Optimization-based reliability control of mechatronic systems

Zur Erlangung des akademischen Grades

DOKTOR DER INGENIEURWISSENSCHAFTEN (Dr.-Ing.)

der Fakultät für Maschinenbau

der Universität Paderborn

genehmigte

DISSERTATION

von

Dipl.-Ing. Tobias Meyer

aus Hildesheim

Tag des Kolloquiums: 15. Dezember 2016
Erstgutachter: Prof. Dr.-Ing. habil. Walter Sextro
Zweitgutachter: Prof. Dr.-Ing. Bernd Bertsche

Acknowledgment

This thesis was written while I was at the Chair for Mechatronics and Dynamics at Paderborn University. During this time, the Collaborative Research Center 614 „Self-Optimizing Concepts and Structures in Mechanical Engineering“ was defining for me and turned out to become the basis of my work. Without the prior works of so many researchers I wouldn't have been able to create my own scientific contribution – you are the giants on whose shoulders I am standing and for this I thank you!

My sincere thanks go to Prof. Dr.-Ing. Walter Sextro for the trust he had in me, for advising me during my research and for being first examiner of this thesis. Also I'd like to thank Prof. Dr.-Ing. Bernd Bertsche for being second examiner and Prof. Dr.-Ing. Detmar Zimmer and Dr.-Ing. Tobias Hemsel for serving on the committee. Above all I thank my colleagues for the wonderful discussions and the many shared publiactions, especially Dr.-Ing. Christoph Sondermann-Wölke, Dr.-Ing. James Kuria Kimotho and Thorben Kaul. At the same time, my work with Paul Eichwald, Andreas Unger, Simon Althoff, Dr.-Ing. Michael Brökelmann and Dr.-Ing. Matthias Hunstig in the field of wire bonding was quite different, but interesting nonetheless. My gratitude also goes to all the other members of our department who made my time at Paderborn University so enjoyable. A large contribution to my personal development was made by all the students who had their final theses advised by me. I thank them for trusting me with such an important part of their studies!

All this would not have been possible without my parents and my brother with his family, who enabled my education and who followed my research work closely and with constant interest. Not to be forgotten is Matilda, who was constant incentive and welcome distraction during my final half year.

Last but not least I'd like to thank Kathi who not only supports me personally, but with whom I shared some very valuable scientific discussions.

Bremen, December 2016

Tobias Meyer

Abstract

Reliability-adaptive systems allow an adaptation of system behavior based on current system reliability. They can extend their lifetime at the cost of lowered performance or vice versa. This can be used to adapt failure behavior according to a maintenance plan, thus increasing availability while using up system capability fully. To facilitate setup, a control algorithm independent of a degradation model is desired.

A closed loop control technique for reliability based on a health index, a measure for system degradation, is introduced. It uses self-optimization as means to implement behavior adaptation. This is based on selecting the priorities of objectives that the system pursues. Possible working points are computed beforehand using model-based multiobjective optimization techniques. The controller selects the priorities of objectives and this way balances reliability and performance.

As exemplary application, an automatically actuated single plate dry clutch is introduced. The entire reliability control is setup and lifetime experiments are conducted. Results show that the variance of time to failure is reduced greatly, making the failure behavior more predictable. At the same time, the desired usable lifetime can be extended at the cost of system performance to allow for changed maintenance intervals. Together, these possibilities allow for greater system usage and better planning of maintenance.

Kurzfassung

Zuverlässigkeitsadaptive Systeme ermöglichen eine Anpassung des Systemverhaltens basierend auf der aktuellen Systemzuverlässigkeit. Sie können damit ihre Lebensdauer und die Leistungsfähigkeit situationsbezogen gegeneinander abwägen. Dies kann genutzt werden, um das Ausfallverhalten an einen Wartungsplan anzupassen, wodurch die Verfügbarkeit erhöht und das Potenzial des Systems voll ausgeschöpft wird. Um die Implementierung einer solchen Anpassung zu ermöglichen, ist ein von einem Verschleißmodell unabhängiger Regelalgorithmus gewünscht.

Ein Zuverlässigkeitsregler auf Basis des health index, eines Maßes für Systemverschleiß, wird entwickelt. Er nutzt dazu Selbstoptimierung als eine mögliche Umsetzung der Verhaltensanpassung. Sie basiert auf der Auswahl von Zielen, die das System verfolgt. Mögliche Arbeitspunkte werden vorab mit Mehrzieloptimierungstechniken berechnet. Der Regler stellt die Priorität von Zielen ein und wägt damit zwischen Zuverlässigkeit und Leistungsfähigkeit ab.

Als Beispielanwendung wird eine automatische Einscheiben-Trockenkupplung eingeführt. Der Zuverlässigkeitsregler wird vollständig aufgebaut und Lebensdauerexperimente werden durchgeführt. Ergebnisse zeigen, dass die Streuung der Ausfallzeit stark reduziert werden kann, wodurch das Ausfallverhalten besser vorhersehbar wird. Zugleich kann die gewünschte Lebensdauer auf Kosten der Leistungsfähigkeit erhöht werden, um verlängerte Wartungsintervalle zu realisieren. Zusammen werden eine bessere Ausnutzung der Systemleistungsfähigkeit und eine bessere Planung der Wartung ermöglicht.

Prior Publications

Journal Articles

Kimotho, J. K., Sondermann-Woelke, C., Meyer, T., and Sextro, W. „Application of Event Based Decision Tree and Ensemble of Data Driven Methods for Maintenance Action Recommendation“. In: *International Journal of Prognostics and Health Management* 4.2 (2013).

Kimotho, J. K., Sondermann-Woelke, C., Meyer, T., and Sextro, W. „Machinery Prognostic Method Based on Multi-Class Support Vector Machines and Hybrid Differential Evolution – Particle Swarm Optimization“. In: *Chemical Engineering Transactions* 33 (2013), pp. 619–624. DOI: 10.3303/CET1333104.

Meyer, T., Sondermann-Wölke, C., Kimotho, J. K., and Sextro, W. „Controlling the Remaining Useful Lifetime using Self-Optimization“. In: *Chemical Engineering Transactions* 33 (2013), pp. 625–630. DOI: 10.3303/CET1333105.

Meyer, T., Sondermann-Wölke, C., and Sextro, W. „Method to Identify Dependability Objectives in Multiobjective Optimization Problem“. In: *Procedia Technology* 15 (2014), pp. 46–53. DOI: 10.1016/j.protcy.2014.09.033.

Reviewed Conference Proceedings

Althoff, S., Meyer, T., Unger, A., Sextro, W., and Eacock, F. „Shape-Dependent Transmittable Tangential Force of Wire Bond Tools“. In: *IEEE 66th Electronic Components and Technology Conference*. 2016, pp. 2103–2110. DOI: 10.1109/ECTC.2016.234.

Eichwald, P., Sextro, W., Althoff, S., Eacock, F., Unger, A., et al. „Analysis Method of Tool Topography Change and Identification of Wear Indicators for Heavy Copper Wire Wedge Bonding“. In: *Proceedings of the 47th International Symposium on Microelectronics*. 2014, pp. 856–861. DOI: 10.4071/isom-THP34.

Kimotho, J. K., Meyer, T., and Sextro, W. „PEM fuel cell prognostics using particle filter with model parameter adaptation“. In: *Prognostics and Health Management (PHM), 2014 IEEE Conference on*. June 2014, pp. 1–6. DOI: 10.1109/ICPHM.2014.7036406.

Kaul, T., Meyer, T., and Sextro, W. „Integrated Model for Dynamics and Reliability of Intelligent Mechatronic Systems“. In: *European Safety and Reliability Conference (ESREL2015)*. Ed. by Podofillini et al. London: Taylor and Francis, 2015. DOI: 10.1201/b19094-290.

Kaul, T., Meyer, T., and Sextro, W. „Integrierte Modellierung der Dynamik und der Verlässlichkeit komplexer mechatronischer Systeme“. In: *10. Paderborner Workshop Entwurf mechatronischer Systeme*. Ed. by Gausemeier, J., Dumitrescu, R., Rammig, F., Schäfer, W., and Trächtler, A. HNI-Verlagsschriftenreihe. Paderborn: Heinz Nixdorf Institut, Universität Paderborn, 2015, pp. 101–112.

Kaul, T., Meyer, T., and Sextro, W. „Modeling of Complex Redundancy in Technical Systems with Bayesian Networks“. In: *Proceedings of the Third European Conference of the Prognostics and Health Management Society 2016*. 2016.

Meyer, T., Hölscher, C., Menke, M., Sextro, W., and Zimmer, D. „Multiobjective Optimization including Safety of Operation Applied to a Linear Drive System“. In: *Proc. Appl. Math. Mech.* Vol. 13. 2013, pp. 483–484. DOI: 10.1002/pamm.201310234.

Meyer, T., Keßler, J. H., Sextro, W., and Trächtler, A. „Increasing Intelligent Systems' Reliability by Using Reconfiguration“. In: *Proceedings of the Annual Reliability and Maintainability Symposium (RAMS)*. Orlando, FL, 2013. DOI: 10.1109/RAMS.2013.6517636.

Meyer, T., Kaul, T., and Sextro, W. „Advantages of reliability-adaptive system operation for maintenance planning“. In: *Proceedings of the 9th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes*. 2015, pp. 940–945. DOI: 10.1016/j.ifacol.2015.09.647.

Meyer, T., Kimotho, J. K., and Sextro, W. „Anforderungen an Condition-Monitoring-Verfahren zur Nutzung im zuverlässigkeitsgeregelten Betrieb adaptiver Systeme“. In: *27. Tagung Technische Zuverlässigkeit (TTZ 2015) - Entwicklung und Betrieb zuverlässiger Produkte*. 2260. Leonberg, 2015, pp. 111–122.

Meyer, T. and Sextro, W. „Closed-loop Control System for the Reliability of Intelligent Mechatronic Systems“. In: *Proceedings of the Second European Conference of the Prognostics and Health Management Society 2014*. Vol. 5. 2014.

Meyer, T., Sondermann-Wölke, C., Sextro, W., Riedl, M., Gouberman, A., et al. „Bewertung der Zuverlässigkeit selbstoptimierender Systeme mit dem LARES-Framework“. In: *9. Paderborner Workshop Entwurf mechatronischer Systeme*. Ed. by Gausemeier, J., Dumitrescu, R., Rammig, F., Schäfer, W., and Trächtler, A. HNI-Verlagsschriftenreihe. Paderborn: Heinz Nixdorf Institut, Universität Paderborn, 2013, pp. 161–174.

Meyer, T., Unger, A., Althoff, S., Sextro, W., Brökelmann, M., et al. „Modeling and Simulation of the ultrasonic wire bonding process“. In: *2015 17th Electronics Packaging Technology Conference*. 2015. DOI: 10.1109/EPTC.2015.7412377.

Meyer, T., Unger, A., Althoff, S., Sextro, W., Brökelmann, M., et al. „Reliable Manufacturing of Heavy Copper Wire Bonds Using Online Parameter Adaptation“. In: *IEEE 66th Electronic Components and Technology Conference*. 2016, pp. 622–628. DOI: 10.1109/ECTC.2016.215.

Sondermann-Woelke, C., Meyer, T., Dorociak, R., Gausemeier, J., and Sextro, W. „Conceptual Design of Advanced Condition Monitoring for a Self-Optimizing System based on its Principle Solution“. In: *Proceedings of the 11th International Probabilistic Safety Assessment and Management Conference (PSAM11) and The Annual European Safety and Reliability Conference (ESREL2012)*. Helsinki, Finland, 2012.

Unger, A., Sextro, W., Althoff, S., Eichwald, P., Meyer, T., et al. „Experimental and Numerical Simulation Study of Pre-Deformed Heavy Copper Wire Wedge Bonds“. In: *Proceedings of the 47th International Symposium on Microelectronics (IMAPS)*. San Diego, CA, US, 2014, pp. 289–294.

Unger, A., Sextro, W., Althoff, S., Meyer, T., Brökelmann, M., et al. „Data-driven Modeling of the Ultrasonic Softening Effect for Robust Copper Wire Bonding“. In: *Proceedings of 8th International Conference on Integrated Power Electronic Systems*. Vol. 141. 2014, pp. 175–180.

Unger, A., Sextro, W., Meyer, T., Eichwald, P., Althoff, S., et al. „Modeling of the Stick-Slip Effect in Heavy Copper Wire Bonding to Determine and Reduce Tool Wear“. In: *2015 17th Electronics Packaging Technology Conference*. 2015. DOI: 10.1109/EPTC.2015.7412375.

Book Contributions

Gausemeier, J., Rammig, F. J., Schäfer, W., and Sextro, W., eds. *Dependability of Self-optimizing Mechatronic Systems*. Lecture Notes in Mechanical Engineering. Contributions to chapters *Introduction to Self-optimization and Dependability*, *Methods of Improving the Dependability of Self-optimizing Systems*, *Case Study* and *Conclusion and Outlook*. Heidelberg New York Dordrecht London: Springer, 2014. DOI: 10.1007/978-3-642-53742-4.

Contents

Nomenclature

To keep this list brief, only the important symbols that are used multiple times are included.

A	Availability
$\boldsymbol{A}$	Dynamics matrix
a	Slope of linear function across Pareto front
$a_{w,t_{end}}$	Rms value of frequency weighted accelerations
a_w	Frequency weighted accelerations
α	Value of behavior parameterization
α_C	Reliability controller output
α_{cur}	Current value of behavior parameterization
α_{des}	Desired value of behavior parameterization
α_{max}	Maximum allowable value of behavior parameterization
α_{min}	Minimum allowable value of behavior parameterization
α_U	User input on behavior parameterization
α_{use}	Used value of behavior parameterization
$\boldsymbol{B}$	Input matrix
b	Orthogonal slope of linear function across Pareto front
C	Area of friction pads
$\boldsymbol{C}$	Output matrix
CDF $(\cdot\|\cdot)$	Cumulative density function of ... given ...
$\boldsymbol{D}$	Feedthrough matrix
D_α	Perturbation of system behavior
D_a	Distance on Pareto front
D_p	Maximum distance across Pareto front
D_R	Disturbance on reliability
d	Virtual vehicle damping
ΔHI	Degradation rate
ΔHI_{lin}	Linearized degradation rate
$\Delta HI_{nominal}$	Nominal degradation rate
ΔHI_{offset}	Offset of degradation rate
$\Delta\omega$	Difference in rotational velocities
$\hat{E}(\cdot)$	Estimator for expected value of stochastic variable
E_f	Friction work
e	Optimization constraint threshold
ϵ	Upper bound in ϵ-constraint method
F	Failure function
f	Objective function(s)
F_N	Normal force of clutch plates
$F_{N,opt}$	Optimal trajectory for normal force of clutch plates

f_1	Objective function 1, for application example: Friction work
f_2	Objective function 2, for application example: Accelerations
$f_{1/2,cur}$	Current experimental value of objective function 1 or 2
$f_{1/2,max/min}$	Maximum or minimum values of objective function 1 or 2
G_α	Transfer function of full behavior control loop
G_c	Transfer function of behavior controller
HI	Health index
$\widehat{HI}$	Estimated health index
$\widetilde{HI}$	Measured health index
HI_0	Health index of new system
HI_{des}	Desired health index
$HI_{des,end}$	Desired health index at specified lifetime t_f
$h_{h,d,b,f,a,c,g}$	Shape parameters of friction model function
$\boldsymbol{I}$	Identity matrix
i_M	Motor current
J	Cost function in model predictive control
J_x	Cost function for states in model predictive control
J_u	Cost function for system input in model predictive control
$\boldsymbol{K}$	Kalman gain
K_i	Integrator amplification factor of behavior controller
K_p	Proportional amplification factor of behavior controller
k	Index value
$k_{m,T}$	Motor constant for torque measurement
k_{use}	Index of used Pareto point in Pareto front and set
l	Friction plate thickness reduction
$\tilde{l}$	Measured friction plate thickness reduction
$\tilde{l}_i$	Measured initial position of friction plate
$\tilde{l}_m$	Current measured position of friction plate
L	Slope of relationship from thickness reduction to health index
l_{max}	Maximum permissible friction plate thickness reduction
$MTTF$	Mean time to failure
$MTTR$	Mean time to repair
m	Number of time steps for simulation in model predictive control
μ	Coefficient of friction
μ_0	Nominal coefficient of friction
n	Number of elements in a set, e.g. number of objective functions
O	Set of all possible objective values
o	objective value(s)
$\Omega_{1,2,3,4}$	Parameters of transfer function for acceleration evaluation
$\hat{\omega}$	Shape parameter for low velocity approximations
ω_1	Drive motor velocity
ω_2	Load motor velocity
$\boldsymbol{P}$	Set of all possible parameter values
$\boldsymbol{P}_0$	Initial estimate covariance
PDF $(\cdot\|\cdot)$	Probability density function of ... given ...
$\boldsymbol{p}$	parameter values
$\hat{\boldsymbol{p}}$	Pareto set

$\boldsymbol{p}_{opt}$	Optimal parameter values
p_f	Wear proportionality constant
P_f	Friction power
$\boldsymbol{Q}$	Process noise covariance
$Q_{1,2,4}$	Parameters of transfer function for acceleration evaluation
$\boldsymbol{q}$	State of generalized degradation model
R	Reliability function
$r_{1/2}$	Gear reduction ratio of motor 1/motor 2
R_{spec}	Specified value of reliability function at desired lifetime t_f
$\boldsymbol{r}$	Measurement residual
RUL	Remaining useful lifetime
S	Combined objective function in weighted sum method
$\boldsymbol{S}$	Residual covariance
$s\left(\cdot\right)$	Forward s-transform
$s^{-1}\left(\cdot\right)$	Inverse s-transform
$\hat{\sigma}\left(\cdot\right)$	Estimator for standard deviation of stochastic variable
T	Duration of optimal control sequence
T_0	Dry friction torque in virtual vehicle
T_p	Torque transmitted at clutch plates
$T_{p,0}$	Velocity invariant torque
t	Time
t_0	Initial time
t_{end}	End time
t_f	Failure time
t_k	Current time step k
t_p	prediction time steps during model predictive control
t_r	Duration of one actuation cycle
t_s	Step size of behavior controller
t_{spec}	Specified lifetime
Θ	Inertia of virtual vehicle
$\boldsymbol{u}$	Input vector
$\boldsymbol{V}$	Estimate covariance
v	Multiplicative fault on generic system degradation model
W	Wear volume
W_k	Transfer function for acceleration evaluation
$\boldsymbol{w}$	Weights
w_x	Weight of states in model predictive control
w_u	Weight of system input in model predictive control
$\boldsymbol{x}$	State vector
$\hat{\boldsymbol{x}}$	Estimate of state vector
$\boldsymbol{y}$	Output vector
z	Discrete z-transform

1 Introduction

Any industrial product undergoes several stages throughout its life. These range from development over manufacturing and usage to disposal or recycling. Development and manufacturing require ressources, which cannot be fully retrieved during recycling, if the product is recycled at all. Limited ressources demand to be used consciously before depletion. The most accessible way to achieve this is to use a product fully and for as long as possible.

When a product is taken out of service and ultimately discarded, it becomes *obsolete*. According to [PDG+16, pp. 45,64], obsolescence can be categorized into four classes[1]:

Psychological obsolescence:

> „[This] kind of obsolescence comprises premature aging and the resulting exchange of functioning products due to fashion, new technical trends and consume patterns“ [PDG+16, p. 64], [BHP+14, p. 60].
>
> Psychological obsolescence is a subjective perception that is individual for each user and mainly relevant for consumer products, not so much for investments in company assets. Investements into assets are usually based on a careful cost-benefit evaluation that yields a decision whether an asset fulfills requirements and whether it is worth investing in or not. Private users, on the contrary, might base a buying decision on subjective feeling without careful reasoning. It is these products that are exchanged well before their actual service life has been reached.

Functional obsolescence:

> „Reasons for functional obsolescence are rapid changing of technical or functional requirements for a product (e.g. interoperability of software and hardware of different electronic devices). Strong influence on functional obsolescence comes from diverging interests of software and hardware manufacturers.“ [PDG+16, p. 64], [BHP+14, p. 60].
>
> Functional obsolescence is closely related to requirements that were (unconsciuously) defined when the product was purchased. If at the time of purchase, these were not known sufficiently well or if they had changed afterwards, the product might not fulfill them after a (too) short amount of time. To be able to use a product for a long time, it is thus of paramount importance to know *how* and *what for* it is to be used. Actual product lifetime can be prolonged by adapting the product to changed requirements, but this also requires the product to have this ability.

Economic obsolescence:

> „Economic obsolescence describes the degradation of usage capacity of a product since the use of product-related ressources, maintenance or repair is omitted for cost reasons and the margin to alternative costs for a new product is too small. Reasons are e.g. short product development cycles, rapid price drop, repair unfriendly design,

[1] Ordering was changed for a conclusive line of argument.

high repair cost, lacking availability of spare parts, tools or services.“ [PDG+16, p. 65].

Economic obsolescence is directly related to repair cost. If repair cost can be lowered, the (operational) remainder of the product can be used for longer periods of time after repair. In order to fully use up system capability, it is thus desirable to lower repair cost and to operate the product until all components are worn out, at best all at the same time.

Material obsolescence:

„Material obsolescence is caused by the insufficiency of material and components. Product aging shows e.g. as (too fast) degradation of strength by environmental corrosion, plastification, setting or conversion processes“ [PDG+16, p. 64], [BHP+14, p. 60]. This is what is commonly regarded as product failure. In order to have a product that can be used for a long time, such failures need to be avoided.

One of the main results of [PDG+16, p. 283] is that two strategies are required for extended usage and to counter obsolescence:

- Strategies to reach a guaranteed minimum lifetime and to prolong product lifetime,
- Strategies to prolong product usage duration.

A product manufacturer's main influence on technical restrictions to product usage duration is fulfillment of requirements. Whether a user's requirements are fulfilled is dependent on system performance. With increased system performance usually also comes increased load on components and in turn reduced lifetime. Finding a solution to this problem usually means deliberately selecting a trade-off between system performance and lifetime during product development.

With more advanced systems also comes the possibility to create adaptive systems, where such a trade-off is selected at runtime by the operator or by the system itself. This is commonly used to adapt performance to actual current requirements, e.g. with different operating modes in sports cars or CPU throttling in computers or cellphones. However, with such adaptation capability also comes the possibility to change degradation behavior of a system by adapting system operation accordingly.

The objective of this thesis is to develop a method that allows to find a balance between lifetime and performance at runtime, in turn contributing to both strategies against obsolescence. On the one hand, a minimum lifetime could be guaranteed at the cost of (possibly) lowered performance. Overall product lifetime could be increased by making maintenance or repair schedulable while using up system capability fully and avoiding early failures. On the other hand, system performance could be increased at the cost of lowered time to failure. If this is used to counteract functional obsolescence, where changed requirements would require replacement before failure, the actual usage duration could be increased. Such adaptation may not decrease system dependability.

1.1 Dependability of intelligent technical systems

The term dependability is commonly used in computer sciences and was well defined in [ALR+04]. According to this article, „the dependability of a system is the ability to avoid service failures that are more frequent and more severe than is acceptable“ [ALR+04, p. 13], with a service failure being „an event that occurs when the

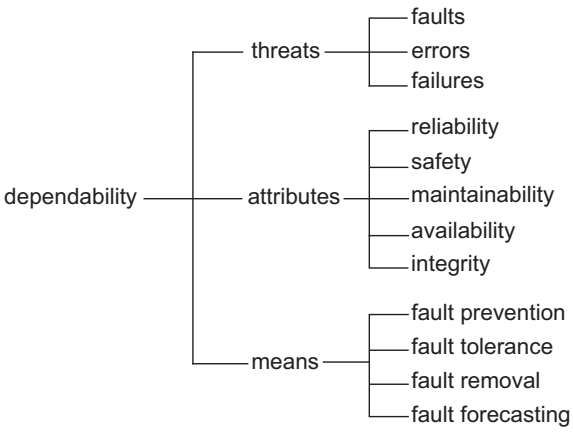

Figure 1.1: Dependability tree according to [ALR+04, p. 14]

delivered service deviates from correct service“ [ALR+04, p. 13]. Dependability is comprised of five attributes. These are availability, reliability, safety, integrity and maintainability[2].

In order to attain dependability in a system, several means are used. These can be grouped into four categories: fault prevention, fault tolerance, fault removal and fault forecasting. The dependability of a system is threatened by failures, which are characterized by the deviation of at least one external state of the system from the correct service state with the deviation being called error and faults being the adjudged or hypothesized root cause of a fault. The complete taxonomy is depicted in figure 1.1.

The five attributes of dependability have to be fulfilled during development. Of these, the attributes reliability and availability are of paramount interest when considering online adaptation processes. According to [Bir07, p. 9], availability expresses the „ratio of delivered to expected service“, i.e.

$$A = \frac{MTTF}{MTTF + MTTR},$$

where $MTTF$ is the *mean time to failure* and $MTTR$ is the *mean time to repair*. It is immediately apparent that the availability of a system can be increased by either increasing the mean time to failure or by decreasing the mean time to repair. An increase in mean time to failure can be achieved by an increase in *reliability* of the system, thus these two attributes are interconnected. According to [Bir07, p. 2], reliability is „the probability that the item will perform its required function under given conditions for a stated time interval. It is generally designated by R“.

To assess system reliability, probabilistic models are employed which usually give a *reliability function* $R(t)$ as result. The reliability function is the probability distribution for the system being fit for operation. The inverse to the reliability function $R(t)$ is the *failure function*[3] $F(t) = 1 - R(t)$. It is the probability distribution for the system *having failed* and takes the whole history of system operation into account as well. $F(t)$ can also be regarded as cumulative density function for the probability that a failure occurs.

[2]Confidentiality, which is mentioned in [ALR+04, p. 14] as well, can be omitted since it is not an attribute of dependability, but one of security.

[3]Depending on source, it is also called *failure probability*. Within this thesis, the term *failure function* is preferred for its disambiguousness.

For the corresponding density function, common probability distributions, e.g. constant failure rate (exponential distribution), normal distribution or Weibull distribution, can be used. Constant failure rate is suitable for random failures, as the probability that a failure occurs is equal throughout the whole operating time. It is commonly used for electronic parts. Normal distribution is suitable for failures due to wear, but can only model a single failure mode [BL04, p. 37]. This limits applicability for systems which additionally also show early failures. Weibull distribution, depending on shape parameter, can be used to model early failures during wear-in, constant failure rate for random failures during operation and finally failures due to wear. To determine parameters that closely fit actual failure behavior, multiple run-to-failure tests are required.

From the reliability function, the mean time to failure can be deduced [Bir07, p. 6]:

$$MTTF = \int_0^\infty R(t)\,\mathrm{d}\,t.$$

Mean time to repair, on the other hand, is highly dependent on many factors including logistic support, human factors and failure time[4]. Generally, the efficiency of a repair is increased if it can be scheduled well in advance. In this case, presence of all required spare parts, equipment and personnel can be planned. Such is the case in many railway applications, where safety-critical maintenance and repair has to be conducted routinely and is scheduled in advance [CYT06; GB06].

In order to achieve high availability, low mean time to repair is desired, making operation until failure undesirable. Instead, maintenance (repair) is conducted *before* failure, thus reducing mean time to *end of operation* to be slightly less than (anticipated) mean time to failure, but reducing mean time to repair considerably. It becomes apparent that the term mean time *to failure* as basis of the definition of availability is problematic. More specialized terms such as the *mean time to preventive maintenance* can be used instead. To cope with this inconsistency, within this thesis, mean time to failure shall always denote the time until a system is taken out of service, whether for (preventive) maintenance or for repair[5].

By not only increasing the mean time to failure, but by making it more predictable and by being able to schedule maintenance early yet flexibly, a decrease in mean time to repair can be achieved. Both factors greatly increase availability. This can be realized by adapting system behavior to current reliability during operation. Systems that allow such adaptation, are called *reliability-adaptive systems*.

1.2 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 thesis, it is assumed that after maintenance, a system is as-good-as-new. Traditionally, maintenance was conducted as either corrective of preventive maintenance [Bir07, pp. 8,112]. In corrective maintenance, system functionality is reestablished once a failure occurs. It directly leads to the term mean time *to failure*, as discussed in section 1.1.

[4] In this case, *wall time*, i.e. the time of day or during the year.

[5] If the actual *mean* time to failure is meant, i.e. the time t at which $F(t) = 0.5$, this is denoted as 50% survival time.

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. Corrective maintenance is commonly followed for many consumer products, e.g. cell phones, notebook computers or bicycles. It comes with the risk of catastrophic failures which makes it unsuitable for many systems, e.g. trains or airplanes. Figure 1.2 shows the effects of these approaches on system health. At first, a system is 100% healthy, but due to degradation, the health index, which is a measure for system health and is indicated with a solid black line, is decreased continuously. After maintenance, the system is restored and the health index starts at 100% and decreases again, indicated by individual gray lines. It becomes obvious that in corrective maintenance, availability 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. An application example is given in [JLF97], where maintenance for a fleet of 642 police vehicles is optimized. Also car manufacturers recommend this approach by specifying maintenance intervals for certain components. 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

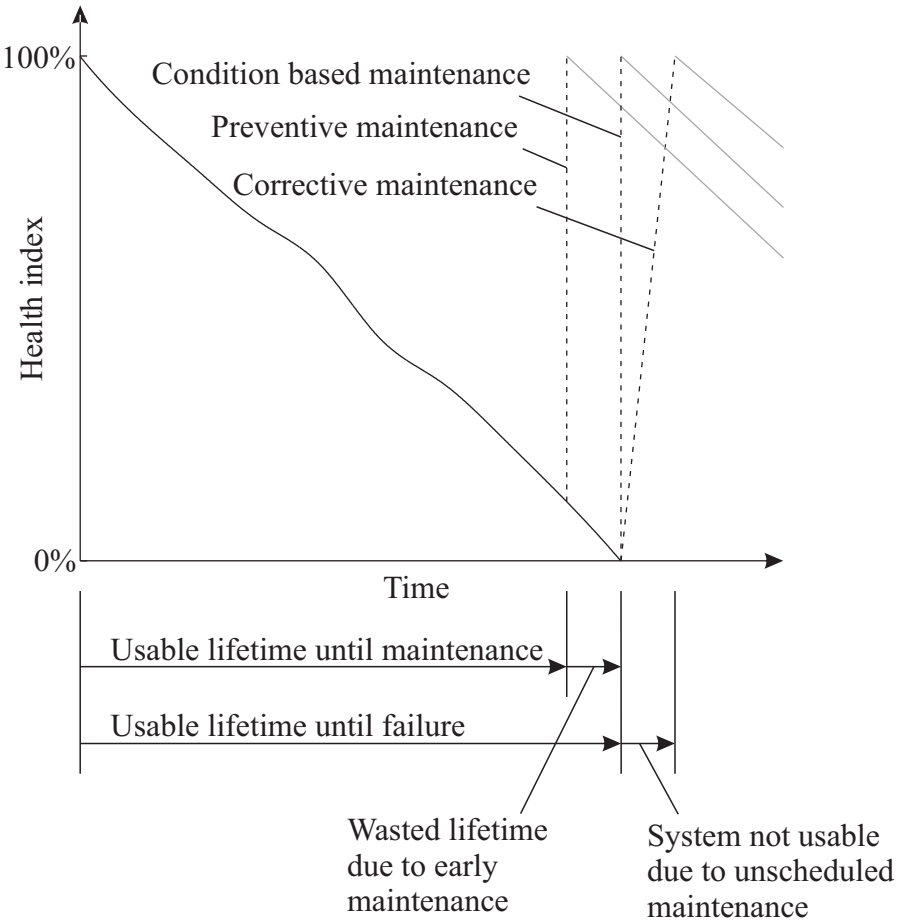

Figure 1.2: Maintenance planning techniques and their effect on usable lifetime.

large fleets of identical systems with identical usage and can hardly be implemented for unique machinery.

In order to overcome these drawbacks, condition based maintenance can be used. According to [JLB06, p. 1484], 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 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, see e.g. [CT05]. Also, condition based maintenance is only possible if maintenance teams are available to perform the required work, which imposes further restrictions on maintenance planning and could possibly decrease system availability.

Feeding information about the current system reliability back into system operation allows to adjust system behavior according to its current reliability. Therefore, the usual approach can be reversed. It now becomes possible to schedule maintenance operations with the system adapting its behavior and its degradation accordingly. The closed loop control proposed in this thesis allows for such operation. It is based on self-optimization, which is a means of influencing system behavior during operation.

1.3 Self-optimizing systems

In today's market, more and more competitive and successful products are so-called mechatronic systems. They contain not only mechanical elements, but also electrical actuators, sensors, at least one micro computer and software. The micro computer acts as an embedded controller, which reacts on sensor signals by controlling system movement through actuators, thus forming a closed control loop. However, these systems are not able to react appropriately in all new situations. Instead, they are limited to those external events that engineers anticipated during development.

With the advent of powerful, yet small and robust computer systems, even more sophisticated systems are possible. These are then not only able to react as has been anticipated during development, but instead, they can change their behavior autonomously, based on either a changed environment or even on changed requirements. Among these systems are *self-optimizing systems*. A basic requirement for a system to be called self-optimizing is that it adapts its behavior by means of objectives [Gau04, p. 25]. For this, desired system operation is expressed in terms of objectives and their priorities. The working point that is required is then selected accordingly. To adapt system behavior, a change of working point may include changes to the structure of information processing, e.g. other closed loop control laws, or parameter changes such as changed controller parameters. To allow for such objective-based system operation, the fulfillment of all objectives needs

to be quantified during operation. To compute suitable working points before operation, fulfillment of objectives is quantified in a model of system operation. This model is then simulated and suitability of working points is evaluated. An efficient process for this is to use multiobjective optimization algorithms, which yield *optimal* working points and where at runtime, selection of working points can be reduced to selection among these pre-computed optimal working points. This way it is ascertained that system behavior is optimal at all time.

A more detailed introduction to self-optimization follows in chapter 2.

However, many current mechatronic systems barely fulfill dependability requirements, and intelligent self-optimizing systems are even more challenging. With added information processing, in some cases with additional hardware to facilitate behavior adaptation, and with increased communication requirements among system components and among separate systems, complexity is even increased. With increased complexity also comes an increased risk of malfunctions, which has to be compensated during development. These aspects are well researched already and the risks can be compensated with suitable methods, e.g. those published in [DDD+14]. With the added complexity of self-optimization also comes the possibility to actively influence dependability by adapting system behavior during operation, which can be used to benefit dependability. This way, self-optimization is used to create a *reliability-adaptive system.*

1.4 Reliability-adaptive systems

According to [Rak05, p. 1633], *reliability-adaptive systems* are defined by the fulfillment of two requirements: „reliability observation“ and „system influence“. The reliability observation requirement is fulfilled if it is possible to estimate the values of reliability measures for many instants of operation and to update them permanently. Within the context of [Rak05] and related works, stochastic reliability measures are used. System influence is comprised of several points. At first, it has to be possible to transfer the system into a state that would result in more reliable operation. Second, individual systems within a fleet can be operated individually so as to react on their own reliability properties. Third, the system control can take reliability aspects into account. Fourth, reconfiguration can be used to benefit from redundant components. Fifth, precise estimation of the time to preventive maintenance is possible. All these can be combined as required for an individual application.

One way to implement reliability-adaptivity is to employ the Safety and Reliability Control Engineering-Concept (SRCE-Concept) [SR97]. This contains closed loop to control the reliability of a system. However, in early introductory papers (i.e. [SR97]), the authors state that „Because of the fact that this control loop is not a pure technical control approach, a part of the necessary connections can not be given yet.“ [SR97, p. 673]. In [Wol08, p. 83], using model predictive control and a model of the degradation behavior is suggested but not implemented. The main drawback of this concept is that reliability is prioritized over all other aspects of system operation. It is not considered to temporarily allow overly degradatory system behavior, which might be required in case of user demands or the current situation. Degradation models exist for some faults, e.g. crack growth, but generally modeling degradation becomes a challenging task that makes implementation of such control schemes infeasible.

In [Pab05], an approach to modeling of reliability control systems is presented. The idea there is to form a closed description including reliability and system dynamics, which can then be treated using general or dedicated methods from system analysis and control theory. However, forming such a system model is difficult and error-prone. Instead, a more generalized and robust method is desirable.

Classifications for prognostics and health management schemes into four *types* of increasing complexity and capability are introduced in [CH08] and [GS14]. [RB15] builds on these but proposes a new *type 5*, which requires an adaptation of the individual system based on type 1 to 4 data and thus encompasses reliability-adaptive systems. It is shown that since type 5 systems directly build on all properties of the lower types, system complexity and the efffort during development is increased.

While reliability-adaptive systems are thus not new, self-optimization techniques as an approach for the implementation of a suitable control strategy has not been discussed in prior works related to reliability-adaptive systems. Also since implementation of reliability-adaptivity had failed so far due to complexity reasons of existing methods, to be interesting for practical usage, a new method needs to be generalizable and adaptable to a multitude of systems with as little effort as possible. Developing a new method based on self-optimization is a promising approach since self-optimizating systems inherently offer the possibility to adapt system behavior and are always operating in an optimal working point.

1.4.1 Multi-level dependability concept

During the course of the collaborative research center 614 „Self-optimizing concepts and structures in mechanical engineering“, the *multi-level dependability concept* was developed. An early introduction can be found in [FGM+07], whereas later publications expanded on details ([SGM+08]) and altered the graphical representation to the one shown in figure 1.3 ([SGH+09]). The following introduction is based on [GRS09b, p. 61], which is an in-depth introduction including application examples and [DDD+14, p. 56], which includes a brief introduction. The multi-level dependability concept is comprised of two main components: first, evaluation and classification of the current system state and second, means to influence system behavior. Originally it was developed to satisfy safety requirements [FGM+07], but later it was extended to the attribute reliability [SS09, p. 17]. This is made possible by the concept's flexibility, and the fact that by increasing reliability, impending faults, which might pose a safety threat, are avoided. The classification is limited to four states as shown in figure 1.3. These are[6]:

Level I:
: The system is in a safe, normal state. Only objectives of self-optimization are pursued. Neither a dedicated prioritization of dependability nor counter-measures against undesired events are required.

Level II:
: A thread was detected, e.g. a threshold was reached. While the system is still operating in a safe state, pursuing objectives like comfort or energy efficiency only might lead it towards an unsafe state. Self-optimization is used to prioritize dependability

[6]The definition of the four levels is closely based on [GRS09b, p. 62] and [DDD+14, p. 56], but is not reproduced verbatim.

over competing objectives. This leads to a behavior adaptation and in turn keeps system operation dependable, but impairs other objectives.

Level III:

A fault that is considered severe has occured. In order to reach a safer state, emergency mechanisms are triggered in real time to reach level I or II. This can be achieved by mainly pursuing objectives that lead to safe behavior and subordinating all other objectives. If this is not sufficient, switching actions, i.e. changes to the structure of the system, can be executed. This might be necessary e.g. to deactivate faulty components.

Level IV:

The system cannot be controlled anymore. A pre-defined fail-safe state has to be reached to avoid fatal consequences. This is achieved by executing emergency routines.

Note that these definitions mix the terms *safety* and *dependability*. If safety, being one attribute of dependability, is increased, dependability is increased as well. In turn, objective functions for dependability need to be able to increase safety. Reliability, which is the main focus of this thesis, influences safety as well.

For each of these four levels, thresholds or characteristic events and counter-measures to affect the system behavior have to be defined. Using self-optimization, the behavior adaptation is achieved by changing the priorities of system objectives, which in turn change system behavior. In order to control the reliability of a mechatronic system by means of self-optimization, the relationship between objective values and resulting reliability of the system has to be known.

To increase reliability, small influences to the priorities of objectives pursued might be sufficient to increase system reliability. Hard switching can be used to compensate for undesired or dangerous system states, but should not be regarded as means to satisfy reliability demands. The switching between states, that is inherent to the multi-level dependability concept, is contradictory to the desire to have small influences over longer periods of operation.

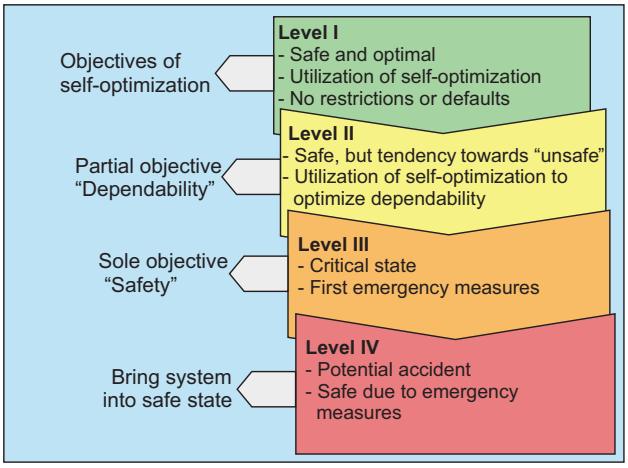

Figure 1.3: Basic characterization of the four levels of the multi-level dependability concept. According to [DDD+14, p. 56]

In [Son15], an approach to implement reliability-based behavior adaptation using self-optimization is presented. This is conducted by the multi-level dependability concept as part of a condition monitoring for self-optimizing systems, which also forms the basis of the thesis at hand. The active guidance control as part of a novel rail vehicle serves as application example [Son15, p. 59]. Objectives include minimization of energy consumption and minimization of wheel flange to rail head contacts, which lead to wear of the flanges. While effectively creating a reliability-adaptive system, this work does not include continuous control of system reliability but instead switches between several predefined states and corresponding control algorithms. Switching is employed to change operating mode at sensor failure, which considerably prolongs usable system lifetime. The undesired effects of switching can also be observed clearly: the current configuration influences wear drastically, while effectively impairing other objective values. After maintenance, fast wear with corresponding good energy efficiency can be observed, while later during usage, wear progession is decreased but energy consumption is increased [Son15, figure 4.28]. While this is an adequate reaction to unexpected failures, to compensate quick wear itself, a continuous control method with a suitable trade-off being selected at all times would be preferred.

1.4.2 Life extending control

The goal of life extending control is to extend the usable life of *mechanical* components, before they fail due to fatigue. So in this context, *life* refers to mechanical fatigue life. Work on life extending control was initially motivated by reusable rocket engines, which would eventually fail due to a crack in the nozzle. Using a model of crack growth, controllers could be designed that increased service life by reducing load on the nozzle. Later, this was extended to other components of the rocket engine and to entirely different systems.

Life extending control was first proposed in [LM91b], which was mainly a concept paper introducing the basic idea. According to this paper, life extending control is always based on multivariable control on a low system level, which makes it possible to select a trade-off among reaching goals for each variable. According to [LM91b], „The fundamental concept of life extending control is to control rates of change of some levels and of some performance variables to minimize damage (or damage rates) [of] critical components while simultaneously maximizing dynamic performance of the plant“ [LM91b, p. 1084]. It is assumed that minimizing forces on critical components extends their life. However, large forces are required for fast control loop response times, so the „time to achieve control performance“, i.e. the speed of a controller, is used to manage life of a critical component.

For the life extending controller, two types of feedback variables are considered: variables that measure dynamic performance and „nonlinear functions of the performance variables representative of the damage variables (stresses, strains, temperature and various rates)“ [LM91b, p. 1086].

Life extending control is subdivided into different classes:

Implicit life extending control: In implicit life extending control, the *best* control algorithm is selected for a set of command transients. It is selected based on an a priori optimization of an overall performance measure, which is composed of two

objective functions that represent dynamic performance and damage respectively. An autonomous adaptation at runtime is not desired.

Life management life extending control: A hierarchical structure composed of two levels is assumed. On the lower level, system dynamics are controlled. On the upper level, an „intelligent control system“ is used to plan which controller to select at the lower level.

On-line minimization of the weighted damage and performance objectives is conducted to select a control for the desired performance and life trade-off. In the proposed life extending controller structure, it is assumed that a given required task has to be fulfilled which can be simulated in full, thus creating an open-loop controller selection trajectory for lower level controls. This trajectory is updated from time to time, forming a closed loop control.

To exemplify implicit life extending control, a basic positioning system that is comprised of a single hydraulic cylinder with the piston rod being the life critical element is modeled in [LM91b, p. 1091]. Damage is given as a function of the number of stress cycles N and the time T it takes to reach the setpoint. Two different controllers are created. One of them is slower, i.e. less actuation cycles per time unit, but also creates less damage per cycle, which in turn yields a higher total number of actuation cycles.

While this structure comes close to the operator controller module structure used for self-optimizing systems, which is introduced in more detail in section 2.2, when and how to switch between pre-determined controllers is not considered. Online-optimization is not feasible for systems with high dynamics or low computational power, so this approach has limited applicability.

Direct life extending control: If it is possible to form a continuous function that allows to predict the incremental damage or damage rate based on measured stresses, strains and temperatures, direct control of life is possible. For this, a setpoint for desired life is compared to the current life and lower level controllers are adapted accordingly. This directly yields a multivariable controller with damage being one control variable. However, such a controller is very complex and the interaction between multiple variables is inherently included.

This is further subclassified in [LM91a, p. 231]:

Measured damage variables life extending control: If damage can be measured directly, it can be used as feedback information for a controller. For this, a continuous damage model is used which is dependent on measured system variables such as measured stress, strains or temperatures. „The control problem then is to minimize damage of the critical life components while maximizing (dynamic) performance of the plant“ [LM91a, p. 231]. It is suggested to permit damage to accumulate at a *setpoint* rate, e.g. linearly over time. „The emphasis here is on obtaining desired system operation by an active, feedback control approach“ [LM91a, p. 231].

Estimated damage variables life extending control: If it is impossible to directly measure damage or damage-driving variables, a model that estimates damage of critital components based on performance measurements and system input can be used. This is similar to an observer used for estimating non-measurable system states. The controller design itself is similar to the one

used for measured damage variables life extending control.

Both approaches classified as direct life extending control require a very good damage model and integrate high system dynamics with low damage accumulation into a single controller. Not only does this complicate system simulation due to different time scales, it also intermixes safety-critical dynamic controllers with auxiliary degradation controllers. Deviations in degradation behavior results in controller input on system dynamics, which could lead to impaired dynamic controller performance and in turn to instability or unsafe operation. For setting up a reliability control system, separation of these components is desirable.

These concepts and the proposed controller structures are based on cyclic stress in mechanical parts and do not form a controller structure that is universally usable for any system. Also, within the scope of life extending control, the controller is limited to two objectives: performance and damage.

1.4.2.1 Life extending control for reusable rocket engine

Initial motivation for life extending control came from a reusable rocket engine which was also used as first application example. In [LM91c], an intelligent control system for the rocket engine that includes durability is presented. The control system objective is to handle durability issues without shutoff of the engine. While no actual implementation is introduced, the basic setup of an intelligent control system for a rocket engine is given. This *could* take system life into account.

An actual implementation for the rocket engine is given in [LSR+92; RDC+94; RDW+94; RWC+93a; RWC+93b]. While the main focus is life prediction using models of crack growth, a controller for system performance is setup as well. The usage scenario is a change of operating point from steady state conditions at 2700 psi and 6.02 oxygen/hydrogen ratio to a new steady state point at 3000 psi and the same ratio. The life is limited by stress in turbine blades due to preburner and turbine operating parameters. A basic open loop feed forward control is compared to one that limits damage rate by taking damage rate constraints into account. The basic performance control yields high damage during transient system operation, whereas total system damage is lowered using damage constrained control without compromising system performance too much. To compute the feed forward control, nonlinear programming is used as optimization technique.

Up until then, the only failure mode was *crack in turbine blade*. [DR96b] builds on this by introducing a damage model for the coolant channel ligament, which is a channel lining the nozzle walls through which fuel is flowing to cool the nozzle walls. In this case, a rocket engine similar to the space shuttle main engine is used as application example. A thermo-fluid model of the engine dynamics and a damage model for the coolant channel ligament is introduced. The accompanying paper [DR96a] combines this with the known failure mode from [RDC+94] and the system dynamics introduced therein. An optimal control problem is formulated, in which system dynamics and both failure modes are taken into account. A feed forward control is computed which, when compared to a pure dynamic control, reduces creep damage of the ligament and fatigue damage of the fuel turbine blades.

A procedure that can be followed to create an open-loop control policy is introduced in [RWC+94a; RWC+94b]. It is meant to be followed by an engineer developing a sys-

tem, but not for automated application. It includes an optimization for determining system parameters suitable for a system design that is able to fulfill a given mission. This is achieved by including upper bounds for damage rate and accumulated damage. Automated evaluation whether system dynamics is good enough is not part of the policy. Instead, it is assured by manually evaluating specific system characteristics after optimization.

In [HTR97], the concept of multiobjective optimization as means to obtain a set of controllers is introduced into life extending control. Several open loop feed forward controls are computed for the known application example, the rocket engine. Using different weights on the individual objectives, three controllers are obtained. A selection among these is not made.

At the same time, damage models were created that specifically addressed the needs of life extending control, such as a generic continuum fatigue damage model [Lor94]. Also in [PR99], the model used for crack propagation in the rocket engine oxygen turbine blades was improved while keeping the basic setup of the control loop.

In the following years, this control was extended with a means for online-selection of the most suitable controller. In [LHR00; LHR98a; LHR98b; LRH01], a performance controller and a damage controller are combined to create a parallel structure. This has the advantage of using individual controllers for both aspects. However, since they are being arranged in parallel with the sum of both outputs acting directly on the plant input, each of them is a perturbation to the other one. This combined with the fact that a highly nonlinear damage model is included makes stability evaluations difficult or even impossible, as the authors note themselves in [LHR98a, p. 15]. For the performance controller, an H_∞ based controller is used. The dynamics of the performance controller are slower than those of the damage controller. This way, the damage controller „corrects“ the performance controller output to reduce damage. The damage controller is a linear filter in state space representation, with optimization variables being the elements of state space matrices. The same maneuver of switching operating points from 2700 psi to 3000 psi is used as characteristic maneuver for optimizing damage and performance at once. Using a weighted sum, a trade-off among these two objectives can be found. Both controllers are thus competing against one another. The authors did not proof stability of life extending control. Also, since life extending control can be regarded as perturbation, proof that it does not de-stabilize system dynamics is be required.

This problem is avoided in [HR01a], where the damage controller directly influences the set point of the performance controller. However, this still influences system dynamics directly and requires one to design and to analyze stability of the whole system at once.

While the rocket engine was the first motivational application example, several others joined in shortly afterwards. In [TCK+99], the step from rocket engine turbines to gas turbines is performed. However, this is mainly a concept paper. Shortly after, [WG01, p. 3707] presented first ideas towards specifically using active clearance control for gas turbine engines as means to improve useful service life using life extending control.

1.4.2.2 Life extending control for aircraft

Aircraft are similar to space craft in the regard that they require to be light but also robust. Life extending control was employed for several structural parts, on which load that is highly dependent on flight dynamics can be lowered by taking damage into account

when setting up flight dynamics controllers. Generally, flight controllers are required to have high bandwidth and fast response, but this also leads to fast loss of fatigue life.

In a helicopter, the main rotor control horn is a highly stressed component for which [RR98] presents a life model. Also flight controllers are designed using a model of helicopter dynamics. The fatigue life for several controllers with different bandwidths is evaluated and correlation between the rotor horn life and controller performance is shown.

In the thesis [Bri03], a crack in the main bevel gear of a helicopter transmission is controlled. Two objectives for flight controllers lead to increased wear and thus higher operational cost. In real rotorcraft, component damage is monitored by a Health and Usage Monitoring System (HUMS), which is implemented to allow for condition based maintenance. Within [Bri03], GENHEL, a commercially available complex nonlinear helicopter model, is used for simulations of an actual helicopter. Additionally, a stress and crack growth model is implemented in GENHEL. Several H_∞ and H_2 controllers with varying objectives for airspeed and damage weight are tested. While it is clearly shown that switching between controllers allows selecting a trade-off between damage and handling qualities, the selection step itself is not considered. The results were also published in [BHR05], but using a linear-quadratic regulator.

A selection process is added in [Tol05]. In this work, two different controller schemes are employed: probabilistic robust control to allow for very responsive controllers with a certain risk of instability and damage mitigating control to find a trade-off between responsiveness and degradation. These form the lower level operating in continuous time. An upper level discrete event supervisor switches among several controllers of both types. This is designed „to mimic human intelligence“ [Tol05, p. 2]. It takes flight maps and the current location characteristics, such as the risk of shots being fired at the (military) helicopter, into account. It does not, however, control lifetime directly. Instead, it only selects an operating point that offers responsiveness as required and tries to keep degradation as low as possible. It is suggested to use health and usage monitoring systems for determining the current system state and to react thereon. An anomaly detection is implemented for the upper level discrete event supervisor, but direct feedback of condition monitoring results is not implemented. For the upper level a discrete system is used which only allows for a limited number of controllers among which selection is possible. Slight adaptations, which might be suitable to keep a specified lifetime but are barely noticeable by users, are not considered.

Similarly, mechanical stresses lead to cracks in the structure of airplanes. In [CRJ01; RC00; SR01] a flight controller is designed such that crack growth in the wings of the aircraft is limited. For this, a simulation model of aircraft dynamics is augmented with a crack growth model. However, instead of feeding back crack growth or size to the controller to form a closed loop, only flight dynamics are controlled. This way, crack growth may still be reduced.

In addition to structural damage to the wing, turbine engines were of concern. Wear on the blades is actively controlled in [BKB+04]. The controller is enabled to blow hot or cold air from the engine onto the outer engine case, thus forcing it to expand or contract. As the engine temperature changes, the clearance between blade tip and engine case might be reduced so far as to lead to wear at the blade tips. The resulting larger clearance of a worn engine, in turn, leads to lower engine efficiency and to increased gas temperatures. By including engine deterioration in active clearance control, efficiency can be increased and usable lifetime can be extended.

The acceleration schedule of an aircraft engine during take-off is optimized in [GCJ04]. It is shown that by including a model of thermal mechanical fatigue damage in calculation of the schedule, damage can be reduced without sacrificing performance.

[Guo01] describes the current state of life extending control at NASA for aircraft engines and problems that will be addressed in the then near future. Here, several current key areas are identified. Among these is the assumption that in the „intermediate term (5-10 years)“, i.e. approximately 2006 - 2011, life calculation is possible closer to the actual feedback loop. This is now the case with condition monitoring techniques being commonly used in several applications. With these, it is now possible to directly control remaining useful lifetime without the need for complex damage models, as was the case at the time life extending control was developed. [Guo01, Figure 2] shows a structure that is close to the operator controller module used within this thesis. However, no implementations or further realizations followed.

In [MMD+10], a six degrees of freedom model of an airplane and a model of a hydraulic actuator are combined. Controllers for flight dynamics, i.e. velocity and altitude, which include degradation, are designed. Then, an objective function, which is a combination of actuator degradation at final time of a given maneuver and output error as performance measurement, is minimized. A „reconfiguration supervisor“ optimizes altitude controller parameters at runtime to keep degradation below a pre-specified level. However, the two layers form a close interaction which makes it hard to allow for temporary user overrides.

1.4.2.3 Life extending control for power plants

Fossil fuel power plants are also interesting subjects for life extending control, as they are designed for a very low number of full operation cycles in a long period of usage. In [KHR97, p. 1101] it is stated that during 40 years of useful life, it is recommended to have only up to 100 cold starts and shutdowns. In this paper, an offline optimization is conducted to obtain a feed forward trajectory that is used to run the plants through transient operations. For this, a structural damage model for the main steam header is created and included in the optimization model.

[HR01b; KR00] build on this but add a more sophisticated control scheme. In addition to a feedforward control, switching between several feedback controllers is employed for adaptation of the power plant. Switching is done based on a supervisory controller, which also includes a fuzzy controller as basis of steady state feedforward computations. This way, the supervisory controller is coupled to system dynamics.

In a tutorial paper, [Ray01], the power plant is used as application example. While this is a good introduction to the field and to life extending control techniques, it does not add any significant modifications or enhancements.

1.4.2.4 Experimental validation of life extending control

The first experimental application for life extending control was published in [TRC95]. The test setup consists of two masses, of which the first one is connected to the testbed by a beam and to the other mass, which in turn is connected to a mechanical shaker. The beam coupling the two masses is weakened by drilling a hole into it. As cracks propagate through the remaining material, it eventually fails. The plant dynamics are modeled using basic dynamic modeling techniques whereas fatigue damage is modeled

in continuous form. A controller displaces the outer mass by means of the mechanical shaker. The controller output is determined using optimal control techniques to guide the system from an initial state to a new state. Then damage rate is included as constraint in the optimization. Performance is degraded, compared to an unconstrained case, but the damage rate descreased considerably. The beams failed with a lifetime that was close to that predicted by the damage model. Also both model and real beam life were increased by a factor of approximately 3.3 by using the constrained optimization results. [THR+98] builds upon this, but adds a closed loop controller for the displacement of the excited mass to allow for better tracking and disturbance rejection. The experiment has also been used in [ZR99], but the test setup was augmented with a third mass and the beams were made from several different materials. Also the tracking controller is of H_∞ design. In [ZRP00], a discrete event supervisory control is added. Ultrasonic measurements for crack detection are introduced into this experiment and into the control loop in [KGR+06]. In all these experiments, crack growth rate is reduced using life extending control.

With life extending control having shown its applicability to several academic examples, it started catching on in other fields as well.

1.4.2.5 Life extending control for other systems

In [LCM+03; LCM+06], a basic life extending control is applied to a boiler-turbine system. Another system is a wind excited antenna mast, [FFC06], for which a controller is designed to limit stress at nodes along the mast height while keeping actuation cost within limits.

The life of polishing pads in a chemical mechanical polishing system for semiconductor wafers is extended using life extending control in [Run01]. A model is used to predict pad conditioning, wafer-scale uniformity and feature-scale planarity. Optimization based on this model determines a „recipe“ for the polishing process.

Wind turbine degradation is controlled in [San07; San08]. A degradation model in continuous form is used in a model predictive control to control dynamic system behavior. This lower control loop directly acts on the wind turbine actuators, whereas an upper control loop specifies which system settings to use. The upper control loop interacts with the lower control loop by adapting objective function weights, and is thus directly linked and only offers complex means to override control loop interactions if a user wishes to do so. The idea of adaptation of system behavior by adapting model predictive control objective function weights for life extending control is also patented in [Ful07], but no rules to alter weights are disclosed.

1.4.3 Other approaches for implementation of reliability-adaptive systems

Another method for controlling a wind turbine is disclosed in [OZZ+13]. This scheme is based on estimating failure mode and remaining useful lifetime of a component of the turbine, then determining one or more control schemes which give different power output and degradation. Total power production or revenue generated is maximized by selecting an appropriate control scheme. However, for determining the control schemes and for the selection itself, no methods are given.

[SL15a] introduces an objective function that quantifies system degradation for an electric motor based on work done by the motor. It is used in [SL15b] to create closed loop control for the drive of a vehicle using model predictive control. To this end, the full Pareto front for objectives *degradation* and *closed loop performance* is computed. An operating point is chosen manually based on user-defined limits. While this general approach fulfills many prerequisites for reliability control as proposed in this thesis, an automatic selection of the working point is not part of the operating strategy.

In [SKK+13], reliability oriented online optimization for the operation of mechatronic systems is introduced. It is comprised of prognostics of remaining useful lifetime and an assessment whether reliability requirements are fulfilled. If this is not the case, degradation is lowered to increase lifetime. System functionality is split up into basic functionality and auxiliary functionality. Basic functionality *has* to be fulfilled, e.g. to pass governmental standards or to fulfill guaranteed system properties. Auxiliary functionality, on the other hand, is designed to e.g. increase user comfort and does not directly contribute to system functionality. Application of this auxiliary functionality is limited to lower degradation of the system. [SKK+13] gives electronic power steering as an application example. It is shown in simulations that different users with different usage scenarios will experience adaptations of auxiliary functionality such that desired system lifetime is fulfilled. During simulation, remaining system lifetime is controlled directly using online optimization that changes the quantity of available auxiliary functionality. This approach requires a degradation model to simulate system lifetime during optimization, which inhibits applicability to arbitrary systems. It also does not allow an adaptation of basic functions, which might ultimately be necessary to satisfy reliability requirements.

[DyL67] presents an adaptive reliability control system for power system control as early as 1967. The basic outline of the control system is comprised of three layers of control. The lowest layer is direct control, which directly interacts in real time with the power system components. The second level is optimizing control, which serves as direct control input, and the uppermost level is adaptive control, which changes power system behavior. It is planned to use a computer for numeric optimization to compute optimal settings of the power system components, but to also include human decision making in the control loop. Reliability of the power system is expressed inherently by means of minimum operating cost. This paper is a very early first publication of a novel research project and is thus mostly conceptual.

The whole operational strategy of a power plant is optimized in [GSA+02]. Goal is a minimization of operating cost, which is equivalent to a maximization of revenue. Operation of a power plant induces direct costs, e.g. fuel and operating personnel, and indirect costs such as degradation of the plant itself. By selecting proper operating points, the indirect costs can be lowered considerably while keeping energy production and thus revenue high. This is achieved using model predictive control for the operation of the plant that aims to minimize overall operating cost. Aging of the plant is expressed by means of crack growth at critical points. While the authors clearly show that using their approach, lifetime consumption can be lowered and revenue can be increased, this requires knowledge about cost of every aspect of system operation. Also the adaptation at runtime is achieved by directly optimizing the operation using model predictive control, which requires slow plant dynamics or huge computing power.

1.5 Objective of this thesis

Prior work has shown that it is possible to use reliability-control to increase the usable lifetime of a system. Controller performance and damage compete with one another and a suitable trade-off has to be found during operation. Current implementations of reliability-adaptive systems are either limited to reactions on discrete events, are highly system-specific, require complex degradation models, or mix dynamic controllers with reliability controllers, thus taking the risk of system instability. While the advantages of continuous adaptation to increase reliability are clear, application is severely limited by the existing methods.

The goal of this thesis is to develop a novel method for reliability control. A flexible solution is desired which can be used for arbitrary systems, yet is capable of controlling even complex systems. It needs to be able to continuously control system behavior such that reliability of the system is ensured by small variations in parameters over a long time. In addition, failures need to be taken into account by discrete reactions. While controlling system behavior according to current reliability, the control must be compliant to user demands. This includes short-term reactions as well as changed basic requirements, which in turn modify maintenance intervals.

The continuous adaptation must not introduce room for new vulnerabilities. For this reason, separation between behavior adaptation control and actual system control, i.e. dynamic controllers which directly interact with the physical system structure, is desired. This way, system control can be validated using established methods, making it robust against failures in the behavior control loop.

The continuous control loop must not be based on a first-principles damage model, as these require considerable knowledge about degradation processes and are prone to errors. The control method is tied into an existing framework for emergency reactions. As solution to these requirements, a behavior adaptation control loop that enhances the multi-level dependability concept introduced in section 1.4.1 is developed. To allow for continuous adaptation, self-optimization is employed. Applicability is proven experimentally.

1.6 Outline of this thesis

This thesis is divided into five chapters. The first chapter motivated the need for reliability control and gave a brief introduction to self-optimizing systems, on which the proposed control method is based. It also includes a discussion of prior works and introduces so-called reliability-adaptive systems.

A more elaborate introduction to self-optimization follows in chapter 2. It is focussed on model-based self-optimization, which is based on a priori computation of possible working points using multiobjective optimization and online selection among those found. This allows for easier setup and for using low computation power during operation. Also a method for the identification of dependability-related objectives, which can later on be used for control, is introduced.

The main contribution of this thesis, development of a novel method for reliability control, is outlined in chapter 3. It builds on a combination of methods from several fields, which are introduced in this chapter. At first, a health index as suitable reference variable

and the generation of a setpoint is discussed. Then the control loop is introduced. It is based on a two-stage controller with a behavior controller forming the inner loop and the actual reliability controller forming the outer loop. The behavior controller is prior art, but due to its importance for this thesis introduced in great detail in section 3.3.2. The whole behavior control loop is reduced to an abstract model in section 3.3.3 and a generic nonlinear degradation model is introduced in section 3.3.5. Parameters for these are estimated using Kalman filtering in section 3.3.6. Setup of the actual reliability controller based on model predictive control follows in section 3.3.7. For comparison with the desired setpoint, the actual value of the health index needs to be estimated. Prerequisites for an estimation using condition monitoring techniques are discussed in section 3.4. Controller stability and incorporation of the controller into existing aspects of self-optimization are discussed at the end of chapter 3.

An experimental validation of the proposed control loop follows in chapter 4. For several reasons, which are discussed in section 4.1, a clutch system as used in automotive applications, is used. The examined use-case and test rig setup are discussed next. For ease of experimental setup, a scaled-down clutch system was built up. Dependability-related objectives are identified, the clutch system is modeled and multiobjective optimization is used to find suitable working points. The whole control loop is setup and validated in individual stages. At first, lifetime experiments using a static working point are conducted. Then system complexity is increased by introducing behavior control to the system. Lifetime experiments show that reliability is not changed considerably. Reliability control is setup and again lifetime experiments are conducted. Results show that reliability becomes more predictably and that deviations in wearing behavior can be compensated and that changed requirements can be realized.

The thesis ends with a conclusion and an outlook in chapter 5. All experiment results are combined and the benefit for system operation is discussed. Ideas for further work are given as well. These include practical applications close to the clutch system example from chapter 4, improvements to the control loop itself and suggestions for other fields of research that go beyond the scope of this thesis.

2 Self-optimizing systems

Self-optimization is one way of implementing intelligence in mechatronic systems. It is based on the idea that system behavior can be expressed in terms of *objectives* that the system pursues and that behavior adaptation can be carried out by adapting objective priorities. This allows for system operation that is optimal with regard to the currently pursued objectives. It forms the basis of the reliability control loop developed within this thesis. For these reasons, a detailed introduction is essential.

2.1 Self-optimization

A system is self-optimizing if it is capable of adapting its behavior to the current situation, to changed external and internal requirements and to changes in the system itself by means of altering the objectives that it pursues [GRS09a, p. 6]. By initiating behavior adaptation through objectives of the system, flexibility is increased over other, similar behavior adaptation approaches. Optimization techniques can be applied to find operating points that are optimal with regard to the defined objectives. By limiting selection to these pre-computed working points, despite changed system behavior, optimality is not compromised. This approach is also denoted as *model-based self-optimization* and is introduced in more detail in section 2.1.2. A common base for all approaches to self-optimization is a cyclic system behavior adaptation process that is composed of three disctinct steps.

2.1.1 Cyclic behavior adaptation process

Self-optimizing mechatronic systems are formed by combination of *classical* mechatronic systems with an advanced signal processing unit. Whereas mechatronic systems have static behavior properties, self-optimizing systems are able to adapt their behavior to the current situation and to user demands. To do so, they select from system objectives or create new system objectives and then find a new working point that is optimal with regard to the currently pursued objectives. This is achieved by continuously cycling three individual steps [Gau04, p. 22], [GRS09a, p. 6], [DDD+14, p. 3], which are depicted in figure 2.1. The three steps are:

1. Analysis of current situation

   During this step, the system state and observations of the environment are taken into account. Observations can also be obtained indirectly by communicating with other systems. The state of the system may also include prior observations. A major aspect of this step is the evaluation of the degree of fulfillment of the pursued objectives.

2. Determination of objectives

   In the second step, new objectives of the system can be selected, adapted or gener-

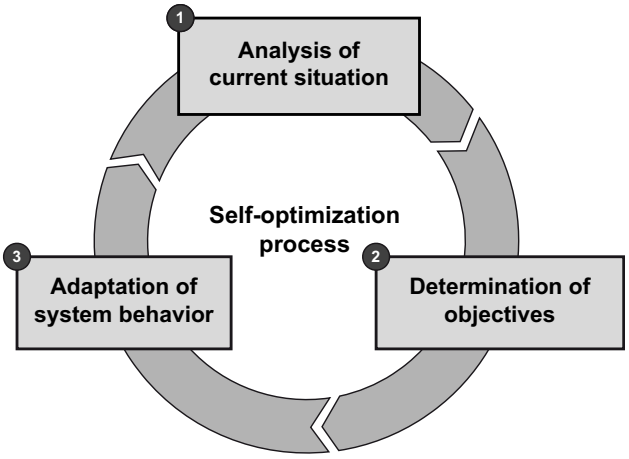

Figure 2.1: Cycle of the behavior adaptation of a self-optimizing mechatronic system [DDD+14, Figure 1.2]

ated from the system's set of objectives. A selection is possible if a finite number of discrete possible objectives exists. An adaptation is carried out if the system's objectives can be altered gradually. A generation of new objectives is performed if new objectives are created independently of known objectives.

3. Adaptation of system behavior
   The system behavior is adapted to account for changes that arise from the determination of new objectives. Changes to parameters of the system as well as changes to the system structure are possible. This action forms the feedback of the self-optimization cycle to the system.

By conducting these three steps, a system adapts its behavior from an initial state to a new state based on outer influences [GRS09a, p. 5]. The steps are repeated continuously, which enables the system to constantly adapt and to always operate in the best working point. With this cyclic behavior adaptation process, self-optimizing systems are capable of pursuing multiple conflicting objectives at the same time.

The cyclic adaptation is similar to that of a digital closed loop control. The three steps of self-optimization can also be regarded as obtaining new sensory and state information („Analysis of current situation“), computing a controller output accordingly („Determination of objectives“) and actuating the system („Adaptation of system behavior“). This similarity gives rise to the idea of viewing the behavior adaptation cycle as closed loop control a, to find a mathematical representation and to design actual controllers for it. This is detailed further in chapter 3.

2.1.1.1 „Analysis of current situation“ for reliability control

Controlling the reliability of intelligent mechatronic systems requires suitable means for all three steps of the self-optimization cycle. In step 1, „Analysis of current situation“, the current reliability of the system needs be determined. Within the scope of this thesis, this comes down to the health state of the system at hand, which can be determined using condition monitoring techniques. Condition monitoring techniques aim at estimating the health state of an individual system using measurement data that is available either during operation or at least during regular maintenance intervals long before failure. The

techniques can be subdivided into two main groups: *Model based condition monitoring* uses a dedicated model for simulating system degradation, whereas *data-driven condition monitoring* relies on a model that has been learned from prior data.

Methods from both groups are suitable for reliability control. In general, model based condition monitoring is advantageous, but requires considerable knowledge about system degradation and yields complex models, which are difficult to setup and validate and might require high computing power. If creating such a model is not possible, data-driven methods offer simpler model setup at the cost of requiring large amounts of learning data, which might need to be obtained using lifetime experiments.

A more detailed discussion of condition monitoring techniques follows in section 3.4.

2.1.2 Model-based self-optimization

Self-optimization can be implemented in two ways: As behavior-based self-optimization or as model-based self optimization [GRS09a]. These are not contradictory, but instead can also augment one another for different aspects of system operation, e.g. using model-based self-optimization for lower level behavior control and behavior-based self-optimization for upper level planning of objective priorities [Gau04, p. 47]. For reliability controlled system operation, model-based self-optimization has proven suitable. As the name suggests, a model of the system forms the basis of this method. The model is generally based on a first-principles model of dynamic system behavior, but could also be learned from data. It is then used to evaluate performance of the system for given system parameters in order to find suitable working points. For reliability control, the model of system performance is augmented with a reliability model, as shown in section 2.4.

Objective functions then quantify performance and reliability measure each as a single value. Using multiobjective optimization techniques, optimal trade-offs among conflicting objectives can be found, which leads to the so-called *Pareto front*. To each point on the Pareto front, parameters for system operation are known, which are commonly referred to as *Pareto set*. During operation, system behavior can then be adapted by changing the priorities of conflicting objectives, selecting an appropriate point from the Pareto front and setting the corresponding parameters from the Pareto set in the actual system, as depicted in figure 2.2. This way, optimization and operation are separated. Optimization is conducted offline during development of the system using a validated model. In certain situations, optimization during operation is advisable, e.g. if an updated model of system behavior is available. This is possible parallel to re-using old results, thus keeping the system operating. Results are Pareto front and set, which are stored for usage during operation. Advantages include low computational requirements during operation, fast reaction time and robustness during operation. A very detailed introduction to model-based self optimization can be found in [Mün12].

To allow for optimization including dependability related objectives, some measure for degradation needs to be included in the optimization model. Degradation is highly system specific and could necessitate models as complex as Finite Element Method models for crack growth. For these reasons, measures as simple and as universal as possible are desired. Section 2.4 gives a method to identify suitable objectives *without* fully setting up a degradation model, which can then be implemented in a multiobjective optimization problem.

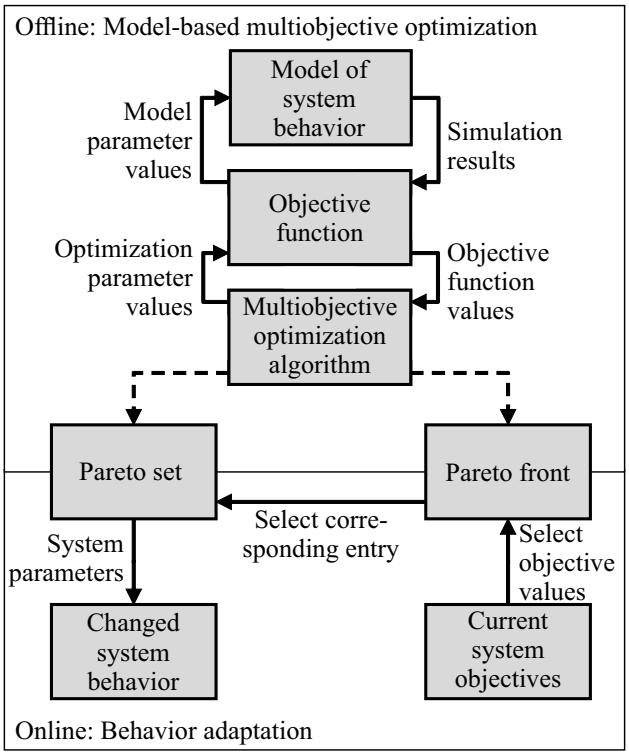

Figure 2.2: Information flow in model-based self optimization

The separation into optimization and selection among results also allows quick behavior adaptations with a small step size, whereas new optimization results become available on a much longer time frame, if at all. To allow such synchronisations of different time frames, the operator controller module was developed.

2.2 Information processing: Operator controller module

To cope with the complexity of computations and the constraints given by system control, a three layer layout was chosen for the information processing of self-optimizing systems. The current layout was first introduced in [HOG04], but it was based on prior work that is referenced therein. Later on, it was updated and changed in [FGM+07], [GRS09a], [GRS14], and other publications. The layout is shown in figure 2.3. The three layers are, based on [GRS09a]:

Controller

The controller ascertains that system dynamics are as desired. To this end, closed loop control is used to interact with the passive structure through sensors and actuators. Usually, digital controllers are employed which must satisfy real time constraints, i.e. a new controller output for the actuators must be available no later than a fixed time after sensory inputs were acquired. The fixed time step needs to be long enough to allow for computation of control laws, but also short enough to not impair system stability. This loop is present in all contemporary mechatronic systems. Behavior changes are rendered possible by changing controller parameters or control strategies. To this end, control switching techniques are implemented, which are initiated by the upper layer, the reflective operator.

Reflective Operator

The intermediate layer initiates controller adaptation and controls the switching itself. This is based on input from the upper layer, called cognitive operator, but also as quick reaction to unforeseen events in emergency situations. In most implementations, discrete configurations among which to switch exist. These are implemented in the configuration control. These parts need to run in real time as well. The reflective operator is not able to directly influence system behavior by means of actuators.

At the same time, the reflective operator is communicating with the cognitive operator. This communication is not bound to real time constraints, since the cognitive operator is working on a much slower time scale. It is necessary to synchronize these time scales. To this end, buffering or asynchronous data transfer are used, which are all part of the reflective operator. Main data that is communicated includes information about the current system state, i.e. (pre-processed) sensor information and the currently desired configuration.

Cognitive Operator

All complex tasks are embedded in the uppermost layer. Information data about the current system state is passed up from controller through reflective operator to cognitive operator. The data is evaluated to assess whether changes to the current

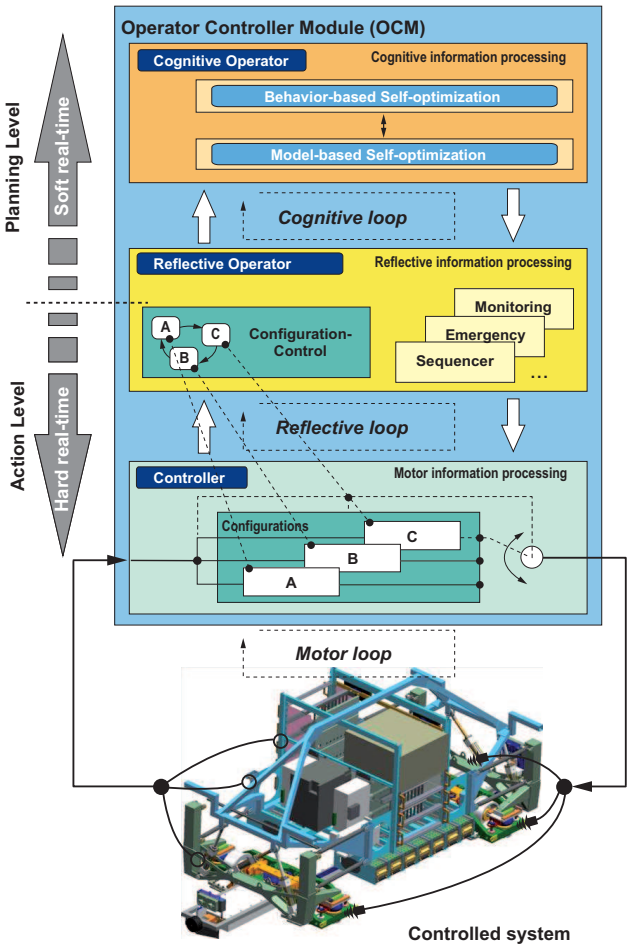

Figure 2.3: Operator controller module as structure for the information processing. According to [DDD+14, Figure 1.4]

operating point are required.

In model-based self-optimization, the model used for multiobjective optimization might not be valid anymore due to e.g. unexpected outer perturbations or inner changes in the system. To obtain new possible working points, a new optimization may need to be conducted. This time-consuming process is running in the cognitive operator. Results from this optimization are required at some point, but restrictions are not as hard as real time constraints on the lower layers. Instead, *soft real time*, i.e. discrete time with longer and possibly changing step size, is sufficient. Selection of the current working point is carried out in parallel.

Continuous reliability control builds on all three layers. While the main control loop is executed in the cognitive operator, it requires measurements from the controller and sets the operating point by means of the reflective operator.

Model-based self-optimization implements a selection of the current working point in the cognitive operator. The new desired working point is passed down to the reflective operator, which then adapts the controller. This separation into different layers makes it possible to adapt system behavior, but at the same time to guarantee that each possible working point is known to be safe and reliable beforehand. This process is based on multiobjective optimization results.

2.3 Multiobjective optimization

When talking about optimization, two different aspects need to be distinguished. In engineering, the term *optimization* is often used to describe a process of iterative improvement of a product. The optimization itself is conducted by changing parameters, re-designing parts or utilizing new materials. However, each iteration often comes with a change of requirements, making it hard to separate actual improvement of the product from merely adapting it to new requirements. In mathematics, on the other hand, the goal of optimization is to find the best solution to a given problem, i.e. *finding the lowest value of a given objective function.* Given a function $f : \boldsymbol{P} \rightarrow O$ with parameters $\boldsymbol{p} \in \boldsymbol{P} \subseteq \mathbb{R}^m$ and objective value $o = f(\boldsymbol{p}) \in O \subseteq \mathbb{R}$. The goal is then to find a solution $\boldsymbol{p}_{opt} \in \boldsymbol{P}$ such that $f\left(\boldsymbol{p}_{opt}\right) \leq f(\boldsymbol{p}) \forall\, \boldsymbol{p} \in \boldsymbol{P}$. The function f is commonly also referred to as *cost function*, *metric*, *energy function* or *fitness function*, but within this thesis the term *objective function* is preferred[7].

During development of a self-optimizing system, these two views need to be combined. At first, the engineering problem is formulated as an objective function, then the mathematical minimum of this function is found. The objective function may contain complex models based on e.g. for multibody dynamics, controllers, structural strength or fatigue life. Simulations of these models require long computation time, which imposes restrictions on optimization algorithms selection. Also if function evaluation requires system simulations, the gradient of the objective function cannot be determined analytically, further restricting optimization algorithm selection.

Multiobjective optimization builds on classical optimization but extends it to more than one objective function at once. For non-conflicting objectives, one common minimum can

[7]The only exception being the *cost function* in model predictive control, section 3.3.7. The differing name was chosen to differentiate it from system objectives and since cost function is commonly used in model predictive control.

be found despite multiple objectives. If instead two or more objectives are conflicting, it is not possible to find a common minimum, so the result is set of optimal compromises that are trade-offs among all objective functions.

Figure 2.4 shows an example for such a multiobjective optimization problem. The two objectives are simple functions:

$$O_1 = (p_1 - 20)^2 + (p_2 - 15)^2 ,$$

$$O_2 = (p_1 + 20)^2 + (p_2 + 15)^2 .$$

Arbitrary points $\boldsymbol{p} = [p_1, p_2]^T$ are mapped to form a cloud in objective function space, which apparently has restrictions so that low values cannot be reached. Since in optimization the minimum is desired, points *to the lower left* are the *best* points. From figure 2.4 it is obvious that no single optimal point exists. Instead, the *lower left* boundary of the cloud of possible objective function values needs to be found. To do so, the general multiobjective optimization problem

$$\min_{\boldsymbol{p}} \left(f_1 \left(\boldsymbol{p} \right), f_2 \left(\boldsymbol{p} \right), \ldots, f_n \left(\boldsymbol{p} \right) \right) \tag{2.1}$$

with parameters $\boldsymbol{p} \in \boldsymbol{P} \subseteq \mathbb{R}^m$ and objective functions[8] $\boldsymbol{f} = (f_1, \ldots, f_n)$ is now considered. An individual feasible working point $\hat{\boldsymbol{p}} \in \boldsymbol{P}$ is then called *Pareto optimal* if there is no other $\boldsymbol{p} \in \boldsymbol{P}$ such that $\boldsymbol{f}(\boldsymbol{p}) \leq \boldsymbol{f}(\hat{\boldsymbol{p}})$ [Ehr05, p. 24]. The set of all solutions $\hat{\boldsymbol{p}}$ is called the *Pareto set*. Corresponding points in objective space $\boldsymbol{f}(\hat{\boldsymbol{p}}) \in \boldsymbol{O} \subseteq \mathbb{R}^n$ are called *Pareto front*.

Numerical algorithms approximate Pareto front and set by a finite number of different possible working points. Each of these points can be tested against safety requirements. This can also be conducted automatically during optimization by including safety mea-

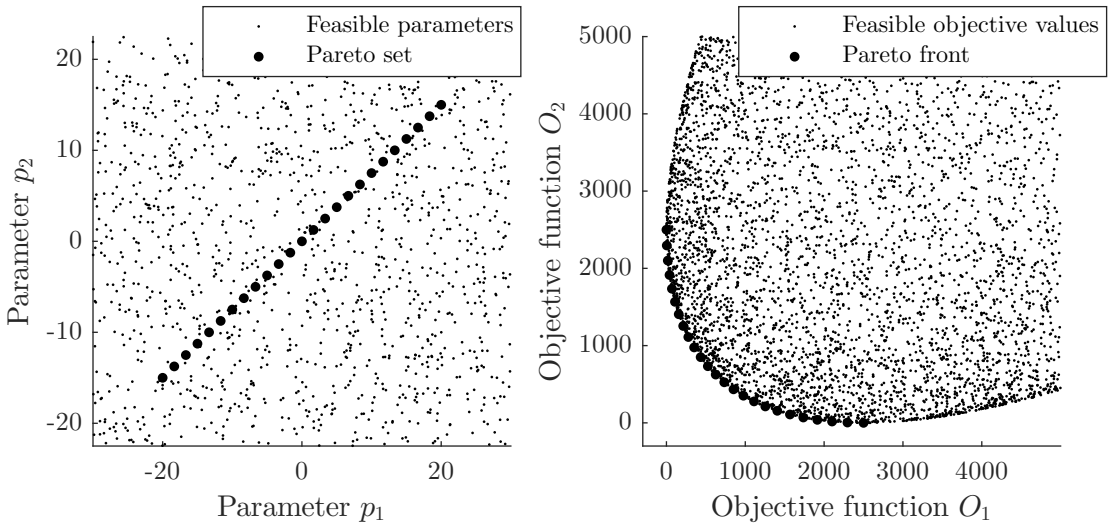

Figure 2.4: Parameters and objective values for possible points and for Pareto optimal points.

[8] In multiobjective optimization, strictly speaking only one objective *function* is utilized, but it has multiple dimensions for multiple objectives. Loosely speaking, each of these dimensions is referred to as an individual objective function.

sures, e.g. stability margins of controllers or maximum forces as constraints that have to be satisfied for a working point to be permitted into the final solution.

For the simple example with two objectives, the Pareto front is visible in figure 2.4 as boundary that forms *to the lower left*. It is optimal with regard to all objectives, but the decision which working point to use comes down to selecting a compromise between them. Selecting a working point that deliberately deviates from the Pareto front is not advised, as it is known to be non-optimal in at least one objective function, i.e. its objective function values are higher than necessary. Such a non-optimal working point is called *dominated* by at least one Pareto optimal working point. The Pareto optimal working points are called *dominating* points. They are found using multiobjective optimization techniques. From this, two goals can be deduced for solving a multiobjective optimization problem [Deb01, p. 22]:

1. Finding a set of solutions that approximates the Pareto front as close as possible,
2. Finding a set of solutions that is as diverse as possible.

The second goal is specific to multiobjective optimization problems and is of utter importance for model-based self-optimization. Since behavior adaptation of the system is based on selection of the current working point from all Pareto optimal solutions, many *different* working points need to be available. These two goals make handling multiobjective optimization problems more challenging than single objective optimization problems.

At this point, only a brief introduction is given. Interested readers are refered to dedicated books, e.g. [Deb01].

For mechatronic systems, the objective functions can become quite complex. First of all, performance as one objective is usually evaluated using a model of system dynamics, which can be given as a set of ordinary differential equations. These are solved using numerical integration schemes before the performance measure is computed from simulation results. Parameters in this case can be set points of controllers, controller gains or even desired state trajectories over time. This model is augmented with reliability-related objective functions.

The solution to such complex optimization problems can generally not be found analytically. A special case is controller design for linear systems, where problems with performance objectives only can oftentimes be solved using controller design techniques such as LQR (linear-quadratic regulator) [Föl94, p. 479] or H_∞ [LRH01, p. 999]. These approaches are limited in applicability to problems that include reliability. For this reason, formulating an optimization problem and solving this with general purpose methods is favored. Multiobjective optimization problems can either be reduced to one-dimensional optimization problems or be solved using dedicated multiobjective optimization algorithms.

2.3.1 Reducing multiobjective problem to single objective problem

A popular method for solving multiobjective optimization problem is to reduce all objectives to a single objective using metrics with changed parameterizations. Instead of solving one multiobjective problem, multiple single objective problems are solved. Several of these approaches exist. At this point, a small selection is presented to explain the key ideas; a more detailed overview can be found in [Ehr05; Mün12].

2.3.1.1 Weighted sum method

In this method, each objective function is assigned an individual weight. To find a point on the Pareto front, a vector $\boldsymbol{w} = [w_1, w_2, \ldots w_n]$ is created with n being the number of weights, which is equal to the number of objectives. Now, weights are varied and for each step j of the variation, objectives are combined as

$$S(\boldsymbol{p}) = \frac{1}{\sum_{j=1}^{n} w_j} \cdot \sum_{j=1}^{n} w_j \cdot f_j(\boldsymbol{p}).$$

Minimizing S yields one optimal solution. By varying weights, different compromises between objectives are found.

This is a simple and commonly used method, but it is not able to find the complete Pareto front for all problems. If the Pareto front is non-convex, no solutions in the region of non-convexity are found, leaving a gap between two parts of the Pareto front. It is difficult to estimate or even know the shape of the Pareto front beforehand. When adapting the working point by means of selection from the Pareto front, such a gap inhibits quasi-continuous adaptation.

2.3.1.2 ϵ-constraint method

A different approach that aims to overcome the shortcomings of the weighted sum method is minimizing just one single objective while keeping the remaining below pre-defined constraints. This way, the multiobjective optimization problem is reformulated to a single objective optimization problem with multiple constraints. The bounds imposed on the remaining objectives are named ϵ. By varying bounds, several optimal solutions can be found. This method is suitable for solving problems that yield a non-convex Pareto front, but selecting constraints ϵ is difficult.

Several other methods exist, but none of them is sufficiently versatile to solve complex multiobjective optimization problems satisfactory [Deb01, p. 75].

2.3.2 Direct multiobjective optimization algorithms

To overcome the shortcomings of reduction methods, direct multiobjective optimization algorithms have been developed. These aim to find the full Pareto front, not just a single point.

2.3.2.1 Evolutionary algorithms

Nature itself is the best optimization that exists: Through mutation and fitness evaluation, living beings evolve. Fit members of the population create more or fitter offspring than inferior members, which might die young or not find a mate. This way, the whole population is optimized. This process is mimicked by *evolutionary algorithms*, as introduced in [Deb01].

In a cyclic process, the population, starting with an initial population, is improved through several steps. The initial population is a number of feasible solutions which might have been computed offline beforehand. At first, population members are evaluated and

fitness values are assigned to each member. After checking whether optimality conditions are satisfied, the population members are reproduced, crossovers are created and new members are mutated. Then, the cycle starts again for the next generation.

During evaluation and fitness assignment, objective function values are computed, for which the user supplied objective function is executed. The evolutionary aspects come into play in the steps of reproduction, crossover creation and mutation.

During reproduction, good solutions are identified and duplicated, while bad solutions are removed from the population instead. This improves the average of all members, but does not increase diversity in the solution and does not find any better new members. The crossover operator then mixes the population by randomly exchanging parts of two different parent solutions between one another, thus creating offspring. In order to introduce entirely new solutions, and thus to increase diversity, existing population members are modified randomly without any external information from other members. These three steps form the basis of all evolutionary algorithms, but might be implemented differently.

A main prerequisite to finding a truly optimal solution is to find a *globally* optimal solution, as opposed to a *locally* optimal solution. Despite being suitable to solve a large number of problems, global optimality cannot be guaranteed for genetic algorithms. Instead, parameters, most importantly population size, must be selected suitably in order to find globally optimal solutions.

2.3.2.2 Box subdivision algorithms

An entirely different approach are box subdivision algorithms [SWO+13]. A large initial box is chosen in parameter or objective space, which is divided into smaller boxes iteratively. In each iteration, fitness of boxes is checked and boxes with dominated solutions are disposed, whereas boxes with non-dominated solutions are further subdivided. Evaluation of box fitness is conducted for several points in each box. Spread of evaluation points can be even or arbitrary, i.e. an individual Monte-Carlo simulation. These can be combined with a descent method, which takes the derivative of objective functions into account when selecting new parameters to evaluate. With these, efficiency can be increased and computation time can be decreased. For applications in engineering with complex objective functions that might include a model of system dynamics, generally the derivative is not known analytically. While methods such as algorithmic differentiation exist, they are not feasible for arbitrary objective functions with complex simulations. in these cases, algorithms that use function evaluations only have to be applied. Despite this shortcoming, computation times are sufficiently fast. An advantage is that all objective function evaluations for one iteration can be computed in parallel, making great use of parallel computers[9].

2.3.3 Optimal Control

Whereas optimization is concerned with finding the minimum of an arbitrary objective function, optimal control aims at guiding a dynamical system from an initial state to a

[9]For this research, the Paderborn Center for Parallel Computing provided computing time on the supercomputer Oculus.

desired state while optimizing one or multiple given objectives. This makes it necessary to take system dynamics into account. There are several ways to achieve this.

The basis is always formed by system dynamics described by a set of ordinary differential equations $\dot{\boldsymbol{x}}(t) = g(\boldsymbol{x}(t), \boldsymbol{u}(t))$ with system state $\boldsymbol{x}(t)$ and input $\boldsymbol{u}(t)$, $t \in [0, T]$. Then the goal is to find an input $\boldsymbol{u}(t)$, such that some optimality criteria $f(\boldsymbol{x}(t), \boldsymbol{u}(t))$ is optimized while fulfilling constraints $c(\boldsymbol{x}(t), \boldsymbol{u}(t))$. As the input $\boldsymbol{u}$ is a function over time, an optimal control problem has infinite dimension.

Several methods for solving optimal control problems exist. The *direct method* is popular for its simplicity of implementation and for its adaptability to a multitude of problems and models. To solve the optimal control problem, it is reduced to finite dimension, i.e. the system input is reduced to k discrete values $u_1(t_1), u_2(t_2), \ldots, u_k(t_k)$ with $0 \leq t_1 < t_2 < \ldots < t_k \leq T$. Then these individual values can be regarded as optimization parameters. The discrete states can either be considered as additional optimization parameters or internally obtained by system simulations. An advantage of simulating system dynamics is that the constraints originating from these are always fulfilled. However further constraints for the state, specifically for the final state at time T, need to be taken into account as optimization constraints.

A more comprehensive discussion of these different approaches can be found in [Obe08, pp. 16-24]. For the remainder of this thesis, a direct optimal control method was combined with a box subdivision optimization algorithm.

To allow an inclusion of reliability in an optimization problem, corresponding objectives need to be included.

2.4 Identifying dependabiliy-related objectives for continuous control

As shown in section 2.1.2 and figure 2.2, model-based self optimization is based on objective functions for the behavior adaptation process. For reliability control, suitable objective functions need to be found. Complexity of objective function formulation has direct impact on applicability and acceptance of the proposed reliability control loop. In order to reduce complexity, general formulations *without* degradation model are desired, as was discussed in section 1.5. In [MSS14], a dedicated method was introduced. It consists of five steps and yields an optimization problem that includes reliability. These steps are outlined in figure 2.5. The individual steps are:

1. Analyze system dependability
   At first, a system dependability model is setup to identify all critical or relevant failure modes, which may need to be addressed using reliability control. For each failure mode, the corresponding critical components are identified.
2. Identify load factors
   A *load factor* is a variable whose value originates from desired operation and which influences the lifetime of a critical component. In mechatronic systems, most critical components are wearing due to mechanical fatigue, abrasion, thermal processes or similar effects. As these are commonly limiting the lifetime of components, degradation is a common topic in research and for many components, degradation models models can be found from literature. Examples are [DIN ISO 281] for roller

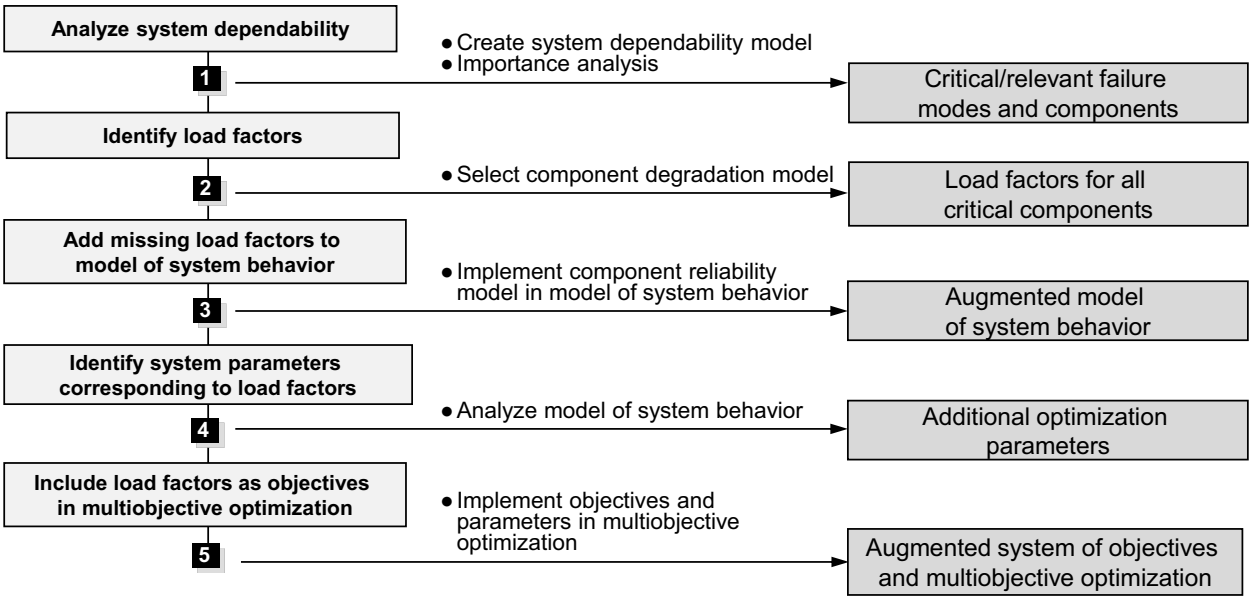

Figure 2.5: Phases and milestones for determining reliability-related objective functions. According to [MSS14, Figure 1].

bearings, [MKH02] for cutting tools, [RS12] for crack growth, or Cox regression models for arbitrary systems, where operating data from similar systems already exists [Cox72]. All of these degradation models require system-usage specific input parameter values such as forces, friction work or dissipated electrical energy. If the magnitude of input parameters is changed, degradation and reliability change. These input parameters are thus considered load factors of components.

3. Add missing load factors to model of system behavior
   Most often, several of the load factors are already included in the performance model, but if not, it needs to be augmented. This might necessitate the creation of new models for effects not considered before, such as heat dissipation.
4. Identify system parameters corresponding to load factors
   If changes in the existing optimization parameters do not reflect as changes in load factors, new optimization parameters need to be identified. This can be simple, e.g. by including controller gain values that can be changed online as variable parameters, but sometimes might also require changes in system structure.
5. Include load factors as objectives in multiobjective optimization
   By including the load factors as objective functions and the new optimization parameters in an existing multiobjective optimization problem, all required prerequisites for reliability control are fulfilled.

Once the optimization problem is fully formulated and solved using suitable algorithms, Pareto front and set are known and can be used for reliability control as well as for performance control.

3 Actively controlling the reliability

From the introduction in chapter 1 it is apparent that actively controlling the reliability of mechatronic systems might have great advantages for system operation. Closed loop control based on self-optimization is desired. This control loop has to be able to actively change system behavior at runtime by means of changing system parameters. A general outline of the desired reliability control loop is shown in figure 3.1. The system and its associated dynamic controllers work in hard real time and are connected as normal control loop. During operation, some information about the system state is obtained and passed to a reliability controller, which then computes desired system behavior and changes the dynamic controllers accordingly. Due to the inherently slow dynamics of system degradation, the reliability controller can work on a slower time scale in soft real time. To cope with the complexity that arises from controlling the system state by means of adapting system behavior, control loop setup is split up into several individual steps. These comprise finding a control variable, residual generation and controller design. At first, a suitable reference input needs to be defined. The controller works on the so-called *health index*[10].

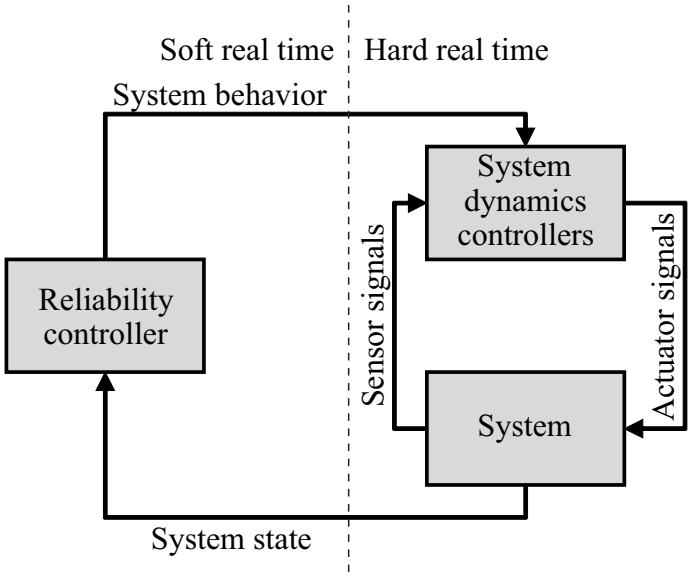

Figure 3.1: General outline of desired reliability control loop.

[10]In literature, *wear margin* can also be found, e.g. in [VDI 2895, p. 5]. Health index and wear margin are essentially synonymous, but wear margin misleadingly suggests limitation to abrasive wear.

3.1 Health index

Health index is a measure for the capability of a component of the system to sustain further degradation. It is defined as

$$\text{Health index} = 1 - \frac{\text{Prior degradation}}{\text{Sustainable degradation}}. \tag{3.1}$$

Degradation is accumulated damage, due to e.g. abrasive wear, crack growth, or chemical decomposition. *Sustainable degradation* is the amount of degradation that a component can sustain before failure. *Prior degradation* is the amount of degradation that the component has already accumulated at a given time.

The health index only captures the current component degradation state and does not include future predictions. It is completely independent from future usage and only depends on prior usage as far as this has influenced degradation. Its value ranges from 1 for a new component to 0 for a worn out component. By definition, the health index is 0 if a failure occurs. Components that show self-healing can have an increasing health index value during self-healing periods, but in general the health index is constantly falling due to ongoing degradation. The health index is specific to a component and also to each failure mode, i.e. a system that is composed of multiple components might have a large number of individual health indices, each of which might limit operating time.

The health index needs to be defined for each failure mode. Since the degradation of a component with regard to a failure mode requires energy for the actual damage accumulation, suitable measures can be based on dissipated energy. A general energy-based formulation of the health index is

$$\text{Health index} = 1 - \frac{\text{Accumulated dissipative energy}}{\text{Total dissipative energy before failure}}.$$

Both accumulated and total dissipative energy before failure need to be determined *for each component*. This poses two challenges: For some components, computing the accumulated dissipative energy is challenging, whereas for other components the total dissipative energy before failure changes during operation, which is difficult to determine.

For example, if a mechanical component fails due to crack growth, the energy that is required to elongate the crack needs to be dissipated in the mechanical component itself. Computing these *losses* proves difficult if component stiffness is high and damping is low. Also the total dissipative energy can change with operating conditions. This is the case for components where the influence of secondary effects on lifetime and that of the intended use of the component have approximately the same magnitude. An example is degradation of fuel cells, where the dissipated energy can be computed easily, but the total dissipative energy before failure increases during non-usage [KMS14]. This effect is also known as self-healing.

However, for some systems the dissipated energy serves as good indicator. Such a health index was used in [SL15a] for electric motors. The remaining useful lifetime RUL, which in this case is equivalent to health index HI, was defined as $RUL = 1 - \frac{W_{actual}}{W_{rated}}$. W_{actual} is the work done by the motor and $W_{rated} = \int_0^{t_{life}} P_{rated}\,\mathrm{d}\,t$ is the rated work. Nominal lifetime t_{life} and rated power P_{rated} are usually given by the manufacturer of a motor. Systems with self-healing are not taken into account.

To keep the reliability control approach developed in this thesis as universal as possible, a generalized health index according to (3.1) is assumed.

3.2 Setpoint generation

In order to setup a closed loop controller for reliability, desired reliability over time needs to be known. The time t is defined in units of system degradation time, i.e. it can either be actual time or it can be a system-specific measure such as usage cycles. Generally, the time value is only increased if the system is actually being used. If degradation occurs even though the system is not used at all, this needs to be taken into account separately in the setpoint generation.

Health index of a new system is named $HI_0 = HI(t_0)$. In most cases, $HI_0 = 1$. Health index at failure time t_f is, by definition, $HI(t_f) = 0$. However, precisely measuring the health index is impossible; instead, an estimated value $\widehat{HI}$ is used. It can safely be assumed that after maintenance, $\widehat{HI} = 1$. After a pre-defined desired lifetime t_{spec}, system failure is expected and desired. In ideal conditions, at this failure time the desired health index should be $HI_{des}(t_{spec}) = 0$ to avoid wasting system capability. Generally, some small deviation due to estimation tolerances cannot be avoided, i.e. $HI \neq \widehat{HI}$. This can become severe if system operation until $\widehat{HI} = 0$ is desired, but the actual health index HI is lower already and the system fails before $\widehat{HI} = 0$ is reached. To avoid such early failures, a safety margin has to be kept when defining the desired health index for the specified system lifetime $HI_{des,end} = HI_{des}(t_{spec})$ to be used as setpoint, thus $0 < HI_{des,end} < HI_0$.

Also the setpoint needs to be strictly monotonically falling during operation time, i.e. desired health index is lowered at all times[11]. If self-healing is evident in the system, health index after periods of rest needs to be correct accordingly. Raising the health index or even keeping the current value is not possible during operation without maintenance. Thus, in order to serve as setpoint, the desired health index HI_{des} needs to fulfill the following constraints:

$$\begin{aligned} HI_{des}(t = t_0) &= HI_0, \\ HI_{des}(t = t_{spec}) &= HI_{des,end}, \\ \frac{\partial HI_{des}(t)}{\partial t} &< 0. \end{aligned} \tag{3.2}$$

Generally, constant behavior and constant degradation are desired at all times. This leads to the setpoint being linearly falling over the desired system usage time, i.e.:

$$HI_{des}(t) = HI_0 - \frac{HI_0 - HI_{des,end}}{t_{spec}} \cdot t, \tag{3.3}$$

with t_{spec} being the specified lifetime.

A problem that remains with this definition is the choice of $HI_{des,end}$. As was shown, it

[11] *Lowered at all times* assumes no recovery during operation. This is true for most systems, but some might require compensation. This is especially true for electrochemical systems, as we have shown for fuel cells in [KMS14]. However, this special case is not considered in this thesis.

should be greater than 0 to avoid early failures, but if an overly large value is selected, sustainable degradation is wasted.

To cope with this problem, error of health index estimation needs to be determined. The easiest way to create a safety margin would be to increase t_{spec} over the *actual* desired lifetime, i.e. creating a safety margin on time. However, this would necessitate knowledge about the variance of time to failure, which is dependent on system wearing behavior, health index estimation and reliability controller performance. Most of these variances are unknown before actually setting up the system and the reliability controller.

From simple run-to-failure experiments with running health index estimation, time to failure of individual systems and the corresponding estimated health index can be obtained. This is shown in figure 3.2[12], upper left.

To determine reliability $R(t)$ of the system, the probability density function of the failure times t_f is evaluated, as shown in figure 3.2, lower left. If the reliability controller is not in action during these experiments, $R(t)$ is not representative of the controlled system, as changing the failure behavior is the main point of reliability control.

Similarly to evaluating failure times, the estimated health index can be evaluated as well. From this, a probability density function PDF $\left(\widehat{HI}|HI=0\right)$[13] for failure over estimated health index $\widehat{HI}$ is obtained. Reliability is given by the corresponding cumulative density function CDF $\left(\widehat{HI}|HI=0\right) = R\left(\widehat{HI}\right)$, shown in figure 3.2, upper right.

Since the reliability function R is the probability of the system being functional, $R\left(\widehat{HI}\right)$ can be evaluated to find a proper safety margin for $\widehat{HI}$. Assuming some specified reliabil-

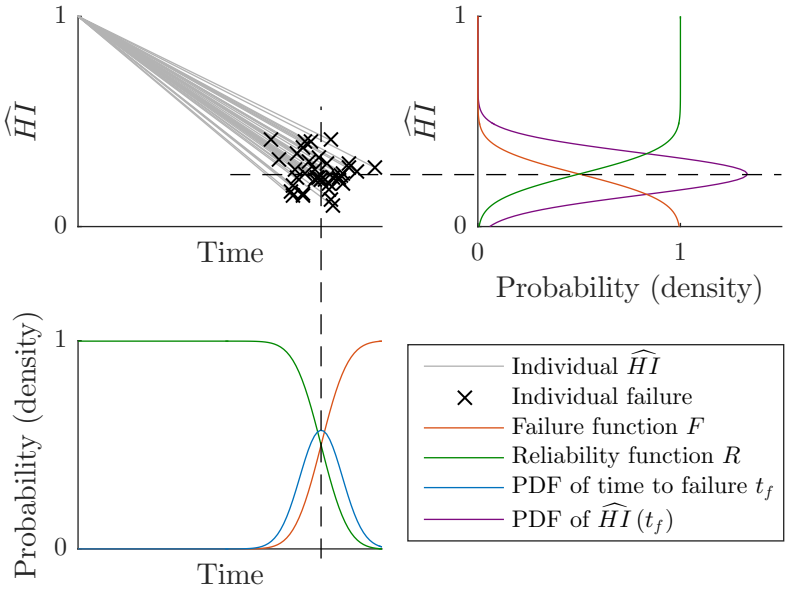

Figure 3.2: Illustration of estimated health index over time (arbitrary values) and resulting probability functions of values at system failure.

[12]The values in the diagram are arbitrary and are not to be taken as precise representation of a particular system. Different probability distributionas could be used as well.

[13]PDF $(A|B)$ denotes the probability density function of a stochastic variable A given *event* B.

ity R_{spec} at some specified time t_{spec}, the problem can be formulated as finding $HI_{des,end}$, such that

$$\text{CDF}\left(HI_{des,end}|HI = 0\right) = R_{spec}. \tag{3.4}$$

This can be found easily by inverting the known cumulative density function. Note that for the type of probability distributions of time to failure and estimated health index at failure, no assumptions have been made. While figure 3.2 shows normal distributions for both, Weibull or other distributions might be just as suitable and can be combined freely, depending on observed system failure and health index estimator behavior.

The resulting value $HI_{des,end}$ is close[14] to 0 and can be either positive or negative. Positive values indicate an estimation method that usually estimates a health index greater than the actual health index, i.e. $\widehat{HI} > HI$, whereas negative values can occur for an estimation method that gives pessimistic estimates, i.e. $HI > \widehat{HI}$. When operating a system with such pessimistic estimation until failure, $\widehat{HI} < 0$ shortly before failure. Usually, pessimistic estimates are preferred to avoid early failures. With these, maintenance is conducted earlier than necessary, thus preventing failure but wasting some sustainable degradation. Since reliability control also serves the purpose of avoiding early failures, it needs to be ensured that the reliability estimator works correctly despite returning negative values which wrongly indicate that degradation at failure is *greater than* sustainable degradation.

3.3 Setup of the control loop

To adapt system behavior in a closed loop control, a manipulating variable needs to be found. It must be capable of adapting system behavior by prioritizing system objectives of a self-optimizing mechatronic system. One way to implement this for systems with two objectives has been introduced by Krüger et al. in [KRK+13] with the so-called *α-parameterization*. It allows a choice among several objectives with abstract values, which are then mapped onto system objective values and in turn on system parameters by the so-called *s-transform*. After introduction of the α-parameterization, the control loop is setup as two stage closed loop control. The inner loop controls system behavior while the outer loop controls system reliability.

3.3.1 Behavior parameterization α as manipulating variable

The α-parameterization is a metric that allows a selection of the current working point from the Pareto front, as illustrated in figure 3.3. The basic idea is that to pair of objective function values, an individual α-value can be computed. This can be reversed as lookup to find a suitable point in objective space for a given α-value. To each point in objective space, i.e. each point from the Pareto front, a corresponding set of system parameters is given in the Pareto set. By setting these in the system, the behavior can be adapted.

Suggestions for the α-parameterization metric are made in [KRK+13, p. 3404], e.g. to use a Simplex-based method or to calculate the ratio among two objectives. The working

[14] If it is not, suitability of the health index estimation method should be questioned or desired reliability R_{spec} should be decreased.

point is determined from used α-value α_{use} by the s-transform[15] as system parameters $\boldsymbol{p} = s(\alpha_{use})$ which are then set in the system. The system behavior is adapted by the new parameters. It is evaluated to obtain actual current objective values $(f_{1,cur}, f_{2,cur})$. These are s^{-1}-transformed to determine the current value $\alpha_{cur} = s^{-1}(f_{1,cur}, f_{2,cur})$ of the α-parameterization. The current behavior parameterization value α_{cur} is fed back into the controller which then corrects α_{use} accordingly.

The system-specific α-parameterization requires system-specific s-transform and inverse s^{-1}-transform as well. While they are directly related, the inverse cannot be deduced from the forward transform without knowledge about Pareto front and set. With this knowledge, definition on the forward s-transform based on inverse s^{-1}-transform and Pareto solutions is possible.

To find system parameters based on used value α_{use} of the α-parameterization, $n \in \mathbb{N}$ points in Pareto front $(f_{1,1}, f_{2,1}), \ldots, (f_{1,n}, f_{2,n})$ and set $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n$ are assumed to be at hand. All corresponding α-values are calculated from the Pareto front using the inverse s^{-1}-transform, which is for now[16] assumed to be known as well:

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} = \begin{bmatrix} s^{-1}(f_{1,1}, f_{2,1}) \\ s^{-1}(f_{1,2}, f_{2,2}) \\ \vdots \\ s^{-1}(f_{1,n}, f_{2,n}) \end{bmatrix}.$$

Then, to find system parameters for an arbitrary α-parameterization α_{use}, the entry $k \in [1, n]$, $k \in \mathbb{N}$ closest to the currently desired value α_{use} is searched:

$$k_{use} = \arg\min_k (|\alpha_k - \alpha_{use}|). \tag{3.5}$$

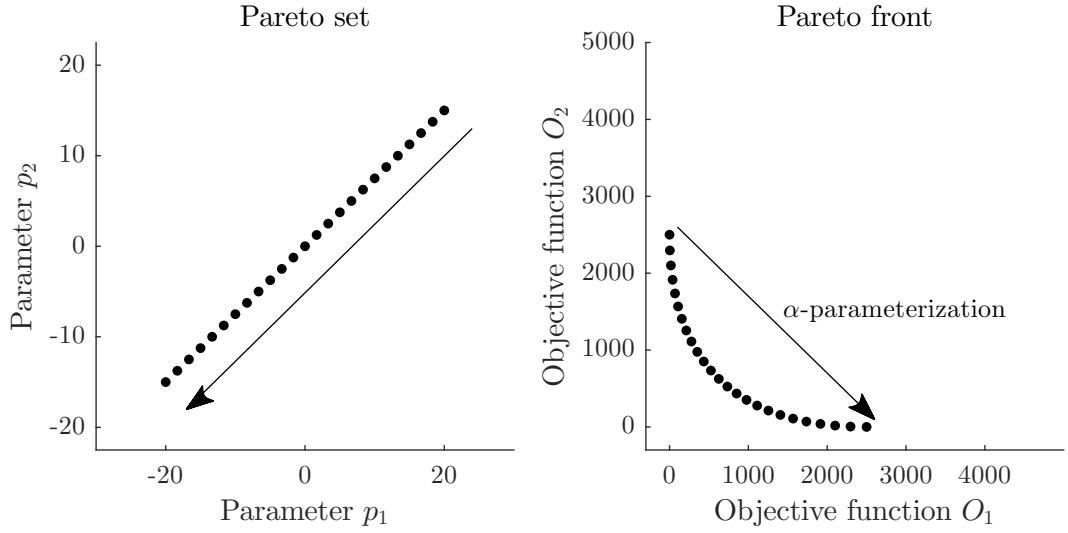

Figure 3.3: α-parameterization as selection metric in Pareto front and set, cf. figure 2.4.

[15] Laplace transform is also sometimes incorrectly called s-transform, but has no connection with the s-transform used in behavior control.

[16] The system-specific α-parameterization with corresponding inverse s^{-1}-transform is setup for an example system in section 4.6.2.

Next, system parameters for working point k_{use} are selected from the Pareto set.

In principle, one could implement (linear) interpolation between α-values and system parameters. This is not suggested though, since optimality cannot be guaranteed for intermediate points and stability of system controllers might be compromised, severely reducing safety of system operation. For pre-computed Pareto points, which are known to be optimal, stability, safety and other important attributes can either be ascertained as constraint during optimization or checked manually afterwards. More details are given in section 3.5.

Finding the s-transform is more involving and its definition is problem-specific. Any function that reduces two objective function values to one common value could be used. In the following, additional restrictions are assumed:

$$\alpha\left(\min\left(f_1\right)\right) = -1, \tag{3.6}$$

$$\alpha_{nom} = 0, \tag{3.7}$$

$$\alpha\left(\min\left(f_2\right)\right) = 1. \tag{3.8}$$

The limited value range makes it possible to setup a reliability controller as proposed in the following sections and the nominal working point $\alpha_{nom} = 0$ allows for robust parameter estimation of the underlying abstract model of behavior adaptation.

The α-parameterization allows to select a system working point based on multiobjective optimization results. These are based on a model of system behavior, which usually does not cover all aspects and has limited precision. For these reasons, deviations between optimization results and actual objective function values evaluated during operation cannot be avoided. To compensate for these deviations and to operate *close to* the desired working point, a behavior control loop based on the α-parameterization is employed.

3.3.2 Behavior controller

The behavior controller reduces the whole system adaptation and objective evaluation to interactions by means of α-parameterization, as shown in figure 3.4. It is implemented to control actual system behavior by evaluating the difference between the desired α-parameterization α_{des} and the current α-parameterization α_{cur}. While priorities of objectives can be selected at will, the system behavior does not necessarily reflect this immediately. On the one hand, an adaptation usually takes some time to take effect; on the other hand, the system model used for multiobjective optimization and the real system might deviate from one another, thus leading to differences between desired

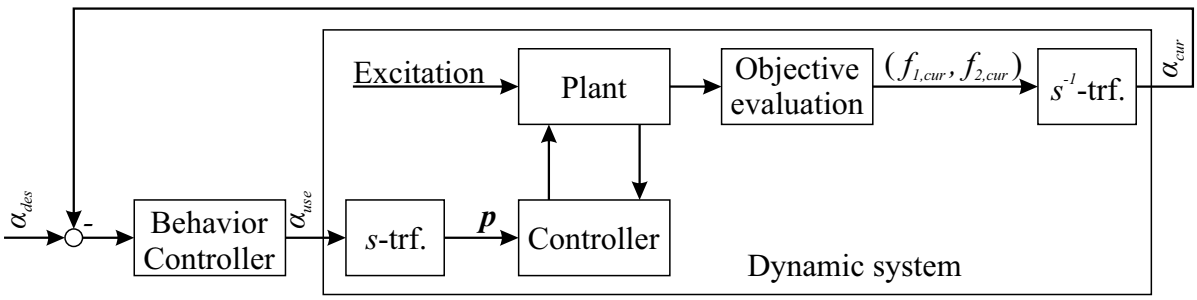

Figure 3.4: Basic outline of behavior control loop.

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 [KRK+13], [Krü14, p. 130]), colloquially also called *Pareto controller* or *behavior controller*. The purpose of this controller is to ascertain that a pre-selected system behavior is actually being achieved, despite of perturbations to the system or deviations between system and optimization results.

[KRK+13] introduces a structure for the behavior controller that is based on a linear system description with open loop feed forward control and closed loop feedback control. Open loop control allows for quick changes in working point while closed loop compensates deviations between desired and current system behavior. The full behavior adaptation control loop is shown in figure 3.5. For closed loop, a PI controller is chosen based on analysation of the linear system description, which is called abstract model. The combined feed forward and feedback controller output is denoted as α_{use} and determines the adapted system behavior parameterization. To allow for such control, two different time scales are assumed: The actual dynamic system operates on a fast time scale with highly dynamic controllers whereas behavior adaptation is slow. Its sample time is mainly based on objective function evaluations, which are required to determine

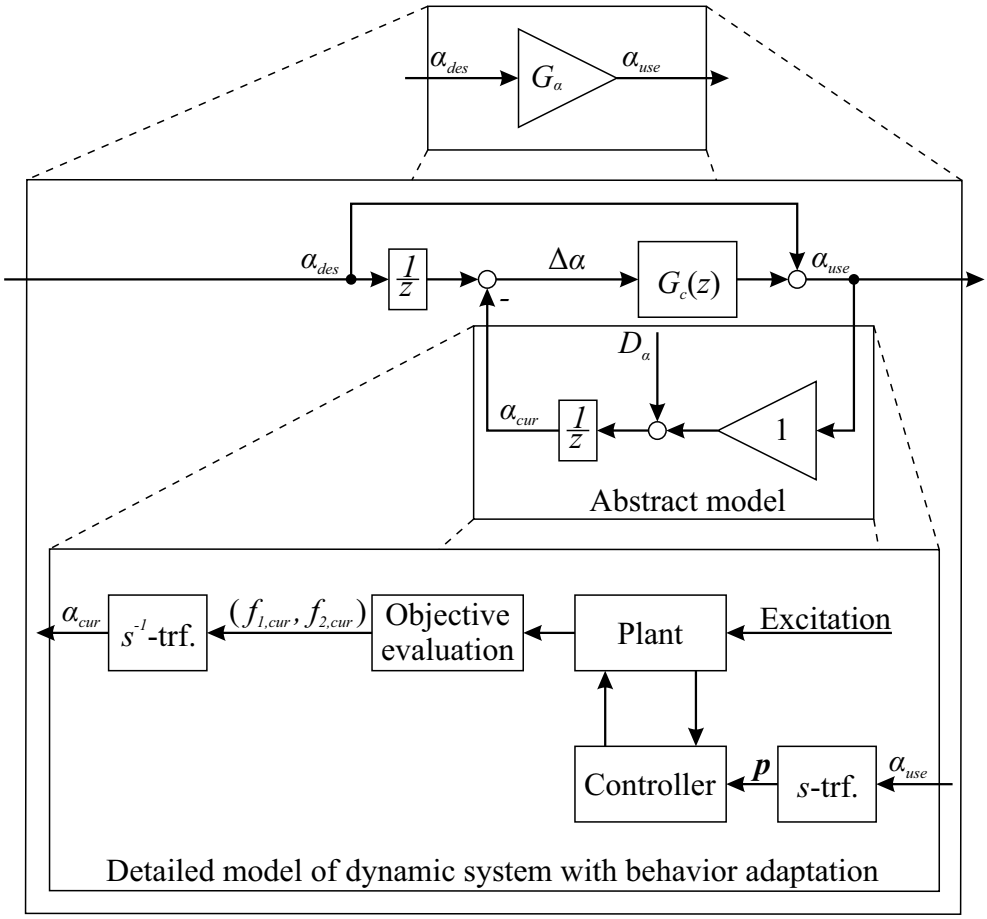

Figure 3.5: Behavior control loop according to [KRK+13, Figures 2, 3][17]. Top: Further reduced model of control loop. Bottom: detailed model of dynamic system.

[17] A sign error at the first sum block in the original of figure 3.5, [KRK+13, Figure 3], has been corrected in this aggregation. Also to obtain clearer equivalence between detailed model and abstract model, some names have been changed.

the currently achieved objective values and from these the current behavior parameterization α_{cur}. To setup a reliability controller, the behavior parameterization control loop is reduced even further.

3.3.3 Reduced model of behavior adaptation

In [KRK+13], an abstract model for the behavior control loop is proposed. In figure 3.5, the abstract model of system dynamics represents the whole process of behavior adaptation including dynamical system behavior, which is shown in the lower part. This starts with the s-transform to determine system parameters $\boldsymbol{p}$ from desired behavior parameterization α_{des}. Then these are set in the controller, which directly interacts with the plant or system to control system dynamics. Outer perturbations occur in the form of excitations to system dynamics, which are generally not or only approximately known. During operation of the system, objectives are evaluated and current objective values $(f_{1,cur}, f_{2,cur})$ are computed. These are fed into the inverse s^{-1}-transform to determine the current value of the behavior parameterization α_{cur}.

As shown in figure 3.5, the detailed model of dynamic system with behavior adaptation is reduced to an ideal transfer function 1, a perturbation $D_\alpha(z)$ and a unit delay $\frac{1}{z}$, where z indicates the z-transform for discrete time. Step size in discrete time is assumed to be 1 *behavior adaptation cycle*, this way the time unit is not coupled to actual time but instead based on the slow behavior adaptation cycle. The additional perturbation $D_\alpha(z)$ is introduced to take deviations due to e.g. outer excitations or deviations between simulation model and actual plant into account. The unit delay $\frac{1}{z}$ represents the time required for objective function evaluation. For the controller itself, a basic PI controller represented by its discrete time transfer function is used in accordance with [KRK+13]:

$$G_c(z) = \frac{K_p(z-1) + t_s \cdot K_i \cdot z}{z-1},$$

with step size $t_s = 1\,\text{cycle}$ and parameters K_p, K_i. The abstract model of the behavior control loop is shown in figure 3.5, center.

Also illustrated in figure 3.5, upper part, is the desired final reduced model of the behavior adaptation control loop. This model is a single transfer function G_α. To find this further reduced abstract model, the block diagram from figure 3.5, center, is recombined into a single equation, which gives:

$$\begin{aligned}
\alpha_{use} &= \alpha_{des} + G_c(z) \cdot \left(\frac{1}{z} \cdot \alpha_{des} - \alpha_{cur}\right) \\
\alpha_{use} &= \alpha_{des} + G_c(z) \cdot \left(\frac{1}{z} \cdot \alpha_{des} - \frac{1}{z} \cdot (D_\alpha(z) + 1 \cdot \alpha_{use})\right) \\
&= \alpha_{des} + G_c(z) \cdot \left(\frac{1}{z} \cdot \alpha_{des} - \frac{1}{z} \cdot D_\alpha(z) - \frac{1}{z} \cdot \alpha_{use}\right) \\
&\Leftrightarrow \alpha_{use} \cdot \frac{1}{G_c(z)} + \frac{1}{z} \cdot \alpha_{use} = \alpha_{des} \cdot \frac{1}{G_c(z)} + \frac{1}{z} \cdot \alpha_{des} - \frac{1}{z} \cdot D_\alpha(z).
\end{aligned}$$

It is now assumed that the model used for objective function evaluations during multi-objective optimization represents actual system behavior perfectly and that no unknown outer excitations occur. This means that in the abstract model, perturbations $D_\alpha(z) = 0$.

This yields the transfer function for the whole behavior control loop:

$$\alpha_{use} \cdot \frac{1}{G_c(z)} + \frac{1}{z} \cdot \alpha_{use} = \alpha_{des} \cdot \frac{1}{G_c(z)} + \frac{1}{z} \cdot \alpha_{des}$$
$$\Rightarrow G_\alpha(z) := \frac{\alpha_{use}}{\alpha_{des}} = \frac{\frac{1}{G_c(z)} + \frac{1}{z}}{\frac{1}{G_c(z)} + \frac{1}{z}} = 1. \tag{3.9}$$

This transfer function now fully models the behavior adaptation control loop. The assumptions that lead to perturbations $D_\alpha(z) = 0$ are quite strong, but since the reliability controller can be considerably slower than the behavior controller, and then behavior controller dynamics become negligible, still holds.

3.3.4 Outline of reliability controller

The behavior control itself does not necessarily improve reliability of the system. In addition, an outer loop that is dedicated to controlling system reliability is required. This reliability controller also uses the α-parameterization to change system behavior, but its output serves as input to the behavior controller, in effect creating two cascaded control loops as shown in figure 3.6. The desired α-parameterization value α_{des} is given as sum of the reliability controller output α_C and a perturbation α_U, which is a user input and allows a user to override the reliability controller output. This serves the purpose of making the system responsive to user demands despite reliability control selecting the working point. The upper stage changes the desired system behavior by altering α_C based on current measured health index $\widetilde{HI}$ and desired health index HI_{des}. Both stages ultimately interact with the system by means of the same manipulating variable α_{use}.

The dynamic system now has two outputs: At first, current α-parameterization value α_{cur} is computed as introduced before. Secondly, the current health index $\widetilde{HI}$ as measure of current system reliability is determined. As shown, the relationship of α_{des} to α_{use} can be reduced to transfer function $G_\alpha(z)$.

As controller, model predictive control is used. In model predictive control, a prediction of future system performance based on future system input is used to determine an optimal system input, of which the first value is then applied to the system. This is done continuously: during each time step, the current system state is evaluated and future performance is simulated. More details follow in section 3.3.7. As the name suggests, a model of the system is required for this control approach. Since health index is the performance measure of the reliability controller, this model needs to map behavior parameterization α_{use} to health index HI. For most systems, such a degradation model cannot be found based on first principles, as it would require lengthy lifetime experiments

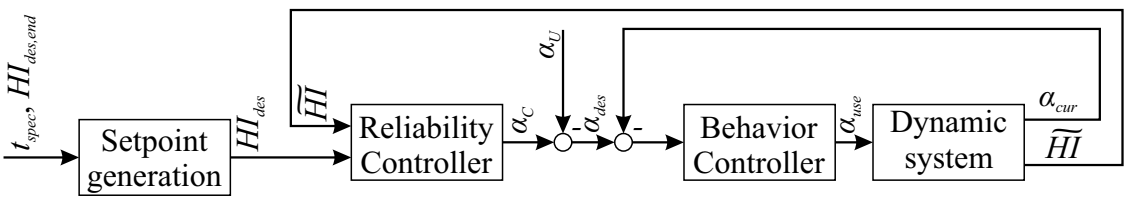

Figure 3.6: Interaction of behavior adaptation and reliability control loop

in addition to full knowledge of all influencing factors. To cope with this discrepancy, a generic degradation model is employed which is based on parameters estimated during operation.

3.3.5 Generic degradation model

The behavior controller and its output α_{use} are invariant to current health index. The only possible way to model relationship from behavior parameterization to health index is to integrate the health index decrease ΔHI, as shown in figure 3.7. Here, a discrete-time integrator $\frac{z}{z-1}$ is used. Additionally, the delay due to health index estimation is taken into account by a unit delay $\frac{1}{z}$ and additional deviations are introduced as unknwon perturbation D_R. The whole reliability control loop is setup around these health index dynamics as shown in figure 3.7.

To find the model for use in model predictive control, a general function is assumed that maps system parameters $\boldsymbol{p}(t_k)$ for the current time step t_k to a difference in health index $\Delta HI(t_k) = \Delta HI(\boldsymbol{p})$. Current system parameters are given by the s-transform. Computing $\Delta HI(t_k)$ is difficult since the underlying function is unknown. Instead, an approximation that can be parameterized during operation is desired.

The additional input α_U represents changed system behavior due to user intervention, D_R is a disturbance on the reliability of the system and α_C is the reliability controller output. With these, also included in 3.7, one finds:

$$HI = \frac{1}{z} \cdot \frac{z}{z-1} \cdot \left(D_R(z) + \Delta HI\left(s\left(G_\alpha\left(\alpha_C - \alpha_U\right)\right)\right)\right),$$

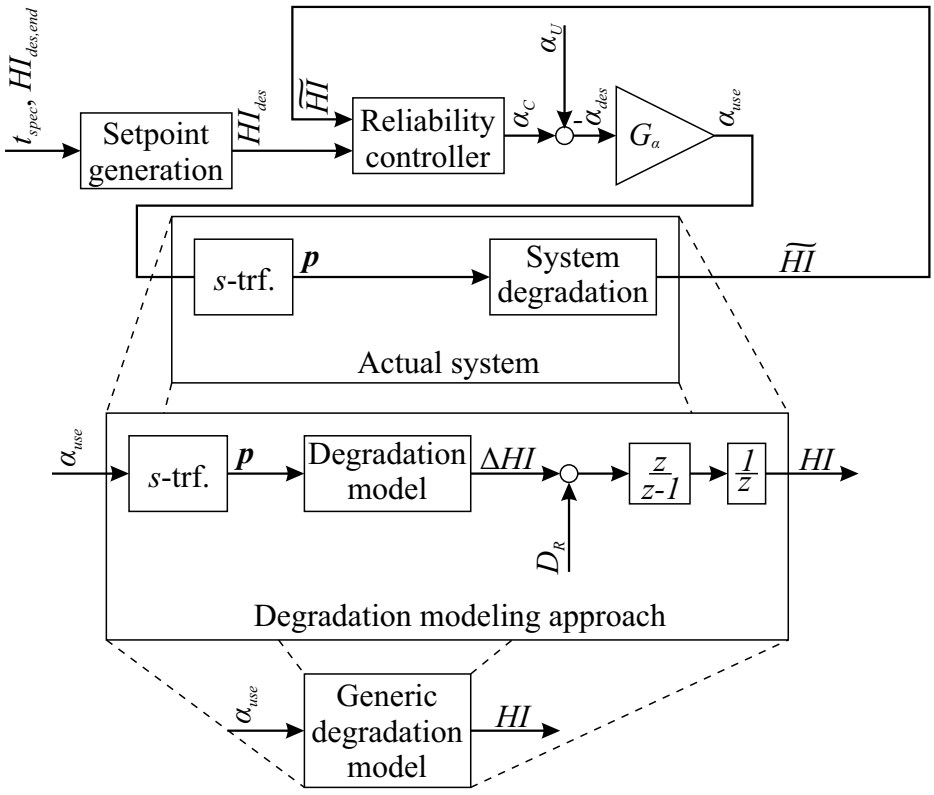

Figure 3.7: Generic degradation model in reliability control loop

where z again is the discrete time z-transform with constant step size 1 cycle. Assuming no perturbations, i.e. $D_R(z) = 0$, $\alpha_U = 0$ and $G_\alpha = 1$ gives:

$$\begin{aligned} HI &= \frac{1}{z} \cdot \frac{z}{z-1} \cdot (\Delta HI\,(s\,(G_\alpha\,(\alpha_C)))) \\ &= \frac{1}{z} \cdot \frac{z}{z-1} \cdot (\Delta HI\,(s\,(\alpha_{use}))) \,. \end{aligned} \tag{3.10}$$

This is essentially an integration and unit delay of the nonlinear function $\Delta HI\,(s\,(\alpha_{use}))$, which describes behavior dependent system degradation for the current time step. With this model, a reduced model for the behavior control loop and for the degradation behavior of the system is obtained.

Since step size in discrete time is set to 1 cycle, it is system-independent. Instead of using the z-transform-based system representation from (3.10), the constant step size can be used to reformulate from discrete time transfer function to difference equation. This results in:

$$\begin{aligned} HI\,(t_k) &= HI\,(t_{k-1}) + \Delta HI\,(s\,(\alpha_{use}\,(t_k))) \\ &= HI\,(t_{k-1}) + \Delta HI\,(t_k) \,. \end{aligned} \tag{3.11}$$

To find a generic degradation model, a linear ansatz for the combination of degradation model $\Delta HI\,(\boldsymbol{p}\,(t_k))$ and s-transformation $\boldsymbol{p}\,(t_k) = s\,(\alpha_{use}\,(t_k))$ is used. Separation of these two is not desired since in the control structure, cf. figure 3.7, they are directly connected and no further parts of the reliability control structure are influenced by the intermediary system parameters $\boldsymbol{p}$. Also the parameters are system-specific and can be arbitrary, whereas the behavior parameterization and health index are variables directly used for reliability control. The ansatz selected is:

$$\Delta HI\,(t_k) := \Delta HI_{lin} \cdot \alpha_{use}\,(t_k) + \Delta HI_{nominal} + \Delta HI_{offset}\,(t_k) \tag{3.12}$$

with some unknown linearity factor ΔHI_{lin}, an unknown constant $\Delta HI_{nominal}$ and an unknown time-variant correction offset $\Delta HI_{offset}\,(t_k)$. The constant term $\Delta HI_{nominal}$ determines the degradation rate at nominal system operation $\alpha_{use} = 0$, cf. (3.7). The multiplicative term ΔHI_{lin} is charateristic for the amount of influence that changed system parameters have on system degradation. The third term $\Delta HI_{offset}\,(t_k)$ is a constant offset with the same effect as the nominal degradation rate $\Delta HI_{nominal}$, but unknown and possibly varying magnitude. To allow simulation of the degradation model, these parameters need to be determined. This ansatz is also illustrated in figure 3.8.

The nominal degradation rate is based on desired and idealized degradation behavior for a perfect system operating at nominal conditions. This is given in continuous time by (3.3) as

$$HI_{des}\,(t) = HI_0 - \frac{HI_0 - HI_{des,end}}{t_{spec}} \cdot t,$$

with t_{spec} being the specified lifetime of the system. It is now assumed that at initial time t_0 the health index is equal to one, i.e. $HI_0 = HI\,(t_0) = 1$ and that at failure time,

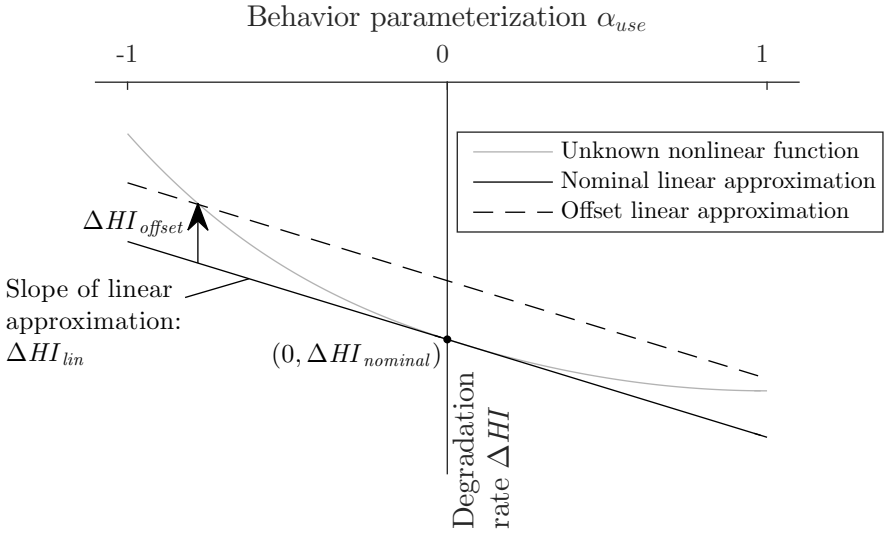

Figure 3.8: Approximation of nonlinear relationship from α_{use} to ΔHI.

which is also the desired end time, the health index is zero, i.e. $HI_{des,end} = 0$:

$$HI\left(t\right) = \overbrace{HI\left(t_0\right)}^{=1} - \frac{\overbrace{HI\left(t_0\right) - HI_{des,end}}^{=1}}{t_{spec}} \cdot t$$
$$\Leftrightarrow HI\left(t\right) - 1 = -\frac{1}{t_{spec}} \cdot t.$$

Finally, derivation and multiplication with the constant time step size t_s gives the time-invariant nominal degradation of the system for each adaptation cycle:

$$\Delta HI_{nominal} := \left(\frac{\partial}{\partial t} HI\left(t\right) - 0\right) \cdot t_s = -\frac{t_s}{t_{spec}} \cdot 1. \tag{3.13}$$

In addition to the constant term, the linear term ΔHI_{lin} needs to be defined. $\Delta HI_{nominal}$ is used as starting point for its value, but an additional multiplicative fault v is introduced:

$$\Delta HI_{lin} = -\frac{t_s}{t_{spec}} \cdot v. \tag{3.14}$$

The actual value of v is unkown. For a value of $v = 1$, the behavior parameterization value range $\alpha_{use} = -1 \ldots 1$ could change system operation from a complete stall of degradation to a doubled degradation rate. This limits the possible range of v to $0 < v < 1$. If additional information is available about the working points and the effect they have on system lifetime, this can also be used to estimate the value of v.

Combination of (3.11), (3.12) and parameter equations (3.13) and (3.14) yields the complete generic degradation model:

$$\begin{aligned} HI\left(t_k\right) &= HI\left(t_{k-1}\right) - \Delta HI_{lin} \cdot \alpha_{use}\left(t_k\right) + \Delta HI_{nominal} + \Delta HI_{offset}\left(t_k\right) \\ &= HI\left(t_{k-1}\right) - \frac{t_s}{t_{spec}} \cdot v \cdot \alpha_{use}\left(t_k\right) - \frac{t_s}{t_{spec}} + \Delta HI_{offset}\left(t_k\right), \end{aligned} \tag{3.15}$$

which fully models system degradation dynamics for a given behavior parameterization $\alpha_{use}(t_k)$. It also includes unknown parameters $\Delta HI_{offset}(t_k)$ and v as linear approximation of unknown nonlinearities and unknown parameter value errors. These need to be estimated.

3.3.6 Parameter estimation

During operation, the linear system model needs to be adapted to adequately represent nonlinear system behavior, effectively forming a linearized system model that is valid for small deviations from the current system state. For this, a parameter estimating Kalman Filter is employed. It is now also assumed that actual health index HI cannot be measured. Instead, all measurements are noisy and have some uncertainty. These noisy measurements are denoted as $\widetilde{HI}$. From the noisy measurements the actual value is estimated, which is denoted by $\widehat{HI}$.

Parameter estimation can be used to detect faults in systems. For this, a fault model is required. Generally, two kinds of faults are distinguished: Additive faults and multiplicative faults [Ise06, p. 63]. To distinguish one from the other, system dynamics need to be excited sufficiently. In case of reliability control, this would mean varying the operating point frequently. Doing so would yield deliberately fluctuating system behavior, which is not desired and considered unacceptable. For this reason, an additional and a multiplicative fault at the same time are difficult to identify and to find parameter values for.

In the degradation model (3.15), two deviations that can be regarded as *faults* with corresponding parameters are included: the influence of behavior parameterization on system degradation changes with multiplicative parameter v and a constant offset $\Delta HI_{offset}(t_k)$. As was already discussed, both faults cannot be estimated during operation. Instead, a constant value for v is set such that parameter estimation can be limited to $\Delta HI_{offset}(t_k)$. To find a suitable value[18] for v, the influence of changed behavior parameterization α_{use} on system degradation needs to be known.

To this end, $\Delta HI_{offset} = 0$ is assumed and (3.12) is rewritten as

$$\begin{aligned}
\Delta HI(t_k) &= -\frac{t_s}{t_{spec}} \cdot v \cdot \alpha_{use}(t_k) - \frac{t_s}{t_{spec}} + \underbrace{\Delta HI_{offset}}_{=0} \\
&= -\frac{t_s}{t_{spec}} \cdot v \cdot \alpha_{use}(t_k) - \frac{t_s}{t_{spec}} \\
\Leftrightarrow v &= -\frac{t_{spec}}{t_s} \cdot \left(\Delta HI(t_k) + \frac{t_s}{t_{spec}} \right) \cdot \frac{1}{\alpha_{use}(t_k)}
\end{aligned}$$

v is assumed to be time-invariant, but apparently it is dependent on $\Delta HI(t_k)$ and $\alpha_{use}(t_k)$. During operation or in experiments, $\Delta HI(t_k)$ changes with $\alpha_{use}(t_k)$ can it be approximated from measurement results. This leads to:

$$v(\alpha_{use}) = -\frac{t_{spec}}{t_s} \cdot \left(\Delta HI(\alpha_{use}) + \frac{t_s}{t_{spec}} \right) \cdot \frac{1}{\alpha_{use}} \tag{3.16}$$

[18] Finding the *correct* value for v is not possible, since it changes with desired parameterization α_{use}. Instead of including this nonlinearity in the model, the estimator corrects ΔHI_{offset} to compensate.

To determine a single value for v, nominal system behavior is evaluated. This is, by definition, $\alpha_{use} = \alpha_{nom} = 0$ (cf. section 3.3.2, (3.7)), for which (3.16) is not defined. Nevertheless, to be able to find a value, behavior parameterizations *close to zero* are evaluated and averaged. For this, a small deviation value ϵ is introduced:

$$\begin{aligned} v^+ &:= v\left(\alpha_{use} = 0 + \epsilon\right), \\ v^- &:= v\left(\alpha_{use} = 0 - \epsilon\right), \\ v &= \frac{v^+ + v^-}{2}. \end{aligned} \tag{3.17}$$

To find value v, running experiments with the actual system, evaluating simulations of a well-suited degradation model[19] or an educated guess is necessary. Absolute precision is not required, as deviations are compensated by an adaptation of the remaining additive fault $\Delta HI_{offset}(t_k)$.

To estimate a value for this parameter, state estimation techniques are employed. A popular method is Kalman filtering, which aims to estimate the state of a linear time-invariant system. A good introduction to the basic concept can be found in [Ath11]. Additional noise inputs are assumed that act as perturbations on the system. *Process noise* acts on the state itself, via some input gain matrix, and *sensor noise* on the sensor output. These two cannot be distinguished, but estimations about their stochastic properties form the basis of Kalman filter parameterization. The Kalman filter itself is closely related to a general observer. The system model is excited by the same input as the system itself, the difference between system output and model output is used to correct the model state in a closed loop. The implementation for reliability control is shown in figure 3.9. In time step k, the system with s-transformation yields measured health index $\widetilde{HI}_k$ and the generic degradation model yields a simulated value HI_k. The residual r_k is formed and the degradation model state is updated with a new estimate $\hat{\boldsymbol{x}}_k$. The Kalman filter also serves the purpose of giving an estimated value for the true health index $\widehat{HI}$.

A drawback to Kalman filtering is the limit to linear time-invariant systems. This is overcome by newer methods like extended Kalman filter or unscented Kalman filter. For parameter estimation of the degradation model, the basic linear filter in its discrete

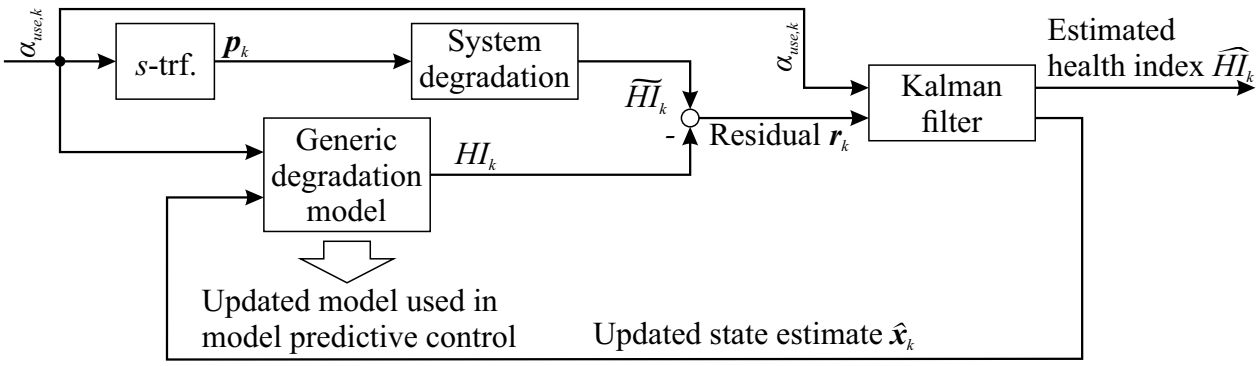

Figure 3.9: Basic outline of Kalman filter

[19] Recall that if a degradation model that is sufficiently precise exists, other means to achieve reliability control are possible. Instead, it is assumed that such a model does not exist!

form is sufficient. The discrete Kalman filter is explained in great detail in [Cat89, pp. 133-140].

To employ filtering for parameter estimation, discrete time state space representation of degradation dynamics (3.15) is desired. For this, the state vector is defined to be health index and constant offset:

$$\boldsymbol{x}_k = \begin{bmatrix} HI\,(t_k) \\ \Delta HI_{\,offset}\,(t_k) \end{bmatrix}.$$

Output vector is scalar health index:

$$\boldsymbol{y}_k = \left[\widehat{HI}\,(t_k)\right].$$

The actual input is just behavior parameterization α_{use}. To include constant terms from degradation dynamics which model nominal degradation, an additional input with constant value 1 is required:

$$\boldsymbol{u}_k = \begin{bmatrix} 1 \\ \alpha_{use}\,(t_k) \end{bmatrix}.$$

With these, (3.15) can now be rewritten as:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A} \cdot \boldsymbol{x}_k + \boldsymbol{B}\,(v) \cdot \boldsymbol{u}_k, \tag{3.18}$$

$$\boldsymbol{y}_k = \boldsymbol{C} \cdot \boldsymbol{x}_k + \boldsymbol{D} \cdot \boldsymbol{u}_k, \tag{3.19}$$

with

$$\begin{aligned} \boldsymbol{A} &= \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \\ \boldsymbol{B}\,(v) &= \begin{bmatrix} -\frac{1}{t_{spec}} & -\frac{1}{t_{spec}} \cdot v \\ 0 & 0 \end{bmatrix}, \\ \boldsymbol{C} &= \begin{bmatrix} 1 & 0 \end{bmatrix}, \\ \boldsymbol{D} &= \begin{bmatrix} 0 & 0 \end{bmatrix}. \end{aligned}$$

Unknown parameter $\Delta HI_{\,offset}\,(t_k)$ is included in (3.18) and (3.19) as separate state vector entry. No direct connection exists from the input $\boldsymbol{u}_k$ to $\Delta HI_{\,offset}\,(t_k)$ and from $\Delta HI_{\,offset}\,(t_k)$ to either $HI\,(t_k)$ or $\boldsymbol{y}_k$. This means that the value could not deviate from the initial value. The Kalman filter forms a state regulator around this model and corrects state vector entries such that model output and system output match up. Doing so, the Kalman filter changes and estimates the current value for additive fault $\Delta HI_{\,offset}\,(t_k)$.

Using the model from (3.18) and (3.19), classic Kalman filter equations are used in each time step to predict a new state estimate $\hat{\boldsymbol{x}}_k$ and a new estimate covariance $\boldsymbol{V}_k$:

$$\hat{\boldsymbol{x}}_k = \boldsymbol{A} \cdot \hat{\boldsymbol{x}}_{k-1} + \boldsymbol{B} \cdot \boldsymbol{u}_k, \tag{3.20}$$

$$\boldsymbol{V}_k = \boldsymbol{A} \cdot \boldsymbol{V}_{k-1} \cdot \boldsymbol{A}^T + \boldsymbol{Q}. \tag{3.21}$$

Next, measurement residual $\boldsymbol{r}$, residual covariance $\boldsymbol{S}$, Kalman gain $\boldsymbol{K}$, state estimate $\hat{\boldsymbol{x}}$

and estimate covariance $\boldsymbol{V}$ are updated:

$$\boldsymbol{r}_k = \widetilde{HI}_k - \boldsymbol{C} \cdot \hat{\boldsymbol{x}}_k, \tag{3.22}$$

$$\boldsymbol{S}_k = \boldsymbol{C} \cdot \boldsymbol{V}_k \cdot \boldsymbol{C}^T + \boldsymbol{R}, \tag{3.23}$$

$$\boldsymbol{K}_k = \boldsymbol{V}_k \cdot \boldsymbol{C}^T \cdot (\boldsymbol{S}_k)^{-1}, \tag{3.24}$$

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k + \boldsymbol{K}_k \cdot \boldsymbol{r}_k, \tag{3.25}$$

$$\boldsymbol{V}_k = (\boldsymbol{I}_2 - \boldsymbol{K}_k \cdot \boldsymbol{C}) \cdot \boldsymbol{V}_k. \tag{3.26}$$

Using this approach, parameter $\Delta HI_{offset}(t_k)$ is estimated by the Kalman filter, in turn adapting the model to fit the current working point.

An initial guess for estimation covariance $\boldsymbol{V}_0$ is required. Generally, this can be considered to be a zero matrix, i.e. $\boldsymbol{V}_0 = \boldsymbol{0}$, with suitable dimensions. Main tuning parameters to achieve a satisfactory estimation are process noise covariance $\boldsymbol{Q}$ and observation noise covariance $\boldsymbol{R}$. To determine suitable values, experiments might be required. To reduce the number of experiments and to obtain a good value, observation noise can be determined from experiments that were conducted to find value v.

With Kalman filter state estimation, the model from (3.15) can be fully parameterized and can be used to simulate future degradation for working points *close to* the current working point. This way, it can be used as part of a closed loop control for reliability using model predictive control.

3.3.7 Model predictive control as reliability controller

The degradation process that needs to be controlled for reliability control is a complex system with nonlinear components and unknown parameters. The controller setup needs to be sufficiently flexible for all these effects, yet still be robust. One control algorithm that fulfills these requirements is model predictive control, for which a good introduction can be found in [CB00]. The aim in model predictive control is to find a system input, such that the system output follows a given desired reference trajectory, as shown in figure 3.10. In a time step t_k, deviations between reference values and actual values are compensated by a new feedforward control, which steers the system back to the desired trajectory. This control is computed for a prediction horizon of m time steps ending at t_{k+m}. Of the computed system input, only the first value is applied at time step t_{k+1}. At this time, the actual system output is re-evaluated and a new input trajectory is computed.

While this control approach is based on open loop feedforward control without feedback, the feedback part is introduced in the form of repeated computations of open loop inputs in regular time intervals. Each feedforward input is computed for a finite time horizon only, but before this limited time is reached, a new feedforward control is computed. This way, if deviations occur, they are compensated by a new corrected computation which is valid for the actual current system state. The computation of a new system input needs to be real time capable, with, for many systems, very short computation durations.

As the name suggests, model predictive control uses a model to find system input parameters for each open loop control sequence. Using optimization algorithms, model input parameters are changed such that a given cost functional is minimized. Complexities for real-world implementations arise from model complexity and the computation

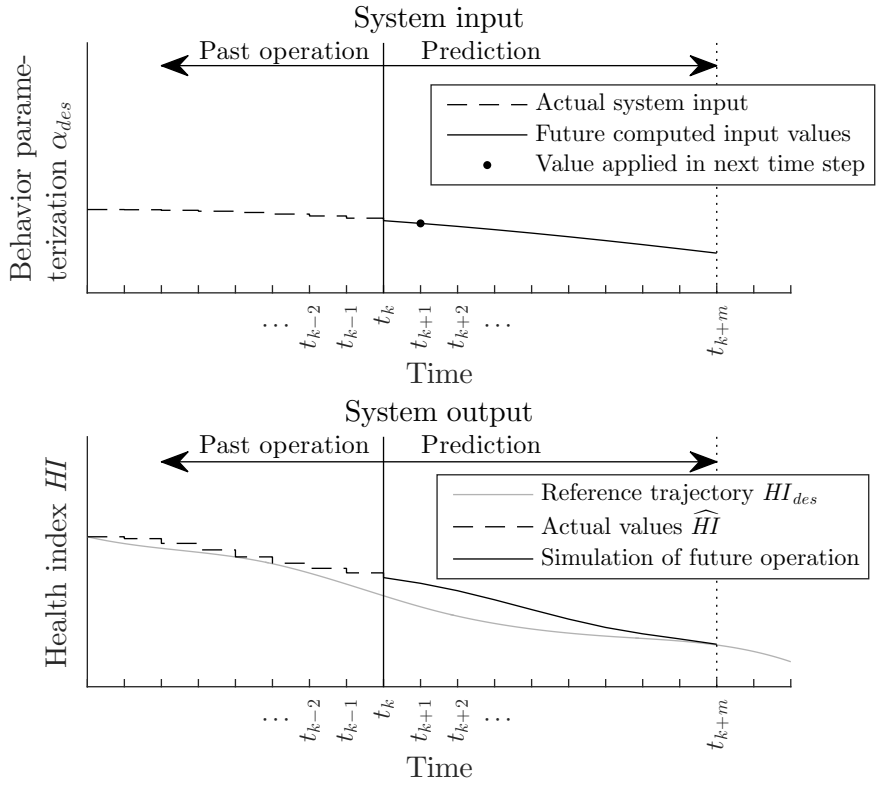

Figure 3.10: Model predictive control

time required for simulations. The optimization requires fast computation times for the model, which limit model precision. Contrary to this, a good model is required to find well-suited system input parameters.

For reliability control with the reduced model described in section 3.3.5, a basic model predictive control algorithm as described in [CB00, p. 76] is sufficient. It uses the abstract model from (3.15) with parameters v from (3.17) and $\Delta HI_{offset}(t_k)$ estimated according to section 3.3.6 to calculate a feed forward input signal $\alpha_C(t_p)$, $t_p = t_k + t_s \ldots t_k + t_s \cdot m, m \in \mathbb{N}$ for the next m time steps starting with the current time step t_k. This abstract model is simple enough to yield fast computation times but also, due to parameters being refreshed in each time step, sufficiently precise for small deviations from the current working point. The model from eqns. (3.18) and (3.19) is parameterized with estimated parameters and used for forward simulation starting at the current time t_k. The result of one such simulation is the predicted health index $HI(t_p), t_p = t_k + t_s \ldots t_k + t_s \cdot m$ for the short future. This prediction can then be evaluated by the cost functional required for solving the optimal control problem. It is:

$$J = w_x \cdot J_x + w_u \cdot J_u$$

with components J_x and J_u, which are weighed individually. J_x is the error in health index. It determines how closely the simulated values HI and the desired value HI_{des} fit:

$$J_x = \sum_{p=k+1}^{k+m} \left(HI_{des}(t_k + t_s \cdot p) - HI(t_k + t_s \cdot p) \right)^2 .$$

Differences in consecutive system input values penalize fast changes, for which cost functional J_u is created. The first value of the system input is for all simulations the current behavior controller input value $\alpha_{des,k}$. New and variable values are reliability controller output $\alpha_{C,k+1\ldots k+m}$:

$$J_u = \left(\alpha_C\left(t_k + t_s\right) - \alpha_{des}\left(t_k\right)\right)^2 + \sum_{p=k+2}^{k+m} \left(\alpha_C\left(t_k + t_s \cdot p\right) - \alpha_C\left(t_k + t_s \cdot (p-1)\right)\right)^2.$$

The optimal system input $\alpha_C\left(t_k + t_s \ldots t_k + t_s \cdot m\right)$ is then found by minimizing the cost functional J over a receding horizon for m time steps coming forth, i.e. solving the problem

$$\alpha_C\left(t_k + t_s \ldots t_k + t_s \cdot m\right) = \underset{\alpha_C(t_k+t_s \ldots t_k+t_s \cdot m)}{\arg\min} J, \tag{3.27}$$

subject to the constraints given by (3.18), (3.19) and the input constraints

$$-1 = \alpha_{min} \leq \alpha_C\left(t_k + t_s \ldots t_k + t_s \cdot m\right) \leq \alpha_{max}, = 1 \tag{3.28}$$

with α_{min} and α_{max} being the minimal and maximal admissible values for the behavior parameterization α_C. From this, the first value $\alpha_C\left(t_k + t_s\right)$ is used as next controller output.

3.4 Condition monitoring to determine current health index

A closed loop controller relies on measuring the difference between a given reference input and the system output. For this, the current system output value needs to be known. In case of the reliability controller, knowledge of the current health index is necessary[20]. This is a complex variable which, for most systems, cannot be measured using traditional sensors. In addition to determining the current value of the health index, the delay between acquirement of data and availability of new value must not be overly long. Controller stability is limited by the drop in phase of the closed control loop. If additional delay is introduced by state measurement, the phase drop is increased, thus severely limiting controller stability[21]. Techniques to estimate the current degradation state of a system are summarized by the term condition monitoring. There are sophisticated methods for condition monitoring available, which all have individual benefits and drawbacks. To decide whether a condition monitoring approach is suitable, accuracy and delay of health index estimation need to be taken into account.

At first, model based and data driven methods can be distinguished. In model based condition monitoring, a degradation model is simulated with input parameters given by system operation and environment and the state of the system or component is obtained. With *model*, a user-defined model, usually first principles based, is meant. For

[20] In [MKS15b], we assumed that estimating the health index and then having a prediction based on this is at a slight disadvantage when compared with estimating the remaining useful lifetime. In the meantime, the advantages of using the health index as basis became clear. The main drawback of remaining useful lifetime is that it includes future system operation, which is adapted dynamically by reliability control.

[21] System dynamics themselves are still stable and safe, as discussed in section 3.5.

most systems, such a model has a high number of input variables, making it difficult if not impossible to setup. Environmental input variables might include temperature, humiditiy, ambient vibrations, uncertain material parameters or similar measures. If it is possible to setup a degradation model, high computational requirements might make it difficult to use it in reliability control. An example is crack growth which can be simulated using FEM models. While there are several drawbacks when it comes to setup of a model based condition monitoring approach, generally the assessment of current health index is quite good, but due to long simulation times, delay might be large. Thus suitability of a model based condition monitoring for reliability control has to be assessed on case-by-case basis.

Data driven condition monitoring is based purely on data and does not require a first-principles based model. Instead, machine learning algorithms are used to automatically construct a model for all information that can be deduced from system data. This is a mostly automated process, making setup of the model easier. Machine learning algorithms are meant to be flexible. Usually, they suit many different kinds of problems of which condition monitoring is only a niche application. Very few algorithms allow to include prior knowledge about physical effects, one possible implementation is described in [USA+14]. This general applicability brings with it the problem of selecting the most appropriate algorithm. We presented a (limited) guide for this in [MKS15b]; a more thorough approach can be found in [Kim16].

All data driven condition monitoring approaches rely on training data and on reference data. From the training data, features are extracted and a machine learning algorithm constructs a model that represents the relationship between features and reference data. This procedure is represented in figure 3.11. In condition monitoring, reference data usually includes some measure of the current health of the system, which is also the output variable of the learned model. The basic approach is to classify the system degradation into several distinct states which correspond to system health. During operation, the current state is estimated using the learned model. By taking into account the time the system is in a given state, the remaining useful lifetime or the health index can be estimated. While this approach has high relatively precision in estimating the current state, it has low precision for state-based estimation of the continuous value health index and is not well suitable for reliability controlled system operation. Instead, it could be used to execute emergency routines based on a system classification according to the four

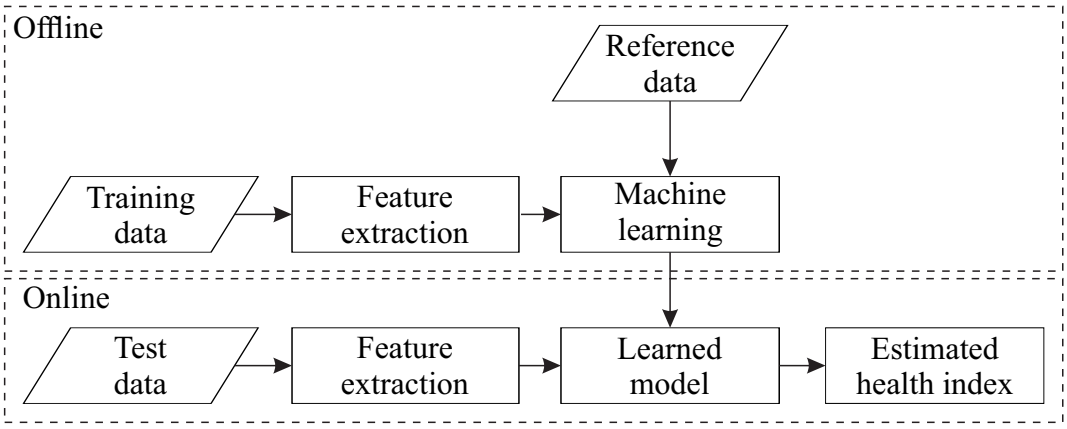

Figure 3.11: Basic procedure used in data driven condition monitoring.

levels of the multi-level dependability concept, see section 1.4.1.

Instead, estimation of the value of a continuously defined health index variable is desired. Most machine learning algorithms have no inherent dynamics, i.e. they directly map extracted condition monitoring features to current system health index. However, features extraction itself might include complex algorithms. These include frequency-based approaches, e.g. determination of the amplitudes of characteristic frequencies in a vibration signal using Fast Fourier Transform. To be suitable for closed loop control, the delay has to be kept to a minimum, i.e. the time horizon on which transformations into frequency domain work needs to be as short as possible while also considering the constraints given by sampling frequency and data acquisition intervals. If these prerequisites are fulfilled, health index obtained from data driven condition monitoring is well suitable for reliability control.

3.5 Stability of the controller

To assure dependable operation, controller stability has to be ascertained. Reliability control is based on model predictive control, making stability theory for it accessible. A good overview over stability for constrained model predictive control is given in [MRR+00].

The full system model would need to include a degradation model and a detailed model for condition monitoring, both of which would yield a highly complex model. *Not* requiring a degradation model was also one of the main motivations for developing a novel reliability control approach. For control purposes, a generic degradation model is setup and parameterized at runtime, as explained in section 3.3. This model only loosely represents actual system behavior and is not sufficient for theoretical stability analysis.

Without a sufficiently detailed model, only empirical stability evaluation remains. This would have to be conducted during setup. Behavior controller or reliability controller instability would *not* directly lead to unsafe system state though! Since only parameters of pre-determined working points are set in the dynamic control loops, full isolation from reliability and behavior control to dynamic control is achieved.

Figure 3.7 shows the basic setup of the whole reliability control loop. One noteworthy specific about the chosen realization is that all controllers for system dynamics are on a low level and the only connection between this lower level and the upper reliability control level is the α-parameterization. It determines the working point used, but does not directly influence system behavior – only parameters are set accordingly on the lower level. While the selection of dynamic controllers obviously influences system reliability and current system reliability determines controller choice, they are not able to interact on the same time scale or even influence the same system states. This way, the two layers of dynamic controllers and reliability controllers are as well isolated from one another as possible.

By using multiobjective optimization, which is based on numerical computations and does not find a continuous Pareto front, individual working points as optimal compromises are found. The α-parameterization is limited to selection from these compromises. With this limitation comes the inherent problem, that a *perfect* working point for the current situation is highly unlikely; instead, the best compromise is selected. One could think of ways to overcome this using interpolation methods, as was also described in [MS14,

p. 5]. However, using such interpolation is ill-advised since the currently selected interpolated working point would be deviating from all pre-computed working points. Not being able to rely on this point being an *optimal* working point with regards to system objectives might not be problematic, as deviations between actually reached objective values and pre-determined theoretical values cannot be eliminated anyway. However, one major benefit of using multiobjective optimization to compute optimal working points is that safety of each point can be ascertained individually. This can be done e.g. by automatically evaluating controller stability during optimization and demanding a certain safety margin as constraint for a working point to be valid. This reduces the necessary additions for complete system safety to proof of safety of the switching process itself. A good overview and possible implementations of safe switching procedures, which can also be setup in an automated process, is given in [Osm15]. To avoid unsafe states in the full system or in collaborating systems, parts of the operator controller module can be modeled as state machine. Continuous parts, i.e. controllers for system dynamics, are included using hybrid states. Using this state chart, safety of all configurations can be proved, as was shown in [GBS+04; GHH+06].

For these reasons, discrete working points are preferred over continuously adaptive parameters. If a sufficiently large number of working points with gradual differences exists, quasi-continuous adaptation becomes possible. The larger the number of points to choose from, the less influence the selection of the nearest discrete working points has on system and failure behavior. One main goal in multiobjective optimization is to find a set of solutions as diverse as possible [Deb01, p. 22]. Generally, multiobjective optimization algorithms fulfill this goal quite well, creating a large number of possible working points that allow for finely adapted system behavior. Due to this, the term *continuous control* is used throughout this thesis for reliability control as proposed, despite being deliberately non-continuous.

3.6 Implementation into operator controller module

The operator controller module serves as basis for the structure of information processing in a self-optimizing system, see also section 2.2. On the lowest level, the controller, dynamic controllers that directly process sensory information and compute new actuator signals are situated. At least some of these must be adaptable by changing parameters or by changing their setpoint. Parameters or setpoint were also used as optimization parameters during multiobjective optimization. Dedicated signal acquisition for condition monitoring might be required, which is also executed on this system level.

The reflective operator may run the signal processing algorithms for condition monitoring. Either using condition monitoring or other means, e.g. signal level thresholds, the current system state is classified into pre-defined classes. This is required for fast reactions in case of failures using the multi-level dependability concept, see section 1.4.1. Also behavior adaptations are initiated by the reflective operator, but actual switching between working points is situated in the controller level.

The cognitive operator selects the currently pursued working point according to requirements imposed by reliability control, by user demands and by other aspects of system operation. The processes required for synchronisation and cooperation with other systems, planning of failure times and maintenance intervals are also embedded into the

cognitive operator.

3.7 Enhanced multi-level dependability concept

The multi-level dependability concept as introduced in [GRS09b, p. 61], [KSR12, p. 6] enables a self-optimizing system to change its behavior based on current dependability, i.e. reliability among other attributes, as was already discussed in section 1.4.1. It does so by classifying the system state into four levels. In the first level, it is assumed that dependability does not need to be taken into account when selecting current system behavior, only in levels two and three dependability becomes an issue. Level four is special, as in this level, the sole goal of the system is to reach a safe state.

In [SMD+12], an example for usage and setup of the multi-level dependability concept is given. The remaining life of railway wheel flanges is used as main factor for degrading reliability. Once the wheels reach 50% or 25% remaining useful lifetime, the current system state is classified into different levels. However, if the remaining useful lifetime reaches 25% after 75% at the same time, there is no need to switch states – degradation is just as desired! In this case, continuous control would be advantageous, as it would take the time-dependancy into account and compensate early, yet only as much as required to reach the desired usable lifetime.

This original definition assumes that dependability is a minor aspect of system operation and that if everything is well, dependability can be neglected. Reliability controlled operation, on the other hand, aims to always find an optimal compromise between reliability as one aspect of dependability and system performance. Without taking system reliability into account, system performance would either be worse than necessary to achieve the desired time of operation until maintenance or the system might fail early. So in order to meet the classifications when using reliability control, the levels need to be redefined.

A basic characterisation of the first level is that system behavior is *nominal*, whereas in the second level dependability requires some adaptation that deviates from the nominal working point. In order to rework the definition of the multi-level dependability concept, this notion has to be included in the definition of the discrete levels. The first two levels are reformulated as follows:

Level I
: The system is in a safe, normal state. Performance and dependability objectives are in balance. Dependability does not need to be prioritized overly and countermeasures against undesired events are not required.

Level II
: A threat was detected, e.g. a threshold was reached. While the system is still operating in a safe state, pursuing objectives like comfort or energy efficiency only might lead it towards an unsafe state. Self-optimization is used to prioritize dependability over competing objectives higher than during nominal operation. This leads to a behavior adaptation and in turn keeps system operation dependable, but impairs other objectives.

Levels III and IV are not subject of this thesis which is focussed on continuous control to be employed in levels I and II. Instead, the interested reader is asked to consult [Son15], which covers these aspects in detail.

From the attributes of dependability, cf. figure 1.1 and section 1.1, this thesis is focussed on reliability. Of these, the attribute safety is similar to reliability. In reliability, the likelihood of an undesired event *system failure* at some point in time is researched and methods to avoid early failure are developed. The resulting failure function $F(t)$ represents probability of the undesired event over time. In safety, consequences and likelihood of catastrophic faults are researched. A catastrophic fault can also be regarded as *undesired event* in the same sense as in reliability, thus making it equivalent to a system failure. Without looking at consequences, the likelihood of a catastrophic, safety-impeding fault can then be assessed using the same probabilistic methods as the likelihood of a system failure in reliability. A comparison between safety measures and reliability measures can be found in [MP10, Tab. 2.4-1]. If this likelihood can be quantified and the measure can be embedded in an optimization, the same reliability control methods can be used to control safety of operation. For this, the main requirements are knowledge about actual safety risks and objectives which can be prioritized to enhance safety.

This was conducted in [MHM+13] for the collision of two parts of a railroad vehicle linear drive system. The undesired event is a collision between vehicle mounted primary part and track mounted secondary part. If the air gap between these two becomes lower or equal to zero, a collision occurs. Additional difficulties arise in safety-related multiobjective optimization from quantification of the safety measure. In [MHM+13], probability distribution of occurences of unknown height deviations of the secondary part is assumed to be known. The air gap is measured by a sensor with additional uncertainty. From these two uncertain measures, the probability of the true sensor value being lower than the unknown height deviation can be found, which is equal to the probability of collision. By computing this collision probability during objective function evaluation, the risk of operation can be included in a multiobjective optimization.

To fully adapt system behavior based on system safety, the current probability of an undesired event would need to be determined at runtime. This is equivalent to obtaining the current health index for control, cf. section 3.4. While condition monitoring is a suitable way to estimate the health index, different means to determine the current risk of operation would need to be found.

The other attributes of dependability are either inherently controlled as well (availability) or cannot be controlled at all (maintainability, integrity). Focussing on reliability as one aspect only allows a more in-depth approach including experimental validation.

4 Application and experimental validation of reliability control

To show that the control approach outlined in chapter 3 is a feasible solution to implement reliability control in a given system, an experimental validation is desired. By conducting experiments, feasibility of controller setup, controller behavior and finally failure behavior of the controlled system can be evaluated.

An arbitrary system could be selected, but to show the desired effects, several constraints have to be taken into account. One is the obvious aspect of being a mechatronic system that fails at some point but allows interaction. Other aspects include ease of setup, cost of the test setup and cost of the failing specimens. At the end, decision was made to develop and assemble a mechatronic single plate dry clutch system.

4.1 Motivation for self-optimizing clutch system

Even though several alternative means of propulsion for passenger vehicles, e.g. fuel cell technology and purely electric vehicles, are in development or have reached acceptance in the automotive market already, the currently most-used power source is still fossil fuels. These are usually converted into mechanical energy using internal combustion engines which have a minimum operating speed. If the vehicle is at low speed or even at rest, the drivetrain needs to be operated below minimum engine speed. This is possible by means of a dedicated connecting element. In most manually shifted gear boxes, a foot-operated single plate dry clutch is employed, while automatic gear boxes are equipped with hydraulic torque converters [KSS+03, Section 5.4]. With the desire for more comfortable vehicle operation and better fuel economy, automated manual transmissions with a single clutch and dual-clutch transmissions were developed and make up a small but steadily growing fraction of the whole market for passenger vehicles, see figure 4.1. Dual-clutch transmissions usually have one clutch that engages odd numbered gears and a second one to engage even numbered gears. By alternating between these two clutches, fast shifting times can be achieved.

Automated manual transmissions generally are equipped with a single clutch and work just the way regular manual transmissions do [KSS+03, p. 271]: An input clutch connects engine to gearbox and internal meshes engage individual gears. A gear change is only possible when no torque is being transferred. This can be achieved by opening the clutch, which usually leads to long shift durations[22]. In commercial vehicles, this is hardly noticeable since acceleration is low anyways [FS07, p. 141]. In high-power supercars, where the drivetrain needs to cope with the high power throughput, dual clutch gearboxes would be too heavy. There, this problem can be solved by controlling motor, gearbox and single clutch in union [Tor12, p. 604].

[22] The early *smart City-Coupé* is a good example of this [Jac09, p. 23].

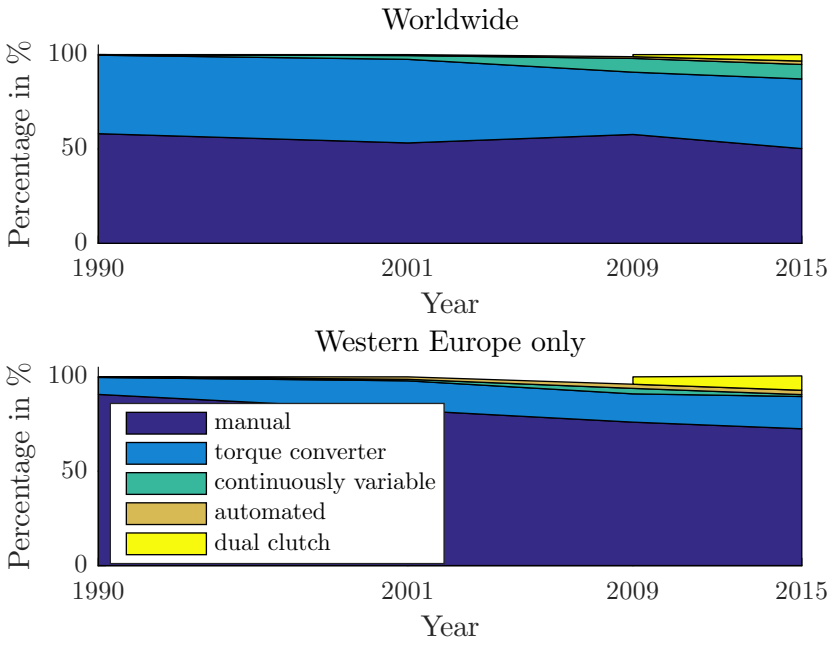

Figure 4.1: Split of transmissions in passenger vehicles. Data from [KSS+03, Tab. 5.4-1], [ABB+13, Tab. 5.4-1].

The acceleration from rest of a vehicle equipped with an automated manual transmission is a prime example for an active control of the reliability. During each acceleration, the clutch system wears and ultimately fails due to being worn out. If the clutch fails early, the whole vehicle is unusable. While for privately used vehicles, unscheduled maintenance is a nuisance, for commercial applications it is also very costly. The conventional approach to avoid this is to schedule maintenance early enough to avoid unexpected failure, while accepting wasting what is left of usable lifetime (cf. section 1.2).

Active control of the reliability of a clutch system is a good way to make scheduling maintenance and using the clutch up to its full potential possible, thus adhering to reliability requirements and increasing availability of the vehicle. In order to show the feasibility of the proposed closed loop control for reliability introduced in section 3, a clutch system test rig has been built. It is embedded into a virtual vehicle, which is accelerated using the real clutch system.

4.2 Usage scenario for clutch system

Controlling the reliability of a clutch system is not possible without taking the surrounding system into account. For this reason, a full vehicle and a usage scenario are required. The usage scenario is made up of a characteristic maneuver, which includes load on the clutch that is representative of actual system operation, cf. [KMS15, p. 2209]. While it might seem advantageous to just use an actual vehicle instead of an experimental test rig, this has several drawbacks. At first there is complexity of the system and of operation, which makes it difficult to achieve repeatability of experiments. Second is lifetime and cost. For the evaluation of clutch plate reliability, lifetime tests need to be conducted. If the plates were as durable as regular automotive clutch plates, large test stands and

long experimental duration would be required.

To overcome these problems, a virtual vehicle is used as application example. This is either simulated with a model of the clutch plates or it is used to compute input signals to the actual clutch system. The simulation model can also be used as optimization model.

4.2.1 Characteristic maneuver: Acceleration from rest

To avoid overly complex modeling, reasonable model simplifications have to be chosen such that the goal of modeling is achieved, but model complexity is as low as possible. For this, the goal of modeling needs to be clear first. As an automotive vehicle moves, the clutch transmits power from the engine to the gearbox and in turn to the wheels. When in motion, the vehicle dynamics can be subdivided into three categories: Vertical (body) dynamics, lateral dynamics and longitudinal dynamics. Lateral dynamics generally have no effect on clutch usage. The longitudinal dynamics interact directly with the clutch plates and their wear. Vertical dynamics are only important at high accelerations, were vertical tire forces and the longitudinal motion interact closely. For this reason, the characteristic maneuver is limited to a longitudinal motion. There are three distinct types of longitudinal motion that need to be addressed: constant or accelerated motion with fully engaged clutch; no motion or purely rolling motion with the clutch disengaged; low-speed operation, gear changes and acceleration with partially engaged clutch.

Of these, only those with the clutch being partially engaged are important, as full engagement or no engagement at all yields no wear. During shifting, power transmitted is low and so is wear. Low-speed operation at constant speed is rare and can be neglected. Then the main driver is acceleration of a vehicle from rest, where both the velocity difference between plates and the torque transmitted is high. There are several objectives relevant for this characteristic maneuver, of which the first one quantifies system performance.

4.2.2 System performance objective

The main objective of operating a motor vehicle is obviously very dependent on the application it is used for. The operators of commercial vehicles primarily pursue the total cost of ownership being as low as possible in order to increase revenue. However, for privately owned vehicles other factors come into play as well. Sports cars or historic vehicles, for example, are luxury items where cost is of lower importance and the operation itself is the goal of operation. Another prerequisite is that comfort of passengers should be high. This also corresponds to reducing vibrations of the vehicle, which might damage cargo.

Within the scope of this thesis, a commercially used small vehicle is assumed. This makes the main objectives with regard to acceleration from rest low energy consumption and high comfort. Energy consumption is difficult to quantify without complex motor models, which would extend the scope of this thesis too far. Instead, performance of the system is only defined as comfort of the passengers.

To determine how comfortable an acceleration maneuver is for the passengers of a vehicle, human perception has to be taken into account. To this end, longitudinal accelerations are evaluated. Since human perception is difficult to model, it was by itself worthy of detailed research.

In [Alb05, pp. 86-119], machine learning is used to determine passenger comfort in simulations. For training, measurement data and training data is required. These are obtained using a specifically built test vehicle which is equipped with an automated clutch system which allows to select five different actuation characteristics while driving. 23 different test drivers were subjected to these different actuation characteristics during multiple test runs on an empty proving ground, i.e. in a controlled environment. Then, these test runs were evaluated. For this, the group of test drivers was subdivided into three groups of customers. After each test run, the test driver was asked „Would you accept this system as it is?“. Answering was possible by sliding a continuously variable sensor into an arbitrary position between the two endpoints *absolutely inacceptable* and *no further improvement necessary*. The data obtained from the slider position as driver acceptance measure and of vibration signals was correlated using machine learning. While using machine learning is not possible without measurement and training data and thus not possible within this thesis, it is noteworthy that as measurement signal, in [Alb05, p. 71], longitudinal acceleration measured under the seat is used.

In the standard [VDI 2057-1], assessment of the effect of mechanical vibrations on the human body is described. However, these cannot be evaluated directly since human perception changes with frequency and direction of the excitation and posture of the body. For the purpose of evaluating simulated acceleration signals, a filter according to this standard could be implemented.

The vibration transmittance of the vehicle body to the driver would be neglected since it would be difficult to model and is highly dependent on the vehicle. Perception is taken into account by filtering the vehicle body acceleration signal which could be implemented in the model of vehicle dynamics. The posture of the driver is assumed to be sitting. To assess the full acceleration maneuver, the rms value $a_{w,t_{end}}$ of the frequency-weighted acceleration signal is to be used. It is computed as

$$a_{w,t_{end}} = \sqrt{\frac{1}{t_{end}} \int_0^{t_{end}} \left(a_w\left(t\right)\right)^2 \mathrm{d}\, t} \tag{4.1}$$

with $a_w\left(t\right)$ being the frequency-weighted acceleration and $t = 0 \ldots t_{end}$ being the duration of measurement. Frequency weighting could be implemented according to [VDI 2057-1, Figure 2], where W_k is the frequency weighting curve to be used for a sitting position and a measurement on the seat[23]. While the standard includes a graphical representation of the frequency weighting curve and transmission factors for several frequencies, no parameters for a continuous time filter are given. However, [RM07, p. 514] gives several transfer functions including filter parameters for filtering according to international standard [ISO 2631-1], on which the national standard [VDI 2057-1] is based. The following transfer function is given for W_k:

$$W_k = H_h\left(s\right) \cdot H_l\left(s\right) \cdot H_t\left(s\right) = \frac{s^2}{s^2 + \frac{\Omega_1}{Q_1} \cdot s + \Omega_1^2} \cdot \frac{\Omega_2^2}{s^2 + \frac{\Omega_2}{Q_2} \cdot s + \Omega_2^2} \cdot \frac{\frac{\Omega_4^2}{\Omega_3} \cdot s + \Omega_4^2}{s^2 + \frac{\Omega_4}{Q_4} \cdot s + \Omega_4^2},$$

with $\Omega_1 = 2 \cdot \pi \cdot 0.4\frac{1}{\mathrm{s}}$, $\Omega_2 = 2 \cdot \pi \cdot 100\frac{1}{\mathrm{s}}$, $\Omega_3 = 2 \cdot \pi \cdot 2\frac{1}{\mathrm{s}}$, $\Omega_4 = 2 \cdot \pi \cdot 2\frac{1}{\mathrm{s}}$, $Q_1 = \frac{1}{\sqrt{2}}$, $Q_2 = \frac{1}{\sqrt{2}}$, $Q_4 = 0.63$.

[23] The identical curve is referred to as W_d in [ISO 2631-1] and in [RM07], while in [VDI 2057-1] it is a different frequency weighting curve used for a measurements on the foot platform.

Root-mean-square a_{wT} of frequency weighted acceleration $a_w(t)$	Description of perception
< 0.01	not perceptible
0.015	threshold of perception
≤ 0.02	barely perceptible
≤ 0.08	easily perceptible
≤ 0.315	strongly perceptible
> 0.315	extremely perceptible

Table 4.1: Perception of different values for the rms value of frequency weighted accelerations (Source: [VDI 2057-1, Tab. 3]).

By calculating $a_{wT}(a_w) = a_{wT}(W_k \cdot a_x)$, comfort of the acceleration maneuver can be assessed. [VDI 2057-1, Tab. 3] also gives descriptions of the perception for intervals of a_{wT}, which are reproduced in Tab. 4.1.

As can be seen, with the simple approach outlined in [VDI 2057-1], some effects would need to be neglected while introducing additional factors of uncertainty. In both the standard evualuation and the sophisticated evaluation from [Alb05], the comfort measure is based on longitudinal acceleration. In keeping with this, for the remainder of the thesis longitudinal accelerations are evaluated *directly* by computing the maximum longitudinal acceleration during vehicle acceleration.

4.2.3 Virtual vehicle

As was discussed in section 4.2.1, a model covering longitudinal dynamics is sufficient. They are mainly influenced by the engine, which transmits its power via clutch and gearbox to the wheels, and the mass of the vehicle. Engine dynamics are omitted from the simulation model. Instead, it is assumed that engine velocity is controlled and thus constant despite perturbations from power demands. The ability to transmit power through friction of the tires is limited by vertical force on each wheel. However, the effect is small and mostly relevant for fast acceleration, light vehicles and a high center of gravity. For simulation efficiency reasons, it is omitted as well.

Thus, pure longitudinal dynamics remain. For optimization and for hardware in the loop experiments, fast simulation is required. To allow for as fast simulation as possible, a basic model with one degree of freedom was implemented. In this, all masses of the vehicle are lumped together into one inertia Θ, which is accelerated by clutch plate torque $T_p(F_N, \Delta\omega, E_f)$:

$$\dot{\omega}_2(t) = \begin{cases} \frac{1}{\Theta} \cdot (-d \cdot \omega_2(t)) & \text{iff } T_p < T_0, \\ \frac{1}{\Theta} \cdot ((T_p(F_N(t), \Delta\omega(t), E_f(t)) - T_0) - d \cdot \omega_2(t)) & \text{otherwise.} \end{cases} \tag{4.2}$$

With ω_1 being constant motor velocity, d being a drag coefficient and T_0 being dry friction torque which models internal losses, e.g. bearing and seal friction. Torque computation inputs[24] are the normal force $F_N(t)$, the difference in rotational velocities $\Delta\omega(t) =$

[24] The reasons for $F_N(t)$, $\Delta\omega(t)$ and $E_f(t)$ as input variables for the torque computation becomes clear in section 4.5.1.

$\omega_1(t) - \omega_2(t)$ and the friction work $E_f(t)$.
This load model is sufficient as basis of optimization and as hardware in the loop load model. For hardware in the loop simulations, it is coupled with a friction clutch testrig, which is introduced in section 4.3. When used in optimization, a friction model of this testrig is required.

4.3 Set-up of a test rig for clutch system

The test rig serves as demonstrator module for the actively controlled reliability. As such, one of the main goals was to create a system that wears fast and has friction pads that are cheap and can be changed quickly. This was achieved by using friction pads made from felt. They are only capable of transferring low torques, thus scaling between the torque applied to the virtual vehicle and to the test rig or scaling of the system itself is necessary.

4.3.1 Structure of test rig

The test rig consists of four main components, also depicted in figure 4.2:

Friction pads

Power from the drive motor to the load motor is transmitted via two clutch plates, which are the main components of the clutch system. The two plates are pressed against one another and transmit power through dry friction, which induces wear in the plates.

The requirements for the clutch plates deviate slightly from regular automotive clutch plates. The main premise is to be able to observe wear effects in experiments of short duration at low power. Short duration is required to be able to conduct lifetime experiments without waiting for weeks or even months as would be required if normal automotive clutch plates were used. Also quick changes of worn out clutch

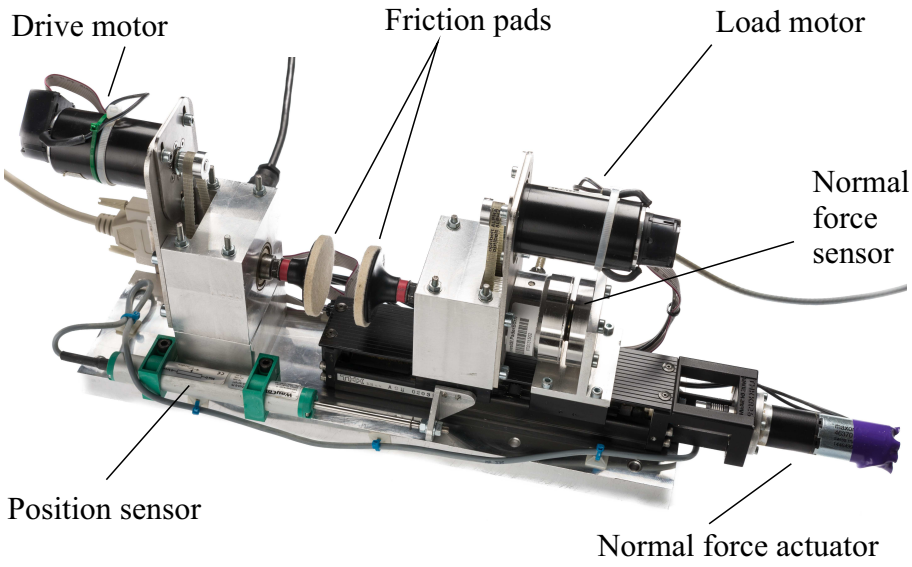

Figure 4.2: Clutch system.

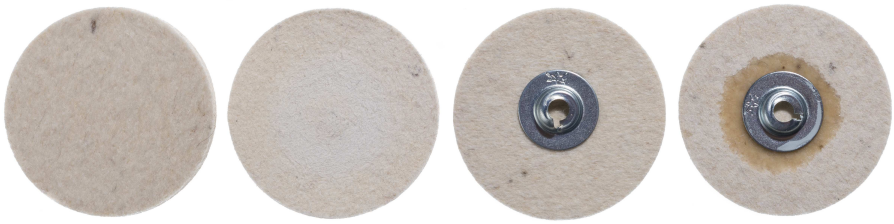

Figure 4.3: Felt pads used as clutch plates. Left to right: Front of new plate, front of worn plate, quick-lock holder on rear, reinforced holder.

plates are necessary to obtain short setup times. The request to use low power comes from the desire of safe operation and easy setup of the whole test rig.

These requirements led to the selection of small clutch plates made from felt[25] which are sold as polishing pads, see figure 4.3. They are equipped with a quick-lock holder which enables unmounting with a single rotation counter-clockwise and are readily available at low cost. Failure modes are identified in section 4.4. To avoid the stochastic and undesired failure mode „Breaking of holder“, the quick-lock holder is reinfored with additional glue. The plate's intended purpose is not to transfer torque, but to be used as a tool for polishing surfaces. As such, the manufacturer does not publish friction properties. For this reason, a dedicated model is parameterized in section 4.5.1.

Drive motor

The drive motor represents the engine of the vehicle. It is used to apply a predefined rotational velocity to the input clutch plate. A high quality DC motor[26] was selected. It is equipped with a digital incremental encoder[27] and controlled by a dedicated controller[28]. The motor controller is highly parameterizable and supports multiple modes of operation. For the drive motor, speed control was selected with the speed set point being the input variable of the drive system. Available outputs are actual speed and current, which is proportional to torque, as shown in section 4.3.4. The motor power is transmitted to the clutch plates with a belt drive with reduction ratio 1:3.

Load motor

The load motor applies the rotational velocity that is computed in the virtual vehicle model to the friction contact. The resulting torque is computed from the current and applied to the virtual vehicle model. The same drive system comprised of DC motor, encoder and controller as for the drive motor was selected.

Both drive and load motor are supplied with 48 V DC from a switching power supply[29]. The load motor converts mechanical energy from the clutch plates into electric energy which is fed back into the power supply circuit. Using the same 48 V supply circuit for both drives, the drive motor uses this electric energy and converts it into mechanical energy which is then transferred to the clutch plates. This way it is possible to drive both motors with a comparatively small power supply which

[25] Pferd Combidisc CD FR 50, 440490
[26] Maxon DC motor RE40, 148877, 48 V, 150 W
[27] Maxon sensor HEDL5540 110514
[28] Maxon motor control ESCON 50/5, 409510
[29] Mean Well S-150-48

only needs to provide the power that is dissipated in friction and in internal losses due to drive efficiency.

Normal force actuator

The normal force actuator is again a DC motor but of smaller size[30]. It is not equipped with auxiliary sensors. The normal force is applied via a planetary gear head[31] and a ball screw with integrated linear guide[32]. The linear guide is equipped with two carts, of which only one is affixed to the ball screw. The motor torque is converted into a linear force acting on this cart, from there it is transmitted into a normal force sensor and onto the second cart which holds the load motor and clutch plate.

The electric motor is driven by the same motor controller as the drive and load motors, but runs at a lower voltage. For this reason it is equipped with its own power supply made up of a transformer with rectifier and capacity for current surge stability reasons. It controls motor current, which is proportional to motor torque and in turn to normal force applied in the clutch plates. The set point is determined by a force controller in the realtime control prototyping unit.

To allow for closed loop control of the normal force actuator, two sensors are included in the system. The first one is a normal force sensor which measures the force acting on the clutch plates. For this, a strain gauge based load cell[33] with dedicated amplifier[34] is used. The second sensor is a linear potentiometer[35] which is used as position sensor that measures displacement of the moving clutch plate. Details about the associated controllers follow in section 4.3.5.

Power supplies and controllers are contained within a control enclosure which connects to the test rig via two dedicated cables. All signals except the force sensor value are routed through this control enclosure. The interior of the control enclosure can be seen in figure 4.4. The rapid control prototyping unit is connected to this box and to the load cell amplifier via analog signal inputs and outputs.

4.3.2 Hardware used to implement the reliability control system

The actual system layout is more complex than the idealized structure shown in figure 3.7. This is mainly due to the requirements imposed by implementing it on actual hardware. The setup is based on the operator controller module, see section 2.2 and figure 2.3. It structures the information processing into three layers: Actual system running in real time, reflective operator as intermediate stage and cognitive operator running in soft real time.

On the lowest level, hardware-implemented controllers are situated. For the clutch system, these are motor controllers and normal force sensor amplifier, which both have internal signal and data processing. The motor controllers run a closed loop control for velocity of the motors. Their setpoint is generated by a rapid control prototyping

[30] Maxon DC motor A-max 26, 110937, 12 V, 11 W
[31] Maxon gear GP 26 A, 406764, nominal reduction ratio 27:1
[32] THK KR26 02B 200, ball screw lead 2mm
[33] HBM U3 0.5 – 10 kN
[34] HBM MGCplus with connection board AP01
[35] Waycon LZW2-S-75

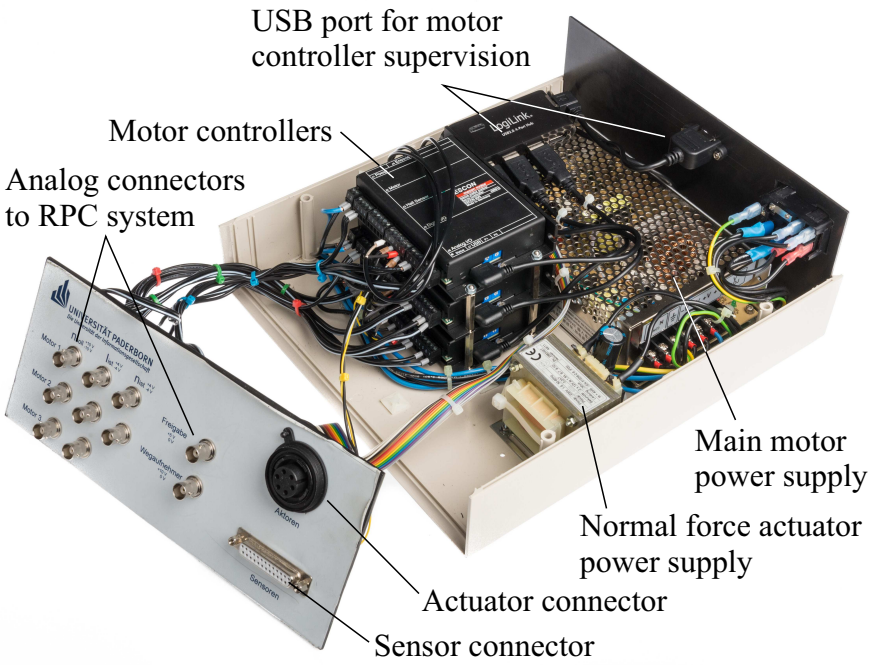

Figure 4.4: Opened control enclosure with individual components.

system[36] running in real time as well. The virtual load, other dynamic controllers, supervisory control and state control are implemented here. Its behavior is influenced by a host system running Windows and Matlab. All higher level routines are implemented in Matlab, which utilizes the dSpace HIL API to interact with the rapid control prototyping system. The HIL API allows to change variable values in the real time code at runtime and to receive measurement data. Reliability control, behavior control and the working point selection are all implemented in Matlab.

4.3.3 Interfaces to virtual vehicle

Embedded within the virtual vehicle is the clutch plate model, which determines the torque transmitted from one friction plate to the other by means of dry friction. Its input variables are the difference in rotational velocity $\Delta\omega = \omega_1 - \omega_2$ and the normal force F_N. This model is exchanged for the test rig, which has to implement $T_p\left(F_N, \Delta\omega, E_f\right)$. For this reason, both motors are velocity controlled with external setpoint, which is transmitted from the rapid control prototyping unit by means of an analog signal. In turn, the current is measured by the controllers and transmitted to the rapid control prototyping unit via an analog output.

However, the test rig is smaller than an actual automotive clutch and not capable of transmitting as high a torque. This is desirable for behavior adaptation testing purposes, but requires an adaptation between virtual vehicle and test rig. Adaptation between *small* test rig and *big* vehicle could be achieved by using constant factors, which scale according to approximate maximum values.

To find the values of scaling factors, similarity analysis can be conducted. One possible way to do this is to use Buckingham's Π theorem, which was introduced in and named

[36] dSpace DS 1005 PPC with I/O-boards DS 2004 and DS 2102

after [Buc14]. An elobarate application to vehicle dynamics, both lateral and longitudinal, can be found in [BA00; BA01; BA99; Bre04]. In these papers, a full vehicle model is setup such that vehicle dynamics are similar to actual vehicle dynamics. Then observations made at the model can be transferred to full-size vehicles. Advantages of using the model include easier handling, safer operation and lower cost.

To adapt clutch system behavior, where pure longitudinal dynamics with a rather simple model are employed, and where transferring experimental results from dynamics is not desired, the approach of using similarity analysis seemed overly involved. Instead, parameters of the virtual vehicle were adapted to suit test rig dimensions, in turn creating a *small* virtual vehicle as well. Since it is not based on an actual vehicle, parameters Θ and d in (4.2) can be chosen arbitrarily.

4.3.4 Torque measurement

For cost and complexity reasons, the clutch system is not equipped with a dedicated torque sensor. During design of the clutch system, it was assumed that the current of the motors could be used instead, which needed additional verification.

For stationary operation ($\dot{\omega}_P = 0$), torque and current are proportional:

$$T_p = k_{m,T} \cdot i_M \cdot \frac{1}{r_{1/2}},$$

with T_p being the torque at the friction plate, $k_{m,T}$ being the proportionality constant for torque detection, i_M being the current and $r_{1/2}$ being the gear reduction ratio.

For verification, experiments using the clutch system were conducted. For these, the fixed motor was removed and exchanged for a lever and a force sensor, which enable measuring the torque. The temporarily rebuilt clutch system is shown in figure 4.5. The normal force applied was $F_N = 0 \ldots 200\,\mathrm{N}$ with 8 individual values and the difference in rotational velocity[37] was $\Delta\omega = 0 \ldots 400 \frac{\mathrm{rad}}{\mathrm{s}}$ with 9 individual values. All 72 experiments

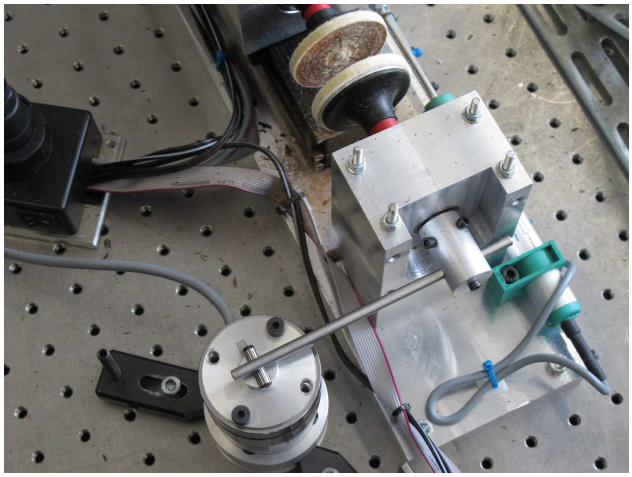

Figure 4.5: Rebuilt clutch system with torque sensor to determine friction model parameters.

[37] For this experiment, $\Delta\omega$ and ω_2 are identical since the secondary plate is fixed.

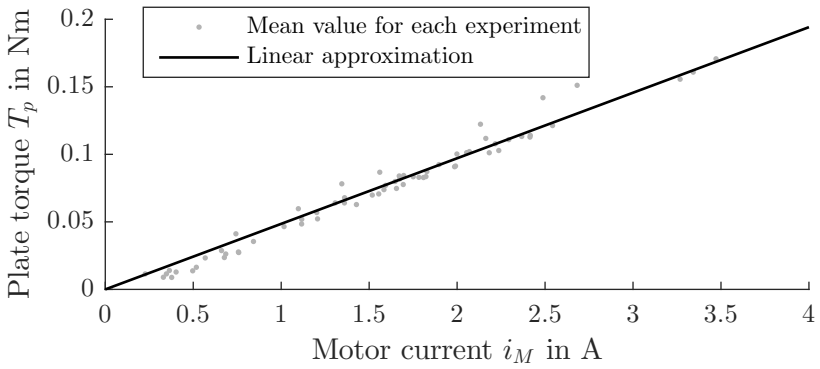

Figure 4.6: Relation of torque to current.

were conducted with the same friction pads. To reduce the influence of systematic errors, e.g. heating-up of the friction pads, the order of the experiments was randomized. For each experiment, the values were set before the drives were enabled. Before storing data, 1 s was given for transient effects such as the motor accelerating and the force controller setting[38] the required normal force. After this, 2 s of continuous operation were recorded before disabling the drives again. The recorded data was averaged for each experiment.

Figure 4.6 shows the mean current and mean torque for each experiment. As can be seen, the linear approximation fits quite well. From this data the proportionality factor was determined to be:

$$k_{m,T} = 0.0566 \frac{\mathrm{N\,m}}{\mathrm{A}}.$$

The manufacturer's nominal value for the motor alone is $k_m = 0.0603 \frac{\mathrm{N\,m}}{\mathrm{A}}$. From these two values, the approximate efficiency of the clutch drive can be computed to be $\eta = \frac{k_{m,T}}{k_m} = 0.94$. While this value is not required for general operation during experiments, it is reassuring to know that no major losses occur in the belt drive.

4.3.5 Normal force control

The normal force acting on the clutch plates serves as system input. Different force trajectories are used to adapt system behavior. For these reasons, normal force of the test rig needs to follow the setpoint precisely, which requires closed loop control. It is measured using a normal force sensor, the corresponding actuator is an electric motor, cf. section 4.3.1, which acts on the clutch plates via the linear guide and spindle. In [Mer14, p. 33], a closed loop control with additional feed forward control for the motor current was proposed. Later on, regular PI control with anti wind up proved sufficient. It computes the desired current of the actuator from the error in normal force. The current is limited by the maximum admissible current specified by the motor manufacturer.

Additionally, a position controller is implemented for auxiliary purposes (e.g. opening and closing of the clutch system for friction pad changes) as PI controller. Since the force controller is only activated at $F_N = 0$ and the position controller tracking accuracy is not

[38] Obviously wrong measurements with e.g. the normal force not having reached a steady state were rejected and repeated.

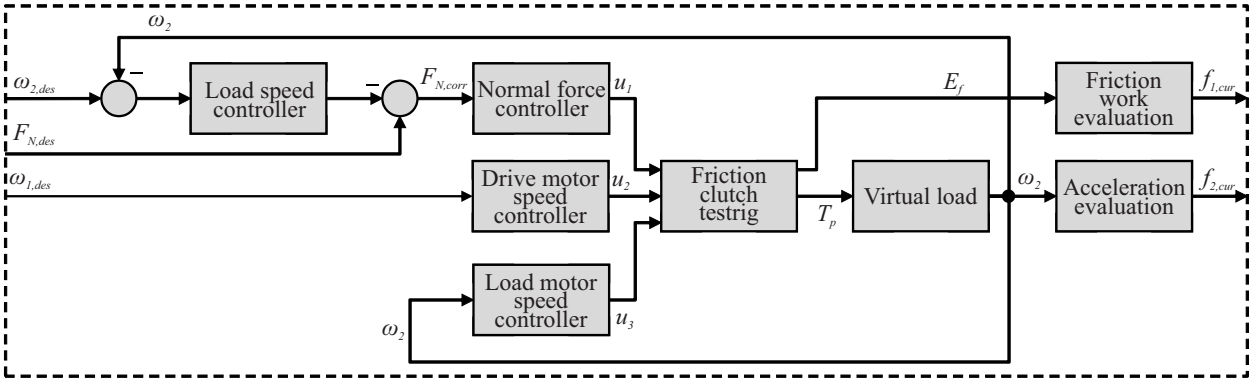

Figure 4.7: Control structure for clutch system.

relevant for actual system operation, hard switching from one to the other is sufficient. At switching, both integrators are reset to avoid errors due to wound-up integrators.

During operation, controlling the normal force alone is not sufficient to guarantee good agreement between optimization model and actual system. Instead, a velocity controller for the load is designed that directly changes the normal force controller setpoint. To allow this, during optimization, nominal velocity values are stored in addition to system input parameters and objective function output values.

Storing nominal values as well allows to compensate perturbations and slight deviations between optimization model and actual system by controlling intermediate values. Without this approach, open loop control, which is highly sensitive against perturbations, would fail or would require a better, more detailed optimization model. The resulting clutch system control structure is shown in figure 4.7.

4.4 Identification of dependability-related objectives

In addition to performance objectives from section 4.2.2, dependability-related objectives that increase dependability when prioritized are required. To find these, a dedicated method was introduced in section 2.4. It consists of five steps, which are now applied to find appropriate objectives for the clutch system in this and in the following section.

In the first step, system dependability is analyzed. A thorough reliability analysis of test rig and clutch plates was conducted [Li15]. One main result was the different possible failure modes of the clutch plates and the conditions necessary to excite each mode. Three modes were found:

Carbonization of clutch plates

> High friction energy leads to heating of the clutch plates, which in turn soften. At first, the surface starts to darken, then plastic deformation sets in and the surface starts melting, and finally the plates start carbonizing. At this point all experiments were stopped for safety reasons, but it can be assumed that driving further friction energy into the plates might also light them.
>
> This failure mode is specific for the material of the plates, felt, and any information gained from this cannot be transferred to actual clutch friction plates. For this reason and for safety reasons, this failure mode is avoided by limiting friction work during each cycle.

Breaking of holder

The felt pad holder, as seen in figure 4.3, is a small sheet metal plate with stamped threads at its center. It transfers most of the torque from motor to plate, friction between holder and plate is secondary. The plate is held to the center of the clutch plate with glue only. This bond breaks at random times during the experiment.

As this is a stochastic failure mode and mainly due to usage far from what the manufacturer designed its product for, and is not at all representative of actual clutch plates, failure due to this failure mode is not desired at all. For this reason, each holder is manually strengthened with additional glue around the perimeter. Regular high viscosity super glue[39] was used.

Low thickness due to abrasive wear

During each clutch cycle, a tiny amount of material is removed from the surface of the clutch plates. This wear is spread evenly over the surface of the plates. It adds up until the plates are very thin and finally tear.

For experiments, failure due to this failure mode is assumed to be reached if combined clutch plate thickness is reduced by 5 mm. All experiments strive to excite this failure mode and reliability control can be designed around it.

The remainder of this thesis is focussed on failure mode *Low thickness due to abrasive wear*. This failure mode corresponds to the components *friction pads*.

The second step is to find a suitable degradation model in order to identify corresponding load factors. The model itself does not need to be parameterized. Wear models for dry friction can be found as far back as [Rey60] or [Arc53]. According to [Rey60, p. 239], „For a given shaft, normal wear of any rubbing surface area element is proportional to friction work it produces[...].“. Similar statements can also be found in [Arc53, p. 985]. The proportionality factor introduced by [Rey60] suggests that to decrease wear of the clutch plates, friction work has to be reduced. It is considered to be a *load factor* for the clutch plates.

In the third step, the load factor needs to be included in the model of system behavior. This model is setup next.

4.5 Modeling the clutch system

For model-based multiobjective optimization, a model of the behavior of the system is required. To form the basis of the required objective functions, fast evaluation time is more important than precision, but it needs to be ascertained that all relevant effects are included. The system model is composed of the virtual load, a friction model which is equivalent to the experimental setup, and all effects required for objective function evaluation.

For modeling the vehicle itself, the virtual load is directly implemented as described in section 4.2.3, specifically equation (4.2). It is augmented with an empirical friction model to compute clutch plate torque T_p.

[39] Toolcraft Ropid 200

4.5.1 Friction model

The torque $T_p(t)$ transmitted in the clutch plates is mainly dependent on the applied normal force $F_N(t)$. A common approach to model this relationship is to assume linear dependency, i.e. $T_p \sim F_N$. However, preliminary measurements shown in figure 4.8 indicate that the torque transmitted by the felt pads used as friction pads decreases over time. This was found to actually be a temperature dependency, the friction pads heat up during each actuation cycle. Since temperature is not measured and a simulation model would increase overall model complexity unreasonably, it is substituted for friction work which forms the basis of heating of the friction pads. This effect needs to be included in a more complex friction model. Due to these limitations, an empirical relationship for velocity-independent torque $T_{p,0}(F_N(t), E_f(t))$ is used. Parameters are normal force $F_N(t)$ and friction work $E_f(t)$.

An empirical fitting of a function with unknown parameters to measurement data[40] is

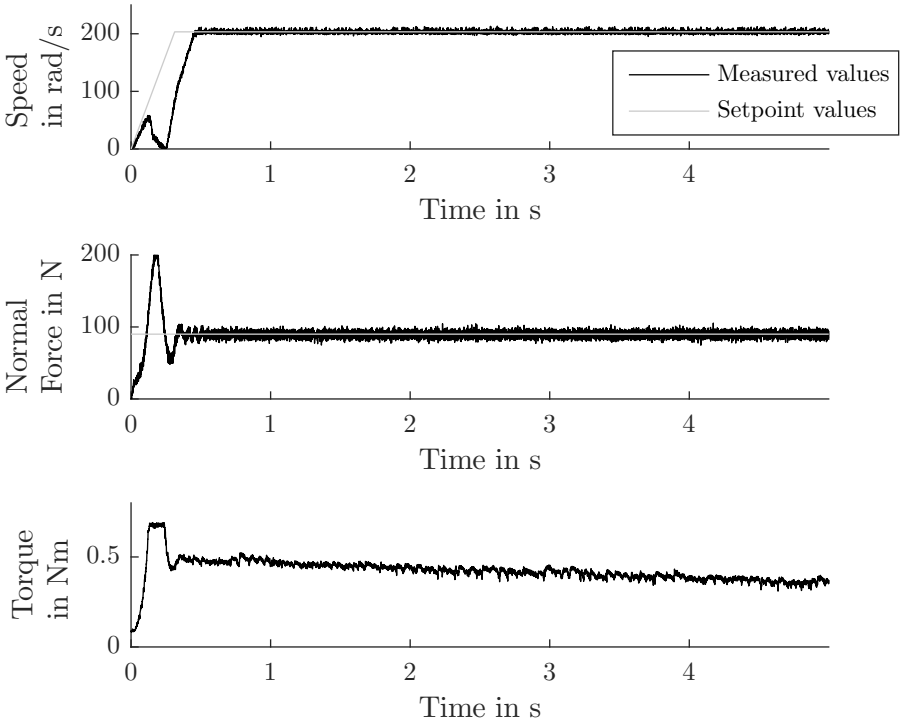

Figure 4.8: Measured velocity of friction plate, normal force (gray: desired values) and resulting torque during one experiment. After an initial overshoot in normal force and subsequent drop in velocity, stationary operating conditions are reached after about 0.5 s. Torque lowers over the time span depicted, which is typical for usage of the clutch plates.

[40]For fitting of function to measured data, Matlab Curve Fitting Toolbox was used.

used. It was found that three main terms can fit measurement data reasonably well:

$$\begin{aligned} T_{p,0}\left(F_N, E_f\right) = &\left(F_N - h_h\right)^2 \cdot h_d \\ &+ h_b \cdot \arctan\left(h_f \cdot F_N - h_a\right) - h_c \\ &+ h_g \cdot F_N \cdot E_f. \end{aligned} \tag{4.3}$$

The first and most important term, which is quadratic in F_N, models the relationship between normal force and torque, which is usually assumed to be linear, but where measurement results showed a slight nonlinearity. The second term $\arctan(\ldots)$ has the purpose of modeling an initial step, since the friction plates transmit a small amount of torque even at $F_N = 0\,\mathrm{N}$. This initial torque probably arises due to the surface of the friction plates: When the plates are in contact, the rough felt transmits torque by means of mechanical interlocking, not by friction. Only for higher normal forces, friction becomes dominant. Third, relationship of normal force and friction work is modeled with a bilinear term. As friction work increases, torque decreases. For excessive friction work, discoloring or even melting and burning of the surface of the friction plates can be observed. During experiments, discoloring cannot be avoided and has little impact on torque, but melting and burning are neither included in the model nor acceptable for reliable operation.

To parameterize and to validate the model, measurement data was obtained with automated experiments. These were conducted with rotational velocity in the range $\Delta\omega = 0 \ldots 300\ \frac{\mathrm{rad}}{\mathrm{s}}$ in 7 steps and normal force $F_N = 0 \ldots 100\,\mathrm{N}$ in 41 steps. The experiment was repeated three times for each pair of values. The resulting time variant data for one of these experiments is shown in figure 4.8. For all 861 experiments, such curves were recorded and friction work E_f of each experiment was computed as

$$E_f(t) = \int_{t_0}^{t_0+t} P_f(\tau)\,\mathrm{d}\,\tau = \int_{t_0}^{t_0+t} T_p(\tau) \cdot \Delta\omega(\tau)\,\mathrm{d}\,\tau.$$

All data was combined and broken down into individual time-independent combinations $\{F_{N,i}, E_{f,i}, T_{p,0,i}\}, i \in \mathbb{N}$, of which one exists for each time step at which measurement data was recorded. Since the resulting number of data points was too large for further evaluations, 10 000 random samples were drawn. Equation (4.3) was fitted to these selected data points by adapting parameters $h_{\{h,d,b,f,a,c,g\}}$ using a least squares fit. The resulting three dimensional approximation is shown in figure 4.9. Friction model values are graphed as shaded surface, where shading also indicates torque. It can be seen that the data points are close to the friction model, but quality of the fitted model cannot be estimated visually. Instead, the error was computed for each selected data point. The error between measurement results and corresponding model values is approximately normally distributed with low variance, as shown in figure 4.10. Correlation between normal force or friction work and error could not be found.

The friction model is created for usage in simulations of vehicle acceleration. During acceleration, the load clutch plate is accelerated until the driving clutch plate velocity is reached. At this time, the difference in rotational velocity $\Delta\omega = \omega_1 - \omega_2$ becomes zero. In coulomb friction, the subsequent lowering of transmitted torque would be modeled as $T_p = T_{p,0} \cdot \mathrm{sgn}(\Delta\omega)$. However, this approach de-stabilizes numeric simulations since slight deviations from $\Delta\omega = 0$ yield fast switching between positive and negative torque. To fully model system dynamics with $\Delta\omega = 0$, switching to an entirely different set

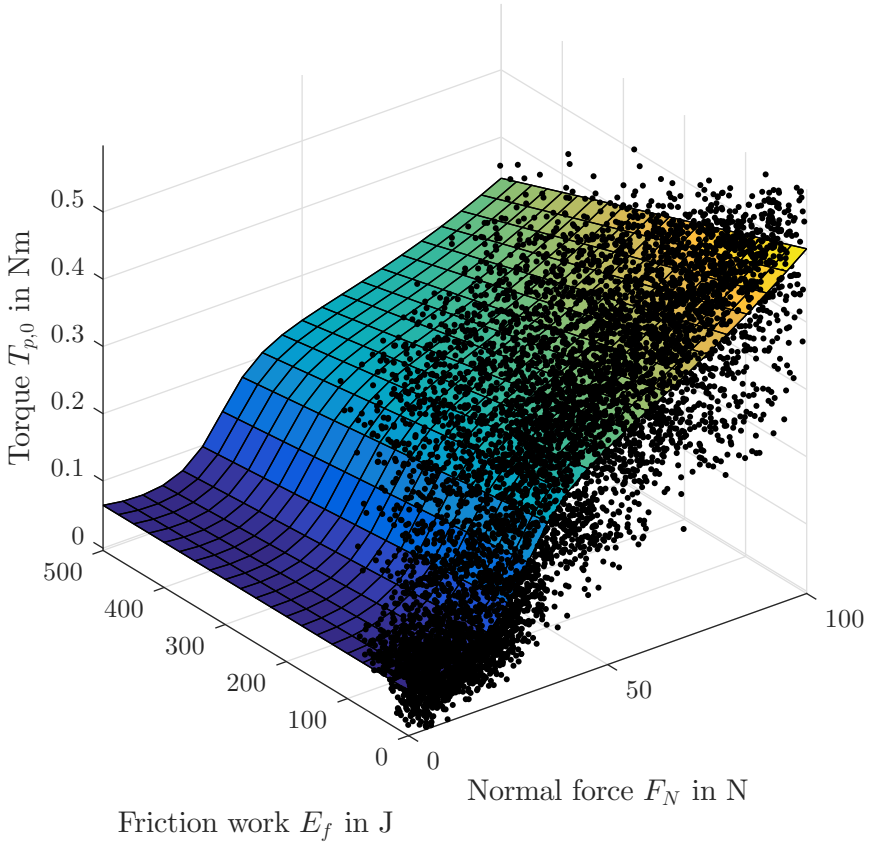

Figure 4.9: Measured data points and fitted approximation function.

of differential equations would be required. To cope with these problems, instead of Coulomb friction, the slightly more complex model according to [Pop10, p. 310] is used. It approximates a coefficient of friction μ, which normally is a fixed value, by a velocity-dependent function $\mu(\Delta\omega) = \mu_0 \cdot \frac{2}{\pi} \cdot \arctan\left(\frac{\Delta\omega}{\omega}\right)$. For both high and low values of $\Delta\omega$,

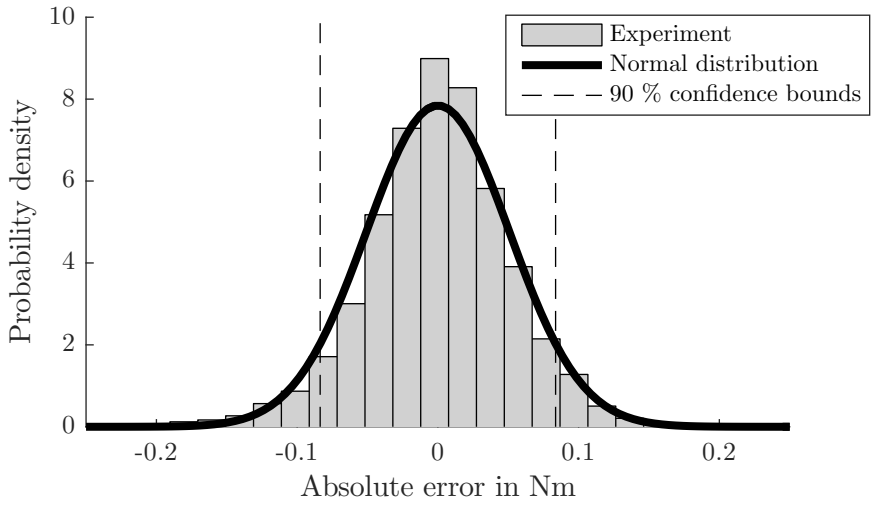

Figure 4.10: Error of fitted approximation function.

a nominal coefficient of friction μ_0 and the velocity-dependent coefficient of friction μ become approximately equal. The less slippage occurs, i.e. close to $\Delta\omega = 0$, the lower the coefficient of friction μ is and the less force or torque is transmitted. The main advantage are continously defined ordinary differential equations if used in conjunction with a model of system dynamics. This idea can also be applied to the more complex model from (4.3). Here, instead of a coefficient of friction, torque is altered directly:

$$T_p\left(\Delta\omega, F_N, E_f\right) = T_{p,0}\left(F_N, E_f\right) \cdot \frac{2}{\pi} \cdot \arctan\left(\frac{\Delta\omega}{\widehat{\omega}}\right). \tag{4.4}$$

The parameter $\widehat{\omega}$ specifies the accuracy for low relative velocities. To balance simulation time with model accuracy, it was set to $\widehat{\omega} = 0.1\frac{\text{rad}}{\text{s}}$.

To use the friction model in simulations, the friction work E_f is required. Similarly to experiment evaluation, it can be computed as

$$E_f\left(t\right) = \int_0^t P_f\left(\tau\right) \mathrm{d}\,\tau, \tag{4.5}$$

with

$$P_f\left(t\right) = T_p\left(\Delta\omega\left(t\right), F_N\left(t\right), E_f\left(t\right)\right) \cdot \Delta\omega\left(t\right). \tag{4.6}$$

Simulation of the full system is conducted in discrete time steps, where integration of P_f to find E_f is computed by the integration scheme. This is only possible for the whole system, which is combined from dynamics of the virtual vehicle (4.2) including friction model (4.4) and friction work (4.5).

These are combined to form a set of differential equations with state vector

$$\boldsymbol{q} = \begin{bmatrix}\omega_2 \\ E_f\end{bmatrix}, \dot{\boldsymbol{q}} = \begin{bmatrix}\omega_2 \\ E_f\end{bmatrix}^{\cdot} = \begin{bmatrix}\dot{\omega}_2 \\ P_f\end{bmatrix}.$$

The full set of differential equations then is

$$\dot{\boldsymbol{q}}\left(t\right) = \begin{cases} \begin{bmatrix} \frac{1}{\Theta} \cdot \left(-d \cdot \omega_2\left(t\right)\right) \\ T_p\left(\omega_2\left(t\right) - \omega_1\left(t\right), F_N\left(t\right), E_f\left(t\right)\right) \cdot \left(\omega_2\left(t\right) - \omega_1\left(t\right)\right) \end{bmatrix} & \text{iff } T_p < T_0, \\ \begin{bmatrix} \frac{1}{\Theta} \cdot \left(\left(T_p\left(F_N\left(t\right), \omega_2\left(t\right) - \omega_1\left(t\right), E_f\left(t\right)\right) - T_0\right) - d \cdot \omega_2\left(t\right)\right) \\ T_p\left(\omega_2\left(t\right) - \omega_1\left(t\right), F_N\left(t\right), E_f\left(t\right)\right) \cdot \left(\omega_2\left(t\right) - \omega_1\left(t\right)\right) \end{bmatrix} & \text{otherwise.} \end{cases}$$

According to section 2.4, step 3, load factors need to be included in the model of system behavior. The load factor for clutch plate degradation that was identified in section 4.4 is friction work E_f. This is already included in (4.5); no further enhancements to the model are required.

4.5.2 Modeling clutch plate degradation

For setup of the reliability control loop, no degradation model is required. However, it is advantageous to have a *full* system behavior model for the evaluation of controller behavior, and for this reason a degradation model is setup.

During the experiments conducted for the friction model in section 4.5.1, the friction

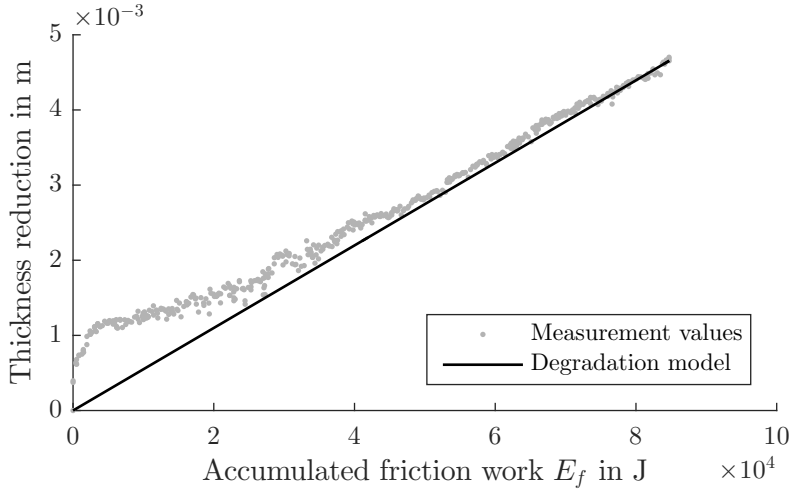

Figure 4.11: Friction work dissipated in the clutch system during the experiments conducted to parameterize the friction model.

pads were wearing down considerably. The main influence for wear due to friction is, according to [Rey60, p. 239], [Arc53, p. 985], the friction work E_f. For each actuation cycle k with time span $t = t_0 \dots t_0 + t_r$, where t_r is the duration of the actuation cycle, the wear volume $W(k)$ is

$$W(k) = p_f \cdot C \cdot E_f(k) \tag{4.7}$$

with proportionality factor p_f and surface area C.

During experiments, homogenuous wear of the surfaces was observed, which implies that energy was dissipated homogenuously as well. The relative lateral velocity of the plates increases with distance from the rotation axis. To obtain homogenuous energy dissipation, a non-homogenuous pressure distribution is required to compensate. While the pressure distribution was not measured, lower pressure towards the rim of the clutch plate holder is highly likely. The holders are made of flexible rubber material, which deforms when the clutch is engaged.

During experiments, wear volume cannot be measured, but plate thickness can be determined quite easily. With homogenuous wear across the surface, the thickness reduction becomes

$$l(k) = \frac{W(k)}{C} = p_f \cdot E_f(k) \tag{4.8}$$

with the proportionality factor p_f. It is assumed that new clutch plates are not worn at all, whereas the plates are fully worn out at $l_{max} = 5\,\mathrm{mm}$[41]. During the experiments, both velocity and normal force were measured. To determine the proportionality factor, the mean for each actuation cycle is calculated. Figure 4.11 shows thickness over accumulated friction work. It can be observed that during the first experiments, fast wearing occured, whereas during later experiments the clutch plates were wearing linearly. From this data, the proportionality constant was found to be $p_f = 4.3687 \cdot 10^{-4} \frac{\mathrm{m}}{\mathrm{J}}$.

Of course, this approach is limited in accuracy since several effects, such as temperature dependency of the proportionality factor p_f, are neglected. However, a good trade-off

[41] The pads are slightly thicker than 5 mm, but to avoid failure of the backing pads, the felt pads are not fully worn out during experiments.

between modeling accuracy and simulation complexity always needs to be found. For multiobjective optimization, simulation speed is critical.

To allow for the setup of a full model of the system including degradation[42], the health index needs to be determined from simulation results. To this end, all prior actuation cycles need to be taken into account. This is achieved by computing the sum of $l(i)$ from cycle $i = 1$ until the current cycle $i = k$. Then the health index for the next cycle $k + 1$ can be calculated by taking the maximum amount of wear l_{max} of the clutch into account. This results in the following relation:

$$HI(k+1) = 1 - \left(\frac{\sum_{i=1}^{k} l_i}{l_{max}} \right). \tag{4.9}$$

4.6 Multiobjective optimization applied to the clutch system

In order to use model-based self-optimization, optimal system parameters need to be determined. This can be achieved using multiobjective optimization techniques, which attempt to minimize several conflicting objective functions at once, cf. section 2.3. The desired system behavior is, according to section 4.2, to engage the clutch and in turn to accelerate the vehicle to engine velocity, i.e. $\omega_1 = \omega_2$, $\Delta\omega = 0$. The optimization parameters which allow an adaptation at runtime are mainly the time it takes to accelerate from rest to full velocity, which is equal to the actuation duration t_r, and the trajectory of the normal force $F_N(t)$ used during actuation. This leads to the multiobjective optimization problem being a multiobjective optimal control problem, see also section 2.3.3. In such a problem, a trajectory over time is required as system input signal.

For the clutch system, a performance objective and a dependability-related objective are required. The performance objective has already been determined in section 4.2.3 to be passenger comfort. The dependability-related objective is given as reduction of load factors, see section 4.4 and equation (4.5). Formulated as objective function, this is

$$f_1 = E_f(t_r). \tag{4.10}$$

Friction work is computed from torque transmitted in the clutch system and relative velocity between plates, which are both already known from system dynamic equations. No further load factors need to be included.

For the fourth step according to section 2.4, additional system parameters might need to be added. Since a change in input parameters does induce a change in load factor E_f, no further optimization parameters are required.

The main performance objective is to have the vehicle accelerate as comfortably as possible. This is expressed based on the comfort evaluation outlined in section 4.2.2:

$$f_2 = \max\left(\dot{\omega}(t)\right), t \in [t_0, t_0 + t_r]. \tag{4.11}$$

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

[42] This is not required for reliability control, but is helpful for first controller evaluations.

input.

Additionally, one important constraint needs to be taken into account: The relative error in velocity $\Delta\omega$ at $t = t_r$ must not be greater than a certain threshold e, so as to avoid the trivial solution of staying at rest:

$$\frac{|\Delta\omega\,(t_r)|}{\omega_1\,(t_r)} < e. \tag{4.12}$$

To conclude the method outline in section 2.4, the objectives and optimization parameters are combined into a multiobjective optimization problem:

$$F_{N,opt}\,(t) = \arg\min_{F_N(t)} \left(f_1\,(F_N\,(t))\,, f_2\,(F_N\,(t))\right). \tag{4.13}$$

4.6.1 Solving the multiobjective optimal control problem

To solve (4.13) subject to constraints (4.12), multiobjective optimal control is used. A direct method was selected, in which $F_N\,(t)$ is discretized into several individual values, of which each is one optimization parameter. This reduces the optimal control problem to a general optimization problem. Additionally, actuation duration t_r is an optimization parameter. For the clutch system with load acceleration, which has comparably slow dynamics, time of the maneuver is discretized into five individual time steps such that one time step has a duration of approximately 1 s to 6 s. For each time step, one force value ist required. Of these, two were set to be $F_N\,(0\ \mathrm{s}) = 0\,\mathrm{N}$ and $F_N\,(t_r) = 100\,\mathrm{N}$, thus $k = 3$ variable force values remain. For times $t_i = \frac{1}{k+1} \cdot i \cdot t_r$, force values $F_{N,i}\,(t_i)$, $i = 1 \ldots 3 \in \mathbb{N}$ are determined by the optimization algorithm. For simulation of system behavior, intermediate values are linearly interpolated.

The discretized multiobjective optimization problem is

$$\boldsymbol{p} = \begin{bmatrix} F_N\left(\frac{1}{k+1} \cdot 1 \cdot t_r\right) \\ F_N\left(\frac{1}{k+1} \cdot 2 \cdot t_r\right) \\ \vdots \\ F_N\left(\frac{1}{k+1} \cdot k \cdot t_r\right) \\ t_r \end{bmatrix}, k = 3, \tag{4.14}$$

$$\boldsymbol{p}_{opt} = \arg\min_{\boldsymbol{p}} \left(f_1\,(\boldsymbol{p})\,, f_2\,(\boldsymbol{p})\right), \tag{4.15}$$

again subject to constraints (4.12).

As numerical solver, GAIO was chosen. It is a software package that implements several algorithms based on box subdivision techniques, which explore parameter space efficiently and are guaranteed to find globally optimal solutions for multiobjective optimization problems. From these algorithms, `k_diskret` was chosen for its reliance on objective function values without the need to supply derivatives as well. More information regarding this optimization algorithm can be found in [SWO+13].

In addition to optimal system parameters $\boldsymbol{p}_{opt}$, the *Pareto set*, objective function values $f_1\left(\boldsymbol{p}_{opt}\right)$, $f_2\left(\boldsymbol{p}_{opt}\right)$, the *Pareto front*, and intermediate values for load velocity $\omega_2\,(t)$, the *nominal values*, are stored for each possible compromise.

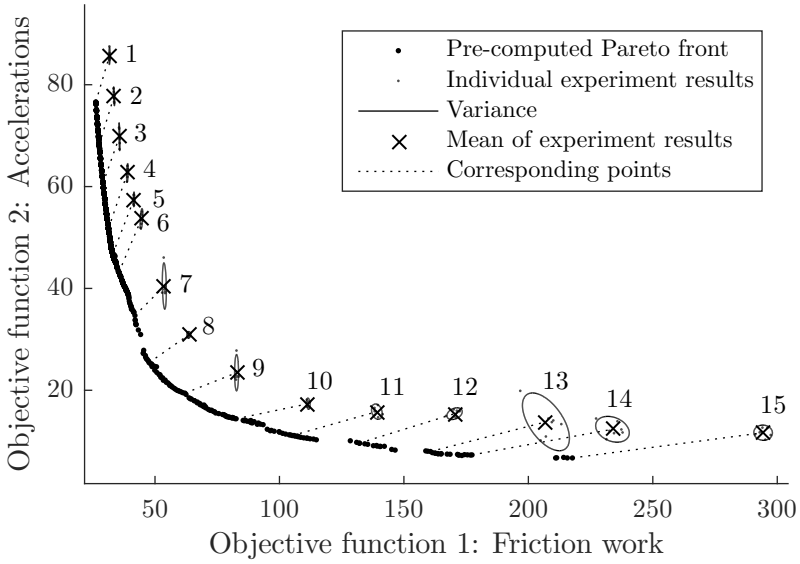

Figure 4.12: Pareto front of clutch system with two objective functions: f_1: minimize friction work and f_2: minimize accelerations. Numbering corresponds to Pareto set numbering in figure 4.13.

Results are shown in figure 4.12. Deviations between system and optimization model cannot be avoided, but in case of the friction model their exact magnitude and their effect on objective function values can only be estimated. To find out whether optimization results are actually valid for the system, they are tested experimentally. For this, fifteen points, which are approximately evenly spaced on the Pareto front, are selected. With each corresponding working point from the Pareto set, multiple experiments are conducted. The normal firce trajectories for the working points are shown in figure 4.13. Parameters and nominal values are applied to the real system and objective function values are computed from experimental data. This yields the *experimental Pareto front*, which is also shown in figure 4.12. For each working point, mean and variance are computed. As can be seen, no experimental result exactly matches the theoretical result. Both a stochastic error and a deterministic error, i.e. a constant offset in any direction, can be observed. The deterministic error is most probably due to model limitations. De-

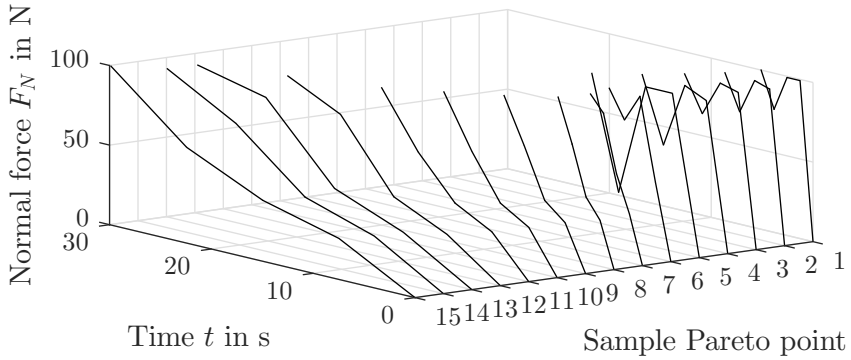

Figure 4.13: Sample trajectories from Pareto set of clutch system.

spite these obvious errors, the shape of the computed Pareto front is closely matched by the experiment result values. Lowering remaining differences would require more complex modeling and in turn more computation time, but an exact match could probably still not be found. For these reasons, the optimization result is assumed to be valid.

4.6.2 α-Parameterization

The basic purpose of the behavior control loop is to control the system behavior based on the so-called α-parameterization, as outlined in section 3.3.2. To implement the α-parameterization, two transforms are required: The s-transform to convert from α-value to system parameters and the inverse s^{-1}-transform to convert actual system objective function values back to α-values. As was shown in section 3.3.2, the forward transform can be deduced from the inverse transform, which needs to be defined system-specific. The definition can be arbitrary as long as it adheres to restrictions (3.6), (3.7), (3.8).

For the clutch system, a definition close to that suggested in [KRK+13, p. 3404] is employed. To this end, a linear function is constructed. Its slope a and offset is given such that it runs through both extrema of the Pareto front.

$$f_2 = f_{2,\max} + a \cdot (f_1 - f_{1,\min}),$$
$$a = -\frac{(f_{2,\min} - f_{,\max})}{(f_{1,\min} - f_{1,\max})},$$

with $(f_{1,\min}, f_{2,\max})$ and $(f_{1,\max}, f_{2,\min})$ being the extrema of the Pareto front. The distance between both extrema is named D_p. The current value of the α-parameterization α_{cur} is the ratio of distance from left extremum to orthogonal projection D_a to maximum distance D_p, scaled such that it matches the given value range $-1 \ldots 1$:

$$\alpha_{cur} = 1 - \frac{2 \cdot D_a}{D_p}. \tag{4.16}$$

A graphic representation of Pareto front with projection and distances is depicted in figure 4.14. To compute distances D_a and D_p, the intersection between orthogonal projection and connecting linear function is required. The orthogonal projection function is

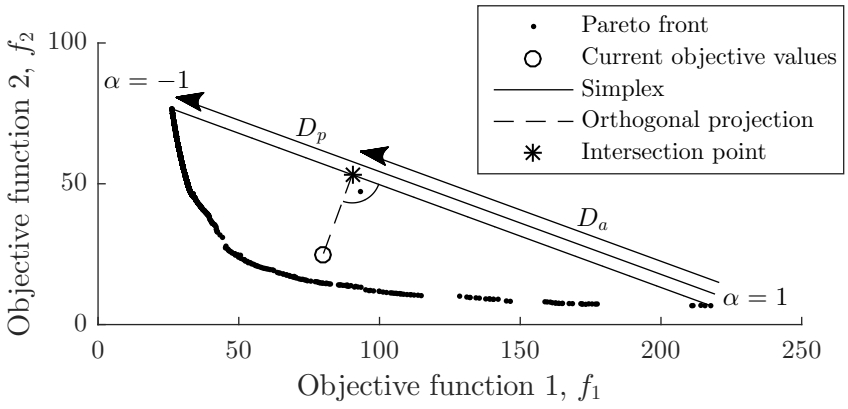

Figure 4.14: α-parameterization.

given by the current objective values $(f_{1,cur}, f_{2,cur})$ and the orthogonal slope b:

$$f_2 = f_{2,cur} + b \cdot (f_1 - f_{1,cur})$$
$$b = -\frac{1}{a}.$$

The intersection point (f_{1s}, f_{2s}) can be found to be

$$f_{1s} = \frac{f_{2,\max} - f_{2,k} + \frac{f_{1,k} \cdot (f_{1,\min} - f_{1,\max})}{(f_{2,\min} - f_{2,\max})} + \frac{f_{1,\min} \cdot (f_{2,\min} - f_{2,\max})}{(f_{1,\min} - f_{1,\max})}}{\frac{(f_{1,\min} - f_{1,\max})}{(f_{2,\min} - f_{2,\max})} + \frac{(f_{2,\min} - f_{2,\max})}{(f_{1,\min} - f_{1,\max})}}$$
$$f_{2s} = f_{2,\max} + a \cdot (f_{1,s} - f_{1,\min}).$$

Finally, distances are:

$$D_a = \sqrt{(f_{1,\max} - f_{1s})^2 + (f_{2,\min} - f_{2s})^2},$$
$$D_p = \sqrt{(f_{1,\max} - f_{1,\min})^2 + (f_{2,\min} - f_{2,\min})^2}.$$

With these, the full inverse s^{-1}-transform is setup. As desired, it reduces both current objective values to a common parameterization value.

4.7 Condition monitoring for clutch plates

Health index of the clutch plates needs to be determined during operation according to requirements as discussed in section 3.4. Since condition monitoring is not a main contribution of this thesis, but a necessity, a simple model-based approach was chosen.

As discussed in section 4.4, the main failure mode is directly related to clutch plate thickness, which can be measured. For this measurement, a dedicated phase was introduced at the beginning of each experiment. In this phase, both motors are at rest and a constant force is applied. Position $\tilde{l}_m$ of the load clutch plate is measured using the built-in displacement sensor. After a change of clutch plates, initial position $\tilde{l}_i$ is measured and set as new reference position. For position measurement, a lower value indicates thinner clutch plates. Clutch plate thickness reduction is computed as:

$$\tilde{l} = \tilde{l}_i - \tilde{l}_m.$$

The measurement phase can be seen in experiment measurement data depicted in figure 4.15. The plates wear faster at the beginning of their lifetime, which is mostly due to wearing in, then a steady state is reached and linear thickness reduction can be observed as shown in figure 4.16. This settling is only brief, but needs to be taken into account when mapping position to health state. For mapping, a linear function is sufficient:

$$\widetilde{HI} = -\frac{1}{L} \cdot \left(\tilde{l} - l_{max}\right).$$

Parameters were determined to be $L = 0.004732$ m, $l_{max} = 5$ mm.

The position determined after each experiment is regarded as one measurement value. As measurements are inherently noisy, filtering is required. This is implicitly done using

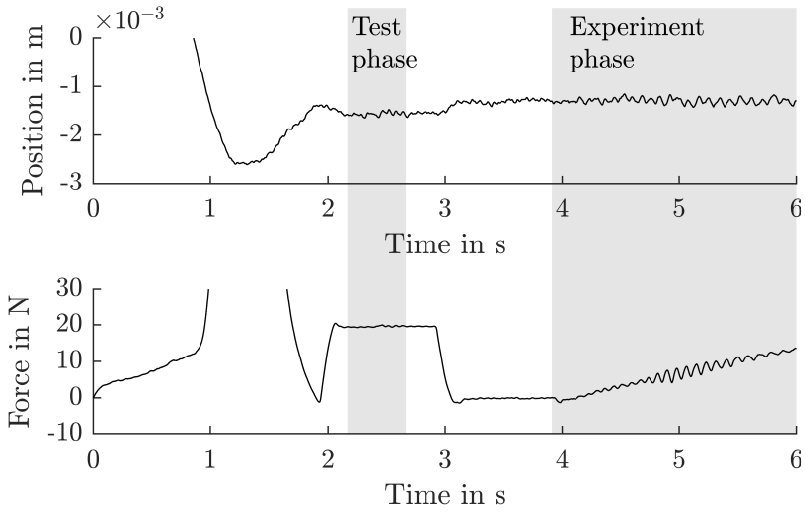

Figure 4.15: Normal force and position of clutch plates over time. After an initial contact, thickness measurement phase is from $t = 2.164 \ldots 2.6640$ s. The actual experiment begins at $t = 3.9135$ s.

the Kalman filtering approach outlined in section 3.3.6. The Kalman filter simulates a model of system degradation parallel to actual system operation, where the system state $\boldsymbol{x} = [HI, \Delta HI_{offset}]^T$ is corrected after each time step such that simulated output complies with actual measured system output. The correction takes both model and measurement into account. This way, measurement noise is removed from the state estimate. Estimation of the state yields an estimate of the error ΔHI_{offset} and a filtered estimate of the true health index HI. The raw recorded signal and the low pass Kalman filtered signal are shown in figure 4.17.

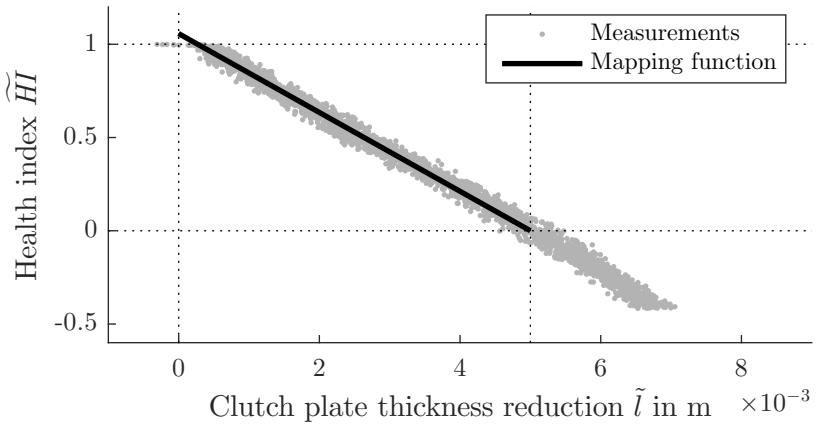

Figure 4.16: Health index of clutch plates over measured thickness reduction $\tilde{l}$.

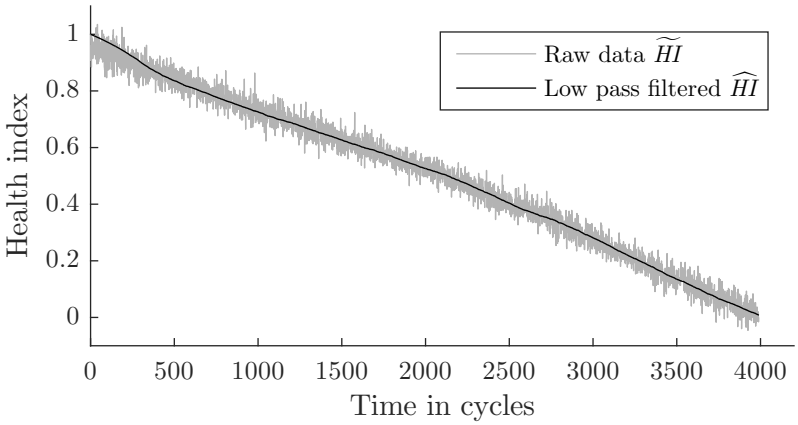

Figure 4.17: Raw and Kalman filter low pass filtered health index data.

4.8 Simulation and experimental results with static working point

To setup and validate controllers, a model of the system was simulated before expensive and lengthy lifetime experiments were conducted. The controllers are setup individually, increasing overall system complexity with each new control loop. Each controller is transferred to the actual system.

To fully assess the benefit obtained with reliability control, a reference failure function of a *normal* system without behavior adaptation is required. To this end, behavior and reliability controllers were fully deactivated and the system behavior was defined by a static working point, which also yielded static system parameters. The nominal working point chosen is $\alpha_{use} = 0$, cf. section 3.2. While the selection of a static working point by means of the behavior parameterization is overly complex and would not be employed for an actual system without behavior or reliability control, it allows for good comparison of the results obtained to those obtained lateron with controllers activated. A simulation and lifetime experiments with a system with this static working point were conducted.

4.8.1 Simulation results

For evaluation of system behavior, the optimization model was used, so no deviations from desired system behavior occured, i.e. $\alpha_{cur} = \alpha_{des}$. Health index decreases continuously over time and the simulated system fails after 4874 cycles. The model is entirely deterministic, and as such, no variance of time to failure is possible. The resulting failure function $F(t)$ is a step function and shows failure probabiliy to be zero for 4874 cycles. After failure, failure probability is one.

4.8.2 Experimental results

As was shown in section 4.6.1 already, slight deviations between computed Pareto front and actually achieved objective values cannot be avoided. In order to obtain results similar to simulation results, an adapted working point has to be selected. To achieve

$\alpha_{cur} \approx 0$, a static working point of $\alpha_{use} = -0.1374$ is required. The constant offset can be explained by the difference in computed Pareto front and in actually achieved objective values, cf. figure 4.12. Behavior parameterization during experiments are shown in figure 4.18. This figure shows desired, used and current behavior parameterization. The mean over all experiments is indicated by a solid colored line, the shaded area indicates variance between experiments. This figure style is utilized for all subsequent behavior parameterization plots; one is shown for each operating mode of the system. To facilitate comparison between different operating modes, axis scaling is equal for all equivalent figures, which makes it difficult to see changes in behavior for the case of a fixed working point. Still, slight deviations of α_{cur} over time can be observed. While variance is small, the mean over all experiments has a minimum at approximately 2500 cycles before increasing again. This can most probably be attributed to inhomogenuous material properties.

Failure behavior for the system with static working point is shown in figure 4.19. It is divided into two subfigures. The upper one shows the health index over time, whereas the lower one shows the failure function. The mean of the estimated health index is indicated by a solid line with the shaded area again being variance. Gray lines in both subfigures show individual failures. In the upper graph, they are merely an event in time; as graphical representation a vertical line was chosen for each individual failure. The failure function is a fit to cumulated failures over time, which are indicated by a stepped plot in the lower graph. After fit, the failure function is evaluated to find the time at which a system has 50% and 95% survival probability. These are indicated with vertical lines. Again, this style of experiment result illustration is used for all operating modes of the system.

To assess stochastic properties of system failure and to find the failure function $F(t)$, a large number of experiments is desired. Due to cost and time constraints, only seven lifetime experiments could be conducted. The margin of error in time to failure is huge, with a 50% survival time of 4345 cycles but a 95% survival probability at just 3848 cycles. It becomes apparent that a great benefit is to be expected from reliability control.

The lifetime found in experiments is considerably less than the deterministic lifetime of the model, a difference that can most probably be attributed to deviations in the degradation model introduced in section 4.5.2. While such a deviation is undesirable,

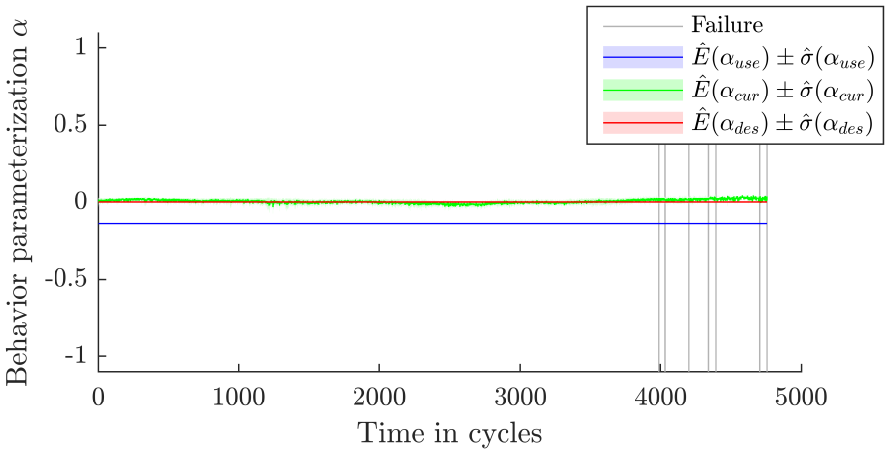

Figure 4.18: Resulting behavior parameterization of system with static working point.

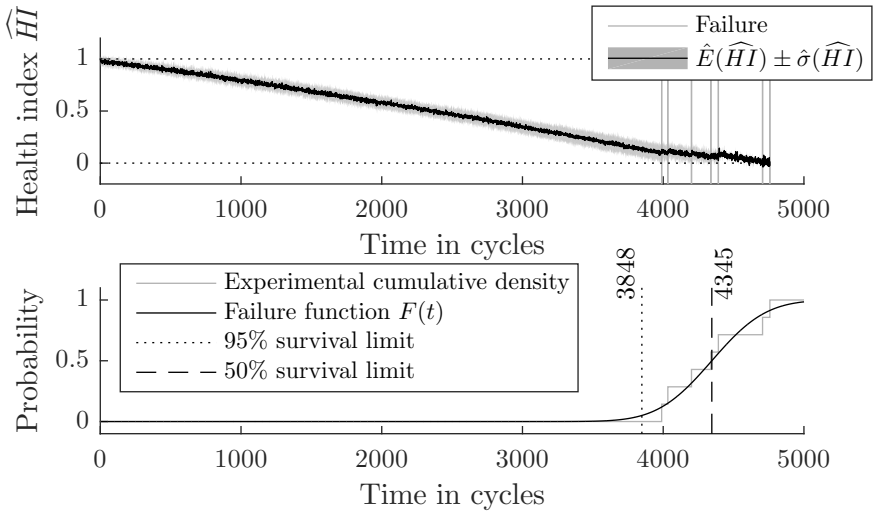

Figure 4.19: Experimental results for systems with static working point $\alpha_{use} = -0.1374$.

it cannot be avoided completely, which would make any kind of reliability control that relied on such a degradation model unreliable.

4.9 Setup and validation of behavior control

In the experiments with static working point, a constant offset was sufficient to obtain close fit of α_{cur} to α_{des}. Deviations, probably due to inhomogeneous material, could be observed though and the static offset works for a single working point only. For dynamic working point selection, behavior control according to section 3.3.2 is implemented. It aims at correcting the discrepancy between computed Pareto front and actual objective values, which were also discussed in section 4.6.1. As this is the inner loop of a two-stage cascaded control loop where reliability control forms the outer loop, separate setup and testing is advisable.

4.9.1 Setup of the behavior control loop

With the s-transformation and the inverse s^{-1}-transformation known from section 4.6.2, the controller can be setup according to section 3.3.2. As was shown in section 3.3.3, specifically (3.9), transfer function of the full behavior control loop is $G_\alpha = 1$ given that no perturbations occur and that optimization model represents system behavior perfectly. This also means that the feedforward part of behavior control would be sufficient, closed loop feedback control would not be necessary and controller parameters became irrelevant. Figure 4.12 shows that a good fit between optimization and experimental validation can be achieved, but that the representation is not perfect, i.e. $G_\alpha = 1$ does *not* hold. Since the model used in lifetime simulations is the same one used for optimization, it cannot be used to determine suitable controller parameters. For this reason, actual controller parameters need to be chosen empirically; parameters of the controller are determined based on observed system behavior. Step response was chosen as controller criteria since system behavior is nonlinear, rendering controller design techniques for linear systems unsuitable. The response to a step in desired behavior parameterization

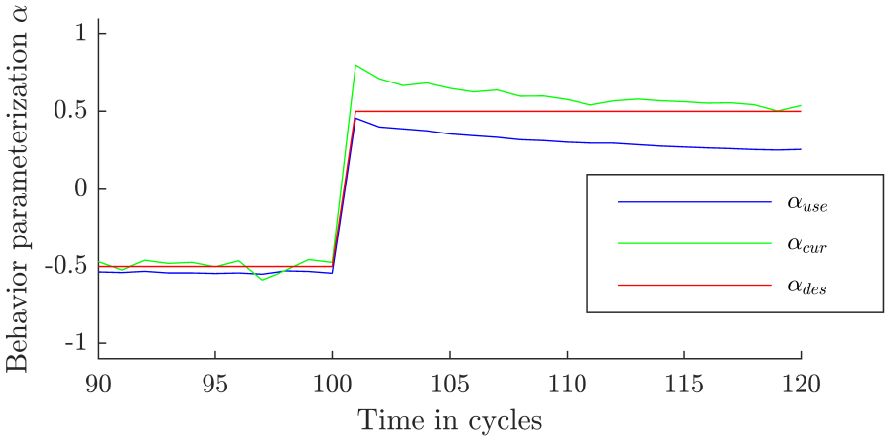

Figure 4.20: Step response of behavior controller.

from $\alpha_{des} = -0.5$ to $\alpha_{use} = 0.5$ of the behavior controller is shown in figure 4.20. At 100 cycles, the desired behavior is changed. The feedforward part of the behavior controller reacts immediately and changes α_{use}, but creates some overshoot in α_{cur}. After approximately 20 cycles, this is reduced by the feedback part of the controller such that $\alpha_{cur} = \alpha_{des}$. No oscillations of the controller can be observed.

4.9.2 Simulation results

The behavior controlled system is simulated as close to prior simulations as possible. To this end, $\alpha_{des} = 0$ was selected as constant setpoint. As expected, simulation results do not differ from the system with static working point, as $\alpha_{use} = \alpha_{des}$. These results do not justify graphical representation.

4.9.3 Experimental results

As was discussed, the behavior controller corrects system behavior such that desired behavior parameterization α_{des} is achieved despite deviations between optimization model and actual system. It does so by adapting the working point such that the current behavior parameterization value is equal to desired behavior parameterization value. In experiments, quick adaptation of α_{use} can be observed to obtain $\alpha_{cur} \approx \alpha_{des}$. Controller parameters were selected such that a steady working point is reached after only about 20 cycles but without overshoot.

Again, 7 lifetime experiments were conducted and the failure function $F(t)$ was determined. Results are shown in figure 4.21.

Similarly to prior experiment results, a deterministic deviation in behavior, this time in the corrected value α_{use}, can be observed at approximately 2500 cycles. Failure behavior, shown in figure 4.22, was changed slightly. 50% survival time is at 4204 cycles and 95% survival probability at 3797 cycles. A small influence of the behavior controller on failure behavior can be observed, but it cannot be estimated whether this is due to advantageuous experimental conditions, which can yield a systematic error, due to a small number of experiments or an actual advantage. This is also not necessary though, as the behavior controller does not serve the purpose of increasing system reliability, it

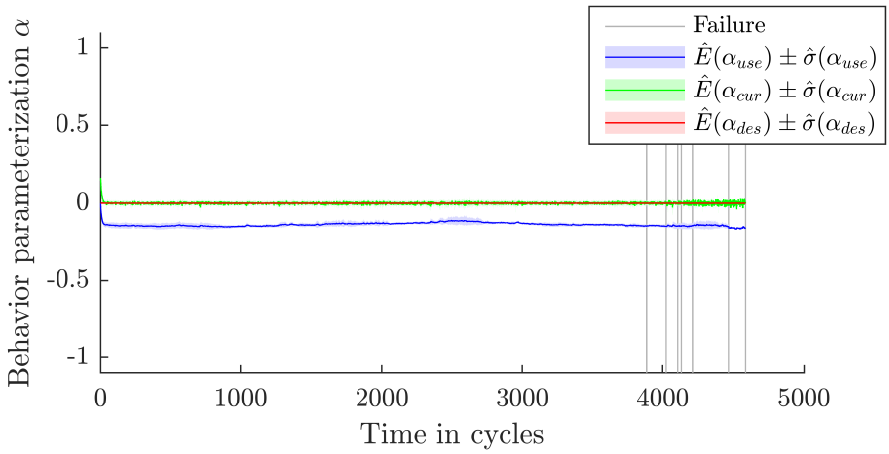

Figure 4.21: Dynamically adapted behavior parameterization of behavior controlled experiments.

only needs to make system behavior more manageable for the reliability controller. But at this point, it can safely be assumed that the controller is working and is able to control system behavior as desired.

4.10 Setup and validation of reliability control

To compensate for non-deterministic failure behavior and in turn to reduce the variance of time to failure while having as good system performance as possible, reliability control is used. It is the outer loop of the full two-stage control loop and interacts with the underlying behavior controller by means of behavior parameterization α_{des}. This controller is based on condition monitoring to determine the current health index, which serves as controlled variable. Since the estimated health index has some uncertainty, a safety margin needs to be found such that early failures, i.e. failures before the desired operating time is reached, are avoided. In a first step, this safety margin is determined

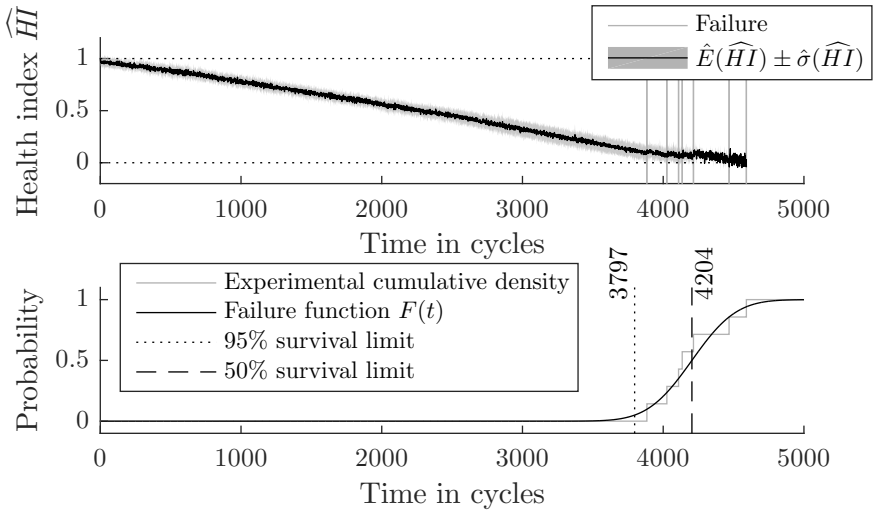

Figure 4.22: Failure behavior of behavior controlled system.

from all prior experiment results.

4.10.1 Health Index at Failure

In accordance with section 3.2, desired estimated health index at specified lifetime $HI_{des,end}$ needs to be defined. This is based on all prior experiments without reliability controller, but could be updated with each new experiment data. Such updating was not implemented, instead a static desired health index $HI_{des,end}$ was used. Each data point is the result of a full lifetime experiment until failure. At this stage of controller setup, only results from systems with static working point and of systems with behavior control are available. These are in total fourteen experiments.

As can be seen in figure 4.23, despite a limited amount of data, thickness of the clutch plates at time of failure is approximately normally distributed. The thickness data serves as basis of the health index estimator, which is not capable of measuring actual health index HI directly. Instead, the algorithm introduced in section 4.7 is used to estimate health index $\widehat{HI}$. The estimation algorithm lets the distribution of the estimated health index at failure $\widehat{HI}(t_f)$, also shown in figure 4.23, deviate from the distribution of thickness. Especially a slight negative skewness of the data, which is shown as gray stepped plot, can be observed. However, while such skewed behavior could be modeled with more complex probability distributions such as the Weibull distribution, it is not clear whether this effect is due to actual skewedness or due to limited data. Since the actual type of distribution is unknown, a normal distribution $\widehat{HI}(t_f) \sim \mathcal{N}\left(\hat{E}\left(\widehat{HI}(t_f)\right), \hat{\sigma}\left(\widehat{HI}(t_f)\right)\right)$ is assumed for both thickness and estimated health index at failure. From the distribution of the estimated health index at failure $\widehat{HI}(t_f)$, the value for $HI_{des,end}$ given an arbitrary desired reliability $R_{spec} = 95\%$ can be computed by solving (3.4).

Later on, results from different reliability controller operating modes were available.

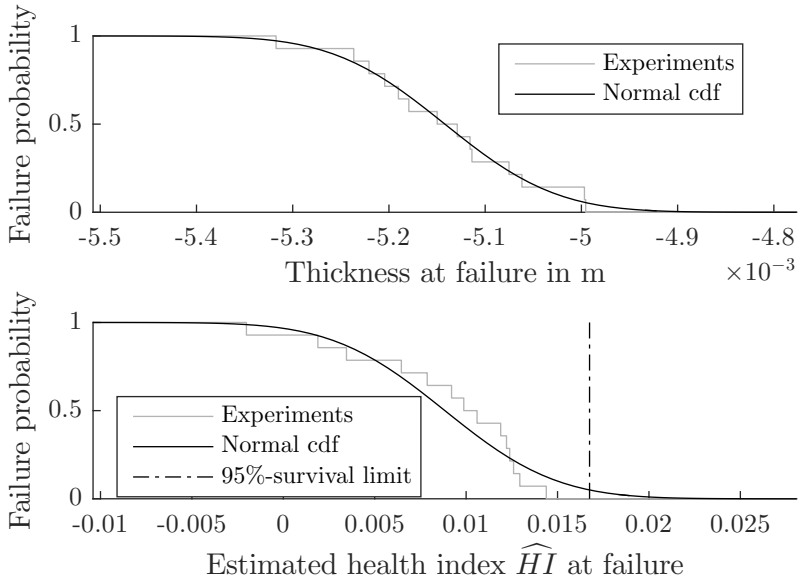

Figure 4.23: Experimental density and fitted cumulative density function of friction plate thickness and of estimated health index $\widehat{HI}$ at failure for systems with static working point and for systems with behavior control.

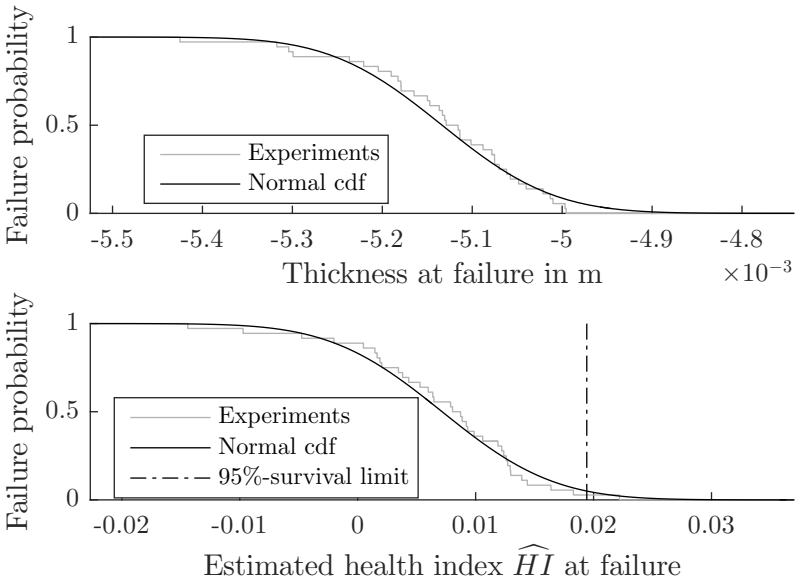

Figure 4.24: Experimental density and fitted cumulative density function of friction plate thickness and of estimated health index $\widehat{HI}$ at failure for all experiments, also those including reliability control.

These increased the number of total data points to 35. With the increased amount of data, a new fit was conducted to validate the assumption that estimated health index at failure $\widehat{HI}\,(t_f)$ is normally distributed. Figure 4.24 shows the full data and is good evidence that normal distribution was a suitable choice.

4.10.2 Setup of the reliability control loop

The reliability control loop was implemented as introduced in chapter 3. Mainly, the model predictive control for system parameterization from section 3.3.7 is implemented. Controller weights were chosen empirically and were selected to be $w_x = 0.95$, $w_u = 0.05$. Prediction horizon for model simulation was selected to be $m = 13$ time steps, i.e. actuation cycles. To solve the optimal control problem, Matlab's `fmincon` with `active-set` algorithm was selected. Model parameter v was determined to be $v = 1.034542$ according to section 3.3.6.

Kalman filter covariance matrices were selected as

$$\boldsymbol{R} = 1 \cdot 10^{-2}$$
$$\boldsymbol{Q} = \begin{bmatrix} 1 \cdot 10^{-7} & 0 \\ 0 & 1 \cdot 10^{-4} \end{bmatrix}$$
$$\boldsymbol{V}_0 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

Kalman filtering steps prediction and update using (3.20), (3.21), (3.22), (3.23), (3.24), (3.25), (3.26) follow after each actuation cycle.

The multiobjective optimal control problem from section 4.6 yields Pareto front and set as required for reliability control. α-parameterization is restricted to values $-1 \ldots 1$.

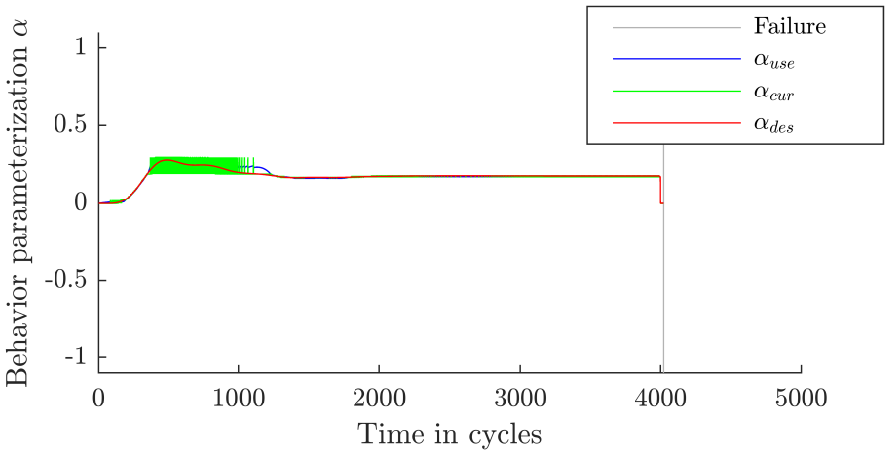

Figure 4.25: Adapted behavior during simulation of full reliability control loop.

4.10.3 Simulation results

To find good controller parameters, the full control loop was simulated. Results of the behavior adaptation are shown in figure 4.25. Slow behavior adaptation with some overshoot before settling for a stationary working point can be observed. Also the effect of discrete working points can be observed in this simulation quite well. The controller output is a continuous variable α_{use}, but the s-transform then selects the *closest* working point from all individual elements in Pareto front and set.

Between ≈ 500 and ≈ 1200 cycles, the controller selected working point is in a region of the Pareto front where the number of points is so low, disconnected parts of the front have formed. These can be seen in figure 4.12 quite clearly. Small controller adaptations result in a jump in working point. Model precision also yields a jump in α_{cur}. While this is a general problem if the total number of working points is very low, for the system at hand actual experiments were not affected by this.

The resulting failure behavior shows a lowered time to failure with the estimated health index $\widehat{HI}$ being very close to desired health index HI_{des}. The deterministic time to failure is slightly larger than 4000 cycles since $HI_{des,end} > 0$.

4.10.4 Experimental results

For experimental validation, the full control loop was implemented for the clutch system. Parameters are not changed from the simulation loop, but due to deviations between model and system, actual system behavior differs from simulations. Figure 4.26 shows the resulting behavior adaptation. It can be observed that at the beginning, the controller initiates intense behavior adaptations, with large variance of desired system behavior parameterization α_{des} between systems. At the same time, the health index, shown in figure 4.27, has large variance during the first ≈ 500 cycles as well. This can most likely be attributed to different wearing behavior of the plates at the beginning of each lifetime experiment.

Clutch plates are not mounted perfectly perpendicular to their rotational axis and their surface has some irregularities. Due to these two effects, at the beginning of their lifetime only a small fraction of the total surface area is in contact. Once the bulges are

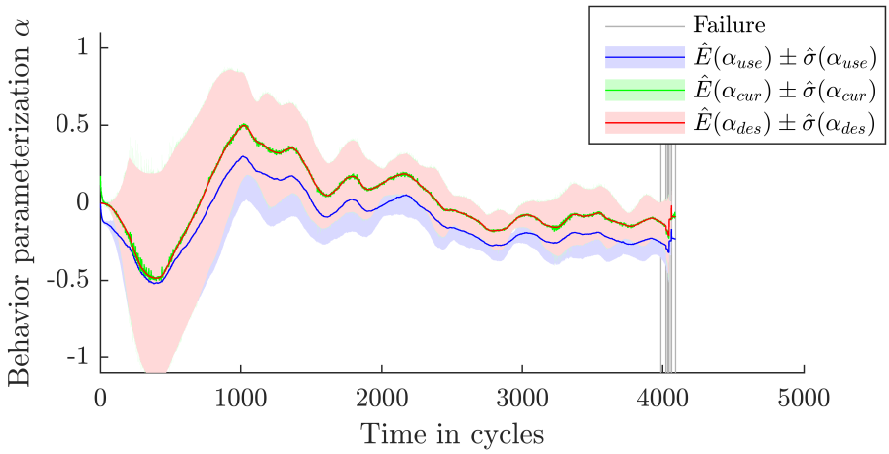

Figure 4.26: Behavior adaptation of experiments with reliability controlled systems.

ground down and the surface is planar, the whole area is in contact. The health index estimator does not take this into account. Resulting thickness reduction, which is proportional to estimated health index $\widehat{HI}$, is fast during the initial wear-in phase, but slows down with increasing contact area. Wear-in is different for each experiment, which reduces estimation accuracy during the first ≈ 500 cycles. During wear-in, the controller compensates deviations between desired health index HI_{des} and estimated health index $\widehat{HI}$ by a lowered desired behavior parameterization α_{des}, which leads to the controller behavior observable in figure 4.26.

Failure behavior, as shown in figure 4.27, is improved dramatically when compared to prior results of uncontrolled systems. 50% survival time at 4047 cycles is only slightly larger than desired lifetime, with the 95% survival limit being at 3995 cycles. From these results, it can be deduced that the reliability controller works as expected.

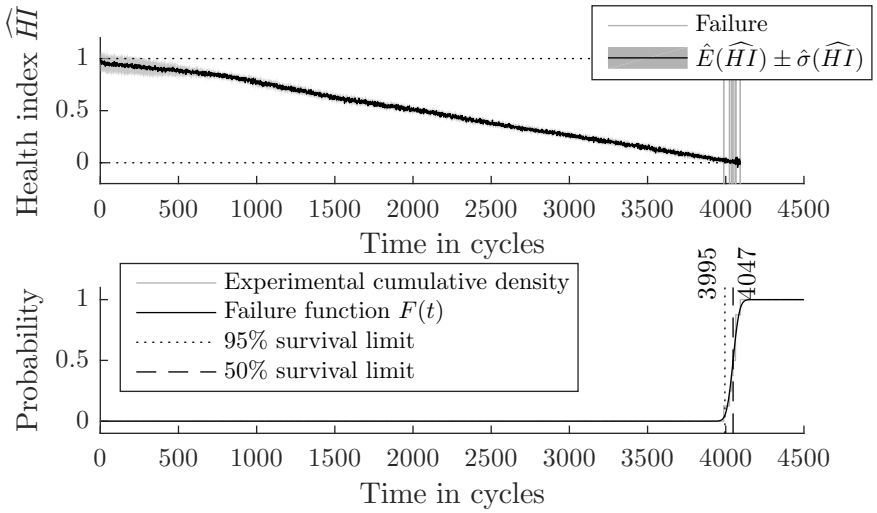

Figure 4.27: Failure behavior of clutch plates with reliability control.

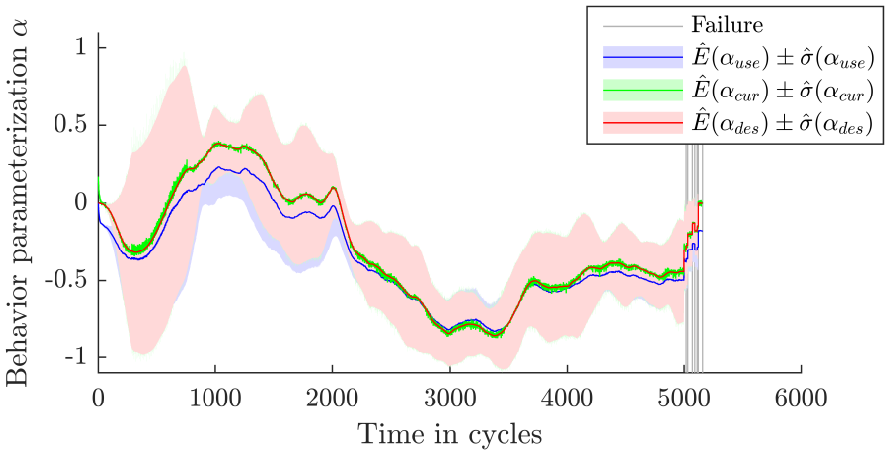

Figure 4.28: Reaction of reliability controlled system to changed requirements.

4.10.5 Reaction to changed requirements

A main goal for reliability control is to permit an inversion of the current approach to maintenance planning: Instead of creating the maintenance plan to suit failures, failures shall suit the maintenance plan. This requires the possibility to change requirements at runtime, most importantly to change the time until failure. To evaluate controller response to such changed requirements, the desired lifetime was increased after 2000 cycles to be 5000 cycles. This necessitates a different working point that is selected by the controller as shown in figure 4.28. While the changed behavior indicates that comfort is compromised, failure behavior is as desired. Results are shown in figure 4.29. While the estimated health index does not differ much from prior experiments before 2000 cycles, afterwards changed wear rate can be observed. 50% survival time is 5086 cycles with 95% survival probability being 5004 cycles, close to the desired 5000 cycles.

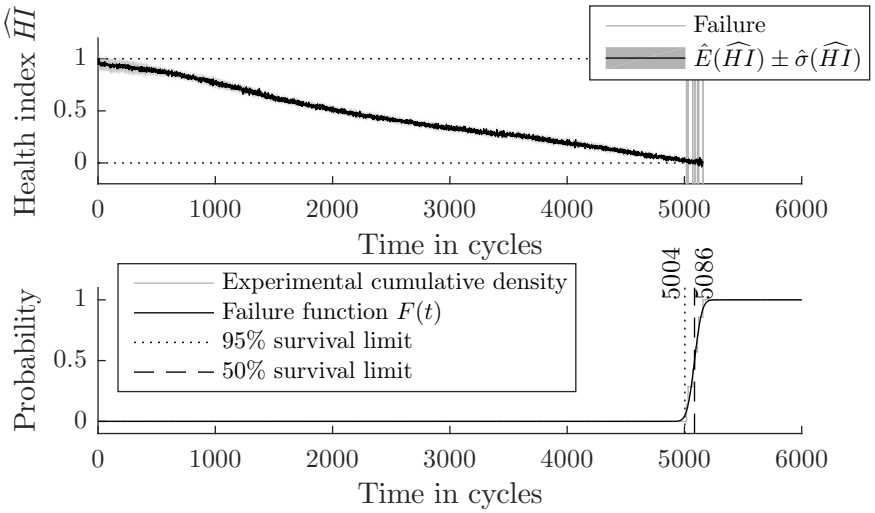

Figure 4.29: Failure behavior of reliability controlled system with changed requirements.

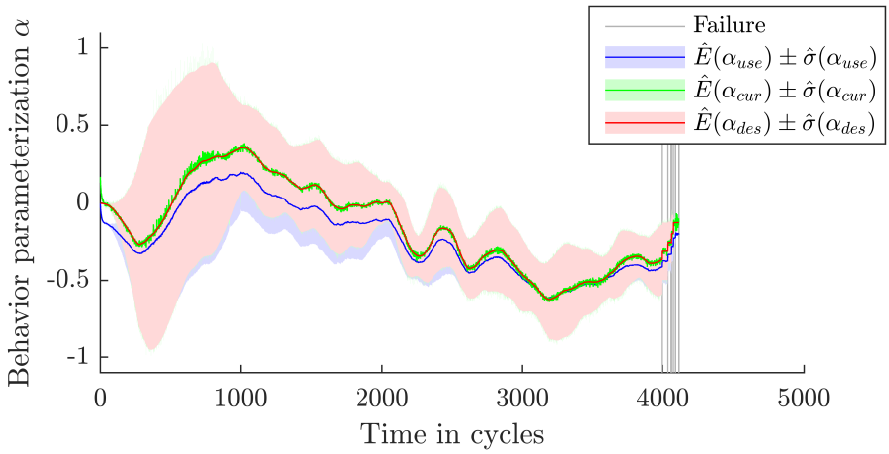

Figure 4.30: Behavior adaptation system with reliability control and changed wearing behavior. At 2000 cycles, part of the clutch plates is cut off.

4.10.6 Reaction to disturbances: Changed wearing behavior

To evaluate suitability for compensation of faults within the system itself, experiments with changed wearing behavior were conducted. This was achieved by stopping system operation after 2000 cycles, cutting part of the felt friction pads off[43], and resuming operation. After cutting, system behavior is adapted to compensate, as shown in figure 4.30. Failure behavior, as shown in figure 4.31, is virtually unchanged from normal reliability controlled failure behavior as demonstrated in section 4.10.4. Any differences visible in the graphs can only be attributed to the limited amount of experimental data that was available for validation.

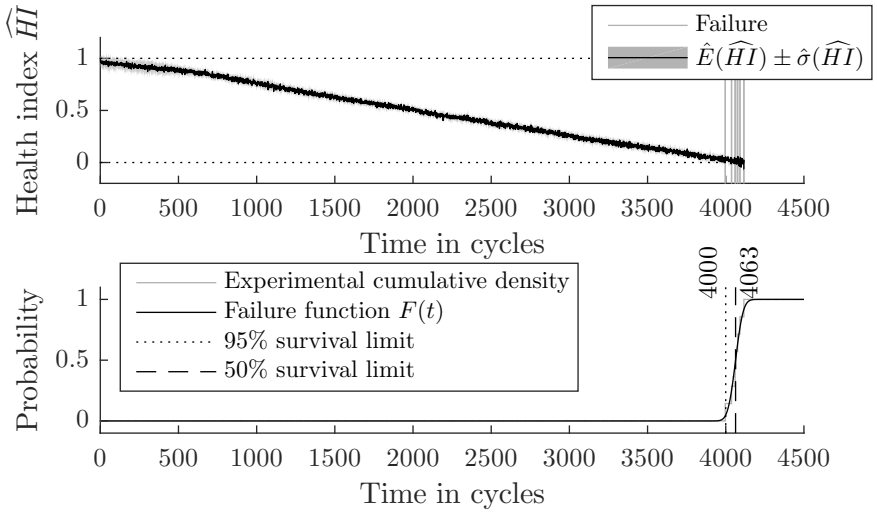

Figure 4.31: Failure behavior of clutch plates with reliability control and changed wearing behavior.

[43] Manual cutting leads to some uncertainty, but in all cases *accelerates* wear.

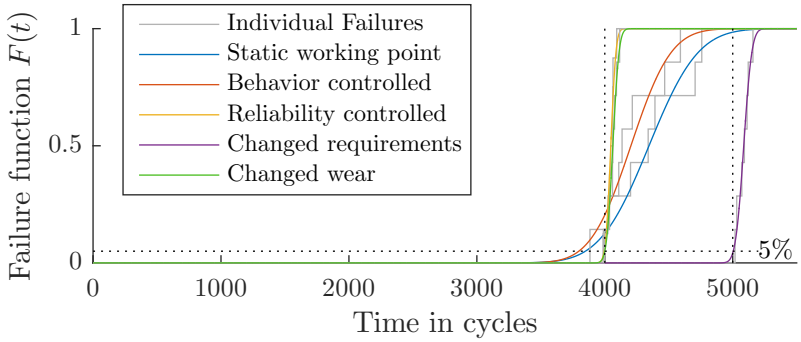

Figure 4.32: Influence of different controller modes on system failure behavior.

4.11 Summary of experimental results

Main results from Figs. 4.19, 4.22, 4.27, 4.29, 4.31 are summarized in figure 4.32. The most important difference between results is the slope of the failure function: With increasing controller complexity, the failure function changes from high variance to being close to the ideal step function. It can be seen quite clearly that the systems with static working point and the behavior controlled system have very high variance. The difference among the two is neglectable and probably not due to behavior control. On the other hand, the reliability controlled system is able to reach the pre-defined 95% survival time precisely. At the same time, 50% survival time is lowered. This cannot even be impeded by changed wearing behavior, as was shown in experiments with reliability control enabled were parts of the friction pads were cut off during experiments. This corresponds to better known failure times and increased system performance. If, on the other hand, longer average system lifetime is desired, this can be achieved as well by selecting an appropriate working point. An example of such operation is given by changing desired system lifetime to 5000 cycles, which are reached with little variance as well.

From these experiment results, it can be deduced that reliability control operates as expected. Variance of time to failure can be reduced with either increased system lifetime or better system performance.

5 Conclusion and Outlook

Self-optimizing systems allow an adaptation of their behavior during operation. They can adapt to changed operating conditions, changed requirements or changes in the system itself. The behavior adaptation is based on the objectives that the system pursues. A new trade-off between objectives is selected during each adaptation cycle, and the working point is always optimal with regard to these objectives. Possible working points to choose from are found using multiobjective optimization, which is conducted offline before operation. This capability can be used to adapt system behavior to the current reliability of the individual system, which is advantageous for maintenance planning. Prior implementations of reliability-adaptive systems not based on self-optimization had individual drawbacks. Mainly they are limited to reacting to discrete events, they are overly system-specific or they require a degradation model, which is difficult to setup.

A self-optimization based reliability controller selects the current working point from multiobjective optimization results based on current health index and desired health index. The current health index is estimated using condition monitoring. As controller algorithm, model predictive control is used. The required model of system degradation is introduced as generic model that is parameterized automatically during operation. This resolves the problem associated with many prior approaches to reliability adaptation, which require an explicit model of system degradation during controller setup.

An automatically actuated single plate dry clutch is introduced as application example. It operates a maneuver that is characteristic for system degradation, which is the acceleration of a motor vehicle from rest. The vehicle and its motor are modeled as virtual load, whereas the clutch plates are physical and actual wearing can be observed. By altering the normal force trajectory, a trade-off between comfort for passengers and wear can be selected. Lifetime experiments were conducted to show the advantages of reliability control.

Experimental results show that the reliability controller works as desired. It is capable of lowering variance of time to failure and to make system failure behavior more predictable. At the same time, it maximizes system performance and allows for an adaptation of desired useful lifetime.

A main prerequisite to controller performance is estimation of the current health index of the system. In the experiments, wearing-in leads to some uncertainty of the health index at the beginning of each experiment. This is in turn compensated by a behavior adaptation, which leads to oscillations during the wear-in phase. This would not be necessary if better condition monitoring was employed.

Self-optimization as basis of a novel method for reliability adaptation helps to contribute against two main reasons for obsolescence:

Strategies to reach a guaranteed minimum lifetime and to prolong product lifetime:

> The reliability controller is able to adapt system behavior such that desired lifetime is reached. If faults occur, e.g. faster wearing of a component as in the experimental

validation, these are compensated by adapted system behavior. Desired product lifetime can also be prolonged, but only at the cost of system performance.

The reliability control introduced in this thesis lowers the margin of error of time to failure by changing system operation. This benefits weaker individual systems with lower than usual wear margin most, as for these systems wear is reduced by the controller and early failures are avoided. However, it can also be used to lower the time to failure of strong systems, i.e. systems with longer than average time to failure. For these systems, usage is increased so that they fail earlier. However, selecting a working point with higher wear also improves other objective function values, which in turn offer a true benefit to system users. So while the reliability control system can also be used adversely to decrease time to failure of strong systems, the user also benefits from this.

Strategies to prolong product usage duration:

Whether an old product is continued to be used is highly dependend on its performance. If the product becomes obsolete due to low performance, the controller could also be used to increase system performance at the cost of lifetime. This could increase the actual usage time and shift the reason for obsolesence to material obsolescence, thus a benefit could be obtained.

Of course both of these aspects contradict one another. Since system operation is optimal with regard to pre-determined objectives, of which one is for system performance and one for reliability, an improvement in both objectives is impossible. If both shall be improved at the same time or product usage duration has to be increased by increasing performance while keeping the lifetime as before, structural changes are required to the system or product.

5.1 Suggested future work

While the work documented within this thesis has reached an end, the development of reliability-adaptive systems is far from finished. Reliability control is one promising way to implement reliability adaptivity, but of course there are further developments and applications.

5.1.1 Improvements to the reliability controller itself

The reliability controller introduced in this thesis is a combination of work from many different fields. First and foremost, self-optimization with all prior research conducted mainly at Paderborn University is to be mentioned, but it also builds on multiobjective optimization, condition monitoring, behavior modeling, model predictive control and parameter estimation techniques. Each of these fields is still ongoing further research, which will further improve the methods and algorithms. All of these could be tied into the existing framework, which is open for further enhancements.

To obtain good controller performance, the health index must be estimated as precisely as possible. In condition monitoring, phase shift of the estimation or the delay introduced by algorithms is largely neglected. Slow decision making without feedback to the process does not suffer from such delay, but controller stability does. For reliability control, *fast* and accurate condition monitoring is required.

Another outer loop that plans failure behavior for each individual system is able to adapt the desired health index such that strong behavior influences at the beginning of operation due to wear-in are avoided. This could also be tied in with maintenance planning and take other systems into account, making it a global supervisory setpoint generator.

At this time, only two objectives were taken into account: performance and dependability. If additional performance objectives exist or if multiple failure modes require reliability controlling, reliability control from this thesis needs to be extended accordingly. The main limiting factor is the behavior parameterization α, which needs to be defined on multiple dimensions. With a suitable definition, one or more objectives can be decoupled from the others. This way, e.g. a trade-off between failure modes could be selected while keeping system performance as desired. Then either one reliability controller could be setup for each individual α-value or a common model predictive control for all values together could be used.

5.1.2 Application as intelligent load balancing

Clutch systems are still required for many vehicles using an internal combustion engine for propulsion. Only recently, innovations such as serial hybrid drives are starting to supersede them in certain applications. Regular clutch systems will most likely remain in use in applications where efficiency of the drive train is critical and where low variations in load are occuring. One such application are agricultural tractors, which need to be able to work efficiently with high loads, low speeds and long stationary operating conditions.

In [GCO13], a shift concept for a powershift transmission is introduced which makes it possible to accelerate an agricultural tractor from rest without the need for a dedicated starting clutch. Regular powershift transmissions have a set of small clutches for the selection of gears and one main clutch for starting up. The small clutches are designed for the frictional losses that occur when changing gears under load, they are not suitable for the high losses at start up. The main clutch is used only for starting from rest and not opened afterwards. The new approach presented in [GCO13] is to use two small gear changing clutches with their individual gear pairings in conjunction. They now work at different velocities, but are acting in parallel until the lower gearing is completely synchronized. At this point, the other clutch is opened and the lower gearing is fully engaged. However, the algorithms introduced in [GCO13] do not take the wear of the clutches into account. Instead, one leading clutch is engaged until its power limit is reached, then a supporting clutch kicks in.

This powershift transmission is a prime example of the possibiliy to level the wear of several components in existing industrial applications: Using condition monitoring techniques, the degradation of each individual clutch could be examined and taken into account for a comprehensive control of both clutches combined. Thus, failures of individual clutches could be avoided, making the system highly reliable. For this, the proposed control algorithm could be used. Compared to the implementation outlined in this thesis, an additional objective for the wear of the second clutch would need to be taken into account, thus forming two reliability objective functions for the multiobjective optimization problem.

5.1.3 Maintenance planning and reliability modeling

In the introduction of this thesis, one major motivation for reliability control was an increase in availability that could be achieved not only by increasing the time until maintenance, but also by decreasing the time to repair. It was assumed that this could be the result of adapting system behavior according to available repair teams. Experiment results show that indeed reliability was increased and made predictable, while adaptation of time to failure is possible.

The resulting advantage for maintenance planning is hard to quantify. In [MKS15a], we made a first attempt at such a quantification. A simulation model for clutch system reliability was setup and a fleet of 56 of these was simulated. Each instance of the simulation model was running on a dedicated core on a high performance computing cluster, requiring a total of about 1000 hours CPU time. A simple maintenance strategy was implemented. The amount of time required for maintenance and the availability of repair crews was included in the full simulation model.

It was shown that a slight increase in availability could be achieved. This was without maintenance-driven lifetime extension though, so one of the major potentials of reliability control was left unused. To make full use of this capability, the maintenance strategy has to be more complex and can actually be self-optimizing itself. Such maintenance strategies do not exist yet.

Also it was shown that common methods for modeling system reliability and availability such as Petri nets fail to model the interaction from maintenance strategy back into the system failure behavior. Current reliability modeling techniques are based on failure driven maintenance, where system failure behavior is static and known. Our approach was a full simulation model, whose computational requirements make it infeasible for general application. Even the limitations of more advanced modeling techniques such as *Lares*, which we have used in [MSS+13] to model reliability of a single self-optimizing system, is reached if multiple systems and their interactions are considered. A combination of intelligent maintenance strategy and adaptable system failure behavior thus requires novel modeling techniques. Developing these is a separate field far from the topic of this thesis, but reliability control can be a good motivation.

Bibliography

[ABB+13] Adomeit, P., Baar, R., Beck, M., Bönnen, D., Dorenkamp, R., et al. „Vieweg Handbuch Kraftfahrzeugtechnik“. In: ed. by Braess, H.-H. and Seiffert, U. 7th ed. ATZ/MTZ-Fachbuch. Springer Vieweg, 2013. Chap. Antriebe, pp. 221–496 (cited on page 58).

[Alb05] Albrecht, M. „Modellierung der Komfortbeurteilung aus Kundensicht am Beispiel des automatisierten Anfahrens“. Dissertation. Universität Karlsruhe, 2005 (cited on pages 60, 61).

[ALR+04] Avizienis, A., Laprie, J.-C., Randell, B., and Landwehr, C. „Basic Concepts and Taxonomy of Dependable and Secure Computing“. In: *IEEE Transactions on Dependable and Secure Computing* 1.1 (2004), pp. 11–33. DOI: 10.1109/TDSC.2004.2 (cited on pages 2, 3).

[Arc53] Archard, J. F. „Contact and Rubbing of Flat Surfaces“. In: *Journal of Applied Physics* 24.8 (1953), pp. 981–988. DOI: 10.1063/1.1721448 (cited on pages 69, 74).

[Ath11] Athans, M. „Control system adavanced methods“. In: ed. by Levine, W. S. 2nd ed. The Control Handbook. CRC Press, 2011. Chap. Kalman Filtering, pp. 13-1–13-10 (cited on page 47).

[BA00] Brennan, S. and Alleyne, A. „The Illinois Roadway Simulator: a mechatronic testbed for vehicle dynamics and control“. In: *IEEE/ASME Transactions on Mechatronics* 5.4 (Dec. 2000), pp. 349–359. ISSN: 1083-4435. DOI: 10.1109/3516.891046 (cited on page 66).

[BA01] Brennan, S. and Alleyne, A. „Using a scale testbed: Controller design and evaluation“. In: *IEEE Control Systems* 21.3 (June 2001), pp. 15–26. ISSN: 1066-033X. DOI: 10.1109/37.924794 (cited on page 66).

[BA99] Brennan, S. and Alleyne, A. „A scaled testbed for vehicle control: the IRS“. In: *Control Applications, 1999. Proceedings of the 1999 IEEE International Conference on*. Vol. 1. 1999, 327–332 vol. 1. DOI: 10.1109/CCA.1999.806652 (cited on page 66).

[BHP+14] Bertling, J., Hiebel, M., Pflaum, H., and Nühlen, J. „Arten und Entstehungstypen frühzeitiger Produktalterung“. In: *UmweltMagazin* 3 (2014) (cited on pages 1, 2).

[BHR05] Bridges, D. O., Horn, J. F., and Ray, A. „Model-Following Control of a Military Helicopter with Damage Mitigation“. In: *AIAA Guidance, Navigation, and Control Conference and Exhibit*. 2005 (cited on page 14).

[Bir07] Birolini, A. *Reliability Engineering*. 5th ed. Berlin Heidelberg: Springer, 2007 (cited on pages 3, 4).

[BKB+04] Baptista, M., Kumar, A., Brunell, B., and Viassolo, D. „Model-Based Life Extending Control for Aircraft Engines“. In: *AIAA 1st Intelligent Systems Technical Conference*. Chicago, Il, 2004 (cited on page 14).

[BL04] Bertsche, B. and Lechner, G. *Zuverlässigkeit im Fahrzeug- und Maschinenbau*. 3rd ed. Springer-Verlag Berlin Heidelberg, 2004. DOI: 10.1007/3-540-34996-0 (cited on page 4).

[Bre04] Brennan, S. „Similarity conditions for comparing closed-loop vehicle roll and pitch dynamics“. In: *American Control Conference, 2004. Proceedings of the 2004*. Vol. 4. June 2004, 3393–3398 vol.4 (cited on page 66).

[Bri03] Bridges, D. O. „Damage-mitigating control of rotorcraft“. Master Thesis. Pennsylvania State University, Aug. 2003 (cited on page 14).

[Buc14] Buckingham, E. „On physically similar systems; illustrations of the use of dimensional equations“. In: *Physical Review* 4.4 (1914), pp. 345–376. DOI: 10.1103/PhysRev.4.345 (cited on page 66).

[Cat89] Catlin, D. E. *Estimation, control, and the discrete Kalman filter*. Vol. 71. Applied mathematical sciences. Springer-Verlag New York, 1989. ISBN: 0-387-96777-X, 3-540-96777-X. DOI: 10.1007/978-1-4612-4528-5 (cited on page 48).

[CB00] Camacho, E. F. and Bordons, C. *Model Predictive Control*. Advanced Textbooks in Control and Signal Processing. Springer, 2000. ISBN: 9780857293985 (cited on pages 49, 50).

[CH08] Coble, J. B. and Hines, J. W. „Prognostic algorithm categorization with PHM Challenge application“. In: *International Conference on Prognostics and Health Management, 2008. PHM 2008*. Oct. 2008, pp. 1–11. DOI: 10.1109/PHM.2008.4711456 (cited on page 8).

[Cox72] Cox, D. R. „Regression Models and Life-Tables“. In: *Journal of the Royal Statistical Society. Series B (Methodological)* 34.2 (1972), pp. 187–220 (cited on page 32).

[CRJ01] Caplin, J., Ray, A., and Joshi, S. „Damage-mitigating control of aircraft for enhanced structural durability“. In: *Aerospace and Electronic Systems, IEEE Transactions on* 37.3 (July 2001), pp. 849–862. ISSN: 0018-9251. DOI: 10.1109/7.953241 (cited on page 14).

[CT05] Chen, D. and Trivedi, K. S. „Optimization for condition-based maintenance with semi-Markov decision process“. In: *Reliability Engineering & System Safety* 90.1 (Oct. 2005), pp. 25–29. DOI: 10.1016/j.ress.2004.11.001 (cited on page 6).

[CYT06] Cheng, Y.-H., Yang, A. S., and Tsao, H.-L. „Study on Rolling Stock Maintenance Strategy and Spares Parts Management“. In: *7th World Congress on Railway Research*. 2006 (cited on page 4).

[DDD+14] Dangelmaier, W., Dellnitz, M., Dorociak, R., Flaßkamp, K., Gausemeier, J., et al. *Dependability of Self-optimizing Mechatronic Systems*. Ed. by Gausemeier, J., Rammig, F. J., Schäfer, W., and Sextro, W. Lecture Notes in Mechanical Engineering. Heidelberg New York Dordrecht London: Springer, 2014. DOI: 10.1007/978-3-642-53742-4 (cited on pages 7–9, 21, 22, 25).

[Deb01] Deb, K. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, 2001 (cited on pages 28, 29, 54).

[DIN ISO 281] DIN Deutsches Institut für Normung e. V. *Roller bearings – Dynamic load ratings and rating life*. National Standard. 2010 (cited on page 31).

[DR96a] Dai, X. and Ray, A. „Damage-mitigating Control of a Reusable Rocket Engine: Part II-Formulation of an Optimal Policy“. In: *Journal of Dynamic Systems, Measurement, and Control* 118.3 (Sept. 1996), pp. 409–415. DOI: 10.1115/1.2801160 (cited on page 12).

[DR96b] Dai, X. and Ray, A. „Damage-mitigating Control of a Reusable Rocket Engine: Part I-Life Prediction of the Main Thrust Chamber Wall“. In: *Journal of Dynamic Systems, Measurement, and Control* 118.3 (Sept. 1996), pp. 401–408. DOI: 10.1115/1.2801159 (cited on page 12).

[DyL67] Dy Liacco, T. E. „The Adaptive Reliability Control System“. In: *IEEE Transactions on Power Apparatus and Systems* PAS-86.5 (May 1967), pp. 517–531. ISSN: 0018-9510. DOI: 10.1109/TPAS.1967.291728 (cited on page 17).

[Ehr05] Ehrgott, M. *Multicriteria Optimization*. 2005. DOI: 10.1007/3-540-27 659-9 (cited on pages 27, 28).

[FFC06] Fravolini, M., Ficola, A., and Cava, M. L. „Life extending robust control of a wind excited antenna“. In: *Computers & Structures* 84.5-6 (2006), pp. 413–430. ISSN: 0045-7949. DOI: 10.1016/j.compstruc.2005 .09.028 (cited on page 16).

[FGM+07] Frank, U., Giese, H., Müller, T., Oberthür, S., Romaus, C., et al. „Potenziale und Risiken der Selbstoptimierung für die Verlässlichkeit mechatronischer Systeme“. In: *5. Paderborner Workshop Entwurf mechatronischer Systeme*. Vol. 210. HNI Verlagsschriftenreihe. Paderborn, 2007 (cited on pages 8, 24).

[Föl94] Föllinger, O. *Regelungstechnik*. 8th ed. Heidelberg: Hüthig, 1994 (cited on page 28).

[FS07] Förster, B. and Steinel, K. „Kupplungsbetätigungssystem ConAct für Nutzfahrzeuge mit automatisierten Schaltgetrieben“. In: *ATZ - Automobiltechnische Zeitschrift* 109.2 (Feb. 2007), pp. 140–146. DOI: 10.10 07/BF03221866 (cited on page 57).

[Ful07] Fuller, J. *Model predictive controller with life extending control*. Patent. US Patent 7,203,554. Apr. 2007 (cited on page 16).

[Gau04] Gausemeier, J., ed. *Selbstoptimierende Systeme des Maschinenbaus*. Vol. 155. HNI-Verlagsschriftenreihe. Paderborn, DE: Heinz Nixdorf Institute, University of Paderborn, 2004 (cited on pages 6, 21, 23).

[GB06] Grimes, G. A. and Barkan, C. „Cost-Effectiveness of Railway Infrastructure Renewal Maintenance“. In: *Journal of Transportation Engineering* 132.8 (2006), pp. 601–608. DOI: 10.1061/(ASCE)0733-947X(2006)132:8(601) (cited on page 4).

[GBS+04] Giese, H., Burmester, S., Schäfer, W., and Oberschelp, O. „Modular Design and Verification of Component-Based Mechatronic Systems with OnlineReconfiguration“. In: *Proc. of 12th ACM SIGSOFT Foundations of Software Engineering 2004 (FSE 2004)*. 2004, pp. 179–188. DOI: 10.1145/1029894.1029920 (cited on page 54).

[GCJ04] Guo, T., Chen, P., and Jaw, L. „Intelligent Life-Extending Controls for Aircraft Engines“. In: *Intelligent systems technical conference*. 2004 (cited on page 15).

[GCO13] Grad, K., Cappellaro, T., and Oberbuchner, T. „Shift Concept for Starting Clutch in Powershift Transmission“. In: *ATZ offhighway* 2 (July 2013), pp. 58–65 (cited on page 95).

[GHH+06] Giese, H., Henkler, S., Hirsch, M., Tichy, M., and Vöcking, H. „Modellbasierte Entwicklung vernetzter, mechatronischer Systeme am Beispiel der Konvoifahrt autonom agierender Schienenfahrzeuge“. In: *Proc. of 4th Paderborner Workshop Entwurf mechatronischer Systeme*. Vol. 189. HNI-Verlagsschriftenreihe. Paderborn, 2006, pp. 457–473 (cited on page 54).

[GRS09a] Gausemeier, J., Rammig, F. J., and Schäfer, W., eds. *Selbstoptimierende Systeme des Maschinenbaus*. Vol. 234. HNI-Verlagsschriftenreihe. Paderborn, DE: Heinz Nixdorf Institute, University of Paderborn, 2009 (cited on pages 21–24).

[GRS09b] Gausemeier, J., Rammig, F. J., and Schäfer, W., eds. *Verlässlichkeit selbstoptimierender Systeme*. Vol. 235. HNI-Verlagsschriftenreihe. Paderborn: Heinz Nixdorf Institute, University of Paderborn, 2009 (cited on pages 8, 55).

[GRS14] Gausemeier, J., Rammig, F. J., and Schäfer, W., eds. *Design Methodology for Intelligent Technical Systems*. Springer-Verlag Berlin Heidelberg, 2014. DOI: 10.1007/978-3-642-45435-6 (cited on page 24).

[GS14] Galván, J. R. C. and Saxena, A. *Electronics Prognostics*. Tutorial at 2nd European Conference of the PHM Society. 2014 (cited on page 8).

[GSA+02] Gallestey, E., Stothert, A., Antoine, M., and Morton, S. „Model predictive control and the optimization of power plant load while considering lifetime consumption“. In: *Power Systems, IEEE Transactions on* 17.1 (Feb. 2002), pp. 186–191. ISSN: 0885-8950. DOI: 10.1109/59.982212 (cited on page 17).

[Guo01] Guo, T.-H. „A roadmap for aircraft engine life extending control“. In: *American Control Conference, 2001. Proceedings of the 2001*. Vol. 5. 2001, 3702–3705 vol.5. DOI: 10.1109/ACC.2001.946210 (cited on page 15).

[HOG04] Hestermeyer, T., Oberschelp, O., and Giese, H. „Structured Information Processing For Self-optimizing Mechatronic Systems“. In: *Proc. of 1st International Conference on Informatics in Control, Automation and Robotics (ICINCO 2004), Setubal, Portugal*. Ed. by Araujo, H., Vieira, A., Braz, J., Encarnacao, B., and Carvalho, M. INSTICC Press, Aug. 2004, pp. 230–237 (cited on page 24).

[HR01a] Holmes, M. S. and Ray, A. „Fuzzy Damage Mitigating Control of Mechanical Structures“. In: *Journal of Dynamic Systems, Measurement, and Control* 120.2 (June 2001), pp. 249–256. DOI: 10.1115/1.2802416 (cited on page 13).

[HR01b] Holmes, M. and Ray, A. „Fuzzy damage-mitigating control of a fossil power plant“. In: *Control Systems Technology, IEEE Transactions on* 9.1 (Jan. 2001), pp. 140–147. ISSN: 1063-6536. DOI: 10.1109/87.896755 (cited on page 15).

[HTR97] Holmes, M., Tangirala, S., and Ray, A. „Life-extending control of a reusable rocket engine“. In: *American Control Conference, 1997. Proceedings of the 1997*. Vol. 4. June 1997, 2328–2332 vol.4. DOI: 10.1109/ACC.1997.609055 (cited on page 13).

[Ise06] Isermann, R. *Fault-Diagnosis Systems*. Berlin, Heidelberg: Springer, 2006. DOI: 10.1007/3-540-30368-5 (cited on page 46).

[ISO 2631-1] International Organization for Standardization. *Mechanical vibration and shock – Evaluation of human exposure to whole-body vibration – Part 1: General requirements)*. International Standard. 1997 (cited on page 60).

[Jac09] Jackson, P. *the little book of smart*. Veloce Publishing Ltd., 2009 (cited on page 57).

[JLB06] Jardine, A. K. S., Lin, D., and Banjevic, D. „A review on machinery diagnostics and prognostics implementing condition-based maintenance“. In: *Mechanical Systems and Signal Processing* 20.7 (Oct. 2006), pp. 1483–1510. DOI: 10.1016/j.ymssp.2005.09.012 (cited on page 6).

[JLF97] Joo, S. J., Levary, R. R., and Ferris, M. E. „Planning Preventive Maintenance for a Fleet of Police Vehicles Using Simulation“. In: *SIMULATION* 68.2 (Feb. 1997), pp. 93–99. DOI: 10.1177/003754979706800202 (cited on page 5).

[KGR+06] Khatkhate, A., Gupta, S., Ray, A., and Keller, E. „Life Extending Control of Mechanical Systems* using Symbolic Time Series Analysis“. In: *American Control Conference, 2006*. June 2006, pp. 3765–3770. DOI: 10.1109/ACC.2006.1657304 (cited on page 16).

[KHR97] Kallappa, P., Holmes, M. S., and Ray, A. „Life-extending control of fossil fuel power plants“. In: *Automatica* 33.6 (1997), pp. 1101–1118. ISSN: 0005-1098. DOI: 10.1016/S0005-1098(97)00014-9 (cited on page 15).

[Kim16] Kimotho, J. K. „Development and Performance Evaluation of Prognostic Approaches for Technical Systems“. Dissertation. Paderborn University, 2016 (cited on page 52).

[KMS14] Kimotho, J. K., Meyer, T., and Sextro, W. „PEM fuel cell prognostics using particle filter with model parameter adaptation“. In: *Prognostics and Health Management (PHM), 2014 IEEE Conference on*. June 2014, pp. 1–6. DOI: 10.1109/ICPHM.2014.7036406 (cited on pages 34, 35).

[KMS15] Kaul, T., Meyer, T., and Sextro, W. „Integrated Model for Dynamics and Reliability of Intelligent Mechatronic Systems“. In: *European Safety and Reliability Conference (ESREL2015)*. Ed. by Podofillini et al. London: Taylor and Francis, 2015. DOI: 10.1201/b19094-290 (cited on page 58).

[KR00] Kallappa, P. and Ray, A. „Fuzzy wide-range control of fossil power plants for life extension and robust performance“. In: *Automatica* 36.1 (2000), pp. 69–82. ISSN: 0005-1098. DOI: 10.1016/S0005-1098(99)00103-X (cited on page 15).

[KRK+13] Krüger, M., Remirez, A., Kessler, J. H., and Trächtler, A. „Discrete Objective-based Control for Self-Optimizing Systems“. In: *American Control Conference (ACC), 2013*. June 2013, pp. 3403–3408 (cited on pages 37, 40, 41, 78).

[Krü14] Krüger, M. „Parametrische Modellordnungsreduktion für hierarchische selbstoptimierende Systeme“. Dissertation. Universität Paderborn, 2014 (cited on page 40).

[KSR12] Klöpper, B., Sondermann-Wölke, C., and Romaus, C. „Probabilistic Planning for Predictive Condition Monitoring and Adaptation Within the Self-Optimizing Energy Management of an Autonomous Railway Vehicle“. In: *Journal of Robotics and Mechatronics* 24.1 (Feb. 2012), pp. 5–15 (cited on page 55).

[KSS+03] Köpf, P., Steinel, K., Sasse, C., Wagner, G., and Drexl, H.-J. „Vieweg Handbuch Kraftfahrzeugtechnik“. In: ed. by Braess, H.-H. and Seiffert, U. 3rd ed. ATZ MTZ Fachbuch. Vieweg & Sohn Verlag, 2003. Chap. Triebstrang, pp. 258–293 (cited on pages 57, 58).

[LCM+03] Li, D., Chen, T., Marquez, H., and Gooden, R. „Damage modeling and life extending control of a boiler-turbine system“. In: *American Control Conference, 2003. Proceedings of the 2003*. Vol. 3. June 2003, 2317–2322 vol.3. DOI: 10.1109/ACC.2003.1243420 (cited on page 16).

[LCM+06] Li, D., Chen, T., Marquez, H., and Gooden, R. „Life extending control of boiler-turbine systems via model predictive methods“. In: *Control Engineering Practice* 14.4 (2006), pp. 319–326. ISSN: 0967-0661. DOI: 10.1016/j.conengprac.2004.12.002 (cited on page 16).

[LHR00] Lorenzo, C. F., Holmes, M. S., and Ray, A. „Nonlinear Life-Extending Control of a Rocket Engine“. In: *Journal of Guidance, Control and Dynamics* 23.4 (2000), pp. 759–762. DOI: 10.2514/2.4599 (cited on page 13).

[LHR98a] Lorenzo, C. F., Holmes, M. S., and Ray, A. *Design of Life-Extending Control Using Nonlinear Parameter Optimization*. NASA Technical Paper 3700. Apr. 1998 (cited on page 13).

[LHR98b] Lorenzo, C., Holmes, M., and Ray, A. „Nonlinear control of a reusable rocket engine for life extension“. In: *American Control Conference, 1998. Proceedings of the 1998*. Vol. 5. June 1998, 2922–2926 vol.5. DOI: 10.1109/ACC.1998.688392 (cited on page 13).

[Li15] Li, R. „Untersuchung einer Reibkupplung hinsichtlich ihrer Zuverlässigkeit und Schätzung der Restlebensdauer“. Advisor: Tobias Meyer. Examiners: Walter Sextro, Tobias Hemsel. Masterarbeit. Universität Paderborn, 2015 (cited on page 68).

[LM91a] Lorenzo, C. F. and Merrill, W. C. *Life extending control: An interdisciplinary engineering thrust*. Tech. rep. NASA Lewis Research Center, 1991 (cited on page 11).

[LM91b] Lorenzo, C. F. and Merrill, W. „Life Extending Control - A Concept Paper“. In: *American Control Conference, 1991*. June 1991, pp. 1081–1095 (cited on pages 10, 11).

[LM91c] Lorenzo, C. and Merrill, W. „An intelligent control system for rocket engines: need, vision, and issues“. In: *Control Systems, IEEE* 11.1 (Jan. 1991), pp. 42–46. ISSN: 1066-033X. DOI: 10.1109/37.103353 (cited on page 12).

[Lor94] Lorenzo, C. F. „Continuum fatigue damage modeling for use in life extending control“. In: *NASA STI/Recon Technical Report N* 95 (Aug. 1994), p. 10857 (cited on page 13).

[LRH01] Lorenzo, C. F., Ray, A., and Holmes, M. S. „Nonlinear Control of a Reusable Rocket Engine for Life Extension“. In: *Journal of Propulsion and Power* 17.5 (Oct. 2001), pp. 998–1004. DOI: 10.2514/2.5861 (cited on pages 13, 28).

[LSR+92] Lorenzo, C. F., Saus, J. R., Ray, A., Carpino, M., and Wu, M.-K. „Life extending control for rocket engines“. In: *NASA STI/Recon Technical Report N* 92 (Aug. 1992), p. 31263 (cited on page 12).

[Mer14] Mers, S. „Berechnung optimaler Steuerungstrajektorien für ein mechatronisches System“. Advisors: Kathrin Flaßkamp, Tobias Meyer. Examiners: Kathrin Flaßkamp, Walter Sextro. Studienarbeit. Universität Paderborn, 2014 (cited on page 67).

[MHM+13] Meyer, T., Hölscher, C., Menke, M., Sextro, W., and Zimmer, D. „Multiobjective Optimization including Safety of Operation Applied to a Linear Drive System“. In: *Proc. Appl. Math. Mech.* Vol. 13. 2013, pp. 483–484. DOI: 10.1002/pamm.201310234 (cited on page 56).

[MKH02] Mamalis, A., Kundrák, J., and Horváth, M. „Wear and Tool Life of CBN Cutting Tools“. In: *The International Journal of Advanced Manufacturing Technology* 20.7 (2002), pp. 475–479. ISSN: 1433-3015. DOI: 10.1007/s001700200180 (cited on page 32).

[MKS15a] Meyer, T., Kaul, T., and Sextro, W. „Advantages of reliability-adaptive system operation for maintenance planning“. In: *Proceedings of the 9th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes*. 2015, pp. 940–945. DOI: 10.1016/j.ifacol.2015.09.647 (cited on page 96).

[MKS15b] Meyer, T., Kimotho, J. K., and Sextro, W. „Anforderungen an Condition-Monitoring-Verfahren zur Nutzung im zuverlässigkeitsgeregelten Betrieb adaptiver Systeme“. In: *27. Tagung Technische Zuverlässigkeit (TTZ 2015) - Entwicklung und Betrieb zuverlässiger Produkte*. 2260. Leonberg, 2015, pp. 111–122 (cited on pages 51, 52).

[MMD+10] Mahulkar, V., McGinnis, H., Derriso, M., and Adams, D. E. *Fault identification in an electro-hydraulic actuator and experimental validation of prognosis based life extending control*. Tech. rep. Air force research lab Wright-Patterson AFB, 2010 (cited on page 15).

[MP10] Meyna, A. and Pauli, B. *Zuverlässigkeitstechnik, Quantitative Bewertungsverfahren*. 2nd ed. Munich: Carl Hanser Verlag, 2010 (cited on page 56).

[MRR+00] Mayne, D., Rawlings, J., Rao, C., and Scokaert, P. „Constrained model predictive control: Stability and optimality“. In: *Automatica* 36.6 (2000), pp. 789–814. ISSN: 0005-1098. DOI: 10.1016/S0005-1098(99)00214-9 (cited on page 53).

[MS14] Meyer, T. and Sextro, W. „Closed-loop Control System for the Reliability of Intelligent Mechatronic Systems“. In: *Proceedings of the Second European Conference of the Prognostics and Health Management Society 2014*. Vol. 5. 2014 (cited on page 53).

[MSS+13] Meyer, T., Sondermann-Wölke, C., Sextro, W., Riedl, M., Gouberman, A., et al. „Bewertung der Zuverlässigkeit selbstoptimierender Systeme mit dem LARES-Framework“. In: *9. Paderborner Workshop Entwurf mechatronischer Systeme*. Ed. by Gausemeier, J., Dumitrescu, R., Rammig, F., Schäfer, W., and Trächtler, A. HNI-Verlagsschriftenreihe. Paderborn: Heinz Nixdorf Institut, Universität Paderborn, 2013, pp. 161–174 (cited on page 96).

[MSS14] Meyer, T., Sondermann-Wölke, C., and Sextro, W. „Method to Identify Dependability Objectives in Multiobjective Optimization Problem“. In: *Procedia Technology* 15 (2014), pp. 46–53. DOI: 10.1016/j.protcy.2014.09.033 (cited on pages 31, 32).

[Mün12] Münch, E. „Selbstoptimierung verteilter mechatronischer Systeme auf Basis paretooptimaler Systemkonfigurationen“. Dissertation. Paderborn: University of Paderborn, 2012 (cited on pages 23, 28).

[Obe08] Ober-Blöbaum, S. „Discrete mechanics and optimal control“. Dissertation. Universität Paderborn, 2008 (cited on page 31).

[Osm15] Osmic, S. „Flachheitsbasierte Methode zum stoßfreien Umschalten von Reglerstrukturen“. Dissertation. Paderborn University, 2015 (cited on page 54).

[OZZ+13] Ong, J., Zhang, T., Zhou, Y., Lim, K., Chen, W., et al. *Method and a system for controlling operation of a wind turbine*. US Patent 8,577,509. Nov. 2013 (cited on page 16).

[Pab05] Pabst, I. „On Modeling, Analysis and Synthesis of Generalized Reliability Control Systems“. Dissertation. Universität Duisburg-Essen, 2005 (cited on page 8).

[PDG+16] Prakash, S., Dehoust, G., Gsell, M., Schleicher, T., and Stamminger, R. *Einfluss der Nutzungsdauer von Produkten auf ihre Umweltwirkung: Schaffung einer Informationsgrundlage und Entwicklung von Strategien gegen Obsoleszenz*. Studie. Umweltbundesamt, 2016 (cited on pages 1, 2).

[Pop10] Popov, I. *Contact Mechanics and Friction*. Berlin, Heidelberg: Springer, 2010 (cited on page 72).

[PR99] Patankar, R. and Ray, A. „Damage mitigating controller design for structural durability“. In: *Control Systems Technology, IEEE Transactions on* 7.5 (Sept. 1999), pp. 606–612. ISSN: 1063-6536. DOI: 10.1109/87.784424 (cited on page 13).

[Rak05] Rakowsky, U. K. „An Introduction to Reliability-Adaptive Systems“. In: *Advances in Safety and Reliability. Proceedings of the European Conference on Safety and Reliability —ESREL 2005*. Ed. by Kolowrocki, K. Vol. 2. Leiden: Balkema, 2005, pp. 1633–1636 (cited on page 7).

[Ray01] Ray, A. „Life extending control of large-scale dynamical systems“. In: *American Control Conference, 2001. Proceedings of the 2001*. Vol. 5. 2001, 3692–3701 vol.5. DOI: 10.1109/ACC.2001.946209 (cited on page 15).

[RB15] Rakowsky, U. K. and Bertsche, B. „Introducing Type 5 to prognostics and health management classification schemes“. In: *Safety and Reliability of Complex Engineered Systems (ESREL 2015)*. 2015, pp. 2451–2455. DOI: 10.1201/b19094-320 (cited on page 8).

[RC00] Ray, A. and Caplin, J. „Life extending control of aircraft: trade-off between flight performance and structural durability“. In: *The Aeronautical Journal* 104.1039 (Sept. 2000), pp. 397–408 (cited on page 14).

[RDC+94] Ray, A., Dai, X., Carpino, M., and Lorenzo, C. F. „Damage-mitigating control: an interdisciplinary thrust between controls and material science“. In: *American Control Conference, 1994*. Vol. 3. June 1994, 3449–3453 vol.3. DOI: 10.1109/ACC.1994.735219 (cited on page 12).

[RDW+94] Ray, A., Dai, X., Wu, M.-K., Carpino, M., and Lorenzo, C. F. „Damage-mitigating control of a reusable rocket engine“. In: *Journal of Propulsion and Power* 10.2 (1994), pp. 225–236. DOI: 10.2514/3.23733 (cited on page 12).

[Rey60] Reye, T. „Zur Theorie der Zapfenreibung“. In: *Der Civilingenieur* 4 (1860), pp. 235–255 (cited on pages 69, 74).

[RM07] Rimell, A. N. and Mansfield, N. J. „Design of Digital Filters for Frequency Weightings Required for Risk Assessments of Workers Exposed to Vibration“. In: *Industrial Health* 45.4 (2007), pp. 512–519 (cited on page 60).

[RR98] Rozak, J. N. and Ray, A. „Robust Multivariable Control of Rotorcraft in Forward Flight: Impact of Bandwidth on Fatigue Life“. In: *Journal of American Helicopter Society* 43.3 (July 1998), pp. 195–201 (cited on page 14).

[RS12] Richard, H. A. and Sander, M. *Ermüdungsrisse*. Springer Vieweg, 2012 (cited on page 32).

[Run01] Runnels, S. *Method and system for modeling, predicting and optimizing chemical mechanical polishing pad wear and extending pad life*. US Patent 6,169,931. Jan. 2001 (cited on page 16).

[RWC+93a] Ray, A., Wu, M.-K., Carpino, M., and Lorenzo, C. F. „Damage-Mitigating Control of Mechanical Systems: Part I – Conceptual Development and Model Formulation“. In: *American Control Conference, 1993*. June 1993, pp. 1172–1176 (cited on page 12).

[RWC+93b] Ray, A., Wu, M.-K., Carpino, M., and Lorenzo, C. F. „Damage-Mitigating Control of Mechanical Systems: Part II - Formulation of an Optimal Policy and Simulation“. In: *American Control Conference, 1993*. June 1993, pp. 3146–3150 (cited on page 12).

[RWC+94a] Ray, A., Wu, M.-K., Carpino, M., and Lorenzo, C. F. „Damage-Mitigating Control of Mechanical Systems: Part I-Conceptual Development and Model Formulation“. In: *Journal of Dynamic Systems, Measurement, and Control* 116.3 (Sept. 1994), pp. 437–447 (cited on page 12).

[RWC+94b] Ray, A., Wu, M.-K., Carpino, M., and Lorenzo, C. F. „Damage-Mitigating Control of Mechanical Systems: Part II-Formulation of an Optimal Policy and Simulation“. In: *Journal of Dynamic Systems, Measurement, and Control* 116.3 (Sept. 1994), pp. 448–455 (cited on page 12).

[San07] Santos, R. A. „Damage mitigating control for wind turbines“. PhD thesis. University of Colorado at Boulder, 2007 (cited on page 16).

[San08] Santos, R. *Control system for a wind turbine and method of controlling said wind turbine*. EP Patent App. EP20,060,122,043. Apr. 2008 (cited on page 16).

[SGH+09] Sondermann-Wölke, C., Geisler, J., Hirsch, M., and Hemsel, T. „Verlässlichkeit im aktiven selbstoptimierenden Spurführungsmodul eines schienengebundenen Fahrzeugs“. In: *Entwurf mechatronischer Systeme*. Ed. by Gausemeier, J., Rammig, F. J., and Trächtler, A. Vol. 250. HNI-Verlagsschriftenreihe. Paderborn, 2009, pp. 231–242 (cited on page 8).

[SGM+08] Sondermann-Wölke, C., Geisler, J., Müller, T., Trächtler, A., and Böcker, J. „The active guidance module of a rail-bound vehicle as an application for the dependability oriented design in self-optimizing systems“. In: *ASME 2008 – International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE)*. New York, 2008 (cited on page 8).

[SKK+13] Stohrer, M., Kemmler, S., Koller, O., Zeiler, P., and Bertsche, B. „Zuverlässigkeitsorientierte Online-Optimierung von Betriebsstrategien mechatronischer Produkte“. In: *Stuttgarter Symposium für Produktentwicklung*. 2013 (cited on page 17).

[SL15a] Samaranayake, L. and Longo, S. „Cost functions for degradation control of electric motors in electric vehicles“. In: *Control Conference (ECC), 2015 European*. July 2015, pp. 660–665. DOI: 10.1109/ECC.2015.7330617 (cited on pages 17, 34).

[SL15b] Samaranayake, L. and Longo, S. „Nonlinear Model Predictive Control for traction motor degradation minimization“. In: *2015 54th IEEE Conference on Decision and Control (CDC)*. Dec. 2015, pp. 3681–3686. DOI: 10.1109/CDC.2015.7402790 (cited on page 17).

[SMD+12] Sondermann-Woelke, C., Meyer, T., Dorociak, R., Gausemeier, J., and Sextro, W. „Conceptual Design of Advanced Condition Monitoring for a Self-Optimizing System based on its Principle Solution“. In: *Proceedings of the 11th International Probabilistic Safety Assessment and Management Conference (PSAM11) and The Annual European Safety and Reliability Conference (ESREL2012)*. Helsinki, Finland, 2012 (cited on page 55).

[Son15] Sondermann-Wölke, C. „Entwurf und Anwendung einer erweiterten Zustandsüberwachung zur Verlässlichkeitssteigerung selbstoptimierender Systeme“. Dissertation. Universität Paderborn, 2015 (cited on pages 10, 55).

[SR01] Sastry, V. and Ray, A. „Online monitoring of fatigue crack damage for life extending control of aircraft structures“. In: *American Control Conference, 2001. Proceedings of the 2001*. Vol. 3. 2001, 2363–2364 vol.3. DOI: 10.1109/ACC.2001.946105 (cited on page 14).

[SR97] Söffker, D. and Rakowsky, U. K. „Perpectives of monitoring and control of vibrating structures by combining new methods of fault detection with new approaches of reliability engineering“. In: *A Critical Link: Diagnosis to prognosis; Proc. 12th ASME Conference on Reliability, Stress Analysis and Failure Prevention*. Ed. by Pursey, H. 1997, pp. 671–682 (cited on page 7).

[SS09] Sondermann-Wölke, C. and Sextro, W. „Towards the Integration of Condition Monitoring in Self-Optimizing Function Modules“. In: *Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns, 2009. ComputationWorld '09.* 2009, pp. 15–20. DOI: 10.1109/ComputationWorld.2009.47 (cited on page 8).

[SWO+13] Schütze, O., Witting, K., Ober-Blöbaum, S., and Dellnitz, M. „E-VOLVE - A Bridge between Probability, Set Oriented Numerics and Evolutionary Computation“. In: ed. by Tantar, E., Tantar, A.-A., Bouvry, P., Del Moral, P., Legrand, P., et al. Vol. 447. Studies in Computational Intelligence. Berlin Heidelberg: Springer, 2013. Chap. Set Oriented Methods for the Numerical Treatment of Multiobjective Optimization Problems, pp. 187–219. DOI: 10.1007/978-3-642-32726-1 (cited on pages 30, 76).

[TCK+99] Tangirala, S., Caplin, J., Keller, E., and Ray, A. „Life extending control of gas turbine engines“. In: *American Control Conference, 1999. Proceedings of the 1999*. Vol. 4. 1999, 2642–2646 vol.4. DOI: 10.1109/ACC.1999.786549 (cited on page 13).

[THR+98] Tangirala, S., Holmes, M., Ray, A., and Carpino, M. „Life-extending control of mechanical structures: Experimental verification of the concept“. In: *Automatica* 34.1 (1998), pp. 3–14. ISSN: 0005-1098. DOI: 10.1016/S0005-1098(97)00145-3 (cited on page 16).

[Tol05] Tolani, D. K. „Integrated health management and control of complex dynamical systems“. PhD thesis. Pennsylvania State University, 2005 (cited on page 14).

[Tor12] Torrelli, C. „Das Automatisierte Schaltgetriebe des Lamborghini Aventador“. In: *ATZ - Automobiltechnische Zeitschrift* 114.7-8 (2012), pp. 600–604. DOI: 10.1007/s35148-012-0393-0 (cited on page 57).

[TRC95] Tangirala, S., Ray, A., and Carpino, M. „Damage-mitigating control of mechanical structures: experimental verification of the concept“. In: *Smart Materials and Structures* 4.2 (June 1995), pp. 139–146. DOI: 10.1088/0964-1726/4/2/011 (cited on page 15).

[USA+14] Unger, A., Sextro, W., Althoff, S., Meyer, T., Brökelmann, M., et al. „Data-driven Modeling of the Ultrasonic Softening Effect for Robust Copper Wire Bonding“. In: *Proceedings of 8th International Conference on Integrated Power Electronic Systems*. Vol. 141. 2014, pp. 175–180 (cited on page 52).

[VDI 2057-1] Verband Deutscher Ingenieure. *Einwirkung mechanischer Schwingungen auf den Menschen — Ganzkörper-Schwingungen (Human exposure to mechanical vibrations — Whole-body vibration)*. National Standard. 2002 (cited on pages 60, 61).

[VDI 2895] Verband Deutscher Ingenieure. *Organisation of maintenance — Maintenance as a task of management*. National Standard. 2012 (cited on page 33).

[WG01] Wiseman, M. and Guo, T.-H. „An investigation of life extending control techniques for gas turbine engines“. In: *American Control Conference, 2001. Proceedings of the 2001*. Vol. 5. 2001, 3706–3707 vol.5. DOI: 10.1109/ACC.2001.946211 (cited on page 13).

[Wol08] Wolters, K. „Formalismen, Simulation und Potenziale eines nutzungsdaueroptimierenden Zuverlässigkeitskonzepts“. Dissertation. Universität Duisburg-Essen, 2008 (cited on page 7).

[ZR99] Zhang, H. and Ray, A. „Robust Damage-Mitigating Control of Mechanical Systems: Experimental Validation on a Test Apparatus“. In: *Journal of Dynamic Systems, Measurement, and Control* 121.3 (Sept. 1999), pp. 377–385. DOI: 10.1115/1.2802485 (cited on page 16).

[ZRP00] Zhang, H., Ray, A., and Phoha, S. „Hybrid life-extending control of mechanical systems: experimental validation of the concept“. In: *Automatica* 36.1 (2000), pp. 23–36. ISSN: 0005-1098. DOI: 10.1016/S0005-1098(99)00114-4 (cited on page 16).

Index

P

R

S

V

W