangmanagement eines Hydraulikbaggers*. Shaker, 2007.

[7] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2003.

[8] J. O. Kephart and D. M. Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, 2003.

[9] H. Mariutti. *Lastkollektive fuer die Fahrantriebe von Traktoren mit Bandlaufwerken (Fortschritt-Berichte VDI)*. VDI Verlag, 2003.

[10] S. Marsland. *Machine Learning - An Algorithmic Perspective*. Chapman & Hall, 2009.

[11] H. Prothmann, F. Rochner, S. Tomforde, J. Branke, C. Mueller-Schloer, and H. Schmeck. Organic Control of Traffic Lights. In Chunming Rong, Martin Gilje Jaatun, Frode Eika Sandnes, Laurence T. Yang, and Jianhua Ma, editors, *Proceedings of the 5th International Conference on Autonomic and Trusted Computing (ATC-08)*, volume 5060 of *LNCS*, pages 219–233. Springer, 2008.

[12] O. Ribock, U. Richter, and H. Schmeck. Using organic computing to control bunching effects. In *Proceedings of the 21th International Conference on Architecture of Computing Systems (ARCS 2008)*, volume 4934 of *LNCS*, pages 232–244. Springer, Februar 2008.

[13] U. Richter. *Controlled Self-Organisation Using Learning Classifier Systems*. PhD thesis, PhD thesis

at the Universitaet Karlsruhe (TH), Fakultaet fuer Wirtschaftswissenschaften, 2009.

[14] U. Richter and M. Mnif. Learning to Control the Emergent Behaviour of a Multi-agent System. In Franziska Kluegl, Karl Tuyls, and Sandip Sen, editors, *Proceedings of the 2008 Workshop on Adaptive Learning Agents and Multi-Agent Systems at AAMAS 2008 (ALAMAS+ALAg 2008)*, pages 33–40, 2008.

[15] U. Richter, M. Mnif, J. Branke, C. Mueller-Schloer, and H. Schmeck. Towards a generic observer/controller architecture for Organic Computing. In Christian Hochberger and Ruediger Liskowsky, editors, *INFORMATIK 2006 - Informatik fuer Menschen!*, volume P-93 of *LNI*, pages 112–119. Bonner Koellen Verlag, 2006.

[16] H. Schmeck. Organic Computing. *Kuenstliche Intelligenz*, (3/05):68–69, 2005.

[17] H. Schmeck. Organic Computing - A New Vision for Distributed Embedded Systems. In *Proceedings Eighth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC 2005)*, pages 201–203. IEEE, IEEE Computer Society, 2005.

[18] D. E. Seborg, T. F. Edgar, and S. L. Shah. Adaptive Control Strategies for Process Control: A Survey. *AIChE Journal*, 32(6):881–913, 1986.

[19] J. Seeger. Untersuchungen von Antriebsstrangstrategien eines TMS. *Landtechnik*, pages 93–98, 1999.

[20] S. Tomforde, H. Prothmann, F. Rochner, J. Branke, J. Haehner, C. Mueller-Schloer, and H. Schmeck. Decentralised Progressive Signal Systems for Organic Traffic Control. In Sven Brueckner, Paul Robertson, and Umesh Bellur, editors, *Proceedings of the 2nd IEEE International Conference on Self-Adaption and Self-Organization (SASO 2008)*, pages 413–422. IEEE, 2008.

[21] H. Wang, C. J. Harris, and M. Brown. *Advanced Adaptive Control*. Elsevier Science Inc., 1995.