Closed-loop Control System for the Reliability of Intelligent Mechatronic Systems

Tobias Meyer[1] and Walter Sextro[2]

[1,2] *University of Paderborn, Paderborn, Germany*
tobias.meyer@uni-paderborn.de
walter.sextro@uni-paderborn.de

ABSTRACT

So-called reliability adaptive systems are able to adapt their system behavior based on the current reliability of the system. This allows them to react to changed operating conditions or faults within the system that change the degradation behavior. To implement such reliability adaptation, self-optimization can be used. A self-optimizing system pursues objectives, of which the priorities can be changed at runtime, in turn changing the system behavior.

When including system reliability as an objective of the system, it becomes possible to change the system based on the current reliability as well. This capability can be used to control the reliability of the system throughout its operation period in order to achieve a pre-defined or user-selectable system lifetime. This way, optimal planning of maintenance intervals is possible while also using the system capabilities to their full extent.

Our proposed control system makes it possible to react to changed degradation behavior by selecting objectives of the self-optimizing system and in turn changing the operating parameters in a closed loop. A two-stage controller is designed which is used to select the currently required priorities of the objectives in order to fulfill the desired usable lifetime.

Investigations using a model of an automotive clutch system serve to demonstrate the feasibility of our controller. It is shown that the desired lifetime can be achieved reliably.

1. INTRODUCTION

Self-optimizing mechatronic systems are a class of intelligent technical systems that are able to autonomously adapt their behavior if user requirements or operating conditions change (Gausemeier, Rammig, Schäfer, & Sextro, 2014). To this end, the current situation is monitored and the objectives of the system are determined. Using model based multiobjective optimization, for which a model of the dynamical behavior of the system is used, optimal system configurations are calculated before operation of the system. To adapt the system behavior during operation, the self-optimizing system selects among these optimal system configurations.

In order to use self-optimization to ensure that the requirements regarding reliability of the system are met, a suitable selection process has to be implemented. To adapt the system behavior advantageously with regard to system reliability, it has to be possible to lower work load or wear on critical components by selecting appropriate optimal system configurations. Thus it is also necessary to include system degradation in the objective functions used for the multiobjective optimization.

To control the remaining useful lifetime, the whole system history has to be taken into account as well. This could not be achieved by directly including remaining useful lifetime in the model used for multiobjective optimization, as then each objective function evaluation would require a simulation of the whole system lifetime. Such a simulation requires a lot of computing effort, rendering this approach impossible. Thus a process to take the system history into account separately during operation is required. For this, our presented self-optimization based remaining useful lifetime controller can be used.

2. MAINTENANCE PLANNING

The big advantage of actively controlling the reliability of a system becomes apparent if the whole life-cycle including maintenance is considered. Within the scope of this section, it is assumed that after maintenance, a system is as-good-as-new. Traditionally, maintenance was conducted as either corrective of preventive maintenance (Birolini, 2007). In corrective maintenance, system functionality is reestablished once a failure occurs. This strategy is cheap at first, but once a failure occurs and the system is unavailable, maintenance has to be conducted as soon as possible, making the repair expensive. It

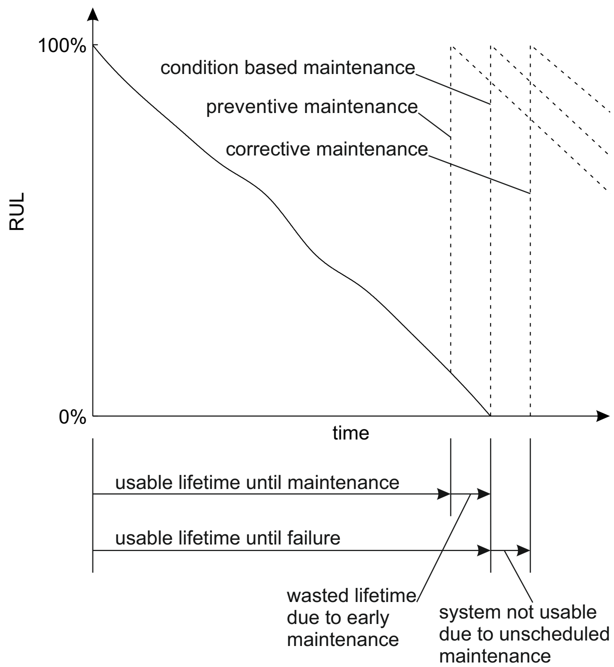

Figure 1. Maintenance planning techniques and their effect on usable lifetime.

also comes with the risk of catastrophic failures which make it unsuitable for many systems. This approach maximizes the usable lifetime, as can be seen in Fig. 1. Availability, however, is limited due to unnecessarily long unscheduled maintenance.

Preventive maintenance, on the other hand, allows a high availability of the system by retaining system functionality. This is achieved by conducting maintenance before a failure occurs, making the maintenance schedulable and thus highly efficient. Usually, suitable maintenance intervals are determined using stochastic models for large fleets of systems (Joo, Levary, & Ferris, 1997). This approach has the advantage of achieving high availability with planned maintenance intervals, but usable lifetime until maintenance is lower than usable lifetime until failure. This increases the cost of operation due to earlier maintenance than necessary. Also it is best suited for large fleets of identical systems and can hardly be implemented for unique machinery.

In order to overcome these drawbacks, condition based maintenance can be used. According to (Jardine, Lin, & Banjevic, 2006), a condition based maintenance program consists of three steps: Data acquisition, data processing and maintenance decision making. In the first two steps, the current state of the system is assessed. After evaluation, efficient maintenance policies are recommended. A condition based maintenance program is comprised of two important aspects: Diagnostics and prognostics. In diagnostics, existing faults are detected, isolated and identified before they lead to a failure. Prognostics, on the other hand, deals with the prediction of future faults. The main objective is to estimate the time until a fault occurs or the probability of it occuring. Using this information, the system can be operated without wasting usable lifetime for overly cautious maintenance intervals and also without requiring unscheduled maintenance. While this is advantageous over corrective and preventive maintenance, it remains a reactive method in which the system degradation drives the scheduling of maintenance operations and which makes planning of inspection and maintenance complex (Chena & Trivedi, 2005).

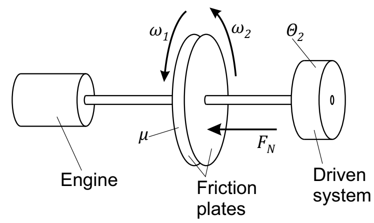

Figure 2. Basic structure of clutch system.

By combining information about the current system reliability with a feedback to system operation, it becomes possible to adjust system behavior according to its current reliability. This allows reversal of the usual approach. It now becomes possible to schedule maintenance operations with the system adapting its behavior and its degradation accordingly. The proposed closed loop control allows for such operation.

3. APPLICATION EXAMPLE

A single plate dry clutch has already been introduced as application example in (Meyer, Sondermann-Wölke, Kimotho, & Sextro, 2013) and is used again in this contribution. This type of clutch is commonly utilized in passenger vehicles to connect an internal combustion engine to the drivetrain. The basic outline of the clutch system is shown in Fig. 2. It consists of two friction plates with coefficient of friction μ, of which the input plate is connected to the engine while the output plate is connected to the driven system, e.g. a gearbox. The input and the output plates are rotating at speeds $\omega_1 = 1\frac{\mathrm{rad}}{\mathrm{s}}$ and ω_2 respectively. To engage the clutch, both plates are pressed against each other by the force F_N, thus transmitting torque T_f from the input plate to the output plate and in turn applying this torque to the driven system.

The system dynamics can be modelled with

$$T_f(t) = F_N(t) \cdot \mu(\Delta\omega(t)) \cdot r_{\mathit{eff}}, \tag{1}$$

$$\dot{\omega}_2(t) = \frac{1}{\Theta_2} \cdot (T_f(t) - d_2 \cdot \omega_2(t)), \tag{2}$$

$$\mu(\Delta\omega) = \mu_0 \cdot \frac{2}{\pi} \cdot \arctan\left(\frac{\Delta\omega}{\hat{\omega}}\right) \tag{3}$$

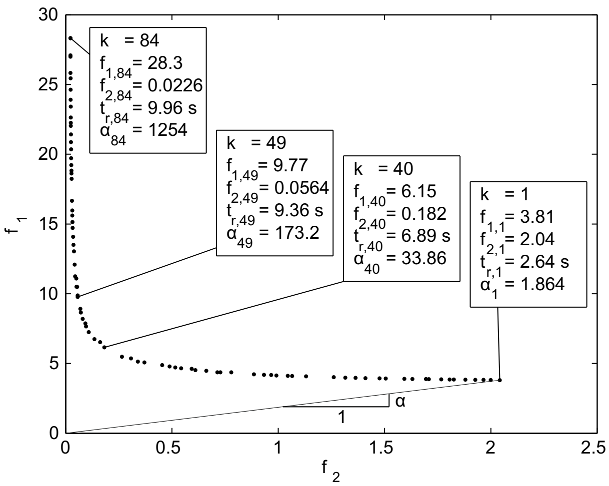

Figure 3. Pareto front of clutch system with two objective functions: f_1: minimize wear and f_2: Minimize accelerations, i.e. maximize comfort. Note that the duration of an actuation cycle t_r has a great effect on both objectives.

where $\mu_0 = 1$ is the nominal coefficient of friction, $\Delta\omega = \omega_2 - \omega_1$ is the difference in revolutionary speed of the plates, $\hat{\omega} = 0.1\frac{\mathrm{rad}}{\mathrm{s}}$ is the accuracy parameter, $r_{eff} = 1\,\mathrm{m}$ is the effective radius of the plates, $\Theta_2 = 1\frac{\mathrm{kg}}{\mathrm{m}^2}$ is the moment of inertia of driven system, $d_2 = 1\frac{\mathrm{N \cdot m \cdot s}}{\mathrm{rad}}$ is the damping factor of the driven system. Arbitrary values, which do not model a particular system, were chosen to demonstrate the proposed control method.

Also in (Meyer et al., 2013) it was shown that using multi-objective optimization techniques, a control trajectory for the actuation force $F_N(t)$ can be computed to actuate the clutch system. Multiobjective optimization techniques attempt to minimize user defined objective functions by adapting system parameters. Typically, it is not possible to minimize multiple objective functions at once, but instead as one objective function value is lowered, another objective function value rises. This leads to the so-called Pareto front, which consists of all optimal compromises between multiple objective functions. To each point on the Pareto front, system parameters are given in the Pareto set. To compute Pareto front and Pareto set, a genetic algorithm which comes with the Matlab global optimization toolbox has been used.

The required objective functions are included in a full model of the system dynamics. For our system, the objective functions are f_1, which represents the power loss in the clutch P_f and in turn corresponds to the wear rate of the clutch plates, and f_2, which represents e.g. comfort of vehicle passengers:

$$f_1 = \int_{t_0}^{t_0+t_r} (P_f(t))^2 \,\mathrm{d}t = \int_{t_0}^{t_0+t_r} (T_F(t) \cdot \Delta\omega(t))^2 \,\mathrm{d}t,$$
$$f_2 = \int_{t_0}^{t_0+t_r} (\dot{\omega}_2(t))^2 \,\mathrm{d}t.$$

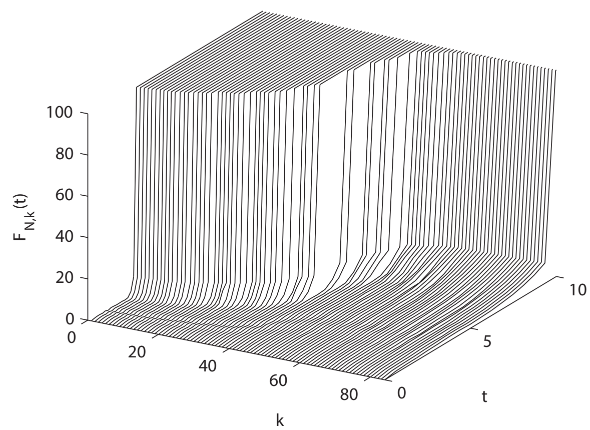

Figure 4. Pareto set of clutch system with 84 possible actuation trajectories $F_N(t)$.

To compute the values of these objective functions, the dynamical model of the system is simulated over the period $t = t_0 \ldots t_0 + t_r$ using trajectories for $F_N(t)$ as simulation input.

The duration of the actuation cycle and the shape of the trajectory are the optimization parameters. To include these in the optimization procedure, the trajectory was subdivided into 16 sections with equal durations. For the trajectory to begin with a completely disengaged clutch and end with a completely engaged clutch, $F_N(t_0) = 0\,\mathrm{N}$ and $F_N(t_0 + t_r) = 100\,\mathrm{N}$ are assumed. The optimization parameters are then the total duration of the actuation cycle t_r and the shape computed by using 15 intermediate values $F_N\left(t_0 + \frac{1}{16} \cdot i \cdot t_r\right)$, $i = 1 \ldots 15$. Linear interpolation is used between these values. This way, the Pareto front shown in Fig. 3 with the corresponding Pareto set shown in Fig. 4 is obtained. A short total duration of the actuation cycle yields low energy losses but high accelerations, as opposed to a long duration, which yields inverse results. Each trajectory is a trade-off between these two objectives.

4. CONTROLLING THE RELIABILITY

In prior works (Meyer et al., 2013), a basic controller for the reliability of the clutch system was presented. However, the approach outlined therein was limited in its effectiveness since it did not take the inherent non-linearities and deviations between multiobjective optimization model and real system into account. It was not capable of handling deviations that required a great change from the nominal working point. The approach presented in the remainder of this contribution overcomes these drawbacks and offers better generalizability to other engineering problems.

A two-stage controller design has been favored for the possiblity to be designed separately for high-frequency perturba-

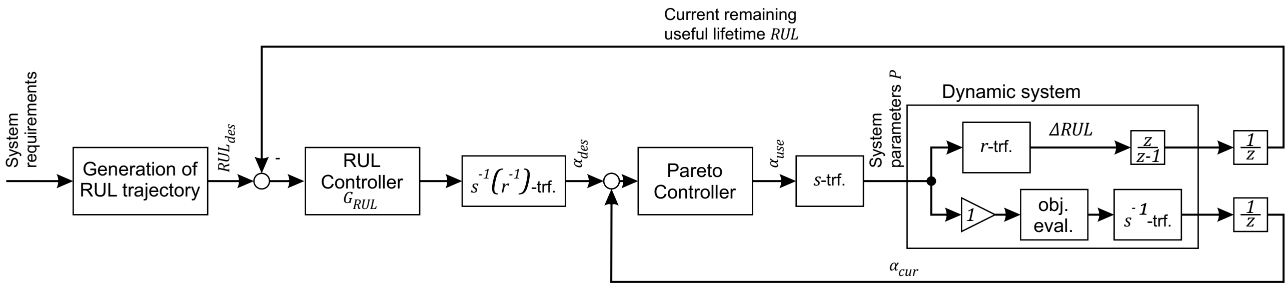

Figure 5. Full two-stage control loop.

tions on the inner loop and for low-frequency perturbations on the outer loop.

While priorities of objectives of a self-optimizing system may be selected arbitrarily, the system behavior does not necessarily reflect this immediately. On the one hand, an adaptation usually takes some time to take full effect; on the other hand the system model used for multiobjective optimization and the real system might deviate from one another, thus if a working point is chosen based an pre-calculated optimal system configurations, which are based on the system model, the actual system might behave differently. This leads to differences between desired objectives and achieved objectives.

To overcome these shortcomings, Krüger et al. developed a closed loop control for the objectives of a self-optimizing system (see (Krüger, Remirez, Kessler, & Trächtler, 2013)), colloquially called "Pareto controller". The purpose of this controller is to ascertain a pre-selected system configuration is actually being used, despite of perturbations or deviations between optimization model and actual system. To this end, the desired system configuration is selected with a so-called α-parameterization, which can be defined individually for each system. Suggestions are made, e.g. to use a Simplex-based method or to calculate the ratio among two objectives. In the course of this paper, we define the α-parameterization as follows, it is also included in Fig. 3:

$$\alpha = \frac{f_1}{f_2}.$$

The desired parameterization value α_{des} is used as controller input. The current value of the α-parameterization, α_{cur} is required for the controller to calculate the used value α_{used} according to the difference between α_{cur} and α_{des}. Once α_{used} has been calculated, the parameters of the system are determined by the so-called s-transform and set in the system. After *a certain time*, the resulting system behavior is evaluated to determine the current value α_{cur} of the α-parameterization.

It is assumed that the behavior adaptation and evaluation of the actual system behavior takes some time to take full effect. For this reason, the Pareto controller works in discrete time on a *slow* time scale, where one discrete time step is the constant time period required for the full behavior adaptation and evaluation process. For this reason, in the abstract model of the system, the output is delayed by the unit delay $\frac{1}{z}$.

This Pareto controller is used as inner loop of the full control loop. It is not able to take the full lifetime information into account and serves the purpose of reliably achieving the desired system behavior.

The outer loop, on the other hand, is responsible for controlling the remaining useful lifetime. For this, an abstract model of the system adaptation process is required. As the inner loop already controls the desired behavior, the outer loop does not need to take actual system parameters into account but instead relies on using the α-parameterization as system input. System output and controlled variable is the remaining useful lifetime RUL. The reference input is denoted by RUL_{des}. However, the relationship between α and RUL is highly nonlinear. The difference in remaining useful lifetime $\Delta RUL\left(\alpha\right)$ over a single actuation cycle i can, however, be approximated using the system model. This is called the r-transform:

$$\Delta RUL\left(\alpha\right) = r\left(s\left(\alpha\right)\right).$$

To obtain the current remaining useful lifetime, an integral element $\frac{z}{z-1}$ in the dynamic system, and a unit delay $\frac{1}{z}$ for the evaluation of the current remaining useful lifetime are added, as shown in Fig. 5.

As controller for the remaining useful lifetime, a P controller was chosen. An integral element is not required to correct for steady state errors due to the integrating properties of the wear process. It calculates the r-transformed desired α-parameterization $r\left(s\left(\alpha_{des}\right)\right)$ according to:

$$G_{RUL} = \frac{r\left(s\left(\alpha_{des}\right)\right)}{RUL_{des} - RUL} = K_{p,RUL} \tag{4}$$

This discrete controller can be implemented in the same discrete time used for the Pareto controller. The controller output is then converted by the inverse r-transform $s^{-1}\left(r^{-1}\right)$ to give α_{des}.

The reference input generated for the RUL-controller needs to be strictly monotonically decreasing. If it was not, an actuation cycle with no or even negative wear would be required, which is physically impossible. The chosen reference input begins with $RUL_{des}\,(new\ system) = 100\%$ and ends with $RUL_{des}\,(end\ of\ specified\ lifetime) = 0\%$. Linear interpolation is chosen for intermediate cycles. The reference input can be altered during operation in case of changed requirements. An adaptation of system behavior is then conducted by means of the control loop.

5. SETUP OF THE CONTROLLER FOR THE APPLICATION EXAMPLE

When controlling the remaining useful lifetime, the system behavior is adapted by changing system objectives. However, it needs to be ascertained that these objectives are met. As was mentioned in the basic introduction of the control loop in section 4, a specifically designed closed loop control by Krüger et al. (Krüger et al., 2013) is used. This control loop as well as the RUL-controller that builds on it are working in discrete time, their stepsize is *big* compared to the system dynamics. Since the clutch system has discrete events, one step corresponds to one full actuation cycle. Due to this, the stepsize of both controllers is $1\,\mathrm{cycle}$.

5.1. Pareto controller

The basic idea of the closed loop Pareto controller is to define the desired system behavior using a so-called α-parameterization. The value of this parameterization is used as reference input α_{des} for the controller. The actual current value α_{cur} is computed from signals or measured variables of the system.

At first, the α-parameterization needs to be defined. For the clutch system, which pursues two objectives only, the fraction of both objective values is used, i.e. $\alpha = \frac{f_1}{f_2}$. This approach has several advantages over more complex parameterizations, e.g. the Simplex-based parameterization suggested in (Krüger et al., 2013). First, it is very simple to calculate, thus requiring low computational time. Second, and more importantly, no knowledge about the Pareto front, such as an approximating function, number of known points or values at the edges, is required. This makes evaluating the currently achieved value α_{cur} independent of any assumptions about other possible working points.

The α-value needs to be transformed into a set of parameters to be used by the actual system. For this, the s-transform is used. It determines the desired Pareto point from the Pareto front and selects the Pareto set, which contains all system parameters, accordingly. For the clutch system, linear interpolation between pre-calculated Pareto points is used. This is done in three steps: At first, the two Pareto points closest to the desired α-parameterization α_{des} are searched. In the next step, the two sets of parameters are selected from the Pareto set P_{set}. Last, linear interpolation is used for each pair of parameters to obtain the final parameter.

To determine the closest Pareto point, the α-paramterization value for each pre-calculated Pareto point is calculated. For this, $k = 1 \ldots n, k \in \mathbb{N}, n \in \mathbb{N}$ Pareto points are assumed:

$$\alpha_k = \frac{f_{1,k}}{f_{2,k}}.$$

The following two steps are conducted at runtime. At first, k closest to the currently desired value α_{des} is searched:

$$\min_k \left(|\alpha_k - \alpha_{des}|\right).$$

Once this is known, linear interpolation among two points with α-values closest to the desired value α_{des} is conducted to find system parameters W from the Pareto set P_{set}:

$$P\left(\alpha_{des}\right) = \begin{cases} \frac{\left(P_{set,k}+P_{set,k+1}\right)\cdot\alpha_{des}}{\alpha_k+\alpha_{k+1}}, & \text{if } 1 < k < n \text{ and } |\alpha_k - \alpha_{k-1}| < |\alpha_k - \alpha_{k+1}|, \\ \frac{\left(P_{set,k-1}+P_{set,k}\right)\cdot\alpha_{des}}{\alpha_{k-1}+\alpha_k}, & \text{if } 1 < k < n \text{ and } |\alpha_k - \alpha_{k-1}| > |\alpha_k - \alpha_{k+1}|, \\ P_{set,k}, & \text{else.} \end{cases}$$

The advantage of this approach is that even though a limited number of Pareto points is known from numerical multiobjective optimization, a *close* approximation for intermediate values can be found. This is important in subsequent steps, since all controllers developed herein have continuous output values and expect the system, i.e. s-transform, clutch system, objective evaluation, and s^{-1}-transform, to accept such and work continuously as well.

With linearly interpolating between Pareto points, it is assumed that all computed possible solutions P_{set} to the optimization problem are *similar*. Proof that this assumption holds is difficult, but clear indications can be seen in Fig. 4.

The s^{-1}-transform, on the other hand, is very simple once current values of the system objectives $f_{1,cur}$ and $f_{2,cur}$ are determined by evaluating measured variables or signals from the system:

$$\alpha_{cur} = \frac{f_{1,cur}}{f_{2,cur}}.$$

With these transformations all set, the actual controller can be parameterized. It was created according to (Krüger et al., 2013) without modifications. The controller parameters were chosen as $K_p = 0.05$ and $K_i = 0.05$.

The controller reference input is the desired α-parameterization α_{des}. It is set by the outer loop which controls the remaining useful lifetime of the system and induces a behavior adaptatation by changing α_{des}.

5.2. RUL controller

The purpose of the outer control loop is to determine the currently required desired α-parameterization α_{des} from the desired remaining useful lifetime RUL_{des} and the current remaining useful lifetime RUL.

At first, the remaining useful lifetime RUL needs to be determined. This is highly application-specific. A model-based approach has been selected to estimate the remaining useful lifetime of the friction plates. It is based on the assumption that clutch plate wear is proportional to friction energy E_f (Fleischer, 1973). For each actuation cycle i with time span $t = t_{0,i} \dots t_{0,i} + t_r$, where t_r is the duration of the actuation cycle, the wear $w\left(i\right)$ occuring during this cycle is:

$$\begin{aligned} w\left(i\right) &= p_f \cdot \Delta E_f\left(i\right) = p_f \cdot \int_{t_{0,i}}^{t_{0,i}+t_r} P_f\left(t\right) \mathrm{d}\, t \\ &= p_f \cdot \int_{t_{0,i}}^{t_{0,i}+t_r} T_F\left(t\right) \cdot \Delta\omega\left(t\right) \mathrm{d}\, t. \end{aligned} \tag{5}$$

The proportionality factor is assumed to be $p_f = 1$ for normal wear behavior. Due to e.g. errors in manufacturing or materials, it might deviate, thus requiring a changed operating point in order to fulfill the specified lifetime.

To estimate the remaining useful lifetime, all actuation cycles need to be taken into account. To do so, the sum of the wear occuring in each cycle $w\left(i\right)$ is summed over all prior m cycles, i.e. $i = 1 \dots m, i \in \mathbb{N}, m \in \mathbb{N}$. The remaining useful lifetime RUL for the next cycle $m+1$ can then be estimated by taking the maximum amount of wear w_{max} of the clutch into account. This results in the following relation:

$$RUL\left(m+1\right) = 1 - \left(\frac{\sum_{i=1}^{m} w\left(i\right)}{w_{max}}\right). \tag{6}$$

To convert the RUL-controller output to a desired value of the α-parameterization, the inverse r-transform $s^{-1}\left(r^{-1}\right)$ needs to be defined next. Since $\Delta RUL\left(\alpha\right)$ can not easily be computed analytically, the main cause of wear needs to be determined. As was shown in eqns. 5 and 6, the remaining useful lifetime mainly depends on the friction energy ΔE_f, thus $\Delta RUL\left(\alpha\right) \sim \Delta E_f$.

To setup the inverse r-transform, the friction energy for each pre-calculated α-parameterization $\Delta E_f\left(\alpha\right)$ is computed by simulating the system model for one full clutch cycle. The resulting relationship between α-parameterization and ΔE_f is shown in Fig. 6.

Using a least-squares approach, an approximating function was fitted to obtain a computationally effective $s^{-1}\left(r^{-1}\right)$-transform. An exponential ansatz was chosen:

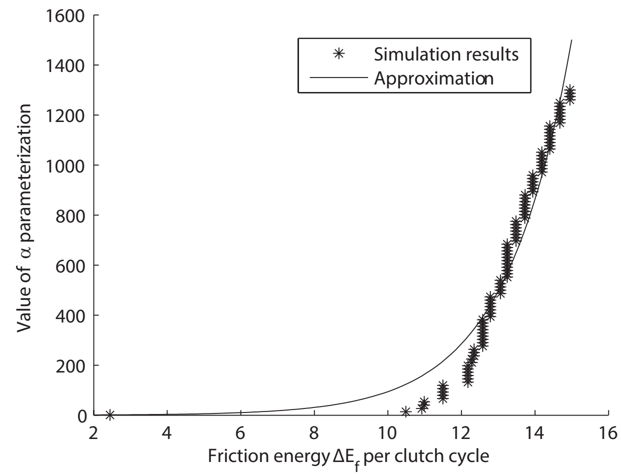

Figure 6. Compensation of nonlinear behavior.

$$\alpha_{approx} = q_1 \cdot e^{q_2 \cdot \Delta E_f}.$$

The parameterized approximating function ($q_1 = 0.3714$, $q_2 = 0.5536$) is also shown in Fig. 6.

The objective-based controller for the remaining useful lifetime is implemented according to eq. 4. As proportional gain parameter, $K_{p,RUL} = 1000$ was chosen.

6. SIMULATION RESULTS

To evaluate the feasibility of the proposed approach, simulations that span the whole lifetime of the clutch system were conducted. For this, a model of the dynamic behavior of the clutch system according to eqns. 1, 2 and 3 was used.

An artifical fault was introduced into the system model: After 200 regular clutch cycles, the wear proportionality factor was changed from $p_f = 1$ to $p_f = 2$. This way, the simulated wearing process was accelerated; the plates wear twice as fast as they did previously. As can be seen in Fig. 7, the system behavior is adapted accordingly. At first, a slight deviation between desired and obtained RUL can be observed; however, the system lasts for the required 500 cycles.

In another test of the behavior adaptation process, the requirements for the system were changed at 200 cycles. The system is now required to last for 600 cycles instead of 500 cycles, as was the initial requirement. As can be seen in Fig. 8, the adaptation process enables the system to successfully adapt its behavior to changed requirements.

As was shown, an adaptation to either changed system degradation processes or to changed requirements is possible. The controlled system fulfills the desired properties regarding reliability.

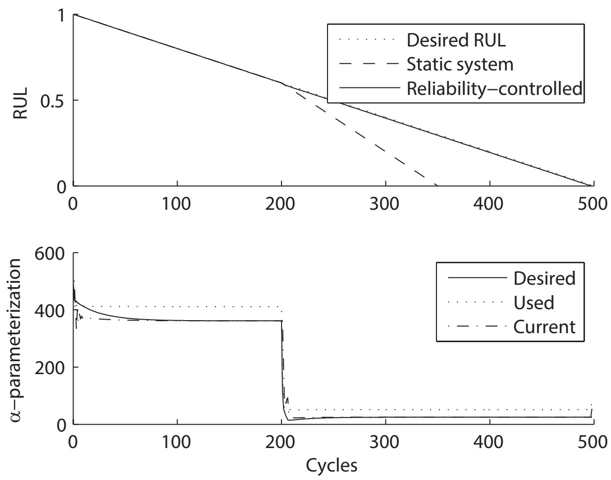

Figure 7. System behavior if a fault occurs. At 200 cycles, the proportionality factor p_f was changed to simulate a clutch system wearing twice as fast as was anticipated for a normal system.

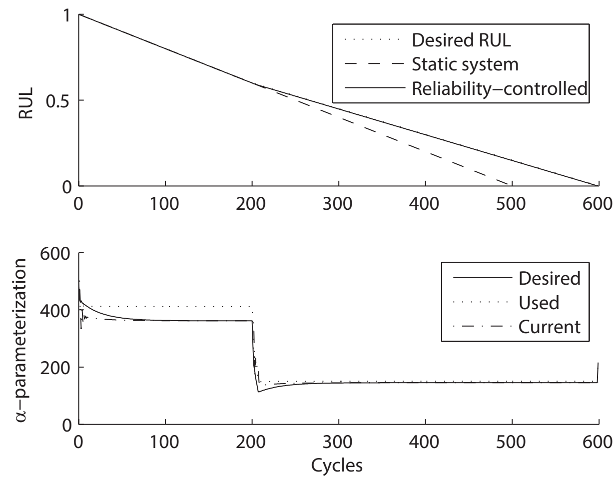

Figure 8. System behavior for changed requirements. At 200 cycles, the system requirements are changed; it is now expected to sustain 600 actuation cycles.

The adaptation to accelerated wear processes and changed user requirements comes at the expense of degrading performance of the system. In case of the clutch system, the value of the α-parameterization is lowered for both adaptations. This leads to lower, i.e. better values of the objective *minimize wear* and to higher, i.e. worse values of the objective *minimize accelerations*. As can be seen in Fig. 4, the main difference between different working points is the duration of the actuation trajectory. A system running in nominal operating mode, i.e. before 200 cycles are reached, has an actuation duration of approximately $9.5\,\mathrm{s}$, giving a comfort value $f_2 = 0.039$. If changed user requirements are to be taken into account, the actuation duration is shortened to approximately $8.9\,\mathrm{s}$, lowering the comfort value to $f_2 = 0.064$. In order to compensate accelerated wear processes, the selected working point requires an even faster actuation duration of approximately $6.9\,\mathrm{s}$ at a comfort value $f_2 = 0.25$. These lower values signify a quicker and less comfortable acceleration maneuver, which is required to react on these great variations in system behavior or requirements. Even though the difference in comfort value suggests severely limited operating potential, the benefit of reaching pre-defined reliability goals will in most cases outweigh the loss in comfort.

7. CONCLUSION & OUTLOOK

The behavior adaptation process of an intelligent system is modelled abstractly. A two-stage control loop was designed with the inner loop controlling the desired system behavior whereas the outer loop controls the remaining useful lifetime. To this end, an existing controller for the inner system behavior is implemented. For the outer loop, a new controller is added. Simulation results show, that the adaptation of the system behavior based on the remaining useful lifetime successfully adapts the behavior if either the system behavior or the requirements change. In both cases, the desired useful lifetime can be accomplished.

While simulations show that the system degrades as desired, experimental validation is still required. Since the behavior adaptation and experiments that span the whole system lifetime are complex, the setup of a dedicated test rig is currently being pursued.

ACKNOWLEDGMENT

This research and development project is funded by the German Federal Ministry of Education and Research (BMBF) within the Leading-Edge Cluster Competition “it’s OWL” (intelligent technical systems OstWestfalenLippe) and managed by the Project Management Agency Karlsruhe (PTKA). The authors are responsible for the contents of this publication.

REFERENCES

Birolini, A. (2007). *Reliability engineering* (5th ed.). Berlin Heidelberg: Springer.

Chena, D., & Trivedi, K. S. (2005, October). Optimization for condition-based maintenance with semi-markov decision process. *Reliability Engineering & System Safety*, *90*(1), 25-29. doi: 10.1016/j.ress.2004.11.001

Fleischer, G. (1973). Energetische Methode der Bestimmung des Verschleißes. *Schmierungstechnik*, *4*(9), 269-274.

Gausemeier, J., Rammig, F. J., Schäfer, W., & Sextro, W. (Eds.). (2014). *Dependability of self-optimizing mechatronic systems*. Heidelberg New York Dordrecht London: Springer. doi: 10.1007/978-3-642-53742-4

Jardine, A. K. S., Lin, D., & Banjevic, D. (2006, October). A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, *20*(7), 1483-1510. doi: 10.1016/j.ymssp.2005.09.012

Joo, S. J., Levary, R. R., & Ferris, M. E. (1997, February). Planning preventive maintenance for a fleet of police vehicles using simulation. *SIMULATION*, *68*(2), 93-99. doi: 10.1177/003754979706800202

Krüger, M., Remirez, A., Kessler, J. H., & Trächtler, A. (2013, June). Discrete objective-based control for self-optimizing systems. In *American control conference (acc), 2013* (p. 3403-3408).

Meyer, T., Sondermann-Wölke, C., Kimotho, J. K., & Sextro, W. (2013). Controlling the remaining useful lifetime using self-optimization. *Chemical Engineering Transactions*, *33*, 625-630. doi: 10.3303/CET1333105

BIOGRAPHIES

Tobias Meyer studied mechanical engineering and mechatronics at the University of Paderborn. Since 2011 he is with the research group Mechatronics and Dynamics at the University of Paderborn. His research focusses on the dependability of intelligent self-optimizing systems.

Walter Sextro studied mechanical engineering at the Leibniz University of Hanover and at the Imperial College in London. Afterwards, he was development engineer at Baker Hughes Inteq in Celle, Germany and Houston, Texas. Back as research assistant at the University of Hanover he was awarded the academic degree Dr.-Ing. in 1997. Afterward he habilitated in the domain of mechanics under the topic Dynamical contact problems with friction: Models, Methods, Experiments and Applications. From 2004-2009 he was professor for mechanical engineering at the Technical University of Graz, Austria. Since March 2009 he is professor for mechanical engineering and head of the research group Mechatronics