SEQUENTIAL UPDATING OF QUANTITATIVE REQUIREMENTS FOR INCREASED FLEXIBILITY IN ROBUST SYSTEMS DESIGN

Funk, Matthias (1); Jautze, Marcus (2); Strohe, Manfred (2); Zimmermann, Markus (3)

1: BMW, Department of Preliminary Design for Vehicle Dynamics; 2: Hochschule Landshut University of Applied Sciences, Mechanical Engineering; 3: Technische Universität München, Laboratory for Product Development and Lightweight Design

ABSTRACT

In early development stages of complex systems, interacting subsystems (including components) are often designed simultaneously by distributed teams with limited information exchange. Distributed development becomes possible by assigning teams independent design goals expressed as quantitative requirements equipped with tolerances to provide flexibility for design: so-called solution-spaces are high-dimensional sets of permissible subsystem properties on which requirements on the system performance are satisfied. Edges of box-shaped solution spaces are permissible intervals serving as decoupled (mutually independent) requirements for subsystem design variables. Unfortunately, decoupling often leads to prohibitively small intervals. In so-called solution-compensation spaces, permissible intervals for early-decision variables are increased by a compensation mechanism using late-decision variables. This paper presents a multi-step development process where groups of design variables successively change role from early-decision to late-decision type in order to maximize flexibility. Applying this to a vehicle chassis design problem demonstrates the effectiveness of the approach.

Keywords: Robust design, Integrated product development, Requirements, Cooperative design, Solution-compensation spaces

Contact:
Funk, Matthias
BMW Group
Department of Preliminary Design for Vehicle Dynamics
Germany
matthias.funk@bmw.de

Cite this article: Funk, M., Jautze, M., Strohe, M., Zimmermann, M. (2019) ‘Sequential Updating of Quantitative Requirements for Increased Flexibility in Robust Systems Design’, in *Proceedings of the 22nd International Conference on Engineering Design (ICED19),* Delft, The Netherlands, 5-8 August 2019. DOI:10.1017/dsi.2019.360

1 INTRODUCTION

The design of large complex systems is strongly associated with uncertainty. Sources of uncertainty and complexity include a high number of interacting design variables, many objectives that have to be reached, and development in distributed teams. Classical approaches of integrated product development prescribe an order for component design in which the most interacting components are to be developed first. Parameter-based DSM (Design Structure Matrix) can be used to describe the relation between design variables and derive a design process (Browning, 2001). Starting with a candidate solution, the less interacting components follow in steps like those shown in Figure 1 (Ehrlenspiel and Meerkamm, 2017). In highly interacting and complex systems, however, this leads to many iterations and possibly to conflicting goals that cannot be resolved. Therefore, robust design principles and techniques were developed to deal with uncertainty in large complex systems and to find a feasible solution (Zhou and Li, 2014). The framework of solution spaces is presented in the following section since it is the basis for the proposed methodology. Instead of one particular component design, a set-based design provides flexibility within ranges and the development work can slowly converge to a solution (Erschen *et al*., 2018; Sobek II *et al*., 1999). This does not necessarily provide a performance optimal solution, however, all requirements are fulfilled, but it is optimized for integrability.
The approach of sequentially updating solution-compensation spaces provides a multi-step development process to maximize flexibility of component design in each step. A structured process is applied to assign groups of design variables to the role of early-decision or late-decision variables and successively change their role during the process. Thus flexibility is maximized and constraints for the component design are provided with respect to the system performance requirements. This approach is applied to a vehicle chassis development problem as an example for a complex product with highly interacting design variables, developed in distributed teams.

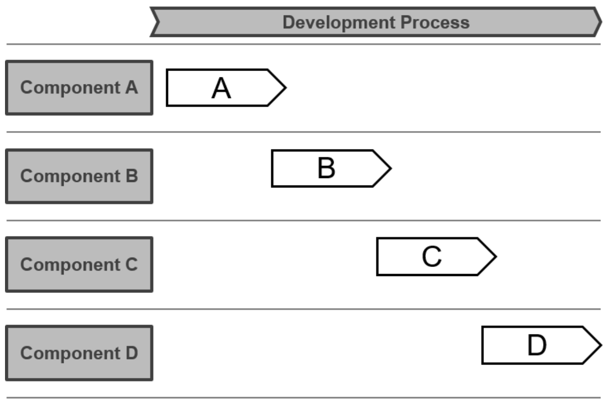

Figure 1. Development Process

2 ROBUST DESIGN

Robust product designs are intended to ensure that the product performance meets all requirements. Activities enabling robust design are applied in an early conceptual design phase (Hendrix *et al*., 1996; Andersson, 1996). Overviews of different design guidelines and design principles have been provided in previous work, e.g. (Ebro and Howard, 2016; Freund *et al*., 2017). The approaches of *solution spaces* and *solution-compensation spaces* are presented in the sub-sections hereinafter. These approaches are the basis for *sequentially updating solution-compensation spaces* presented in this paper.

2.1 Solution spaces

Solution spaces enabling robust design were introduced by Zimmermann and Hoessle (2013). They address the development of systems comprising many interacting design variables in an early development phase of a distributed process with, for example, many design teams. During the development process, these systems are characterized by uncertainty, which is caused by increased complexity and a large number of objectives needing to be reached.

https://doi.org/10.1017/dsi.2019.360 Published online by Cambridge University Press

In contrast, in order to iteratively improve a point based design, solution spaces provide target regions for each design variable. The goal is to decouple the design variables and make these target regions as large as possible so as to make design work easy in distributed teams and satisfy all requirements at the same time.
Therefore, a so-called *complete solution space* is sought within the given design space. The design space is defined as the set of all possible design points x of the system. The requirements give boundaries indicating whether the evaluated combination of design variables is a good design or not. Regions of poor designs in the design space are shown as grey areas in Figure 2. Regions having only good designs are identified thereby, and the complete solution space is provided as illustrated by the shaded area in Figure 2. By realizing a component design and setting design variables x_i to a fixed value within the complete solution space, values for design variables x_j are not independently selectable. *Box-shaped solutions spaces* or *solution boxes* can be calculated to decouple the target regions of all design variables. An orthogonal box of the largest expansion is sought within the complete solution space. A range I_i is then specified for each design variable that is decoupled from all the other design variables. Design teams can develop solutions for their subproblem independently within these ranges, and the system performance will still satisfy all given requirements.

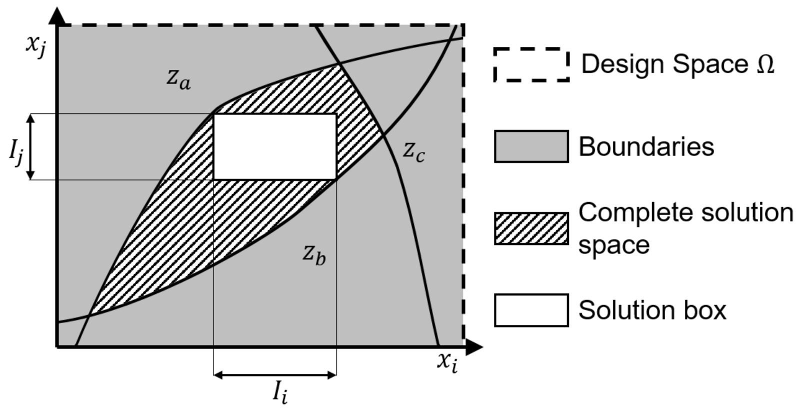

Figure 2. Complete solution space and solution box

Zimmermann and Hoessle (2013) introduced an algorithm that always produces a result for non-linear and high-dimensional problems and simultaneously trying to maximize the size of the solution box. For this reason, a solution box may always be identified (provided that a solution exists) and decoupling in this sense will always be possible. The disadvantage of decoupling all design variables by expressing independent target regions for separate design teams, however, is a loss of solution space, shown as the shaded area in Figure 2.

2.2 Solution-compensation spaces

Small intervals for decoupled design variables, which are provided by classical solution spaces as described in sub-section 2.1, can be avoided by what are known as *solution-compensation spaces* (Vogt *et al.*, 2018). Using this approach, the design variables are divided into two groups.
The first group is called early-decision variables x_a. In an industrial development process, these are characterized by a large uncertainty and a strong influence on the system's performance. As a result, they have to be fixed in early phases of the development process. The approach of *solution-compensation spaces* allows the computation of an admissible interval for each early-decision variable, in which it may assume any value.
The second group are late-decision variables x_b. Due to their characteristic of being easily adjusted, they can be specified late in the development process. They act thereby as tuning values compensating for the decisions made regarding the values of early-decision variables and ensuring the required system performance. When computed using the approach in (Vogt *et al.*, 2018), every late-decision variable must be able to assume any value in its given interval.
Presuming that the provided intervals for the design variables x_a in classical solution spaces are too small, the intervals will be enlarged by *solution-compensation spaces* and changing the character of x_b. To accomplish this, a maximized box-shaped solution space is computed for all design variables x_a

https://doi.org/10.1017/dsi.2019.360 Published online by Cambridge University Press

such that there exists at least one set of values for the late-decision variables x_b resulting in a design that fulfils all requirements.

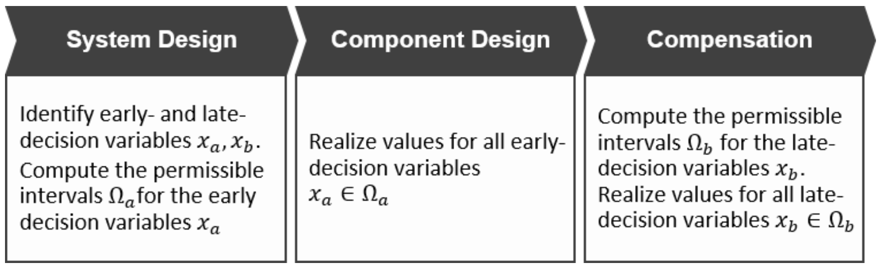
System Design
Identify early- and late-decision variables x_a, x_b. Compute the permissible intervals Ω_a for the early decision variables x_a
Component Design
Realize values for all early-decision variables $x_a \in \Omega_a$
Compensation
Compute the permissible intervals Ω_b for the late-decision variables x_b. Realize values for all late-decision variables $x_b \in \Omega_b$

Figure 3. Procedure using solution-compensation spaces, according to (Vogt et al., 2018)

The steps for application are shown in Figure 3. With regard to the system design, we have to identify early- and late-decision variables. These variables, the assigned design space and the system performance, which provide the constraints for permissible solutions, give the solution space for design variables x_a. In the next step, which is the component design, the values for all early-decision variables are set to a fixed value. A component design is the result thereby. All intervals for late-decision variables x_b are calculated by the last step. With a suitable choice of values for design variables x_b within these intervals the decisions made for variables x_a are compensated with respect to the system performance requirements.

As shown on the left side of the two-dimensional example in Figure 4, the interval for early decision variable x_a is significantly increased in comparison to the box-shaped solution spaces shown. Visible on the right side of Figure 4 is a fixed value for design variable x_a which is due to the component design step. Provided thereby is a given interval for design variable x_b (Ω_b^*) which leads to a compensation for the decision such that a solution for the system design remains within constraints and the required goals are reached.

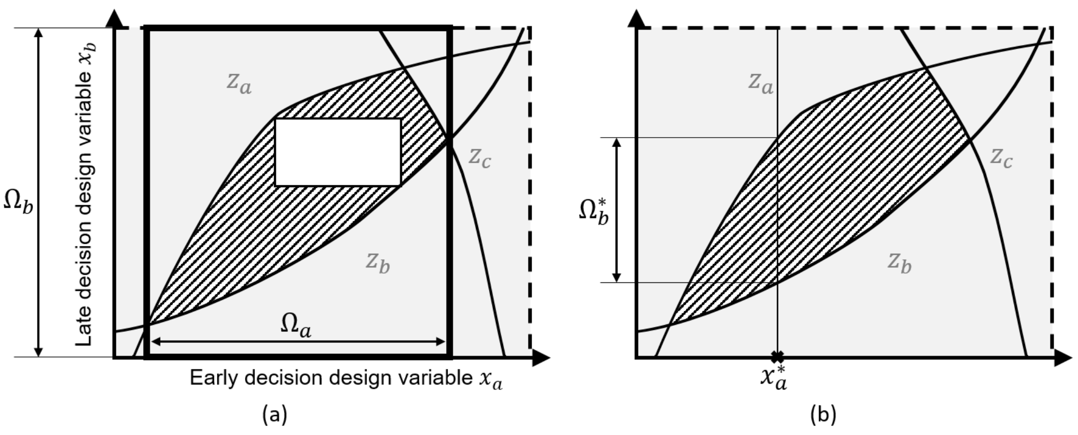

Figure 4. (a) Solution-compensation space (thick line) and (b) new solution space Ω_b^* created by choosing $x_a = x_a^*$, according to (Vogt et al., 2018)

Using solution-compensation spaces, the number of possible designs associated with the group of early-decision variables is greatly increased, and it may thus be easier to find a solution that satisfies all requirements. However, so far this benefit is provided to just one specific subproblem in the design process. The solution space provided for all late-decision variables after the realized design variables x_a may still be very small. Therefore, it is again divided into groups of early- and late-decision variables.

3 SEQUENTIAL UPDATING OF SOLUTION-COMPENSATION SPACES

Solution spaces can be used to reduce iteration in a design process for large and complex systems. When taking all requirements of different teams and disciplines involved into account, a solution space may become very small. Although formulating a requirement as space or set of designs provides more flexibility in creating a satisfactory design than providing a point target. Decoupling requirements is associated with a significant loss of solution space. Solution-compensation spaces were introduced to address this problem. The design variables are divided into two groups: early-decision and late-decision variables. Using the procedure described in sub-section 2.2, a larger range specifically for early-decision variables can be provided to the development team. The team designing the components related to the late-decision variables is able to compensate for the decisions made in the first step and to develop an overall system that achieves all goals.

But how to divide the system variables into early- and late-decision variables? The development process of a system defines the order in which the components are designed. This process has evolved during recent decades. The components with the largest impact and strong interdependency are designed in the beginning. Doing so fixes values of design variables with strongest influence on system performance, and subsequent in design activities by other teams for subsystem components have to deal with and iteratively improve this first design as mentioned in section 2.

In order to apply the robust design of solution spaces and solution-compensation spaces, the development process is utilized to divide the process into early- and late-decision variables. In the first step, all design variables in process step one are early-decision variables denoted by a vector $x_{a1} = (x_{a1,1}, x_{a1,2,\ldots})$. All design variables in the following design steps are used for compensation; these are regarded as the late-decision variables x_{b1}. The permissible intervals for variables x_{a1} are then computed, and a specific component design is realized within this interval. This *realization* involves choosing specific values for all design variables associated with this component.

Instead of one compensation step including all late-decision variables x_{b1} which could, by decoupling, again lead to very small solution boxes, the compensation step includes a further division into early- and late-decision variables. All design variables that will be specified in the second design step assume the role of early-decision variables x_{a2}. All design variables in the subsequent steps are still available for compensation, i.e., they are late-decision variables x_{b2}. This change of roles may be repeated: design variables that were treated as late-decision variables turn into design variables that are treated like early-decision variables. This is done, until all design variables are specified.

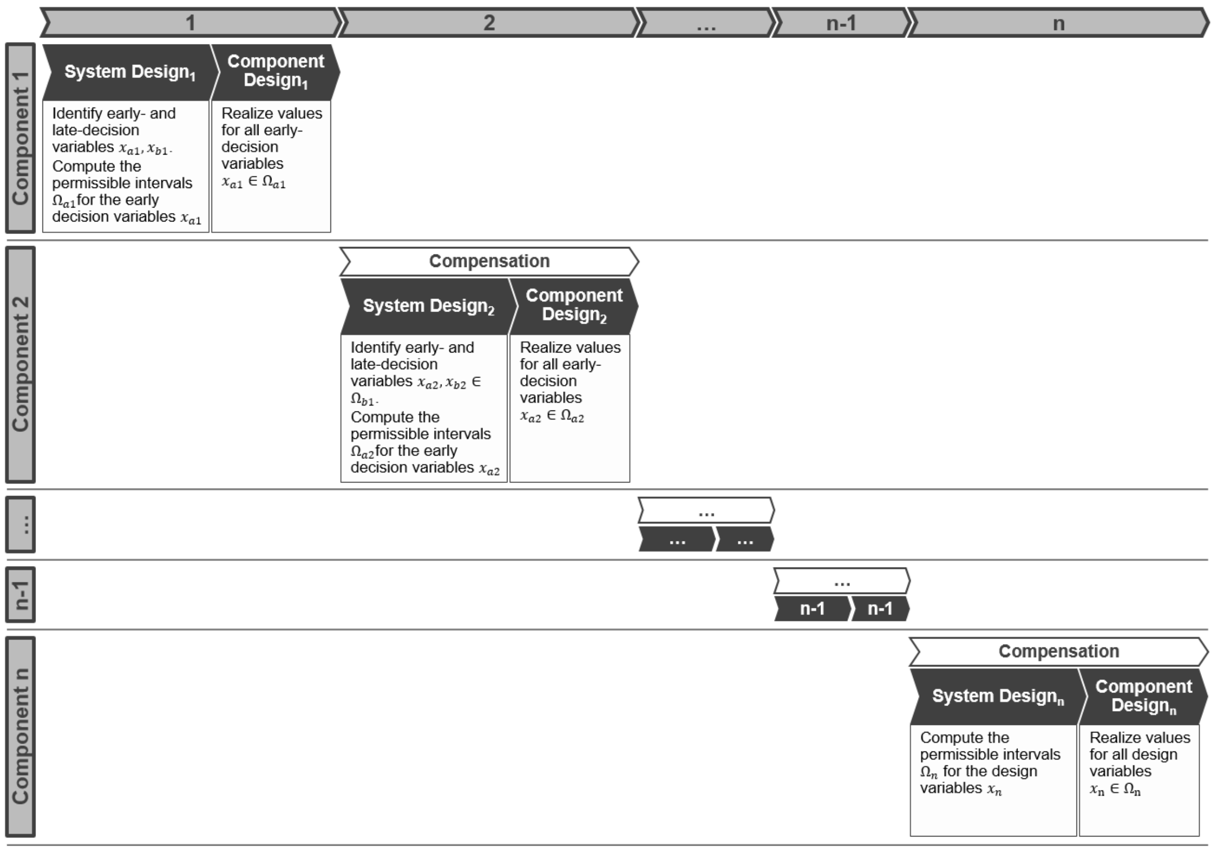
1
2
...
n-1
n
Component 1
System Design$_1$
Component Design$_1$
Identify early- and late-decision variables x_{a1}, x_{b1}. Compute the permissible intervals Ω_{a1} for the early decision variables x_{a1}
Realize values for all early-decision variables $x_{a1} \in \Omega_{a1}$
Component 2
Compensation
System Design$_2$
Component Design$_2$
Identify early- and late-decision variables $x_{a2}, x_{b2} \in \Omega_{b1}$. Compute the permissible intervals Ω_{a2} for the early decision variables x_{a2}
Realize values for all early-decision variables $x_{a2} \in \Omega_{a2}$
...
n-1
Component n
Compensation
System Design$_n$
Component Design$_n$
Compute the permissible intervals Ω_n for the design variables x_n
Realize values for all design variables $x_n \in \Omega_n$

Figure 5. Sequential updating solution-compensation spaces

By sequentially updating solution spaces, there is always an enlarged range for current focused design variables x_{ai} with respect to compensation possibilities regarding the succeeding design variables. By

https://doi.org/10.1017/dsi.2019.360 Published online by Cambridge University Press

contrast, in order to iteratively improve a single point design throughout the development process, all requirements are taken into account all of the time along with the remaining ways of compensating for realized component designs, thus realizing a solution for the overall system that ultimately achieves all system performance goals.

4 APPLICATION AND EVALUATION

4.1 Application to chassis design for vehicle dynamics

The development of driving dynamics behaviour for the chosen components in this application follows a waterfall process. The waterfall model is a sequential design approach. Each activity in a development phase must be completed before the next phase starts. The results of a phase are inputs for the next phase (Haberfellner *et al.*, 2012). A simple illustration of this is shown in Figure 6. A specific task for designing subsystem components is required in each of the rows. Each of the tasks is assigned to a different department or supplier. According to Heissing *et al.* (2011) the suspension design follows on the package, containing the determination of the weight distribution. The tires are developed, typically by suppliers, subsequent to that. With the design of the vertical components there is the possibility to adjust the characteristics in the end. The goal for this sub-process of vehicle development is to ensure a specific dynamic behaviour of the vehicle. In order to demonstrate the application of the methodology provided in the example here, only one design variable per group is considered, representing the component design problem. Note however that this can be easily extended to larger groups.

Table 1. Design variables

Variable	**Unit**	**Subsystem/Component**	**Description**
n_{RA}	%	Architecture	Weight distribution front/rear axle: m_{RA}/m_{totla}
$\sigma_{Y,RA}$	min/kN	Suspension	Gradient of toe angle over lateral force at the rear axle
μ_Y	-	Tire	Maximum lateral friction coefficient of tires at rear axle
$c_{arb,RA}$	N/mm	Vertical Components	Stiffness of the anti-roll bar at the rear axle

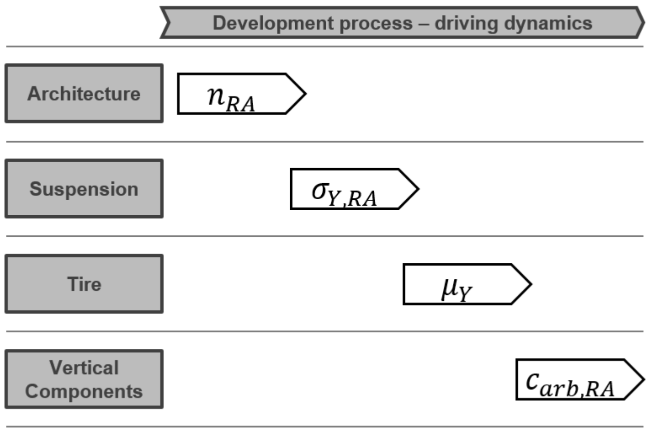

Figure 6. Development process for chassis component design

The system performance is simulated using a modified two-track model by Heissing *et al.* (2011). To evaluate whether the goals are reached, system performance measures are evaluated with respect to system requirements. These performance measures are determined in four simulated driving manoeuvres:

- Quasi-steady state cornering (QSSC): the vehicle follows a circular trajectory with a specified radius. The velocity is increased slowly enough that the state can be presumed to be constant at any time. The velocity is increased up to a speed at which the vehicle can no longer follow the specified radius.
- Weave: the vehicle maintains a constant velocity, and the steering is moved sinusoidally at a constant frequency
- Constantly increasing sinusoidal steering (CSST): the vehicle maintains a constant velocity, and the steering is moved sinusoidally with an increasing frequency

https://doi.org/10.1017/dsi.2019.360 Published online by Cambridge University Press

- Sine with dwell (SWD): the vehicle maintains a constant velocity, and the steering input is a sinusoid with a frequency of 0.7 Hz that pauses for 0.5 seconds after reaching the second peak.

The evaluated performance measures are shown in Table 2.

Figure 7 shows two-dimensional scatter plots of the vehicle performance evaluation. Each dot represents one design. The values for those design variables that are not shown on the axes of the plot are randomly chosen from their associated permissible interval shown on the other plots. For example, in the first column, all design variables have the entire design space as their associated interval. In column *2-start*, $\sigma_{Y,RA}$ is restricted to the smaller band between the two arrows.

A light grey dot represents a good design that fits all performance requirements. If a dot is marked by a dark symbol the design misses at least one goal, symbolized by the particular marker. By making an interval I_{xi} smaller, the random value is chosen from a smaller interval. If it finally has zero width, i.e., the design variable assumes the value x_a^*, this value is used for designs shown in the plots.

Table 2. Performance measures

Performance measure	**Lower limit**	**Upper limit**	**Unit**	**Description, driving manoeuvres**
z_{zy}	9.0		m/s²	Maximum lateral acceleration, QSSC
z_{ϕ}	2.85	3.15	°	Roll angle of the body at a specific lateral acceleration, QSSC
z_{ψ}	0.32	0.37	1/s	Maximum yaw rate gain due to steering wheel angle at a specific frequency, WEAVE
z_f	1.0	1.6	Hz	Natural frequency of yaw rate, CSST
$z_{v_{\psi}}$	100	135	%	Relative increase of yaw rate at a natural frequency compared to quasi static condition, CSST
z_t	70	78	ms	Delay between lateral acceleration and yaw rate at a specific frequency and lateral acceleration, CSST
z_{δ}	4.7		-	Maximum steering angle factor before loss of stability, SWD

In column *initial situation*, all of the requirements are provided and all design variables are uncertain. Design variables associated with the vehicle architecture are the first to be fixed. This means that n_{RA} is the first early-decision variable x_{a1} and $\sigma_{Y,RA}$, μ_y and $C_{arb,RA}$ are the late-decision variables x_{b1}. Calculating the solution-compensation spaces in this step shows that there is no feasible solution for high values of n_{RA} in the given design space (Figure 7). The range $I_{n_{RA}}$ provided for design variable n_{RA} is restricted to a maximum value in the design space. At the end of phase one, the weight distribution is set to a fixed value due to a decision on the architecture.

For example, when designing a fully electric vehicle, the battery, which has a certain size and weight, must be placed within the car's structure. This component has a large effect on the weight distribution. Different concepts can be discussed, e.g. T-storage or underbody storage, and can be evaluated if n_{RA} is within the specified range. Finally, a specific design is chosen and n_{RA}, which is no longer uncertain, becomes a fixed value.

The second design phase then begins, and all other design variables have to be fixed, thus finding a feasible solution and thereby compensating for the decision made for the architecture. In order to arrive at the coordinated process for designing the succeeding components and have appreciable breathing space for finding a suitable solution, the succeeding decisions to be made are cascaded. The design variable for the suspension $\sigma_{Y,RA}$, which is used in order to calculate the solution-compensation space, is the early-decision variable x_{a2}, and the remaining variables μ_Y and $C_{arb,RA}$ are the late-decision variables x_{b2}. The provided range to design $\sigma_{Y,RA}$ is restricted from the lower limit as well as from the upper limit. Nevertheless, there is a range $I_{\sigma Y,RA}$ in which this design variable can be chosen and which compensates for the decision made on the architecture in phase one. By the end of phase two, the design process for the suspension is finished and the variable $\sigma_{Y,RA}$ therefore set to a fixed value.

The methodology is continued in design steps three and four for tire variable μ_Y and anti-roll bar variable $C_{arb,RA}$. By the end of the development process, a feasible solution for the system emerges which fulfils all requirements without iteration steps. A sufficient range for designing the components without concurrent designs is provided at the same time.

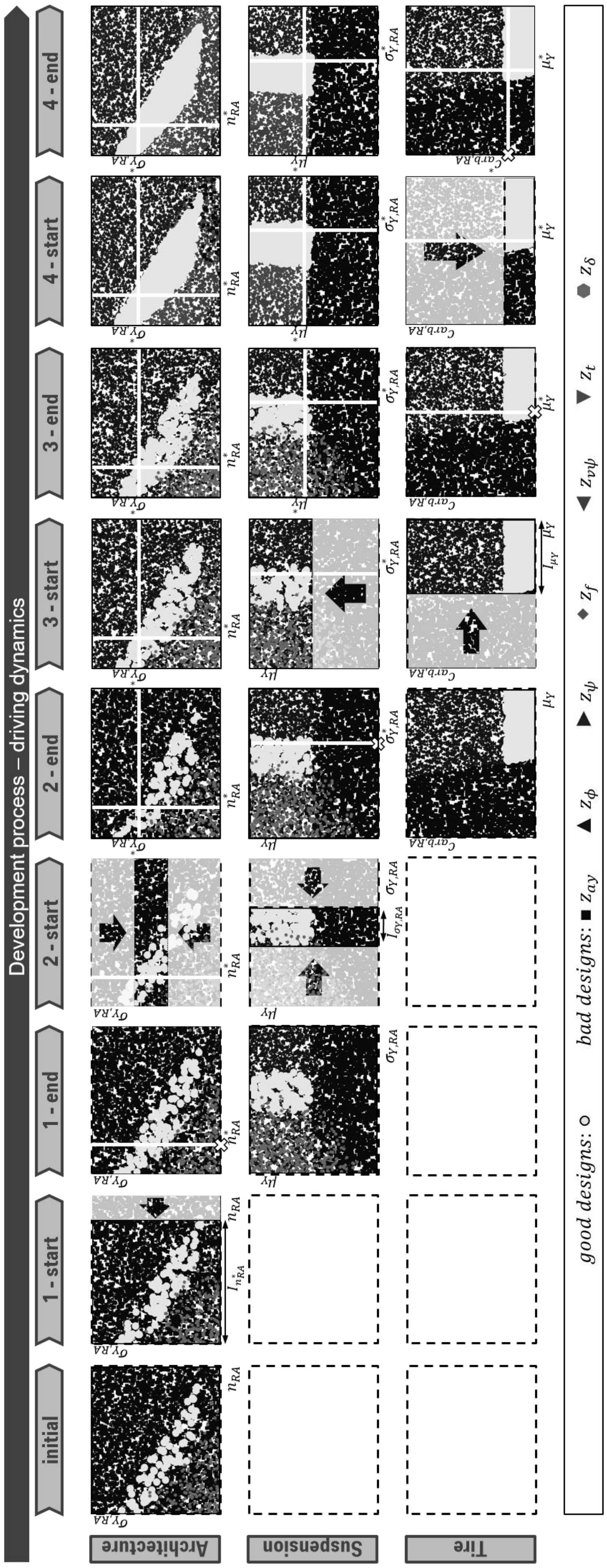
Development process – driving dynamics
initial
1 - start
1 - end
2 - start
2 - end
3 - start
3 - end
4 - start
4 - end
Architecture
Suspension
Tire
good designs: ○
bad designs: ■ z_{ay} ▲ z_{ϕ} ▲ z_{ψ} ◆ z_f ▼ $z_{v\psi}$ ▼ z_t ⬢ z_{δ}

Figure 7. Sequential design steps applied to chassis component design

https://doi.org/10.1017/dsi.2019.360 Published online by Cambridge University Press

4.2 Evaluation

In comparison to the classical box-shaped solution spaces, ranges provided here are significantly larger. Since the range always depends on the previous choice for a design variable valuable, an average interval size was determined for the evaluation. The left side of Figure 8 shows the normalized interval size with respect to the design space provided for each design variable ($I_{i,max} - I_{i,min})/\Omega_{\mathrm{ds,i}}$. The black bars are the intervals for calculating classical box-shaped solution spaces. The grey bars show the mean intervals of sequentially updated solution-compensation spaces. The whiskers at the grey bars show the standard deviation, calculated on the basis of various values for the previous step. This is why there is no deviation for the weight distribution (n_{RA}); it is the first choice to make, and all other design variables are always available for compensation.
The intervals associated with sequentially updated solution-compensation spaces are significantly larger in the first two steps. This is because more design variables used to compensate for the decisions will follow in the design process. The intervals in late design phases can also be smaller, especially when earlier decisions have chosen a solution variant at the edge of the provided space. However, larger intervals are also possible, as the upward deviation shows. A comparison of the mean values for the design variables in late phases shows no appreciable loss of interval size. The right side of Figure 8 shows the total minimum to total maximum from all trials as a grey surface, and the intervals for box-shaped solution spaces as a black line. It can be seen that, even when the provided intervals in the late phases are sometimes smaller, a larger part of the design space can be used to find a solution that leads to the desired result.

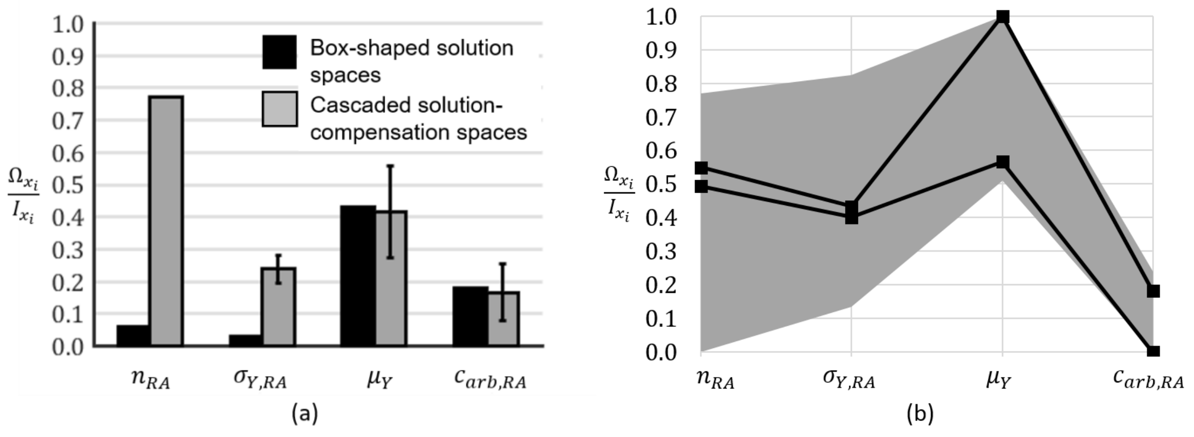

Figure 8. Performance of approaches based on solution spaces and sequentially updated solution-compensation spaces. (a) Fraction of complete design space (b) Location of possible solutions within the design space

5 CONCLUSION

In this paper, a methodology was presented for designing complex systems with strongly interacting design variables. It is based on solution spaces that were introduced to reduce the number of iterations and provide design teams sufficient space or flexibility for component design. By sequentially updating solution-compensation spaces during development, flexibility is further increased as permissible ranges for early-decision variables are significantly enlarged by the prospect of compensating with late-decision variables.
In contrast to Axiomatic Design this approach deals with highly interacting, complex systems. Rather than de- or uncoupling functional requirements, as Suh (1998) requires, sequentially updating solution-compensation spaces takes requirements as they are and provides independent, i.e. decoupled, target regions for component properties that ensure system requirements are satisfied.
In future work, the methodology will be applied to higher dimensional problems and be analysed due to the effect on the multi-dimensional solution spaces and compensation design variables provided.

https://doi.org/10.1017/dsi.2019.360 Published online by Cambridge University Press

REFERENCES

Andersson, P. (1996), *A process approach to robust design in early engineering design phases*, Lund Institute of Technology, Lund.

Browning, T.R. (2001), “Applying the design structure matrix to system decomposition and integration problems: a review and new directions”, *IEEE Transactions on Engineering Management*, Vol. 48 No. 3, pp. 292–306.

Ebro, M. and Howard, T.J. (2016), “Robust design principles for reducing variation in functional performance”, *Journal of Engineering Design*, Vol. 27 No. 1-3, pp. 75–92.

Ehrlenspiel, K. and Meerkamm, H. (2017), *Integrierte Produktentwicklung: Denkabläufe, Methodeneinsatz, Zusammenarbeit, 6., überarbeitete und erweiterte Auflage*, Hanser, München.

Erschen, S., Duddeck, F., Gerdts, M. and Zimmermann, M. (2018), “On the Optimal Decomposition of High-Dimensional Solution Spaces of Complex Systems”, *ASCE-ASME J. Risk and Uncert. in Engrg. Sys., Part B: Mech. Engrg.*, Vol. 4 No. 2, pp. 21008.

Freund, T., Würtenberger, J., Lotz, J., Rommel, C. and Kirchner, E. (2017), *Design for robustness - Systematic application of design guidelines to control uncertainty*, 21.-25.08.2017, Vancouver, Canada.

Haberfellner, R., Fricke, E., Weck, O.d. and Vössner, S. (2012), *Systems Engineering: Grundlagen und Anwendung, Orell Füssli*, Zürich.

Heissing, B., Ersoy, M. and Gies, S. (2011), *Fahrwerkhandbuch: Grundlagen, Fahrdynamik, Komponenten, Systeme, Mechatronik, Perspektiven, ATZ-MTZ-Fachbuch, 3., überarbeitete und erw. Aufl.*, Vieweg + Teubner, Wiesbaden.

Hendrix, E.M.T., Mecking, C.J. and Hendriks, T.H.B. (1996), “Finding robust solutions for product design problems”, *European Journal of Operational Research*, Vol. 92 No. 1, pp. 28–36.

Sobek II, D.K., Ward, A.C. and Liker, J.K. (1999), “Toyota’s principles of set-based concurrent engineering”, *Sloan Management Review*, Vol. 40 No. 2, pp. 67–83.

Suh, N.P. (1998), “Axiomatic Design Theory for Systems”, *Research in Engineering Design*, Vol. 10 No. 4, pp. 189–209.

Vogt, M.E., Duddeck, F., Wahle, M. and Zimmermann, M. (2018), “Optimizing tolerance to uncertainty in systems design with early- and late-decision variables”, *IMA Journal of Management Mathematics*, Vol. 4, pp. 469.

Zhou, J. and Li, M. (2014), “Advanced Robust Optimization With Interval Uncertainty Using a Single-Looped Structure and Sequential Quadratic Programming”, *Journal of Mechanical Design*, Vol. 136 No. 2, pp. 21008.

Zimmermann, M. and Hoessle, J.E.v. (2013), “Computing solution spaces for robust design”, *International Journal for Numerical Methods in Engineering*, Vol. 94 No. 3, pp. 290–307.

https://doi.org/10.1017/dsi.2019.360 Published online by Cambridge University Press