Netherlands Institute
for Metals Research

virtual tool reworking

new strategies in die design using finite element forming simulations

R.A. Lingbeek

Samenstelling van de Promotiecommissie:

voorzitter en secretaris	
Prof. dr. F. Eissing	Universiteit Twente
promotor	
Prof. dr. ir. J. Huétink	Universiteit Twente
Prof. dr. ir. F.J.A.M. van Houten	Universiteit Twente
assistent-promotor	
Dr. ir. V.T. Meinders	Universiteit Twente
leden	
Prof Dr.-Ing. S. Reese	Technische Universität Braunschweig
Prof. dr. R.H. Wagoner	Ohio State University
Prof. dr. ir. R. Akkerman	Universiteit Twente
Dr. Ir. A.H. van den Boogaard	Universiteit Twente
Dr.-Ing. habil. Dipl.-Phys. Stephan Ohnimus	INPRO GmbH Berlin

ISBN 978-90-77172-38-4
First printing March 2008
Keywords: metal forming, Finite Element Method, optimisation, springback, tooling

This thesis was prepared in LaTeX by the author and printed by Print Partners Ipskamp, Enschede, from an electronic document

Cover design and photography by R.A. Lingbeek

VIRTUAL TOOL REWORKING

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit Twente, op gezag van
de rector magnificus, prof. dr. W.H.M Zijm,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op vrijdag 18 april 2008 om 13.15 uur

door

Roald Arnoud Lingbeek
geboren op 19 maart 1980
te Vilvoorde (België)

Dit proefschrift is goedgekeurd door de promotoren:
Prof. dr. ir. J. Huétink
Prof. dr. ir. F.J.A.M. van Houten
en de assistent-promotor:
Dr. ir. V.T. Meinders

Summary

Computer-aided engineering (CAE) has significantly expedited product development in the automotive industry. In the process design and planning of deep drawing processes, computer-aided design tools and finite element (FE) simulations are used together in order to achieve a high-quality product within an acceptable time-span. Here, finding the right shape for the forming tools is one of the most important tasks. However, when the tools are manufactured and tested on the prototype press the quality of the prototype parts rarely satisfies the requirements straightaway. Therefore, manual reworking of the forming tools is required. Because reworking is highly time-consuming and because a lot of experience is required by the tool technicians, this is the most significant bottleneck in the process-planning today.

The two phenomena that cause problems in the product quality are the *deformation of the press and forming tools* during forming, and the *springback* of the product after release of the tools. Especially when high-strength steels are used, both phenomena cause significant problems. To a large extent, they cannot be avoided and therefore they have to be compensated in the shape of the forming tools. In this thesis, various algorithmic methods are developed to carry out this compensation in a numerical context. Ideally, the goal is to avoid tool reworking altogether, and to achieve this goal, three problems need to be solved: Firstly, the accuracy of the forming simulation must be improved in order to obtain a reliable representation of the forming process. Secondly, an algorithmic framework needs to be developed for the geometrical compensation of the forming tools. Thirdly, the proposed shape changes must be transferred back to the CAD description of the tools.

The deformation of the press and tools can be divided into two categories. Firstly, the global deformation of the bed-plate, slide and forming tools and secondly the local deformation of the forming tool surface. In this thesis, these deformations are demonstrated for the cross-die forming process. This is a blank-material testing process and the results of the material test were reported to vary due to tool deflection. Indeed, both global and local deformations can be reproduced with a forming simulation, using deformable tool models. For this, the general purpose FE code ABAQUS is used. Unfortunately, the calculation time has increased tremendously. This implies that carrying out such simulations is not feasible for full-scale industrial processes. A more efficient way of modeling tool elasticity needs to be found.

Static condensation, a well-known technique for reducing the size of finite element models, does not bring the anticipated decrease of numerical cost. The principle of the method is to pre-solve a part of the system of equations so the deformation is only calculated at locations in the tool geometry that are actively required during the forming simulation. However, the reduced set of equations turns out to be

much harder to solve. In contrast, the so-called Deformable Rigid Bodies (DRBs) *do* provide a tremendous reduction in the calculation cost. Here, the deformation of a body is approximated as a linear combination of pre-calculated deformation modes. When the load is global, a small number of modes provides sufficient accuracy. This makes it possible to include the entire press structure into the forming simulation. A DRB module has been developed and it is implemented in the FE simulation code DiekA. As a test, the tools of the cross-die forming process are modeled as DRBs. The simulation results show the same phenomena as the regular ABAQUS simulation, however the increase in numerical cost due to the elastic tool models amounted 8% only.

Springback is the deformation of the blank that occurs when the forming tools are opened. This shape deviation may cause problems in the assembly process for the car-body. In order to produce parts with the correct shape, the forming tools must be compensated. In tube-bending, compensation is achieved by overbending: the tube is bent further than the desired angle to obtain the right shape after springback. The mathematically generalized description of this idea is called the Displacement Adjustment (DA) method. For a simple forming process, the stretch-bending of a bar, it is shown why DA compensation is a nonlinear procedure. This analysis reveals why a different compensation is required for different forming geometries, material or process parameters. In industrial processes, the compensation is different at the various locations in the product, therefore, optimal compensation cannot be achieved in one step. The iterative application of DA does lead to the optimal tool shape in only a few iterations.

In any case, the quality of the tool surfaces must be maintained during compensation. As an addition to the discrete DA principle, the smooth displacement adjustment (SDA) algorithm has been developed. The algorithm leaves the blankholder area of the tools and the gap width between them unchanged. Undercuts that could occur during compensation are automatically removed. Three industrial springback problems are been solved using this method in combination with a commercial forming simulation program.

The previously mentioned SDA compensation principle has been developed for mesh geometries. Mesh geometries suffice to show the effectiveness of the compensation, however, the shape changes must finally be transferred to the tool CAD description. The quality and smoothness of the tool surfaces determines the appearance of a sheet metal product and therefore they must comply to very strict tolerances. When the tool surfaces are manually modified, problems occur. Either the surface quality is lost, or the details of the compensation cannot be transferred fully.

Therefore, a CAD geometry compensation algorithm has been developed. A grid of sampling points is projected onto the geometry. These sampling points are compensated using the SDA algorithm, and then the surfaces are simultaneously re-fitted to the points. During this calculation, smoothness boundary conditions can be applied to the surface transitions in order to preserve the surface quality. This is also possible for complex trimmed surfaces. The algorithm has proven to work well in several academic examples. In fact, small defects in the initial geometry were automatically removed during compensation.

Samenvatting

In de autoindustrie heeft computer-aided-engineering (CAE) het productontwikkelingsproces significant verkort. Bij het procesontwerp en de planning van dieptrekprocessen worden zowel computer-ondersteunde ontwerpwerktuigen als eindige elementen (EE) simulaties gebruikt om in een acceptabele tijdsperiode een product van hoge kwaliteit te ontwikkelen. Hierbij is het vinden van de optimale vorm van de omvormgereedschappen één van de belangrijkste opgaven. Ondanks het gebruik van geavanceerde programma's voldoet het product zelden aan de gestelde kwaliteitseisen als het voor de eerste keer wordt geproduceerd op de prototype-pers. Daarom is handmatige nabewerking van de werktuigen noodzakelijk. Omdat deze nabewerkingen veel tijd in beslag nemen en veel ervaring vereisen, is dit een van de grootste bottlenecks in de procesplanning.

Twee fenomenen zorgen voor kwaliteitsproblemen in het omgevormde product: De deformatie van de pers en de werktuigen gedurende het dieptrekken, en de terugvering van de platine na het openen van de matrijs. Deze problemen zijn groot, met name wanneer hoge-sterkte stalen worden gebruikt. Voorkoming is in het algemeen niet mogelijk en daarom moet de vorm van de omvormwerktuigen gecompenseerd worden. In dit proefschrift worden algorithmische methoden ontwikkeld om deze compensatie al in een numerieke context uit te voeren. Het ultieme doel is, nabewerkingen compleet te vermijden. Hiervoor moeten drie problemen worden opgelost: Ten eerste moet de nauwkeurigheid van de omvormsimulatie worden verbeterd opdat het dieptrekproces realiteitsgetrouw wordt afgebeeld. Ten tweede moet een algorithmisch raamwerk worden ontwikkeld voor de compensatie van de omvormwerktuigen. Tenslotte moeten de berekende vormveranderingen weer in de CAD-beschrijving van de werktuigen worden teruggevoerd.

De deformatie van de pers en omvormwerktuigen kan in twee categoriën worden ingedeeld. Ten eerste de globale deformatie van de perstafel, stoter en omvormwerktuigen, en ten tweede de locale deformatie aan het oppervlak van de werktuigen. Deze beide soorten deformaties worden in dit proefschrift getoond voor het cross-die omvormproces. Dit proces wordt als materiaaltest gebruikt. In publicaties wordt gemeld dat de resultaten afhankelijk zijn van de werktuigdeformatie. Deze afhankelijkheid kan met een eindige elementen-simulatie gereproduceerd worden wanneer deformeerbare werktuigmodellen worden toegepast. Dit is mogelijk met universele EE software, zoals ABAQUS. Helaas veroorzaken de deformeerbare werktuigmodellen een enorme stijging in de rekentijd. Dit betekent dat het uitvoeren van dergelijke simulaties voor omvormprocessen van industriële schaal onmogelijk is. Een efficiëntere methode voor de modellering van werktuigdeformaties moet worden ontwikkeld.

Statische condensatie, een bekende methode om de grootte van EE modellen te reduceren, leidt niet tot reductie van de numerieke kosten. Het principe van de methode is de eliminatie van een gedeelte van het systeem van vergelijkingen, zodanig dat de deformatie van de werktuiggeometrie alleen berekend wordt in die locaties die actief gebruikt worden in de omvormsimulatie. Het oplossen van het gereduceerde vergelijkingssystem blijkt echter zeer inefficiënt. De zogenaamde Deformable Rigid Bodies (DRB) zijn *wel* in staat de numerieke kosten significant te reduceren. De vervorming van een lichaam wordt hier berekend als een lineaire combinatie van voorberekende deformatie-moden. Wanneer de belasting relatief uniform is, levert een klein aantal moden al voldoende nauwkeurigheid. Dit maakt het mogelijk om de gehele pers-setup in de simulatie mee te nemen. Een DRB-module is ontwikkeld en geimplementeerd in de omvormsimulatiecode DiekA. In een testberekening zijn de werktuigen van het cross-die proces als DRB gemodelleerd. De simulatieresultaten tonen dezelfde fenomenen als de reguliere ABAQUS simulatie. De toename in rekentijd, als gevolg van de elastische werktuigmodellen, bedraagt slechts 8%.

Terugvering is de deformatie van de platine die optreedt wanneer de omvormwerktuigen teruggetrokken worden. De vormafwijking kan problemen veroorzaken in het assemblageproces van de autocarosserie. Om onderdelen te produceren met een correcte geometrie moeten de werktuigen gecompenseerd worden. Bij het buigen van buizen wordt compensatie bereikt door overbuigen: de buis wordt tot een kleinere hoek gebogen dan gewenst, opdat deze na terugvering de juiste vorm verkrijgt. De wiskundig gegeneraliseerde beschrijving van dit idee is de Displacement Adjustment (DA) methode. Gebruikmakend van een simpel omvormproces, het strekbuigen van een staaf, wordt aangetoond dat compensatie een niet-lineaire procedure is. Deze analyse laat zien waarom de compensatie verschillend is bij verschillende productgeometrieën, materialen en procesparameters. In industriële omvormprocessen varieert de compensatie over de productgeometrie, en daarom kan de optimale werktuigvorm niet in één stap worden gevonden. Het iteratieve gebruik van de DA-methode leidt tot een goede werktuigvorm in een gering aantal iteraties.

Het is belangrijk, de kwaliteit van de werktuigoppervlakken te waarborgen gedurende de compensatie. Als toevoeging bij het DA algorithme is de Smooth Displacement Adjustment (SDA) methode ontwikkeld. Dit algorithme laat het blankholdergebied en de spleetbreedte tussen de werktuigen onveranderd. Daarnaast worden ondersnijdingen, die eventueel bij compensatie optreden, automatisch verwijderd. Drie industriële terugveer-problemen zijn opgelost met deze methode, in combinatie met een commercieel omvormsimulatieprogramma.

Het SDA compensatieprincipe is ontwikkeld voor mesh-geometriën. Met deze meshes kan in een simulatie de het resultaat van een compensatie gecontroleerd worden. Uiteindelijk moeten de geometrieveranderingen echter worden doorgevoerd naar de CAD beschrijving van de werktuigen. De kwaliteit en gladheid van de werktuigoppervlakken bepalen de uiterlijke verschijning van het gelakte carrosseriepaneel en moeten daarom aan zeer strenge toleranties voldoen. Wanneer de geometrie handmatig aangepast wordt, treden problemen op: De kwaliteit van de oppervlakken neemt af, of details van de compensatie kunnen niet volledig overgebracht worden.

Daarom is een CAD-compensatiealgorithme ontwikkeld. Een net van punten wordt daarbij op de geometrie geprojecteerd. De projectiepunten worden gecompenseerd

met het SDA principe, waarna de CAD-oppervlakken simultaan aan de gemodificeerde punten aangelegd worden. Gedurende deze berekening kunnen gladheidsrandvoorwaarden opgelegd worden tussen de oppervlakken, opdat de kwaliteit van de geometrie behouden blijft. Dit is ook mogelijk voor complexe getrimde oppervlakken. Het compensatiealgorithme functioneert uitstekend bij diverse tests met academische voorbeeldgeometrieën. Zelfs initiële gebreken in de geometrie worden verwijderd gedurende de compensatie.

Preface

The research described in this thesis was carried out in the framework of the Strategic Research Programme of the Netherlands Institute for Metals Research in the Netherlands (www.nimr.nl) and was carried out at INPRO Innovationsgesellschaft für fortgeschrittene Produktionssysteme in der Fahrzeugindustrie mbH, Berlin, Germany.

Roald Lingbeek, Berlin, January 2008

Notation convention

Within one chapter variable characters have one meaning. If certain variables are used in more than one chapter, or when variables have similar meanings in different chapters, the author has strived to use consistent naming as much as possible. However, a glossary is provided at the end of each chapter for convenience .

For the variable-characters, the following conventions have been adopted:

Scalar: Upper and lowercase italic and Greek fonts
Examples: α, a, B

Vector: lowercase bold font or bold Greek font
Examples: $\mathbf{d}, \boldsymbol{\theta}$

Matrix or tensor: Uppercase bold font, stress $\boldsymbol{\sigma}$ and strain $\boldsymbol{\varepsilon}$
Examples: $\mathbf{D}, \boldsymbol{\Theta}, \boldsymbol{\sigma}$
Note: since only well known tensors are used in this thesis and since tensors do not play an important role, they are not marked individually.

Cartesian coordinate, location vector: lowercase italic font with arrow
Example:

$$\vec{c} = \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix}$$

Note that equations using this type of variable can be seen as three independent scalar equations for each component x, y and z

Vector of coordinates: lowercase bold font or bold Greek font with arrow
Example: $\vec{\mathbf{f}}$
Note that, in principle, this type of variable is a matrix, however, equations using it should be regarded as three independent equations for each component x, y and z

Matrix of coordinates: uppercase bold font or bold Greek font with arrow
Example: $\vec{\mathbf{H}}$
Again, equations using this type of variable should be regarded as three independent equations for each component x, y and z

The mathematics in this thesis have been produced from an engineer's point of view. As an introduction into mathematical science the author *highly* recommends the paper by Renteln and Dundes [57].

Contents

Summary **1**

Preface **7**

1 From product design to production process **11**
1.1 Introduction . . . 11
1.2 The deep drawing process . . . 12
1.3 The traditional process design . . . 13
1.4 Opportunities for the digital factory concept . . . 15
1.4.1 Finite Element forming simulations for tool optimization . . . 16
1.4.2 The influence of press and tool deformations . . . 17
1.4.3 Virtual compensation of forming tools . . . 17
1.4.4 Tool CAD geometry modification . . . 17
1.5 Research hypotheses . . . 18

2 Efficient modeling of tool and press deformation **19**
2.1 Introduction . . . 19
2.1.1 Categorizing tool and press deformation . . . 19
2.1.2 Application potential for simulations with deformable tools . . . 21
2.2 Tool deformation in the cross-die benchmark . . . 22
2.2.1 FE simulation of the process . . . 22
2.3 Static condensation . . . 27
2.4 Deformable Rigid Bodies . . . 30
2.4.1 The principle . . . 30
2.4.2 Calculating modes . . . 31
2.4.3 Approximation error analysis . . . 32
2.4.4 Interactions between DRBs . . . 33
2.4.5 Examples . . . 35
2.4.6 Reducing the mesh-dependent error . . . 39
2.5 Including DRBs in the forming simulation . . . 42
2.6 Simulating the cross-die benchmark using DRBs . . . 44
2.7 Conclusion and outlook . . . 45

3 Computer-aided Springback Compensation **49**
3.1 Introduction . . . 49
3.2 Handling springback in industry . . . 50
3.2.1 Springback measurement and assessment . . . 50
3.2.2 From manual to numerical springback compensation . . . 52
3.3 Compensation algorithm principles . . . 53
3.3.1 Displacement Adjustment . . . 53

3.3.2 Spring Forward . . . 55
3.3.3 Analytical verification of iterative and one-step DA . . . 58
3.4 Springback compensation for industrial processes . . . 67
3.4.1 Retaining tool surface quality . . . 68
3.4.2 Retaining the blankholder surface . . . 71
3.4.3 Gap-width preservation and undercut avoidance . . . 74
3.4.4 Implementation . . . 75
3.5 Examples . . . 75
3.5.1 Process 1: Free forming . . . 75
3.5.2 Process 2: Inner panel drawing . . . 78
3.5.3 Process 3: Outer panel drawing . . . 84
3.6 Conclusion . . . 85

4 Modification of tool CAD geometries 89
4.1 Introduction . . . 89
4.2 Surface qualification and quality . . . 90
4.2.1 Class A surfaces . . . 90
4.2.2 Class B surfaces . . . 92
4.3 Global surface modification algorithms . . . 92
4.3.1 Surface compensation principle . . . 93
4.3.2 Transitions between surfaces . . . 97
4.3.3 Multiple surfaces with simple boundary conditions . . . 98
4.3.4 Trimmed surfaces . . . 101
4.3.5 General continuity boundary conditions . . . 102
4.3.6 Results . . . 108
4.4 Conclusion and future work . . . 110

Conclusion and recommendations 113

Acknowledgements / About the author 117

Bibliography 119

Chapter 1

From product design to production process

1.1 Introduction

In recent years, the economic pressure on the car industry has increased, and at the same time, consumer demand has become more diverse. The car manufacturers are faced by the challenge to develop cheaper and more environmentally friendly cars, and to fill smaller market niches with exciting designs. Therefore the time-to-market for new vehicles needs to be reduced for the company to remain competitive in the automotive marketplace [38]. This issue is of particular importance in the production of sheet metal parts.

Since the 1980s, the time required for the design and development process has roughly halved [61]. This has become possible thanks to two major innovations:

- Modular concepts, parallel/concurrent engineering
- Computer-aided engineering

The construction of modern cars is divided into different functional units. Subframes, engines and wheel suspension assemblies in particular are designed as separate modules that can be used for different types of cars. This is commonly called *platform strategy* and saves on development time and production cost. The savings are substantial when a company builds many different models under various brand names. Another advantage is that well-defined modules can be developed concurrently, so different development teams can work on one project at the same time.

Since the 1960s, Computer-aided engineering (CAE) has started to play a more and more prominent role in the development process. Figure 1.1 provides a brief overview of the advances of computer-based development tools. Computer-aided design (CAD) enables the engineers to create more complex shapes (1) and it provides a platform for project management because different parts can be archived, reviewed and updated easily. Ideally, the data of all car parts are coupled as a digital assembly, so geometry changes can be carried out without causing errors, even in the complex concurrent engineering context. CAD was initially used as a drawing tool only, but already in the 1960s, the digital data were applied in production processes (2) such as numerically controlled (NC) machining. This is called Computer-aided

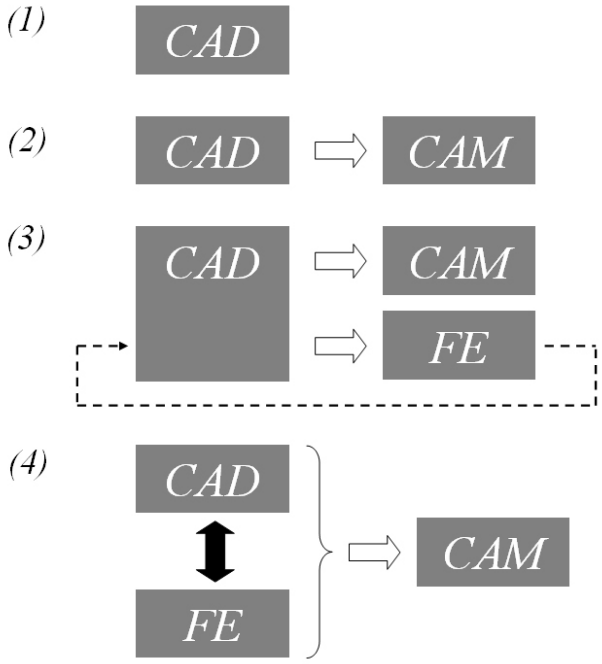


Figure 1.1: Four stages in virtual product development

manufacturing or *CAM*. However, this integration goes much further today.

Finite Elements (FE) software allows the engineer to analyze the stiffness, strength and dynamic properties of a part in great detail, before a prototype part is produced (3). Today it has also become possible to model production processes such as polymer injection moulding processes, the forging of gears or, the main subject of this thesis, the forming of sheet metal parts. The conclusions from such analyses are transferred back to the CAD system in order to optimize the designs.

The final goal is to fully integrate FE and CAD systems, which makes it possible to design, test and automatically optimize the part entirely in the *virtual factory* [67] (4). Ideally, this results in a 'first-time-right' product and production process. The deep drawing of sheet metal products is a very complex process that presents major challenges, but also many opportunities for this virtual factory concept.

1.2 The deep drawing process

Car body structures are made almost exclusively of sheet metal parts. The predominant manufacturing method for these products is the deep drawing process, shown schematically for a double-action press in Figure 1.2. A sheet of metal, the blank, is clamped between the blankholder and die. The die, blankholder and blank move downwards, onto the punch, which forms the product [21].

The process is physically complex and highly sensitive to process parameter changes. The blank is subject to extremely high pressures and large friction forces. When the blank slides into the die cavity and is formed into the product, it is bent as well as stretched. The way the sheet metal is plastically deformed influences many

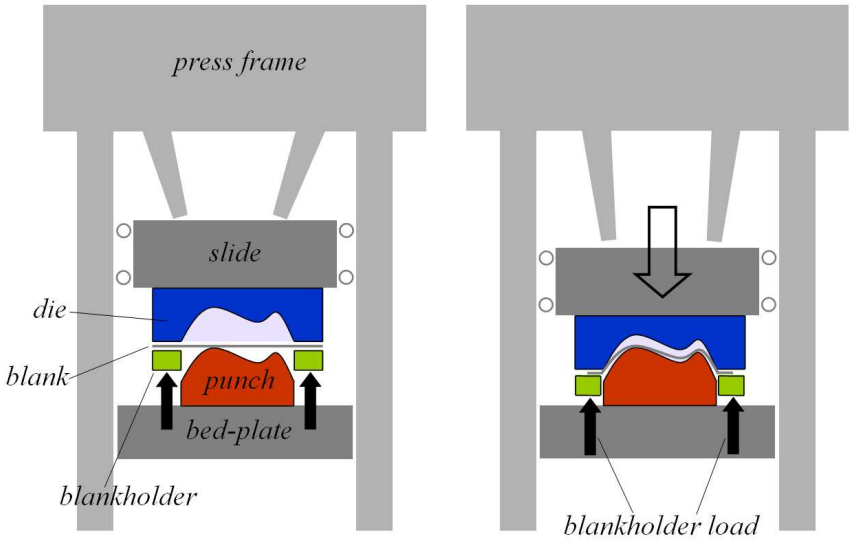


Figure 1.2: The deep drawing process

of the product's quality measures, such as the thickness of the blank, rupture and wrinkling. Obviously, the geometry of the product needs to meet the requirements too. In this respect *springback*, the change of the blank shape after retraction of the tools, has become one of the most important challenges for process-design engineers.

1.3 The traditional process design

The process design for a deep drawn product, shown in Figure 1.4, is a complex task. Process engineers need to find a satisfactory compromise between product quality and process stability. Over time the geometrical tolerances have become smaller, for example to allow the part to be laser-welded. At the same time the materials have become more complex: To save weight, high-strength steels and aluminium are used more frequently. For some parts even tailor-made blanks, consisting of different material grades, are used. As an example, Figure 1.3 shows the variety of steel grades that are used in a modern car body.

The most important requirements for the product and process are:

- Sufficient thickness throughout the product
- Avoidance of rupture in the product
- No visible wrinkles in the product
- The shape of the product should be within geometrical tolerances
- The press force should be within the press limits

The blank material and initial thickness are generally defined by the product-designer, so the process engineer needs to achieve a satisfactory process by the following means:

1. Forming stages and sequence

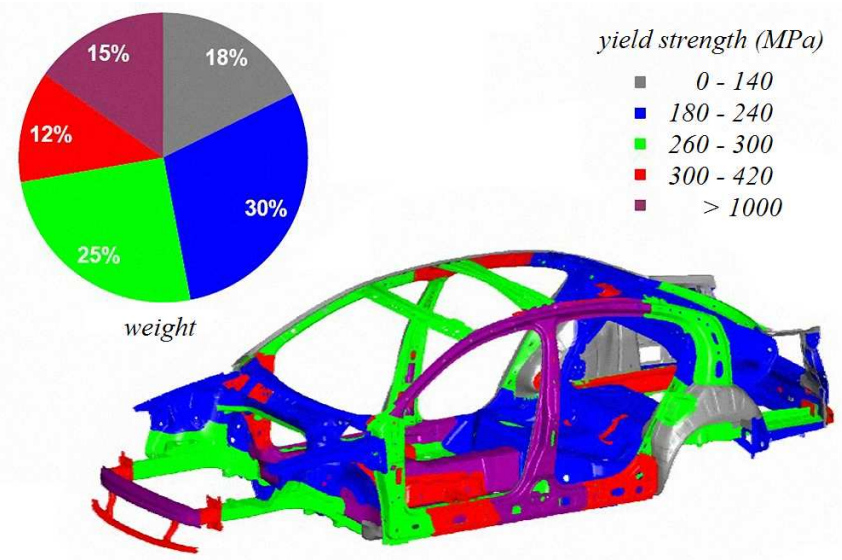


Figure 1.3: Steel grades in a modern car (Picture courtesy of Volkswagen AG)

2. Geometry of the die addendum
3. Blankholder force
4. Drawbeads
5. Geometry of the product-area of the tools
6. Lubrication

Generally, the goal is to invoke as much plastic strain in the product as possible whilst avoiding rupture. The larger the in-plane tensile stresses become relative to the bending stresses, the more stable the shape of the product will be [41]. In Chapter 3, this will be discussed in more detail. The drawing of the product can be carried out in several stages. This way, the flow of the blank into the die-cavity can be controlled more accurately, allowing larger deformations without tearing. The different forming and cutting stages and the optimal sequence are determined in the first phase of Figure 1.4, *tool surface and process design*.

Changing the blankholder force is probably the most intuitive way to influence the blank draw-in. A low force allows the blank to flow into the die cavity easily, likely resulting in wrinkles and an increase in the amount of springback. A high force might result in rupture. The addition of drawbeads and the design of the die-addendum are also powerful methods to influence the blank flow. Nowadays, the tool surface and process design are carried out in the digital domain, using fast FE simulations [62] to check the behavior and feasibility of the process.

When the process and the shape of the tool surfaces have been defined with sufficient confidence, the process is feasible and radical changes to the process are not expected anymore, the three-dimensional design of the tool-structure is produced. The tools are heavy molded structures which are generally designed using strict company guidelines [2] to avoid problems with the molding process. The structural

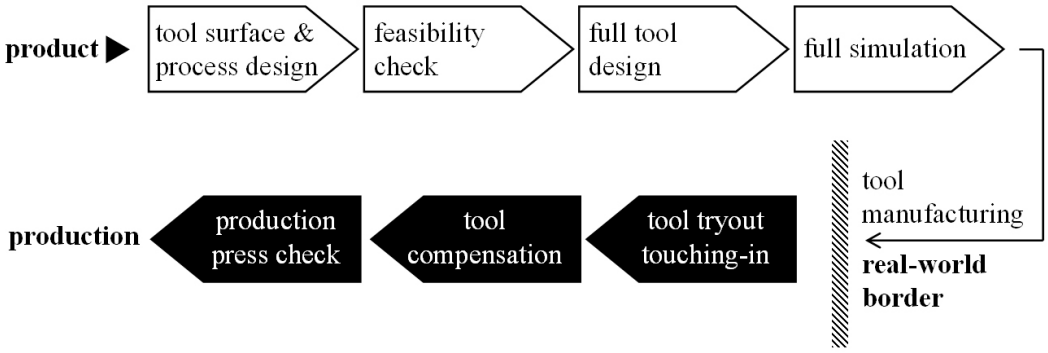


Figure 1.4: The process planning

stiffness of the tools is important too [38], but is generally not yet optimized numerically [51].

At the same time, more detailed simulations are carried out for the process [62]. The accuracy of the results is considerably higher than the previous simulations and therefore the calculation times are much longer. Sometimes the simulation also includes a prediction of the springback of the product. The goal is to apply more detailed optimizations to the tool and process design.

When the *real world border* is crossed, the tools are actually manufactured and an extensive try-out phase is started in the press workshop. Touching-in means that the tool surfaces are carefully ground by hand. This is done because the surfaces need to be very smooth, but more importantly, to allow the process engineer to control the blank flow exactly. In some cases lubrication is also applied to the blank, but this is impractical in most production processes.

To solve remaining problems with the process or product, global changes are applied to the tool surfaces to compensate for springback or for the deflection of the tools under the press load. This is very time consuming, as it requires a redesign by CAD engineers, and the corrections then need to be applied to the forming tools. This involves additional machining or even welding operations. When the process runs smoothly on the prototype press, the tools are transferred to the production press. Due to differences in press-behavior, slight changes to the process settings and tools are carried out even at this phase. For example, the deflection of the press-frame and different kinematics of the slide influence the deformation of the blank already.

1.4 Opportunities for the digital factory concept

Kim et al. [38] state that the use of CAE tools have reduced the process design phase from 16.5 to 8.5 months already. However, many of the die-trial tasks could also be carried out before the real world border, reducing the planning effort even further. Figure 1.5 shows the ideal process planning. The main target of this thesis is to avoid the time consuming tool reworking and to carry out necessary tool compensation in the digital domain already.

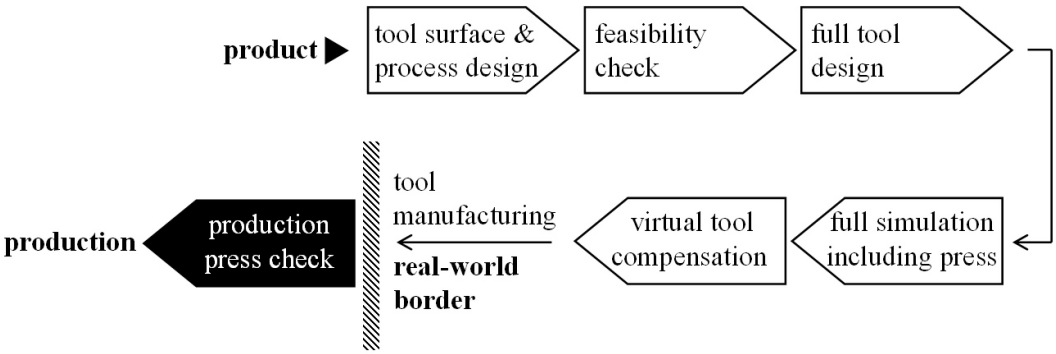


Figure 1.5: The digital factory process planning

1.4.1 Finite Element forming simulations for tool optimization

Before the scope of this thesis is discussed in more detail, the idea behind Finite Element (FE) forming simulations needs to be explained briefly, because this method is the foundation of almost all analyses and procedures presented. The deformation of bodies, such as the blank and the deep drawing tools are governed by the equations of the continuum mechanics theory. The FE method is a way to approximately solve these equations for a body with an arbitrary shape in relation with its boundary conditions and loads. From an 'engineering' point of view, the basic idea of the FE method is to divide the geometry of the body into a set of small *elements*, which are interconnected via *nodes*. Because the deformation of each individual element in relationship with the load on its nodes can be described in a manageable set of equations, it is possible to calculate the deformation of the entire structure by coupling all element equations in a large matrix equation. For an in-depth treatment of the FE method [33] and [9] are highly recommended.

The simulation of deep drawing using the FE method presents a major challenge, it is a large area of active research. Paper [77] provides a historical overview. An in-depth introduction can be found in [72]. The three major topics in forming simulations are:

- Modeling elasto-plastic material behavior [32, 8, 69].
- Modeling contact and friction between the blank and the forming tools [76, 79, 39].
- Time integration schemes, element technology and other numerical topics [72, 54, 73, 13].

An interesting point was made by Meinders [49]: In the case of a benchmark study, different results were obtained by different users even when the FE software was identical. This shows that carrying out forming simulations is no trivial task. It is helpful to consider the recommendations in guideline documents such as [14].

Even though remarkable improvements are still being achieved in the three modeling challenges, the focus of recent conferences such as ESAFORM, IDDRG, NUMIFORM and NUMISHEET has broadened to include the *application* of FE simulations. Instead of just using the results of a simulation for (feasibility) checks, they

can be used actively in optimization techniques. In the thesis of Bonte [12] it is shown how, for example, the blankholder force can be optimized numerically. The virtual reworking of tool surfaces based on simulation results is a different challenge. In numerical optimization, the number of parameters is limited and much too small to directly treat a CAD surface description, or FE mesh. So, in order to obtain the optimal tool surface shape, specific compensation algorithms are required. In this thesis, methods for the compensation of tool deflection and springback have been developed.

1.4.2 The influence of press and tool deformations

Obviously, a reliable prediction of these two phenomena is of vital importance for the results. At the moment, tool and press deformations are not taken into consideration during the FE simulation at all. The reason for this is the numerical cost of the simulation. When the tools, or even the press, are modeled as FE meshes, the size of the matrix equations increases tremendously and instead of a couple of hours, several days might be required for the calculation. There are techniques to reduce this numerical cost. These will be the main focus of Chapter 2. The so called deformable-rigid bodies have been combined with static reduction to provide a highly efficient means of modeling tool deformation. A software module has been developed and included in the FE code DiekA.

1.4.3 Virtual compensation of forming tools

Springback, the deformation of the blank after release of the tools, is another challenge for both process-engineers and computer simulations and will be discussed in Chapter 3. However, for springback the emphasis will be on the geometric compensation and not so much on the physical phenomenon, as recent research has improved the accuracy of the prediction tremendously.

The most effective compensation algorithm is called 'Displacement Adjustment' or DA [24] and is based on the springback displacement. Even though the principle of DA is conceptually simple, the interaction with the forming process turns out to be complex, as Section 3.3.3 reveals. There, the method is demonstrated for a simple forming process that can be modeled analytically.

Many other challenges in springback compensation are encountered when industrial parts are compensated. An industrially applicable springback compensation strategy called *SDA* is developed. This will be the subject of the final section of the chapter. The compensation methods can also be applied to solve geometrical deviations in the product that are due to tool/press deformation. The approach is completely identical to the springback compensation strategy, so it is not treated in detail.

1.4.4 Tool CAD geometry modification

As stated in the introduction, the integration of FE and CAD systems is one of the main assets of computer-aided product development. However, because of the differences between the CAD and FE descriptions for geometry, this integration is mostly a one-way solution: it turns out to be very hard to return the geometry changes

from an FE-based compensation or optimization to a CAD system. The parametric surface functions used in CAD systems are very flexible during the design-phase. This flexibility makes them also unpredictable when the surface parameters need to be modified, and this problem is aggravated when surfaces are interconnected with smoothness boundary conditions. Chapter 4 provides some views on the mathematics behind the modification of CAD geometries.

1.5 Research hypotheses

The above topics can be summarized in three research hypotheses

Hypothesis 1a
The deflection of the press and forming tools influences the quality of the deep-drawn product.

Hypothesis 1b
Press and tool elasticity can be included in the forming simulation at an acceptable cost.

Hypothesis 2
Using FE simulation results, the surface of the forming tools can be compensated automatically for springback and tool deflection to produce a geometrically accurate product.

Hypothesis 3
Mesh-based shape modifications can be automatically transferred back to the tool CAD geometry.

Chapter 2

Efficient modeling of tool and press deformation

2.1 Introduction

Deep drawing is an incredibly sensitive process. Even small phenomena that are hard to measure may influence the blank flow and therefore the quality of the final product. The deformation of the press and tools during forming is such a phenomenon. The deformations are small, but they are unavoidable and need to be taken into account during the process planning, preferably in the FE simulation phase. Currently the solving of problems related to tool and press deformations depends on experience and experiments only, and it requires a lot of time.

The following subsection will show where unwanted deformations occur in the press during forming, and how the related process problems may be reduced or solved. In Section 2.2 these problems are demonstrated in the simulation of a relatively simple forming process: the cross-die benchmark. It has been found that the numerical cost of including tool and press deformation is so high that industrial application is unfeasible, even when increasing computer processing power is taken into consideration. Therefore, the following sections are dedicated to techniques to decrease the numerical cost required for including tool and press deformations in an FE simulation. Section 2.3 shows the advantages and disadvantages of static condensation, a well-known technique to reduce the size of the system of FE equations. Then, in Section 2.4 the principles behind Deformable Rigid Bodies (DRBs) are explained. Section 2.5 shows how the final solution, a combination of static reduction and modal decomposition, was implemented in a forming simulation code. Finally, the method is tested in Section 2.6, using the cross-die as an example.

2.1.1 Categorizing tool and press deformation

Deformations of the press and tools occur at various locations. Four different categories are identified schematically in Figure 2.1.

Press-frame deformation (1) is the deflection of the press frame. The press frame is an extremely heavy structure and its deformation is therefore not significant for the results of the forming process. Press deformation (2) includes the deformation of the bed-plate and slide. In contrast to the press frame, these components can deform substantially, the deformations are in the order of magnitude of several

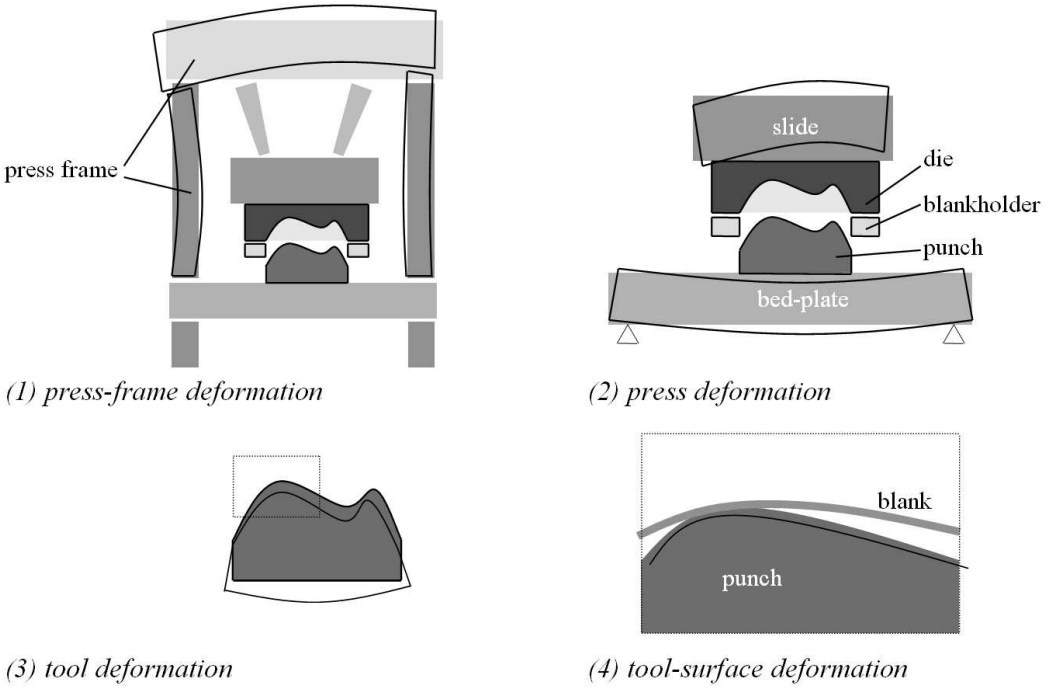


Figure 2.1: Different types of tool and press deformations [46]

millimeters [31], especially in large presses with a large tonnage that are used for parts made of high-strength steels. The slide not only deflects, but it can also tilt slightly, which has an influence on the drawing process as well. Production presses are tested and measured regularly to avoid unexpected production problems [27]. Wiemer [78] gives an extensive overview of press deformations and shows a detailed spring damper model that could also be used in a simulation context.

The heavy loads on punch, blankholder and die results in deformation of these tools (3), even though these are stiff structures. As an additional complexity, it is impossible to separate the tool deformation from the press deformation, as the bed-plate and slide support the tools. Hayashi [31] has measured the deflection of a set of deep-drawing tools on three different presses and obtained very different results for each press. The interaction between press and tools is also reflected in the fact that the touching-in of the tool, which was carried out on the prototype press, needs to be repeated on the production press. Because the press and tool deflection are linked so strongly, it is hard to optimize the tool structure separately (as proposed in [51]). Instead, in industry tool deflection is generally minimized by using heavy ribbed structures that are designed following company guidelines [2].

All previously mentioned press and tool deformations are regarded as macro-deformations, which means that they are global deformations with a large 'wavelength'. Tool-surface deformation (4) is a local deformation, caused by high contact pressure between blank and tool, for example at the die-shoulder and in the blankholder area. The order of magnitude of these deformations is considerably lower (max. 0.1mm). Since friction depends strongly on small changes in the pressure field between the blank and the tools, small changes in the tool geometry have a large influence on the simulation results, blank draw-in and springback.

Tool surface deformation is the focus of almost all present publications on elastic deformations in tools, for example [18], [60] and [37]. All authors report considerable influence, however, in most publications the tools were meshed coarsely, and numerical issues were discovered in modeling the two-sided contact between the tools and the blank [60]. This made it impossible to state whether taking tool deformations into account brought a provable advantage.

2.1.2 Application potential for simulations with deformable tools

For the remainder of the thesis, the deformation categories are summarized in global (press frame, press and tool deformation) and local (tool surface) deformations. The goal of this chapter is to investigate the influence of both categories and to include them in the FE forming simulation at acceptable cost. The tool deformation prediction can be used in various ways.

Firstly, when unacceptable global deformations are encountered during the FE simulations, the results can be used to compensate the shape of the tool surfaces.

A second area of application involves the local deformations: the touching in of tools. The touching in of the forming tools currently requires about 350 to 500 hours on the prototype press and then another 100 to 200 hours on the mass-production press [31]. When tool and press deformations are taken into account during FE analysis, the blank draw-in prediction will become better, avoiding unexpected changes to the tools during the tool tryout phase.

The elastic properties of the tools can also be exploited in a positive way. Some advanced blankholder designs were deliberately constructed to be flexible [19, 29, 30, 11], so varying blankholder loads can be applied at different locations. Normally, the press cushion exerts a uniform pressure onto the blankholder. Varying the pressure on the blankholder brings a new way of control for the forming process. This method has the following advantages:

- Time is saved in the tool testing phase, the amount of tool reworking is reduced[29]
- Increased control over blank-flow makes the production of more complex parts possible, sometimes a forming stage can be omitted
- Increased control over the forming process during production allows for correcting the process when a new batch of sheet material has (slightly) different material properties

Despite these advantages and the fact that most modern presses are equipped with multiple pressure groups, flexible blankholders are not applied much because of the added complexity is not backed up by process planning experience. As a third application, elastic tool modeling in FE simulations might provide the flexible tools with a better acceptance. When the effect of variable blankholder loading can be tested in the FE testing phases, it is much easier to gain understanding and experience with these advanced tools. This is desirable since the process control parameters for deep drawing are generally fixed and limited, whereas flexible tools provide an instantaneous means to influence the process.

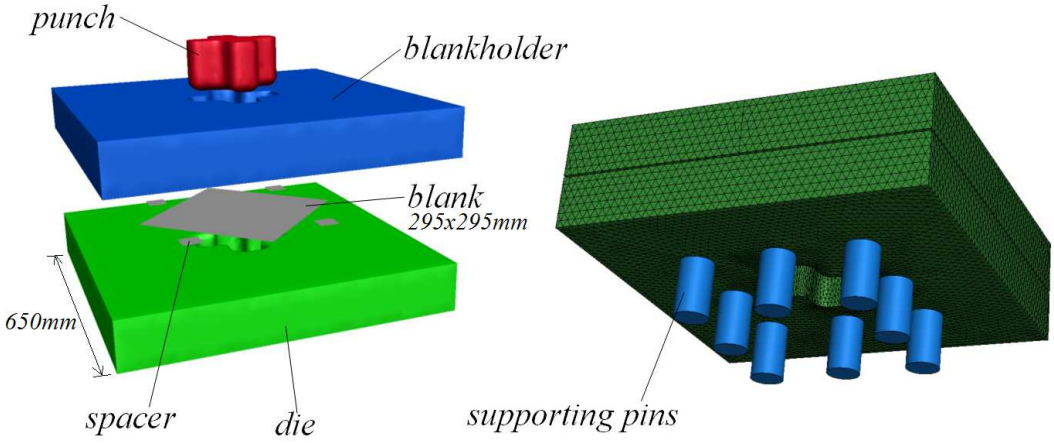


Figure 2.2: The cross-die process

2.2 Tool deformation in the cross-die benchmark

Before the previously mentioned applications are taken into consideration, the first goal is to actually show the effects of tool and press deformations in a manageable forming process. The cross-die benchmark is a forming process that clearly shows these effects. It will serve as the vehicle for most of the analyses in this chapter.

The cross-die process is shown in Figure 2.2. It is used industrially as a material test [6] and provides insight in the formability of a steel grade: The idea is to increase the size of the blank in a series of forming tests until fracture occurs. The maximum allowed blank size is defined as the cross-die benchmark value.

In this thesis, the blank has the size of 295 by 295mm and is made from St14 steel with a thickness of 0.7mm. The blankholder load totals 300kN. During the experiments, the process revealed a high sensitivity to tool deformation. In the prototype press, the tools are supported by a set of pins that allow more tool deflection than a regular bed-plate. Depending on configurations of these pins different benchmark results were found [6]. In order to reduce this sensitivity, small squares called spacers were placed around the blank [6]. These spacers are made from the same sheet-material as the blank. The experimenters intended to make the gap between blankholder and die more even, because due to tool deformations, the gap-width had become nonuniform. Unfortunately, the spacers made the problem worse.

2.2.1 FE simulation of the process

A FE analysis is a good way to analyze the process and show the influence of tool deformation and the use of spacers. In [6] tool deformation was calculated in a separate structural FE simulation and then transferred to the DiekA forming simulation by using a variable blankholder force function. The reason for this was, that modeling deformable tools was not possible in this FE forming code. However, it is possible to carry out such a simulation with a general purpose FE code, at a substantial additional cost. ABAQUS has been used in this project to compare a regular simulation with rigid tool models to a simulation using deformable tool models. Table 2.1 shows the settings of both simulations. Note that for simplicity,

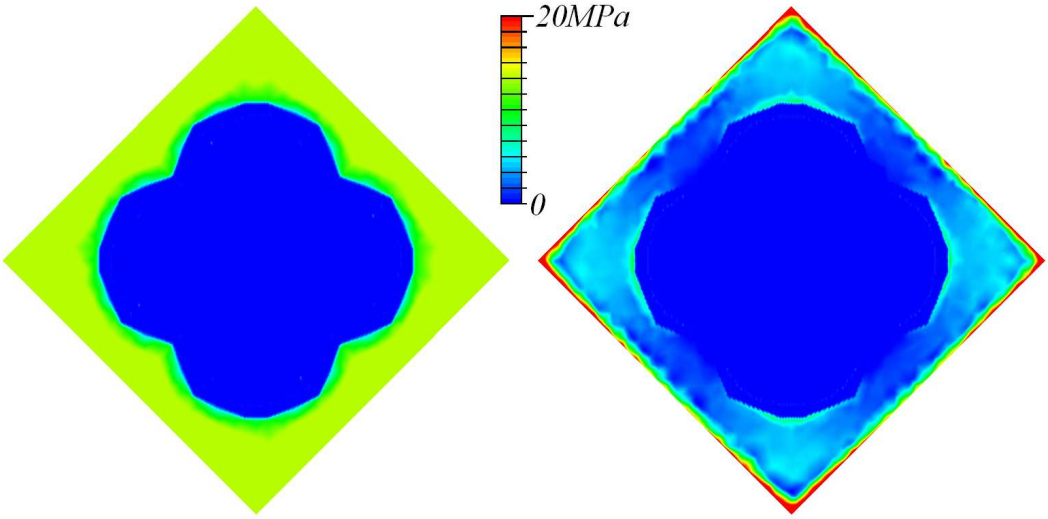


Figure 2.3: Contact pressure, rigid (left) and deformable (right) tools

regular tetrahedral elements were used for the tool models. These elements are too stiff and it is recommend to use ABAQUS C3D10M elements in future applications.

	Rigid tools	Deformable tools
Calculation type	Static implicit	
Blank elements	4-node red. integration shell S4R 8-node solid-shell SC8R	
Tool elements	4-node rigid-body elements R3D4	Tetrahedral solid elements C3D4
Contact	Penalty Default stiffness Node-to-surface Contact stabilization	
Number of elements	33124	187298

Table 2.1: Settings for the ABAQUS forming simulations

The forming process consists of two phases; blankholder loading and forming. The contact pressure from the tools onto the blank defines the amount of friction and therefore the amount of draw-in. The contact pressure distributions for deformable and rigid calculations are compared after the blankholder closing phase in Figure 2.3. Note that there is no pressure in the middle area of the blank, as forming has not yet started. In this process the blankholder area, the part of the blank where it is clamped between die and blankholder, is completely flat. For this reason, a homogeneous pressure distribution was expected.

This is the case for the calculation with rigid tools. However, when the tools are allowed to deform, even the slightest deflection of the tools results in a localization of the pressure field to the edge of the blank. The reason for this is made clear

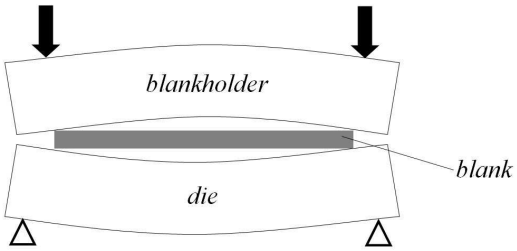


Figure 2.4: Bending of the die and blankholder (schematically)

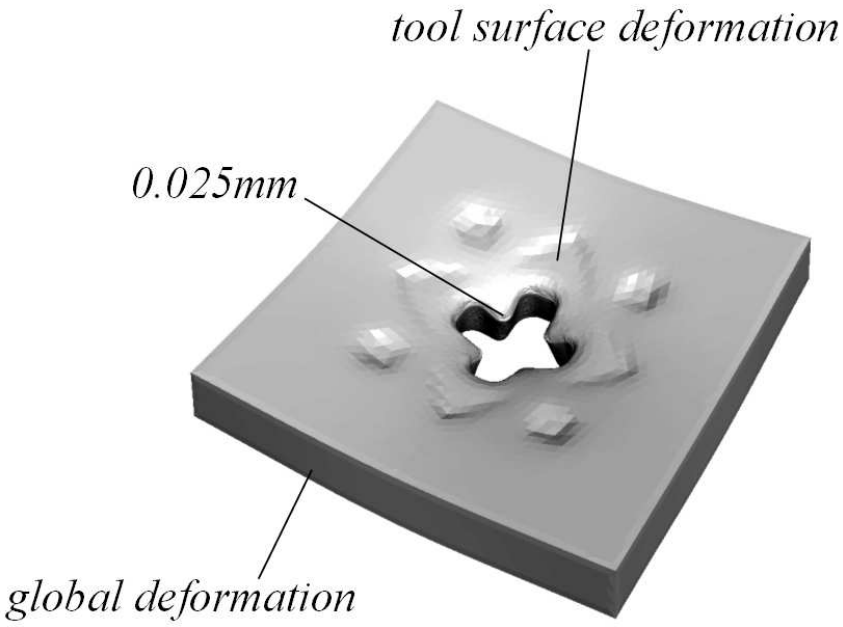


Figure 2.5: Deformed die (x5000)

schematically in Figure 2.4.

The deformation of the die after the completed forming stage is visualized in Figure 2.5. The deformation was multiplied by 5000 for visualization purposes. The figure shows that the deformation is a combination of global tool deformation and local tool-surface deformation. Note that the spacers also cause deformation in the tools, they carry a part of the blankholder load.

Due to the in-plane compression of the blank in the blankholder area, it thickens considerably during draw-in. The contact pressure maximizes at the thickest spots, lifting up the blankholder slightly thereby relieving the spacers. These thickening spots can be observed on an experimental blank as shiny spots in the photograph 2.6. In these areas the blank was 'polished' due to the high friction.

The ABAQUS calculation shows the same pressure spots (see Figure 2.7). In the left picture, rigid tools were used. Because the blankholder is rigid, it is lifted up entirely, almost completely relieving the spacers. However, when the tools are allowed to deform, they do overtake a considerable amount of the blankholder force from the blank. In the right picture, this can be seen clearly: There is a high pressure

Figure 2.6: Shiny spots on the blank reveal high-pressure zones (picture courtesy of Corus RD&T)

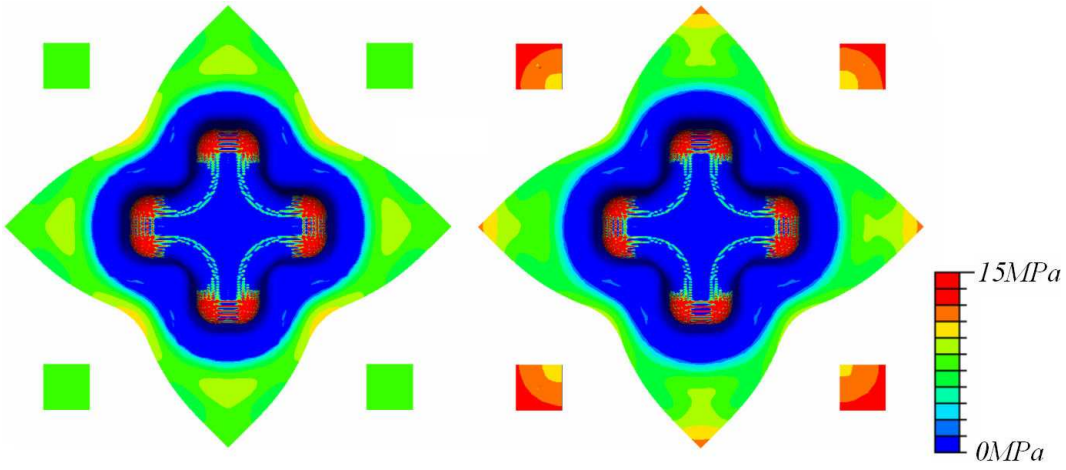


Figure 2.7: Pressure distribution for rigid (left) and deformable tools (right)

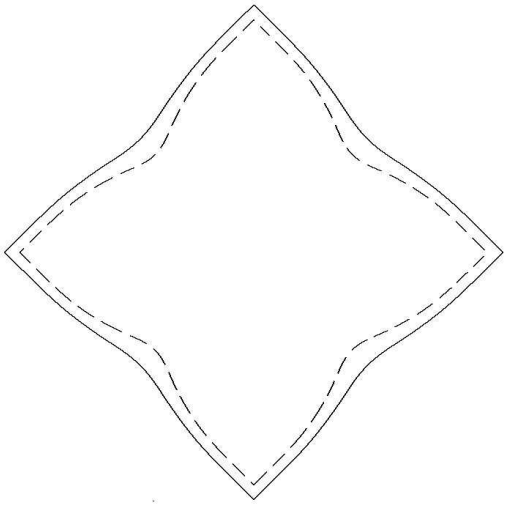


Figure 2.8: Draw-in of the blank for rigid tools (solid line) and deformable tools (dashed line)

on the spacers, and the size of the high pressure spots is reduced. Because of the reduction in blankholder pressure on the blank, the draw-in is larger.

Figure 2.8 shows the draw-in for both deformable and rigid simulations, and the difference is considerable. Due to the larger draw-in, the calculation with deformable tools shows a higher tendency for blank-wrinkling, whereas the calculation with rigid tools predicts a higher risk for rupture. This is visualized by the forming-limit curves (FLC) in Figure 2.9. The left FLC is recorded at 70% of the punch stroke, the right FLC at the end of the punch stroke. In the case of rigid tools, the strains become extremely large at full punch stroke, which would mean complete failure. The results of the calculation with deformable tools also exceed the safety limit, but the result is not as bad.

It would be interesting to compare the different supporting-pin configurations and to compare the simulation results with the experiments. However, this requires much more detailed measurements and a more accurate FE modeling (especially for the contact conditions) and is beyond the scope of this thesis.

The example does, however, strongly confirm hypothesis 1a *The deflection of the press and forming tools influences the quality of the deep-drawn product.* Based on the results that were achieved by using deformable tool models this simple example process, an increase in simulation accuracy is expected for industrial products as well.

However, especially in the case of complex and large car body parts, the size of the required tool meshes increases the cost of the simulation immensely. Due to the complex tool designs, tool meshes with several millions of DOFs are no exception. Because the cost of an FE simulation increases more than linearly in relationship with the amount of DOFs, this would result in simulations that are several orders of magnitude too large to carry out in an industrial context, even considering the rapid increase in computer performance.

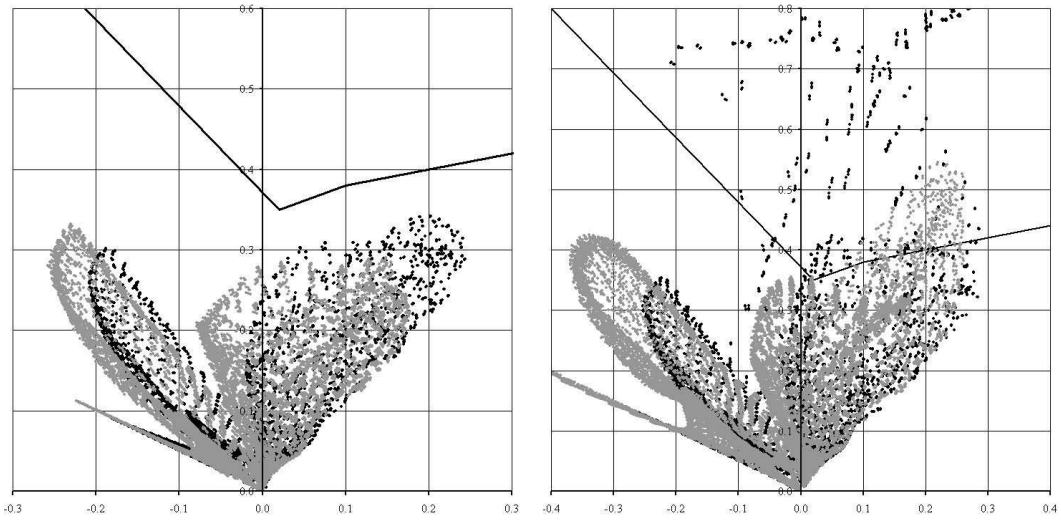


Figure 2.9: FLC at 70% (left) and full punch-stroke (right). Deformable tools in grey, rigid tools in black

Therefore, more efficient ways of modeling tool elasticity have to be found. Due to the small deformations, tool and press deformation can be considered as a linear elastic problem. Two interesting strategies to increase the efficiency of such problems are static reduction [33] and the so-called Deformable Rigid Bodies [10, 46]. In the following section, these methods are explained and their usability is verified.

2.3 Static condensation

Static condensation is a technique for reducing the size of a linear FE calculation. The FE model is reduced to generate only the output for a specific set of (retained) degrees of freedom (DOF). There is no loss of accuracy. This technique was demonstrated for tool elasticity modeling in [59].

The principle
The idea of static condensation is to speed up the calculation by *pre-solving* a part of the equation, and by doing so removing unnecessary DOFs from the active equations. This pre-solving needs to be carried out only once, and the results can be used in many consecutive deformation calculations. In the case of tool meshes, a substantial amount of DOFs can be saved, as the response of the tools and press-components is only required at the spots where they come into contact, or have connections with other bodies. This means that all nodes at the interior of the tools are not actively required either. This is clarified in Figure 2.10

Before the condensation technique is demonstrated, the linear-elastic FE problem needs to be derived briefly. A more in-depth treatment can be found in [33, 40]. The (static) equilibrium condition forms the basis of the system of equations.

$$\begin{cases} \overrightarrow{\nabla}\boldsymbol{\sigma}\left(\tilde{\mathbf{u}}(x)\right) = \mathbf{0} & \text{at } \Omega \\ \boldsymbol{\sigma}\cdot\mathbf{n} = \tilde{\mathbf{t}} & \text{at } \Gamma_t \\ \tilde{\mathbf{u}}(x) = \mathbf{0} & \text{at } \Gamma_u \end{cases} \tag{2.1}$$

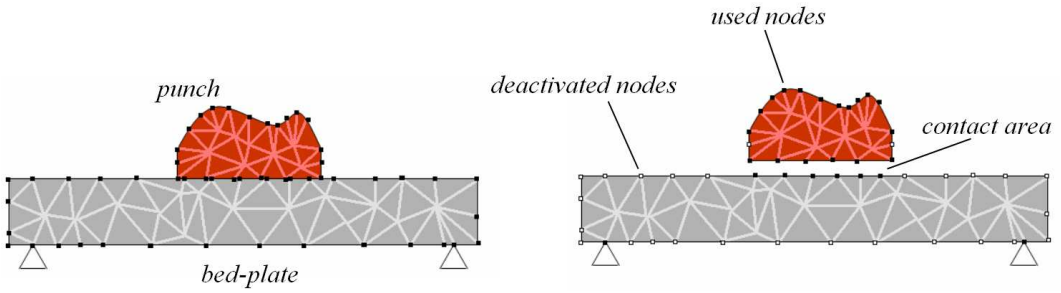


Figure 2.10: Required and internal nodes in punch and bed-plate (schematically)

Here, the stress distribution is $\boldsymbol{\sigma}$, the displacement field is $\tilde{\mathbf{u}}(x)$. Ω defines the deformable body in space. Γ_t designates the part of the boundary of the body where traction loads $\tilde{\mathbf{t}}$ are applied. The normal to the boundary surface is $\mathbf{n}$. As a boundary condition, the displacements are zero at Γ_u. The volume force is omitted for the sake of clarity. An ansatz-space with a set of shape functions $\mathbf{N}$ is introduced for the displacement field:

$$\tilde{\mathbf{u}}(x) = \mathbf{N}\mathbf{u} \tag{2.2}$$

The discretized (nodal) displacements are $\mathbf{u}$. As the tool deformations are very small, the strain tensor can be calculated as:

$$\boldsymbol{\varepsilon}(\tilde{\mathbf{u}}) = \frac{\tilde{\mathbf{u}}\overleftarrow{\nabla} + \overrightarrow{\nabla}\tilde{\mathbf{u}}}{2} = \left(\frac{\mathbf{N}\overleftarrow{\nabla} + \overrightarrow{\nabla}\mathbf{N}}{2}\right)\mathbf{u} = \mathbf{B}\mathbf{u} \tag{2.3}$$

The constitutive equation is

$$\boldsymbol{\sigma}(\tilde{\mathbf{u}}) = \underline{\mathbf{C}}\boldsymbol{\varepsilon}(\tilde{\mathbf{u}}) = \underline{\mathbf{C}}\mathbf{B}\mathbf{u} \tag{2.4}$$

The tensor $\underline{\mathbf{C}}$ defines the material behavior. Using Gauss' Theorem and Galerkin's method, Equation 2.1 can be rewritten in the familiar form with the stiffness matrix $\mathbf{K}$ and load vector $\mathbf{f}$.

$$\underbrace{\int_\Omega \mathbf{B}^T\underline{\mathbf{C}}\mathbf{B}\mathrm{d}\Omega}_{\mathbf{K}}\ \mathbf{u} = \underbrace{\int_{\Gamma_t}\mathbf{N}\mathbf{t}\mathrm{d}\Gamma}_{\mathbf{f}} \qquad \mathbf{u} = \mathbf{f} \tag{2.5}$$

Static condensation is now used to reduce the size of the system of equations. In the next equation, the 'master' DOFs with the subscript r are to be retained, the DOFs with subscript c are to be condensed out:

$$\begin{bmatrix} \mathbf{K}_{rr} & \mathbf{K}_{rc} \\ \mathbf{K}_{cr} & \mathbf{K}_{cc} \end{bmatrix} \begin{Bmatrix} \mathbf{u}_r \\ \mathbf{u}_c \end{Bmatrix} = \begin{Bmatrix} \mathbf{f}_r \\ \mathbf{f}_c \end{Bmatrix} \tag{2.6}$$

In case of the elastic tools the load is zero on the nodes that are condensed out, $\mathbf{f}_c$=0. Therefore Equation (2.6) becomes:

$$\mathbf{K}'\mathbf{u}_r = \mathbf{f}_r \tag{2.7}$$

with

$$\mathbf{K}' = \mathbf{K}_{rr} - \mathbf{K}_{rc}\mathbf{K}_{cc}^{-1}\mathbf{K}_{cr} \tag{2.8}$$

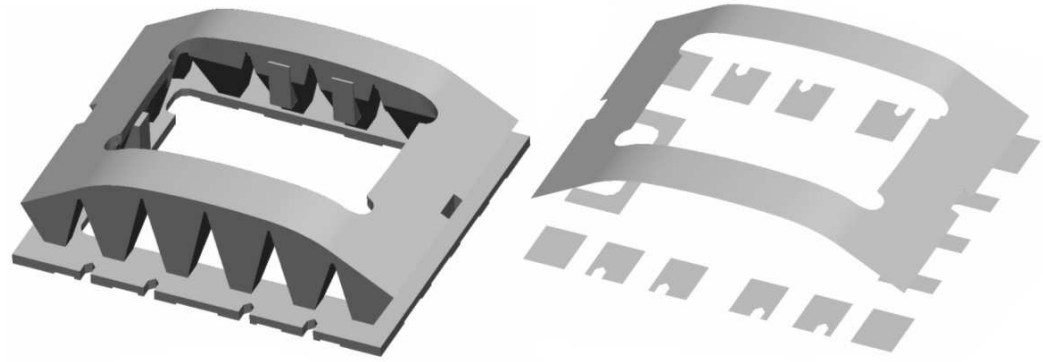


Figure 2.11: The blankholder mesh (left) and the retained regions (right)

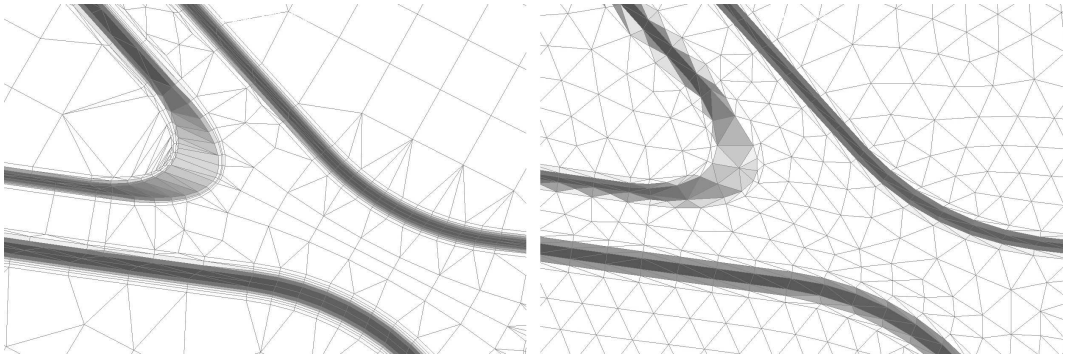


Figure 2.12: A typical rigid surface mesh (left) and a deformable solid mesh (right)

$\mathbf{K}'$ has a smaller dimension, and it should be more efficient to solve.

Example
For a blankholder of a roof panel deep drawing process by Daimler AG [11, 30], shown in Figure 2.11, static condensation reduced the amount of DOFs by 62%. A set of load-cases was carried out with and without static condensation in ABAQUS/standard, and the CPU time for solving the system *increased* by more than a factor of 10 for the statically reduced calculation. The reason for this is that the bandwidth of the condensed matrix $\mathbf{K}'$ has become much larger than the bandwidth of $\mathbf{K}$. As the bandwidth is a major factor in the cost of the solution of the problem, static condensation is useful only when the amount of retained DOFs is an order of magnitude lower than the initial amount of DOFs. In the case of elastic tool modeling, still 20 to 40% of the DOFs are retained because the entire finely meshed contact surface has to remain available. Therefore static condensation actually makes the calculation slower rather than faster.

The reason why so many nodes have to remain is the meshing of solid bodies. In regular deep drawing simulations, the rigid tool surfaces are meshed with a large amount of elements that vary heavily in size and shape. This is done to keep the mesh size minimal, while still capturing the fine geometrical details of the tool surfaces and to retain the surface smoothness. In fact, because the elements are not deforming they are also called 'segments' and there are no requirements to their shape at all. In contrast, when the tool is meshed as a deformable solid, the element

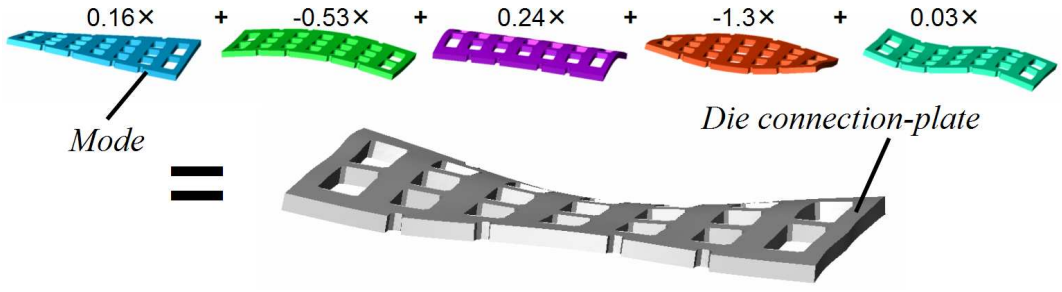


Figure 2.13: Principle of the DRB approach

shape has to meet more geometrical conditions. To obtain the same smoothness on the contact surface, the mesh has to be dense at that location, as Figure 2.12 shows, and all these DOFs are retained.

2.4 Deformable Rigid Bodies

Instead of modeling the tools and press as full FE models, they can also be modeled as Deformable Rigid Bodies (DRBs). The term Deformable Rigid Body, introduced by Bitzenbauer [10], seems self-contradictory but it actually describes the nature of these bodies quite well. The deformation of such a body under a certain load is calculated as a linear combination of a limited set of so-called *modes*, as shown in Figure 2.13. Therefore, the procedure is also referred to as *modal methods*. Especially when the deformation is global, a sufficiently accurate calculation can already be carried out with as little as $20 - 100$ modes. The cost of such a calculation is almost negligible compared to a regular FE calculation.

Bitzenbauer proposed to use the method in the context of crash testing, which requires correct modeling of the dynamic (transient) response of the car body structure. The deformation of deep drawing tools can be regarded as quasi-static, and therefore the previously derived *static* linear-elastic FE equation (2.5) was used instead.

2.4.1 The principle

For the sake of convenience this equation is repeated here:

$$\mathbf{Ku} = \mathbf{f} \tag{2.9}$$

Instead of solving the system (for example by using Gauss-elimination) to calculate $\mathbf{u}$, the displacement is calculated as a linear combination of so-called modes. In other words, the basis of the system is changed so that the solution becomes trivially simple. This is called spectral decomposition, or eigen-decomposition [7]. The stiffness matrix $\mathbf{K}$ is decomposed into two matrices $\mathbf{P}$ and $\mathbf{D}$:

$$\mathbf{K} = \mathbf{PDP}^{-1} \tag{2.10}$$

$\mathbf{D}$ is a diagonal matrix, containing the eigenvalues of the stiffness matrix

$$D_{ii} = \lambda_i \text{ (no sum)} \tag{2.11}$$

During the remainder of the chapter, the eigenvalues and their corresponding modes will be ordered from low to high, i.e. $\lambda_i \leq \lambda_{i+1}$. When $\mathbf{K}$ is positive definite, which is the case in a linear-elastic FE problem, the matrix $\mathbf{P}$ contains the eigenvectors or modes $\mathbf{v}_i$:

$$\mathbf{P} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n] \tag{2.12}$$

The eigenvectors are normalized so that $|\mathbf{v}_i| = 1$. In order to calculate the displacements $\mathbf{u}$ for a given load $\mathbf{f}$, the decomposed system of equations can be solved easily. Firstly, because each eigenvector is orthogonal to the others, the matrix $\mathbf{P}$ is an orthogonal too and $\mathbf{P}^{-1} = \mathbf{P}^T$. Secondly $\mathbf{D}$ is diagonal, and it can be inverted by inverting each diagonal entry. Therefore:

$$\mathbf{u} = \mathbf{K}^{-1}\mathbf{f} = \left(\mathbf{P}\mathbf{D}\mathbf{P}^T\right)^{-1}\mathbf{f} = \mathbf{P}^{-T}\mathbf{D}^{-1}\mathbf{P}^{-1}\mathbf{f} = \mathbf{P}\mathbf{D}^{-1}\mathbf{P}^T\mathbf{f} \tag{2.13}$$

The aforementioned procedure would be a very inefficient way of solving a linear system: Calculating all modes is costly and storing the resulting (full) $\mathbf{P}$ matrix would require an enormous amount of computer memory.

However, the advantage of spectral decomposition is that the displacement $\mathbf{u}$ can be approximated by taking into account only the lowest m eigenvalues and modes. This reduces matrix $\mathbf{D}$ to a m by m sized matrix $\tilde{\mathbf{D}}$ and the reduced $\mathbf{P}$ is now n by m-sized

$$\mathbf{u} \approx \tilde{\mathbf{P}}\tilde{\mathbf{D}}^{-1}\tilde{\mathbf{P}}^T\mathbf{f} \tag{2.14}$$

With an increasing amount of modes, the outcome will approach the exact solution. As will be shown later, a satisfactory approximation can already be achieved with an amount of modes that is several orders of magnitude lower than the amount of DOFs in the system.

2.4.2 Calculating modes

Matrices $\tilde{\mathbf{D}}$ and $\tilde{\mathbf{P}}$ can be constructed by calculating the first m eigenvalues and modes. For very small matrices these can be calculated analytically by solving the following problem:

$$\mathrm{Det}(\mathbf{K} - \lambda\mathbf{I}) = 0 \tag{2.15}$$

Matrix $\mathbf{I}$ is the unity matrix. This results in a polynomial of degree n, which is solved for λ, the roots are the eigenvalues λ_i. For each λ_i the corresponding eigenvector v_i can be found by solving

$$(\mathbf{K} - \lambda_i\mathbf{I})\mathbf{v}_i = \mathbf{0} \tag{2.16}$$

These equations become very impractical and analytically unsolvable for large systems, in these cases the eigenvalues and -vectors are calculated numerically. This is a highly developed area of mathematics and there are numerous algorithms available, each with different strengths, weaknesses and specific applications. Examples can be found in [26, 56]. For the eigenvalue calculations required for the DRBs, the commercial MATLAB software was applied, which uses the Implicitly Restarted Arnoldi Method (IRAM).

2.4.3 Approximation error analysis

When not all eigenvectors, or modes, are taken into consideration, the solution is only approximately correct. To evaluate the magnitude of the error, the elastic energy is the most convenient parameter. Equation (2.14) can also be written in the following form:

$$\mathbf{u} = [\mathbf{v}_1\lambda_1^{-1}, \mathbf{v}_2\lambda_2^{-1}, ..., \mathbf{v}_m\lambda_m^{-1}] \begin{bmatrix} \mathbf{v}_1^T\mathbf{f} \\ \mathbf{v}_2^T\mathbf{f} \\ \vdots \\ \mathbf{v}_m^T\mathbf{f} \end{bmatrix} = \sum_{i=1}^{m} \frac{\mathbf{v}_i^T\mathbf{f}}{\lambda_i}\mathbf{v}_i \tag{2.17}$$

This emphasizes the fact that the displacement is a linear combination of modes: Each mode introduces a displacement $\mathbf{u}_i$, which is the mode $\mathbf{v}_i$ multiplied by a scalar α_i:

$$\mathbf{u} = \sum_{i=1}^{m} \frac{\mathbf{v}_i^T\mathbf{f}}{\lambda_i}\mathbf{v}_i = \sum_{i=1}^{m} \alpha_i\mathbf{v}_i = \sum_{i=1}^{m} \mathbf{u}_i \tag{2.18}$$

The goal is to calculate the amount of elastic energy that is absorbed by this displacement $\mathbf{u}_i$

$$E_i = \frac{1}{2}\mathbf{u}_i^T\mathbf{K}\mathbf{u}_i \tag{2.19}$$

and to compare it to the total amount of elastic energy. The stiffness matrix $\mathbf{K}$ is now spectrally decomposed as in Equation (2.10), and for each mode the following is valid: $\mathbf{K}\mathbf{u}_i = \lambda_i\mathbf{u}_i$. Therefore

$$E_i = \frac{1}{2}\mathbf{u}_i^T\mathbf{u}_i\lambda_i \tag{2.20}$$

Filling in Equation 2.17:

$$E_i = \frac{1}{2}\mathbf{u}_i^T\mathbf{u}_i\lambda_i = \frac{1}{2}\left(\frac{\mathbf{v}_i^T\mathbf{f}}{\lambda_i}\right)^2 \mathbf{v}_i^T\mathbf{v}_i\lambda_i \tag{2.21}$$

And as $|\mathbf{v}_i| = 1$ it follows:

$$E_i = \frac{1}{2}\frac{(\mathbf{v}_i^T\mathbf{f})^2}{\lambda_i} \tag{2.22}$$

With this formula, it is possible to calculate the amount of energy that is absorbed by each particular mode, and to construct an *energy spectrum*. When a relatively uniform load is applied to the DRB, the amount of energy absorbed by the lowest modes will be several orders of magnitude larger than the higher modes. When a mode does not absorb a significant amount of energy, it can be omitted. On the other hand, depending on the loading condition, it is theoretically possible that an omitted mode absorbs most energy. To make an absolutely safe assessment, it would be necessary to know the total elastic energy of the exact displacement solution. In general, only a certain set of modes $\mathbf{v}_i$ is available, and this value cannot be obtained. However, it will be shown for an example project that analyzing the energy spectrum already provides enough insight in practical situations.

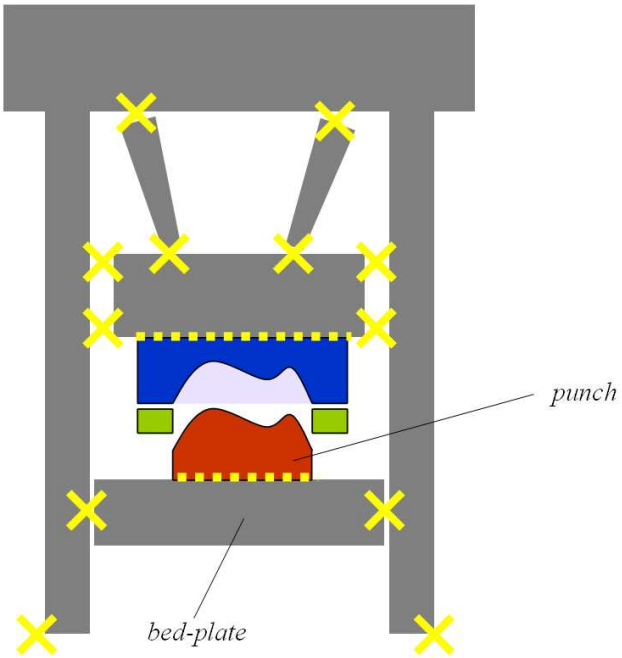


Figure 2.14: Regular boundary conditions and contact between press components and tools

2.4.4 Interactions between DRBs

In the specific case of press and tool deformation, the goal is to calculate the deformation of several components at the same time, for example the punch, which is bolted to the bed-plate. As pointed out in the introduction, the deformation of the tools cannot be treated individually. Different tool and press interactions are visualized schematically in Figure 2.14. The crosses indicate regular boundary conditions, connecting two bodies. The dotted lines indicate contact between two bodies.

The first step to model interactions between press components and tools in a DRB context is to include position (Dirichlet) boundary conditions on the deformation. It will be shown how the penalty method can be used to enforce boundary conditions on linear elastic FE problems and how these can be transferred to the DRB approach. There, two possibilities exist:

- The boundary conditions are embedded in the modes. They cannot be removed in consecutive deformation calculations. This is useful when the DRB is fixed to the outside world with boundary conditions
- The boundary conditions are added after spectral decomposition. New boundary conditions can be added and removed in each deformation calculation. This can be used in, for example, a contact algorithm where boundary conditions are constantly changed.

Fixed boundary conditions
For constrained FE problems, the challenge is to find a solution for the following problem: Find the deformation vector $\mathbf{u}$ for a given force $\mathbf{f}$, under the condition that

the difference in displacement between DOF α and β equals d.

$$\begin{cases} \mathbf{K}\mathbf{u} &= \mathbf{f} \\ u_\alpha - u_\beta &= d \end{cases} \tag{2.23}$$

The following analysis is a variation on the calculation in [33], pages 194-197. The reader is referred to this book for a more extensive explanation. The boundary condition function is rearranged into the following form:

$$\mathbf{l}^T\mathbf{u} = d \tag{2.24}$$

In this particular example the **l**-vector looks like this:

$$\mathbf{l} = [0, \ldots, 0, l_\alpha = -1, 0, \ldots, 0, l_\beta = 1, 0, \ldots, 0]^T \tag{2.25}$$

Then this function is added to the main equation and integrated to a potential I:

$$I(\mathbf{u}) = \frac{1}{2}\mathbf{u}^T\mathbf{K}\mathbf{u} - \mathbf{u}^T\mathbf{f} + \frac{k}{2}(\mathbf{l}^T\mathbf{u} - d)^2 \tag{2.26}$$

Here, k is the penalty-factor, a large scalar. The vector that minimizes this function is an approximate solution of the constrained problem [33]:

$$\begin{aligned} 0 = &\quad \tfrac{\mathrm{d}I}{\mathrm{d}\mathbf{u}} \\ 0 = &\quad \mathbf{K}\mathbf{u} - \mathbf{f} + k\mathbf{l}\mathbf{l}^T\mathbf{u} - kd\mathbf{l} \end{aligned} \tag{2.27}$$

with $\mathbf{M} = k\mathbf{l}\mathbf{l}^T$ and $\mathbf{f}_{\mathrm{bc}} = \mathbf{f} + kd\mathbf{l}$ this can be written as:

$$(\mathbf{K} + \mathbf{M})\mathbf{u} = \mathbf{f}_{\mathrm{bc}} \tag{2.28}$$

This new equation can be solved with spectral decomposition as well, by decomposing the matrix $\mathbf{K}_{\mathrm{bc}} = \mathbf{K} + \mathbf{M}$.

$$(\mathbf{K} + \mathbf{M})\mathbf{u} = \mathbf{K}_{\mathrm{bc}}\mathbf{u} = \mathbf{P}_{\mathrm{bc}}\mathbf{D}_{\mathrm{bc}}\mathbf{P}_{\mathrm{bc}}^T\mathbf{u} = \mathbf{f}_{\mathrm{bc}} \tag{2.29}$$

Flexible boundary conditions
With this procedure, fixed boundary conditions can be applied. In the following part, the boundary conditions will be implied *after* the spectral decomposition.

The decomposition of the stiffness matrix can be regarded as a change of basis for the equations. The boundary condition matrix can also be transformed to this new basis.

$$(\mathbf{P}\mathbf{D}\mathbf{P}^T + \mathbf{M})\mathbf{u} = \mathbf{f}_{\mathrm{bc}} \tag{2.30}$$

Now, the parameters $\mathbf{u}$ and $\mathbf{f}_{\mathrm{bc}}$ are transformed using the $\mathbf{P}$ matrix:

$$\mathbf{u} = \mathbf{P}\hat{\mathbf{u}} \tag{2.31}$$

$$\mathbf{f}_{\mathrm{bc}} = \mathbf{P}\hat{\mathbf{f}} \tag{2.32}$$

With this transformation, Equation (2.30) can be rewritten as

$$\mathbf{P}^T(\mathbf{P}\mathbf{D}\mathbf{P}^T + \mathbf{M})\mathbf{P}\hat{\mathbf{u}} = \hat{\mathbf{f}} \tag{2.33}$$

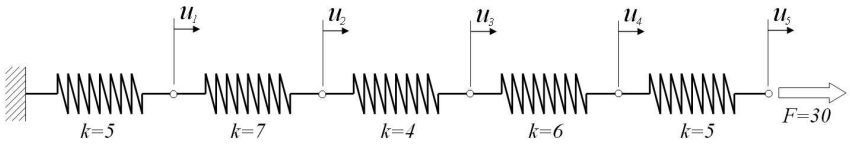


Figure 2.15: A simple FE model

Because $\mathbf{P}$ is orthogonal, $\mathbf{P}^T\mathbf{P} = \mathbf{I}$ so this can be rewritten as

$$(\mathbf{D} + \mathbf{P}^T\mathbf{M}\mathbf{P})\hat{\mathbf{u}} = \hat{\mathbf{f}} \tag{2.34}$$

When all modes would be taken into consideration the above system of equations is equally expensive to solve as the original system. However, when again only the first m modes and eigenvalues are calculated $\mathbf{D}$ is now reduced to a m by m matrix $\tilde{\mathbf{D}}$. $\mathbf{P}$ is a n by m matrix $\tilde{\mathbf{P}}$, so $\tilde{\mathbf{P}}^T\mathbf{M}\tilde{\mathbf{P}}$ also becomes m by m. Generally the number of calculated modes m is significantly lower than the number of DOFs n, typically around 100, therefore the solution of Equation (2.34) is very inexpensive.

When the DRB is constricted severely using this second approach, the simulant must make sure that the calculated modes are able to capture the deformation with sufficient accuracy. The previously introduced energy spectrum is recommended. When it is possible to include the boundary conditions in a fixed manner, it is always recommended to use the first approach and embed them in the modes.

2.4.5 Examples

To demonstrate the previously presented framework, a simple example problem is evaluated. The set of springs is shown in Figure 2.15. The stiffness matrix is:

$$\mathbf{K} = \begin{bmatrix} 12 & -7 & 0 & 0 & 0 \\ -7 & 11 & -4 & 0 & 0 \\ 0 & -4 & 10 & -6 & 0 \\ 0 & 0 & -6 & 11 & -5 \\ 0 & 0 & 0 & -5 & 5 \end{bmatrix} \tag{2.35}$$

The force vector is $\mathbf{F} = [0, 0, 0, 0, 30]^T$. The displacement vector was calculated by inverting $\mathbf{K}$. This results in $\mathbf{u} = [6.0, 10.29, 17.79, 22.79, 28.79]^T$.

Now, the $\mathbf{P}$ and $\mathbf{D}$ matrices are calculated by spectral decomposition:

$$\mathbf{D} = \begin{bmatrix} 0.43 & 0 & 0 & 0 & 0 \\ 0 & 3.58 & 0 & 0 & 0 \\ 0 & 0 & 8.20 & 0 & 0 \\ 0 & 0 & 0 & 16.70 & 0 \\ 0 & 0 & 0 & 0 & 20.10 \end{bmatrix} \tag{2.36}$$

$$\mathbf{P} = \begin{bmatrix} 0.18 & -0.52 & 0.44 & 0.49 & -0.52 \\ 0.30 & -0.62 & 0.24 & -0.33 & 0.60 \\ 0.47 & -0.25 & -0.60 & -0.39 & -0.46 \\ 0.55 & 0.15 & -0.34 & 0.65 & 0.37 \\ 0.60 & 0.52 & 0.53 & -0.28 & -0.12 \end{bmatrix} \tag{2.37}$$

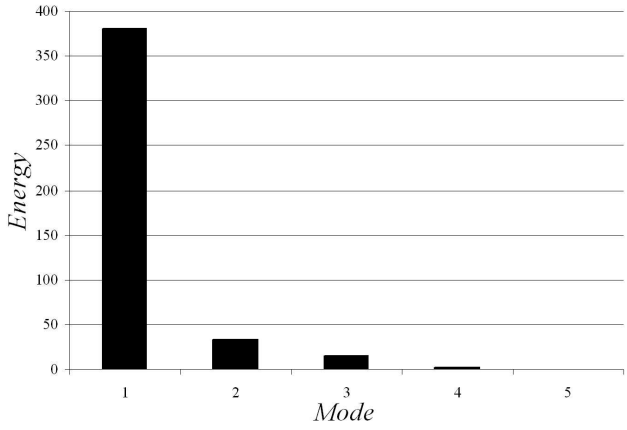


Figure 2.16: The energy spectrum for the spring example

For the approximated solution, the last eigenvalue of 20.10 and the accompanying mode in the last column of $\mathbf{P}$ will be omitted. The resulting displacement field is $\tilde{\mathbf{u}} = [5.90, 10.40, 17.70, 22.85, 28.76]^T$. The relative error in the vector norm amounts a negligible 0.4%. This was expected from the elastic energy assessment, which was introduced in Section 2.4.3. In Figure 2.16, the elastic energy for each mode is shown for this particular load vector. The first mode captures almost 90% of the total elastic energy already. It is not really necessary to know the total amount of energy: The fourth and fifth mode require less than one percent of the energy for the first mode, so they can be left out safely.

Now a boundary condition is applied to the original system:

$$u_4 - u_2 = 10 \tag{2.38}$$

When the penalty factor is 10^5, this results in a boundary condition matrix:

$$\mathbf{M} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 10^5 & 0 & -10^5 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & -10^5 & 0 & 10^5 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.39}$$

and a new force vector:

$$\mathbf{f}_{\text{bc}} = [0, -10^6, 0, 10^6, 30]^T \tag{2.40}$$

The result is a displacement vector $\mathbf{u}_{\text{bc}} = [6.0, 10.29, 16.29, 20.29, 26.29]$. Note that the boundary condition has been satisfied.

The final goal is to apply the boundary condition and leave out the highest mode at the same time using the flexible approach. Therefore the boundary condition matrix needs to be transformed:

$$\mathbf{P}^T\mathbf{M}\mathbf{P} = \begin{bmatrix} 6468 & 19502 & -14640 & 24937 \\ 19502 & 58805 & -44145 & 75192 \\ -14640 & -44145 & 33140 & -56447 \\ 24937 & 75192 & -56447 & 96145 \end{bmatrix} \tag{2.41}$$

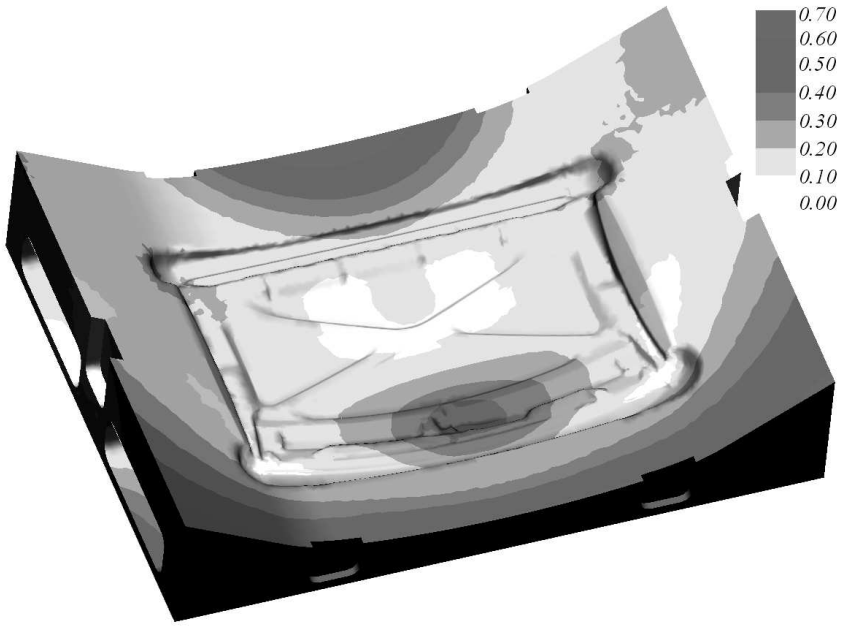


Figure 2.17: Normalized error (%) in displacement

The force vector is also transformed (Equation (2.32)), $\hat{\mathbf{f}} = [254342, 766863, -575664, 980530]^T$. Now equation 2.34 is solved resulting in $\hat{\mathbf{u}} = [38.69, 3.06, 2.35, -0.85]^T$.

Finally, the displacements can be transformed back and yield

$$\tilde{\mathbf{u}}_{\text{bc flexible}} = [5.94, 10.35, 16.25, 20.35, 26.30]^T \tag{2.42}$$

which means a relative error of 0.3% in the vector norm. By leaving out two modes the relative error still remains below 5%, and the results become marginally worse with only two modes left. The error amounts 12% then.

For larger systems, acceptable accuracy can be achieved using a very small fraction of the available modes. For example, a die from an industrial forming process [11] is loaded with the contact forces at the end of the forming stage, in this case derived from a PAM-STAMP simulation. The die is a solid mesh with 180.000 DOFs. In the DRB-approach only *10* modes were used. The error in the nodal displacement is shown as a percentage of the maximum displacement (0.4mm) in the contour plot of Figure 2.17. The accuracy of the DRB approach is so high, because the load is uniformly distributed, and the boundary conditions allow a global deformation. When the bottom of the die is fixed, to model a very stiff bed-plate, the deformations become very local and the approximation error will become much larger.

The interaction between two DRBs was also investigated. A simple contact algorithm was implemented [46], using the penalty method and the previously described flexible boundary condition scheme. In this case, a connection plate, which supports the die, is added to the system. Figure 2.18 shows the deformed bodies. In this DRB calculation, each body was modeled with 20 modes only.

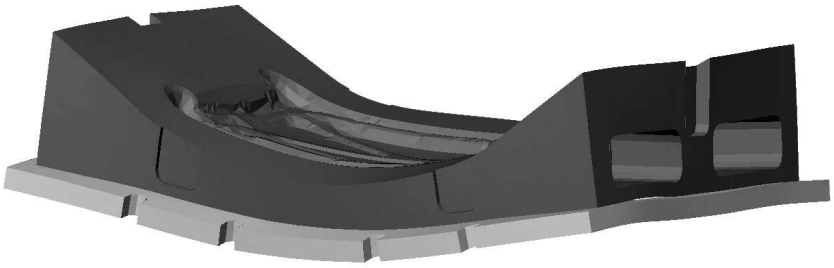


Figure 2.18: Deformation of the die and connection plate (exaggerated)

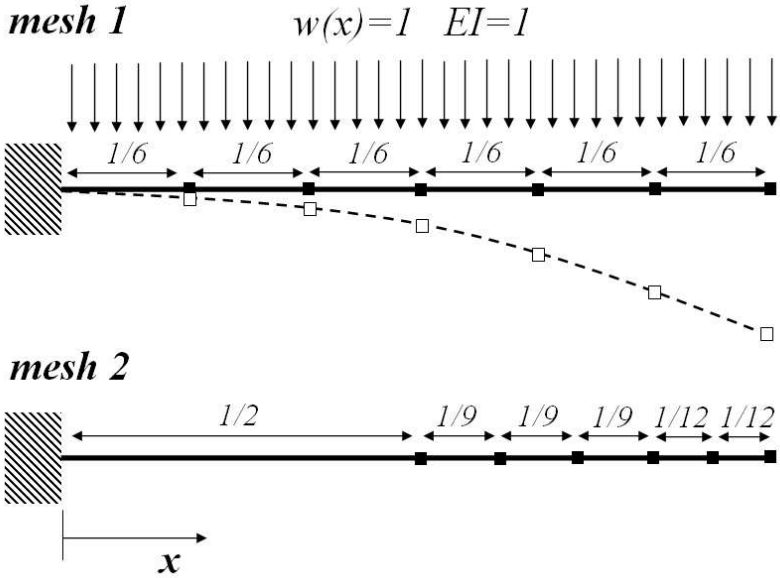


Figure 2.19: The Bernoulli beam example

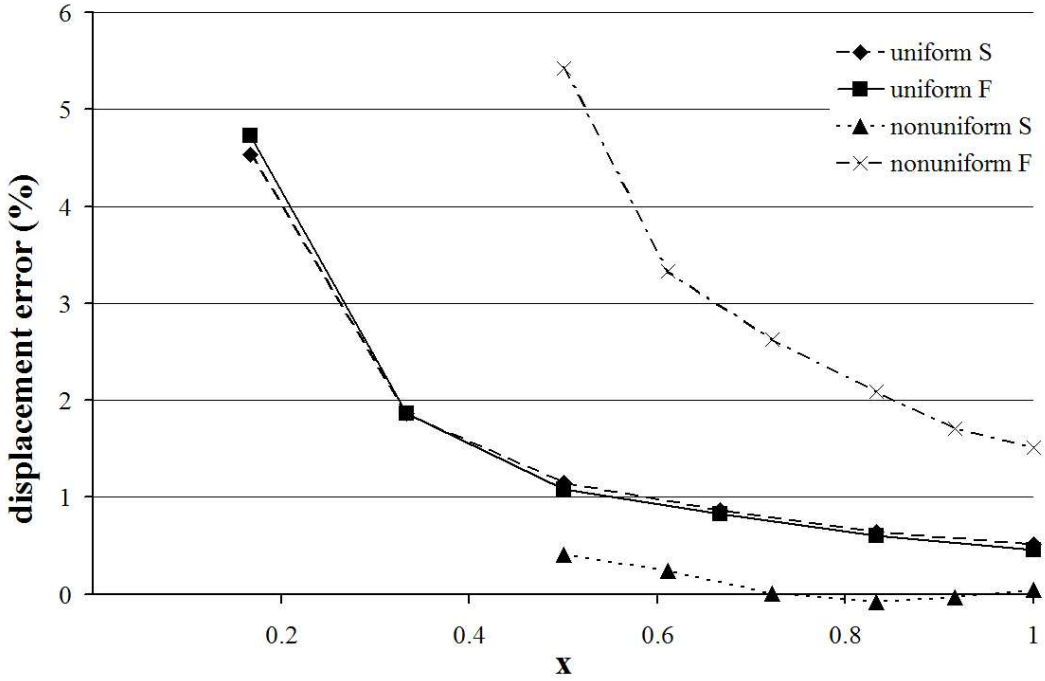


Figure 2.20: Displacement error in the Bernoulli beam displacements

2.4.6 Reducing the mesh-dependent error

Even when the results clearly demonstrate the potential of the DRB approach, the accuracy can be improved further. In the method presented above the modes and eigenvalues are directly dependent on the discretization. When the geometry is not meshed uniformly, the approximation error varies for different mesh-discretizations. It is most convenient to clarify the problem with an example problem: A beam with a length of 1 is clamped at $x = 0$ and bent downwards due to an uniformly distributed force with a value of 1. For reasons of simplicity the beam's bending stiffness EI is also taken as 1. The problem, shown in Figure 2.19 can be described with the differential equation by Bernoulli:

$$EI\frac{\mathrm{d}^4u(x)}{\mathrm{d}x^4} = w(x) \tag{2.43}$$

and the boundary conditions:

$$\left.\frac{\mathrm{d}^3u(x)}{\mathrm{d}x^3}\right|_{x=1} = 0 \quad \left.\frac{\mathrm{d}^2u(x)}{\mathrm{d}x^2}\right|_{x=1} = 0 \quad \left.\frac{\mathrm{d}u(x)}{\mathrm{d}x}\right|_{x=0} = 0 \quad u(x)|_{x=0} = 0 \tag{2.44}$$

Instead of finding the displacement $u(x)$ by solving the problem analytically, an FE approach was chosen, using Bernoulli beam-elements, see [33], p.48 and [40], p.301. Two meshes were investigated; a uniform mesh and a nonuniform mesh, both with 6 elements. Because each node has both displacement and rotational DOFs, there are 12 DOFs. Regardless of the topology of the mesh, the FE solution is exactly equal to the analytical solution at the nodes.

The solution is now approximated using the DRB-approach, using the lowest 4 modes of the 12 available. For the two meshes, the relative displacement error due to the modal approximation was calculated and visualized for each node in Figure 2.20 as

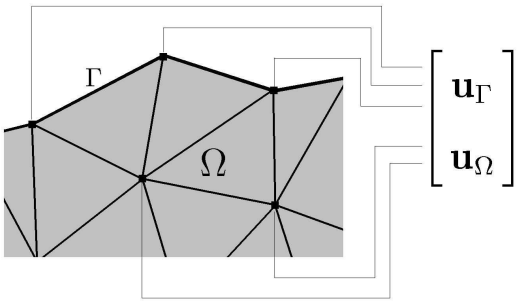


Figure 2.21: Nodes on Γ and Ω

the lines *uniform F* and *nonuniform F*. Note that the relative error is indefinite at the node of $x = 0$ because the displacement is zero there. It is clear that the error is significantly larger for the nonuniform mesh.

This problem will also be present in real tool/press deformation problems. Even when highly non-uniform meshes are unusual for solid geometries, a mesh dependent error is undesirable. The solution to this problem is to take the discretization into consideration during the modal approximation. In the following subsection, an improved method is presented, using the so called *S-modes*.

S-modes
In the previous DRB-formulation, the load force vector was linked to the displacement using the stiffness matrix. The resulting modes will be called Force-modes or *F-modes*. Instead, it is better to use a load *stress* vector in determining the modes, as the following procedure will show. Returning to the Galerkin form (Equation (2.5)): The traction load on the body is now also discretized using the shape functions $\mathbf{N}$

$$\tilde{\mathbf{t}}(x) = \mathbf{N}\mathbf{t} \tag{2.45}$$

$$\underbrace{\int_\Omega \mathbf{B}^T \underline{\mathbf{C}}\mathbf{B} \mathrm{d}\Omega}_{\mathbf{K}} \quad \mathbf{u} \quad = \underbrace{\int_{\Gamma_t} \mathbf{N}^T \mathbf{N} \mathrm{d}\Gamma}_{=\mathbf{S}} \quad \hat{\mathbf{t}} \tag{2.46}$$

The $\mathbf{S}$-matrix is quite similar to the mass matrix used in dynamic analyses, however, the integral is taken over the boundary only. $\mathbf{S}$ and in fact the entire Equation (2.46) is split up in two sections. The DOFs with subscript Γ are on the surface of the body, the remaining displacements are designated with an Ω.

$$\begin{bmatrix} \mathbf{K}_{\Gamma\Gamma} & \mathbf{K}_{\Gamma\Omega} \\ \mathbf{K}_{\Omega\Gamma} & \mathbf{K}_{\Omega\Omega} \end{bmatrix} \begin{bmatrix} \mathbf{u}_\Gamma \\ \mathbf{u}_\Omega \end{bmatrix} = \begin{bmatrix} \mathbf{S}_{\Gamma\Gamma} & \mathbf{S}_{\Gamma\Omega} \\ \mathbf{S}_{\Omega\Gamma} & \mathbf{S}_{\Omega\Omega} \end{bmatrix} \begin{bmatrix} \mathbf{t}_\Gamma \\ \mathbf{t}_\Omega \end{bmatrix} \tag{2.47}$$

It is clear that the value of the boundary integral of the $\mathbf{S}$-matrix:

$$\mathbf{S} = \int_{\Gamma_t} \mathbf{N}^T \mathbf{N} \mathrm{d}\Gamma \tag{2.48}$$

is nonzero at the outside nodes only. Therefore $\mathbf{S}_{\Gamma\Omega}$,$\mathbf{S}_{\Omega\Gamma}$ and $\mathbf{S}_{\Omega\Omega}$ are zero. A traction-load can only be present on the outside of the body, therefore $\mathbf{t}_\Omega = 0$.

Equation (2.47) can be split to calculate the displacements on the outside only in the following way:

$$\begin{aligned} \underbrace{\left(\mathbf{K}_{\Gamma\Gamma} - \mathbf{K}_{\Gamma\Omega}\mathbf{K}_{\Omega\Omega}^{-1}\mathbf{K}_{\Omega\Gamma}\right)}_{\overline{\mathbf{K}}} \quad \mathbf{u}_\Gamma &= \left(\mathbf{S}_{\Gamma\Gamma} - \mathbf{K}_{\Gamma\Omega}\mathbf{K}_{\Omega\Omega}^{-1}\mathbf{S}_{\Omega\Gamma}\right)\mathbf{t}_\Gamma \\ \overline{\mathbf{K}} \qquad\qquad\quad \mathbf{u}_\Gamma &= \mathbf{S}_{\Gamma\Gamma}\mathbf{t}_\Gamma \end{aligned} \tag{2.49}$$

Note that the above procedure resembles static reduction. In this case, the procedure is necessary because otherwise the system would be singular. This implies that the (dis-)advantages of static reduction apply too: The size of the system is reduced, however, solving it is more expensive due to the increased bandwidth of $\overline{\mathbf{K}}$. This is not a big problem since the modal calculations need to be carried out only once.

Equation (2.49) needs to be transformed to a single matrix equation so it can be approximately solved with the aforementioned modal procedure. When $\mathbf{S}_{\Gamma\Gamma}$ is (Cholevsky) decomposed

$$\mathbf{S}_{\Gamma\Gamma} = \mathbf{L}^T\mathbf{L} \tag{2.50}$$

Equation (2.49) can be rewritten as:

$$\begin{aligned} \overline{\mathbf{K}}\mathbf{u}_\Gamma &= \mathbf{L}^T\mathbf{L}\mathbf{t}_\Gamma \\ \mathbf{L}^{-T}\overline{\mathbf{K}}\mathbf{u}_\Gamma &= \mathbf{L}\mathbf{t}_\Gamma \\ \mathbf{L}^{-T}\overline{\mathbf{K}}\mathbf{L}^{-1}\mathbf{L}\mathbf{u}_\Gamma &= \mathbf{L}\mathbf{t}_\Gamma \end{aligned} \tag{2.51}$$

resulting in:

$$\overline{\overline{\mathbf{K}}}\overline{\mathbf{u}} = \overline{\mathbf{t}} \tag{2.52}$$

with the symmetric matrix $\overline{\overline{\mathbf{K}}} = \mathbf{L}^{-T}\overline{\mathbf{K}}\mathbf{L}^{-1}$, $\overline{\mathbf{u}} = \mathbf{L}\mathbf{u}_\Gamma$ and $\overline{\mathbf{t}} = \mathbf{L}\mathbf{t}_\Gamma$. Because this equation has the same structure as a regular linear-elastic FE equation (see Equation (2.9)), it can also be approximated in a modal way, as explained in Equations (2.10)-(2.14):

$$\bar{\mathbf{u}} = \overline{\overline{\mathbf{P}}}\,\overline{\overline{\mathbf{D}}}^{-1}\overline{\overline{\mathbf{P}}}^T\overline{\mathbf{t}} \tag{2.53}$$

The modes in $\overline{\overline{\mathbf{P}}}$ are now called *S-Modes*.

Verification with the Bernoulli beam problem
The Bernoulli beam problem was also solved using S-modes. Again, 4 out of 12 modes were used. Figure 2.20 shows the results in the lines *uniform S* and *nonuniform S*. In comparison with the F-modes the error has decreased significantly for the nonuniform mesh, even below the error of the uniform mesh.

For the uniform mesh, there is only a small difference. Theoretically, the results should be completely identical because the loading stress field will be just as uniform as the loading force field so there is no discretization difference to take into account. It is suggested that the small differences are due to the lumped calculation of the boundary surface matrix.

The fact that the error for the S-modes approach is even lower for the nonuniform mesh than the uniform mesh is an artefact of this particular model. The first mode coincidentally captures the exact deformation very well.

Again, the elastic energy that is absorbed per mode can be used as a check for the error of the approximation. The total elastic energy of a deformed body can be calculated in a way that is principally identical to the calculation for the F-modes in Equations (2.19)-(2.22):

$$E_i = \frac{1}{2} \frac{(\overline{\overline{\mathbf{v}}}_i^T \overline{\mathbf{t}})^2}{\overline{\overline{\lambda}}_i} \tag{2.54}$$

2.5 Including DRBs in the forming simulation

In the previous section, the S-modes were derived and it was shown how a deformation calculation can be carried out for a simple example. The goal of this section is to include the DRB-approach into the FE forming simulation. There are two ways to achieve this:

- Coupled
- Decoupled (staggered)

These methods are shown schematically in Figure 2.22. The best way is to include the tool deformation in the forming simulation code directly. The additional DOFs from the DRBs are added to the system matrix. The contact algorithm needs to be adapted for the two-sided-deformable situation. In order to benefit from the numerical efficiency of the DRBs, the contact boundary conditions need to be transformed to the modal basis, as demonstrated previously. The implementation of these features in a FE code for forming is complex, both in terms of algorithms and the data structure. However, the advantage is that, like in an implicit FE analysis, an equilibrium state is calculated exactly in each increment.

Since the deformation of the tools can be characterized as quasi-static, such a complex implementation is not necessary. From an implementation point of view, a more efficient method is to decouple the tool/press deformations from the simulation, as proposed in [37]: The simulation is halted at a certain point in time. The pressures, acting on the (rigid) tools, are exported and used as a load-inputfile for an external tool deformation calculation. The deformed tool geometries are then transferred back into the calculation and the simulation continues up to a next halting point. The procedure is repeated for each halting point until the product is finished.

A DRB-module was implemented in the DiekA FE code, developed at the University of Twente [3]. The mesh-independent S-modes approach was applied. The tool deformation calculation is carried out after each forming increment. The main features of the code, like the contact algorithm, were left unchanged. Due to the decoupling, the deformation lags the load by an increment, and instabilities rise in the tool deformation. For example, Figure 2.23 shows the growing oscillations in the multiplication factor α for the 3rd mode. These oscillations need to be reduced numerically. This is achieved by calculating the deformation as a weighted average over the last 5 steps.

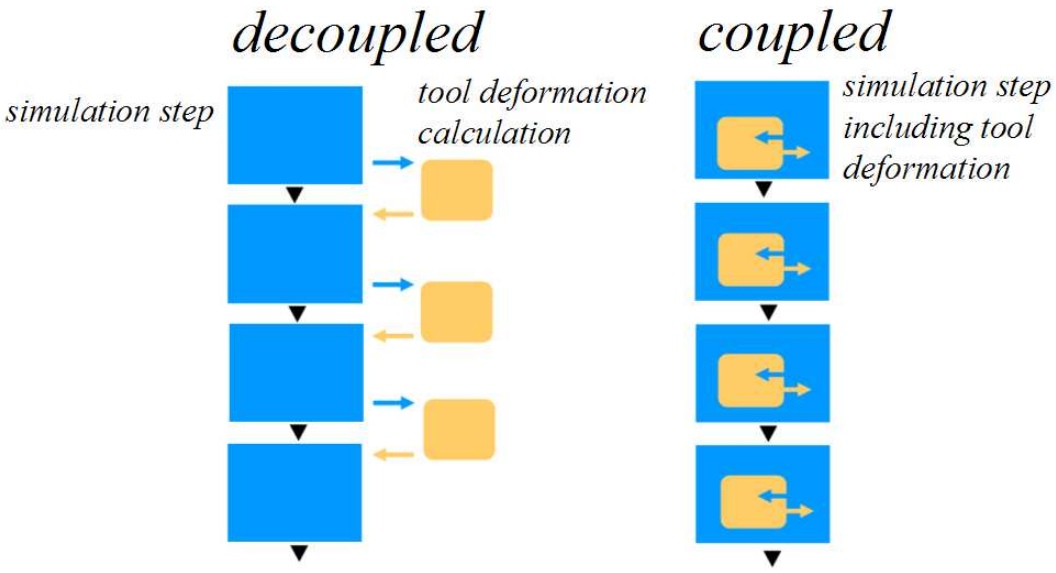


Figure 2.22: Two ways to incorporate tool deformations

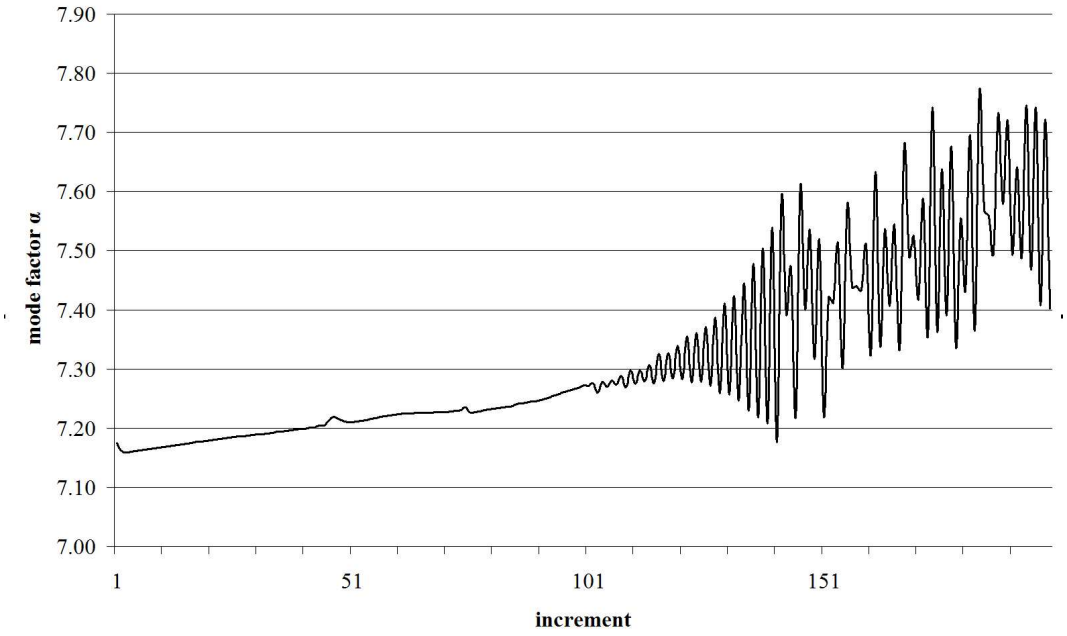


Figure 2.23: Instabilities occur during forming

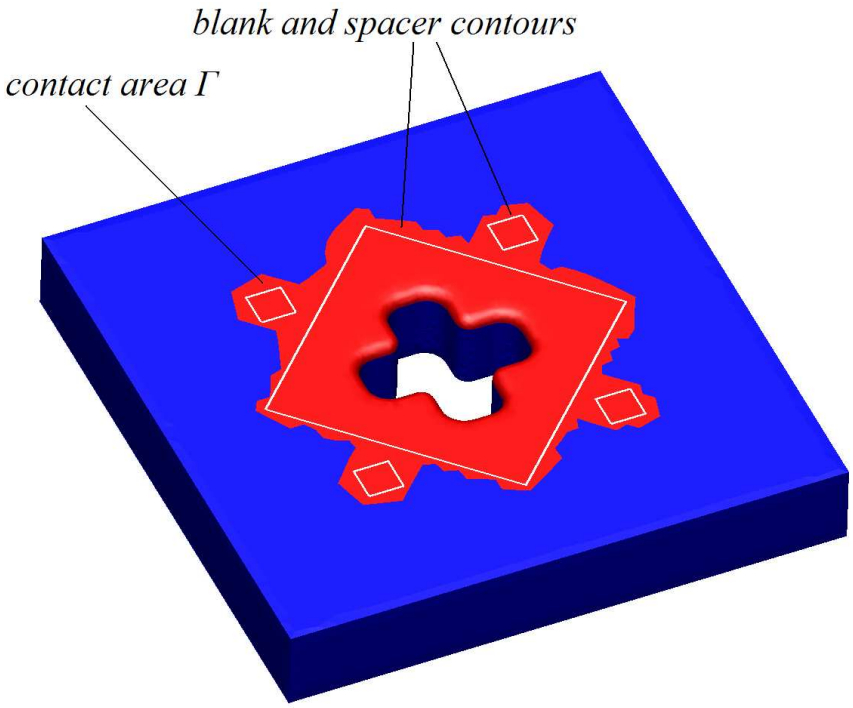


Figure 2.24: The contact area of the die used in the modal approach

2.6 Simulating the cross-die benchmark using DRBs

All three tools, blankholder, die and punch were modeled as DRBs. The supporting pins were modeled as fixed nodal boundary conditions on the die and blankholder. This is not entirely realistic, as in the experiment the tool might lift off a pin, but modeling the pins with contact conditions would make the analysis very complex.

Only the deformations of the contact area of the tool are included in the active set of DOFs $\mathbf{u}_\Gamma$, the other deformations are not calculated and set to zero in the visualization. This area is shown in Figure 2.24.
A set of 100 S-modes was calculated for each tool. In Figure 2.25 the first four modes of the die are shown. For clarity, a square contact area was used for this picture. The first two modes are tilting deformations, followed by a global bending mode and a saddle mode. As the eigenvalue increases, the shape of the mode becomes more complex.

The deformation of the die in the last increment of the forming simulation is investigated: The energy spectrum (Figure 2.26) shows that the third mode absorbs the largest amount of energy. The 8th mode also has a significant contribution, the other modes could be omitted already. This is easily explained: due to the relatively uniform load and the boundary conditions modeling the supporting pins, the deformation can be captured well by the third mode, which is a global bending mode.

Numerical cost
The DiekA simulation was run on a single-processor SPARC machine using Sun Solaris and required 6.5 hours for the rigid tools, and 7 for the deformable DRB-tools, an increase of only 8%. The preprocessing and eigenvalue-solving required an addi-

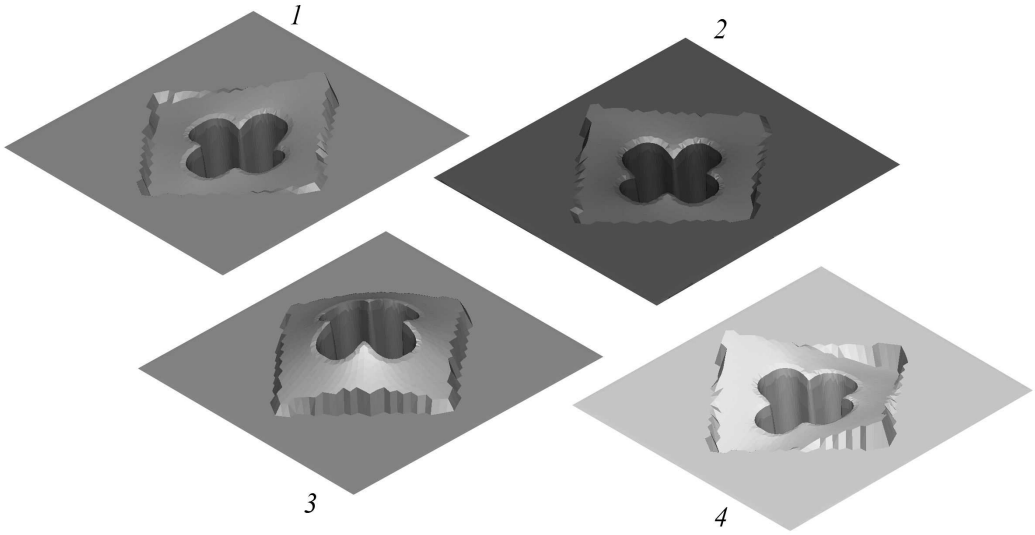


Figure 2.25: The first 4 modes of the deep drawing die

tional 30 minutes for each tool. In comparison, the reference ABAQUS calculations were carried out on a 2-Processor HP-UX system, and the simulation required 24 CPU-hours for the rigid tools and 72 hours for the deformable tools, an increase of 200%.

Accuracy of the DRB approach
Such a time-saving is only useful if the reliability of the results is retained. In Figure 2.27 the pressure distributions are compared between an ABAQUS reference calculation with and without deformable tools [46] and the DiekA calculations with rigid and DRB-tools. It is clear that there are quantitative differences, but the DRB-approach shows the same differences to the calculation with rigid tools. At *(a)*, the spacers partially overtake the blankholder load and at *(c)* the blank corners also experience a larger pressure. In exchange, the high-pressure spot at *(b)* decreases in size, allowing a larger blank draw-in.

The quantitative differences between ABAQUS and DiekA simulations cannot be overseen. However, they already occur when rigid tools are used in both codes. There are always implementation differences in different FE-codes, especially when they, like DiekA, are optimized specifically for forming. However, in this case, the main cause is the use of solid-shell elements in ABAQUS. These elements are required when the thickness change is to be taken into consideration. This is highly desirable for any forming simulation and essential for this particular process. For unknown numerical reasons, the solid-shell elements (SC8R) show a considerably smaller thickness change than the regular shell elements, as Figure 2.28 shows. Therefore the high pressure spots are too small, as experiments reveal that the simulation by DiekA is closer to reality [46].

2.7 Conclusion and outlook

Even when tool and press deformations are small, the influence on the blank draw-in and consecutively many other quality issues, such as the risk of rupture, wrinkling

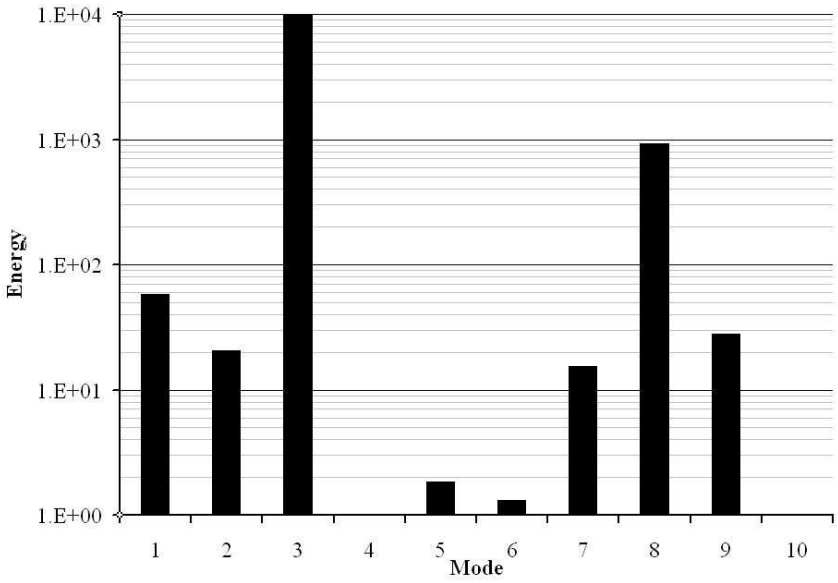


Figure 2.26: Energy spectrum of the first 10 modes

and springback, is clear. Deformable Rigid Bodies provide a possibility to take these deformations into account during the forming simulation. Because the added numerical cost (8% in the case shown) is very small there is no reason to *not* take the tool deformation into consideration. This confirms hypothesis 1b, *Press and tool elasticity can be included in the forming simulation at an acceptable cost.*

In the cross-die example, only the tools need to be modeled elastically. However, in industrial forming processes the press plays an important role too. Due to the efficiency of the DRB approach, the entire press/tool structure could be taken into account. This would enable the process engineer to do a true virtual factory test of the forming process and reduce the amount of tool reworking.

Despite promising results, the existing planning process may not be ready for FE simulations with deformable tools yet. The reason for this is that the design of the tools might not be completed when the forming simulation is carried out. The planning procedure could be changed, but then the advantage of deformable tool models should be proved in much more detail, with experiments as well as simulations.

However, the method will be of great use when a flexible blankholder is used. The DRB approach will be able to show the changes in the blankholder pressure distribution as a result of nonuniform loading. This can already be made clear in the blankholder closing phase, before forming is started. It is easier to verify the results in this context, as a blankholder loading simulation is less complex than a full forming simulation, and the pressure distribution can be measured experimentally, using pressure foil under the blank. Further research in this direction is highly recommended, because here the approach includes a clear practical use and a clear verification procedure, and the process planning does not need to be changed.

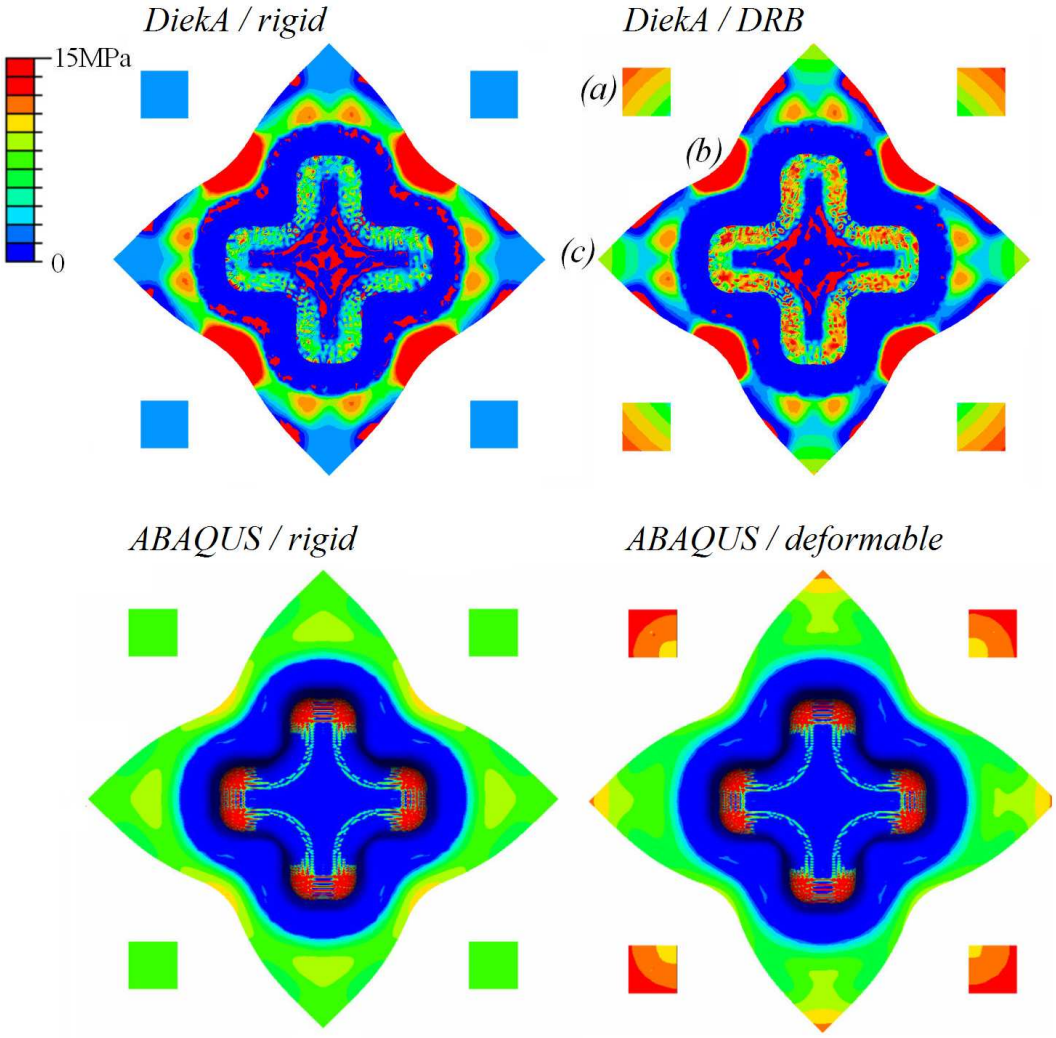


Figure 2.27: Contact pressure on the blank at the end of the forming process for ABAQUS and DiekA simulations

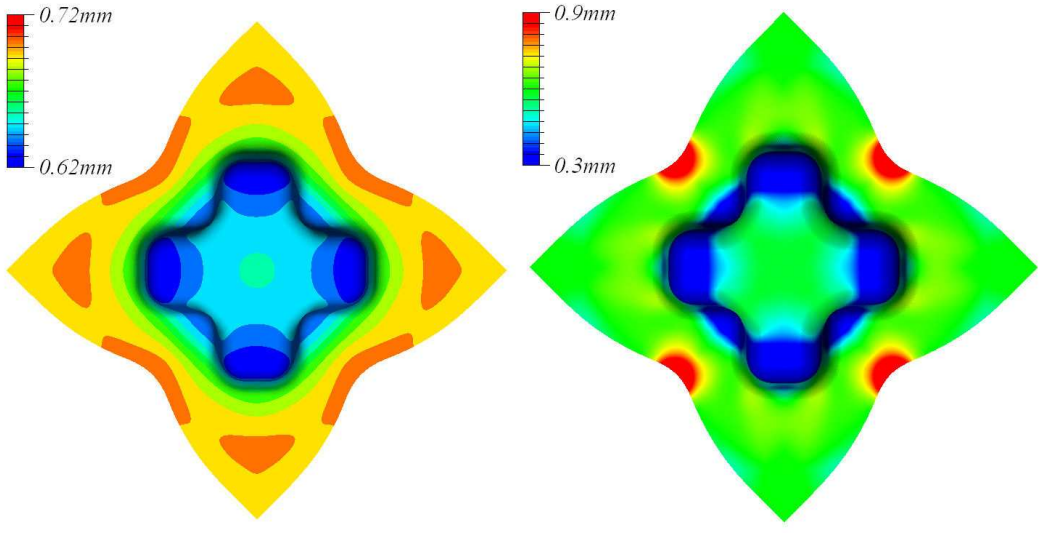


Figure 2.28: Blank thickness, ABAQUS solid-shells (left) and regular shells (right)

Glossary

α mode factor
Γ boundary surface
$\boldsymbol{\varepsilon}$ strain
λ eigenvalue
Π potential
$\boldsymbol{\sigma}$ stress
Ω body
$\mathbf{B}$ B-matrix (FE)
$\underline{\mathbf{C}}$ material behavior matrix
d distance in boundary condition
$\mathbf{D}$ eigenvalue matrix
$\boldsymbol{\varepsilon}$ strain
E elastic deformation energy
EI bending stiffness (beam theory)
$\mathbf{f}$ load vector
$\mathbf{I}$ unity matrix
k penalty factor
$\mathbf{K}$ stiffness matrix
$\mathbf{l}$ boundary condition vector
$\mathbf{L}$ decomposed boundary surface matrix
$\mathbf{M}$ boundary condition matrix
$\mathbf{n}$ surface normal
$\mathbf{N}$ shape functions matrix
$\mathbf{P}$ mode matrix
$\mathbf{S}$ boundary surface matrix
$\tilde{\mathbf{t}}$ traction load
$\mathbf{t}$ discretized traction load
$\tilde{\mathbf{u}}$ displacement field
$\mathbf{u}$ discretized displacements
$u(x)$ deflection of beam (beam theory)
$\mathbf{v}$ eigenvector
$w(x)$ distributed load (beam theory)
x 1-dimensional coordinate
$\mathbf{x}$ coordinate in cartesian space

Chapter 3

Computer-aided Springback Compensation

3.1 Introduction

Springback is the deformation of the blank that occurs when the forming tools are opened. In Figure 3.1 the numerically predicted springback of a front fender is shown. As the figure shows, the deformation can be considerable, leading to problems in the assembly process of the car-body. In order to produce parts with the correct shape, the forming tools must be compensated. As pointed out in the introductory chapter, springback is hard to predict and as a result the compensation requires a lot of trial-and-error, which is time-consuming and costly. Industrial sources state that springback-related issues cost the American car industry over $50 million per year [70].

In scientific publications, springback is in many cases analyzed using two-dimensional geometries, for example the bending of a bar [68], or the forming of a hat-profile, both experimentally [80, 15] as well as in simulations [55]. The influence of in-plane stretching is generally neglected in the analysis. Springback is then mainly characterized by *unbending* due to internal moments, the in-plane stresses are neglected. An extensive overview of these publications can be found in [70, 54], including both experimental and numerical results.

Industrial products, such as the front fender of Figure 3.1, have complex three-dimensional shapes. The geometry of the panel is generally double-curved and non-developable, and due to the (repeated) bending-unbending and in-plane stretching of the material the stress distribution becomes more complex. This may cause the panel to twist, rather than just unbend. As an additional complexity, high-strength steels and aluminium used in today's car-body parts, show large springback. Heavy springback and twisting are also likely to occur when a trimming stage is present in the process, because the large internal stresses that are present in the relatively stiff cup-shaped drawing product are released to the more flexible cut-out blank.

These phenomena make springback hard to handle for the die engineer, as well as the simulation code. The current procedure in solving springback problems will be discussed in Section 3.2. Section 3.3 explains the principles and problems of the two best known springback compensation algorithms: Displacement Adjustment (DA)

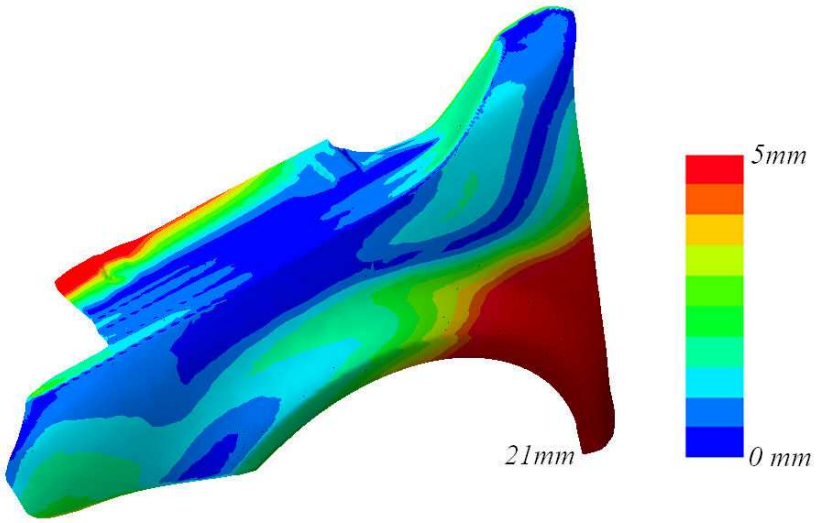


Figure 3.1: Springback (normalized) on a front fender

and Springforward (SF). Rather than using FE results, the research is based on an analytical model for a stretch-bending process. This model also suffers from the limitations of a 2D model, but it does show the effect of in-plane stretching, an important aspect of the physics of the deep drawing process. This provides a clear view on the principles and problems of each algorithm. The goal, however, remains the compensation of springback for industrial products. The tools have to comply with many practical considerations that provide new challenges. These are the focus of Section 3.4, and the resulting algorithm is demonstrated for different processes in Section 3.5.

3.2 Handling springback in industry

3.2.1 Springback measurement and assessment

Due to its unpredictability, springback is rarely compensated beforehand in the the CAD design and FE testing phases. Instead, the key point is to *avoid* springback as much as possible by introducing large plastic strains in the product (as will be explained in Section 3.3). Springback compensation should only be applied to resolve geometrical errors as a last resort. The question whether compensation is required can be answered by assessing the following three points

- Measurement of the geometrical error (see Figure 3.2)
- Measurement of the restraining forces
- Functional analysis

Careful analysis is required because of the flexibility of most sheet metal parts. Depending on the position of the product during measurement and the way the product is held in position, deformations already occur due to gravity and the fixation clamps. Even handling procedures may influence the springback assessment [20].

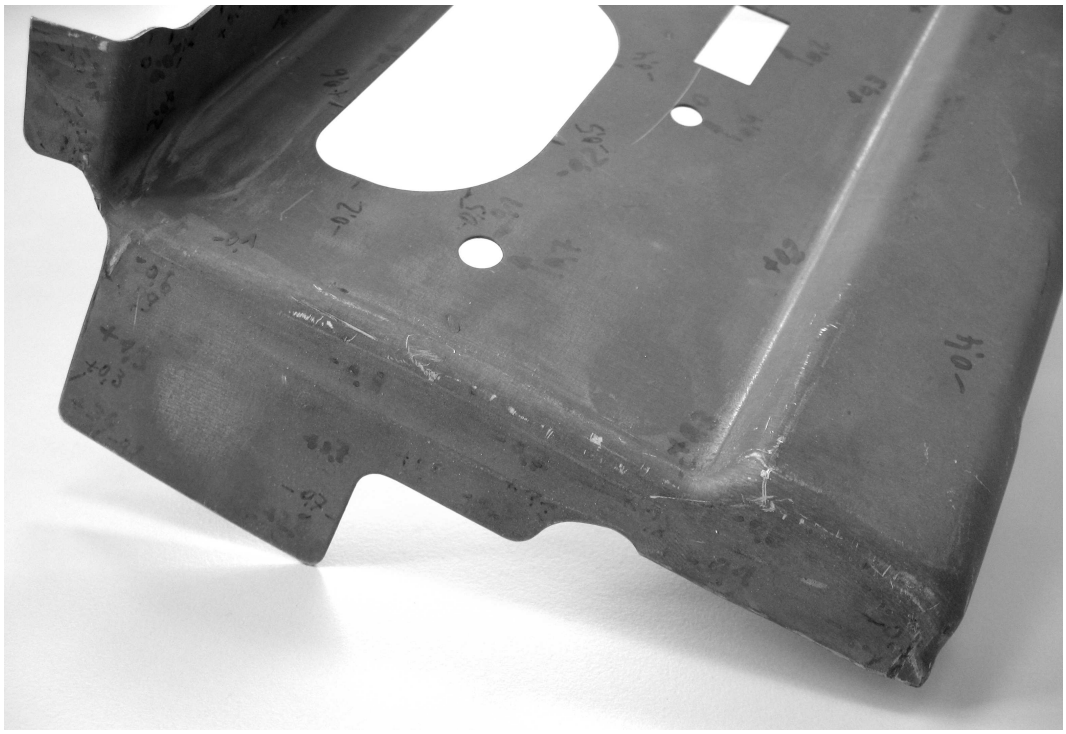


Figure 3.2: Springback measurement on a structural part (Daimler AG)

When the product is used in a larger assembly, two situations might occur. Consider an outer car body part consisting of an inner panel and an outer panel, for example the front fender. When the inner panel has a major function in the stiffness or crashworthiness of the car body it is generally made of thicker sheet material. The geometry of this part should be exactly right, and correct springback compensation is essential. The outer panel is generally more flexible. So, even when the geometrical error is high, the product can be pushed back into the correct shape before joining, and a satisfactory assembly is obtained. Compensation is not required in this case. However, to make sure the assembly is not distorted by the push-back forces, these need to be measured. As a rule of thumb, 30N is the maximum admissible force for fixing body panels during welding [1].

In other cases, both the inner and outer panel are flexible and the final assembly obtains satisfactory stiffness only after the parts have been joined. In this case, there are also possibilities to avoid springback compensation, for example by optimizing the joining fixation: the parts are slightly deformed by the joining fixation clamps and joined under strain. After release, a satisfactory geometry is obtained. This is, however, a very sensitive procedure and when errors occur, large planning problems will emerge. The outer body panel generally has to meet very tight geometrical tolerances (see 4) and remaining internal stresses might distort the outer surface appearance to an undesirable extent.

Material and process variability
An important aspect to consider when or when not to apply compensation is the variability of the material. Due to variations in the batches of sheet metal, considerable differences may occur in the product's springback. For example, according

to DIN norm 10079 the thickness of the sheet material has a tolerance margin of 10%, and mechanical properties such as the yield-stress are mainly regarded as lower bounds, so the actual value can vary considerably. At the moment, manual springback compensation is based on products made on a prototype press in very small series. There, rather large deviations occur in the final part geometry, making the compensation less reliable, even when it is based on real measurements. This shows that the springback challenge is greater than the numerical aspect alone. As a side-note, this issue is attracting more and more attention from an FE simulation perspective [81, 12].

3.2.2 From manual to numerical springback compensation

In almost all cases, the measurements on the prototype parts indicate a remaining springback issue, either a geometrical error or large push-back forces, and compensation is required. The most simple form of springback compensation is applied in a (tube) bending process. Generally, the product needs to be bent further for it to spring back into the desired shape. Because the tube has been bent further, springback has also increased slightly. So, in the case of the tube, the amount of compensation is larger than the amount of springback. A correction factor, the so-called *springback compensation factor a*, is applied. This factor is well-documented in tables for bending processes [**?**].

The same principle can be applied to more complex geometries. The compensation is carried out by CAD engineers of the tool design department. Based on 3D geometrical measurements, the surfaces are displaced in the direction opposite to the geometrical error. For this type of compensation, the compensation factor cannot be predicted and documented well. The problem is even more complex: To obtain maximum accuracy, the compensation needs to be carried out differently at different spots on the product. Consequently, a large number of experiments and a lot of experience are required to find a satisfactory tool shape.

Chu et al. [20] were the first to replace experimental springback measurements with FE predictions. This brings considerable time-savings for the process planning, because computer simulations require much less time than tool reworking followed by a forming experiment. However, the compensation was still carried out manually by Chu et al. The objective of this chapter is to increase the efficiency and accuracy of numerical springback compensation by developing a dedicated compensation algorithm.

When FE springback prediction is used as a basis for compensation, the compensation can only be as accurate as the simulation, and as reliable as the process itself. The topic of this chapter and thesis is on the compensation of springback, and *not* specifically the springback prediction itself. In any case, it must be pointed out that efficiency increases even when the FE springback prediction is only moderately accurate: The remaining problems that need to be solved on the real press are reduced significantly.

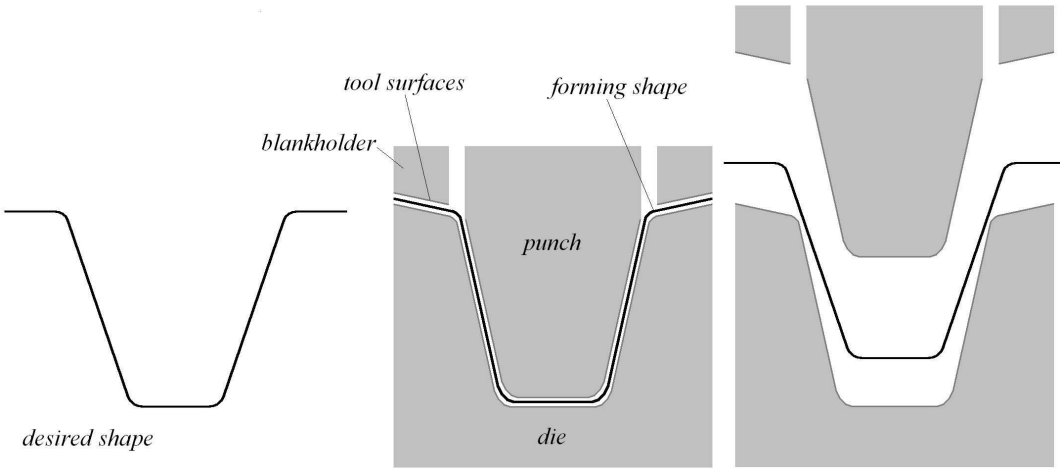


Figure 3.3: The forming shape

3.3 Compensation algorithm principles

As explained in the previous section, there is no clearly defined procedure to find the correct tool geometry, either in the case of measurement-based or numerically-based springback compensation. A straightforward approach is to define a set of geometrical parameters, for example using the part's CAD description, and to optimize these parameters to minimize the shape deviation between the sprung-back geometry and the desired geometry. This was carried out in [25] for a simple u-profile process. Industrial products feature complex shapes, and their CAD geometries are defined by thousands of parameters. It is unfeasible to take all these parameters into consideration in an optimization procedure.

Therefore, two specific springback compensation algorithms have been published: Displacement Adjustment, or DA, and Spring Forward (SF). DA, developed by Gan and Wagoner [24, 23, 71] is based on the geometrical principle that was explained with the tube bending process in the previous section. SF was developed by Karafillis and Boyce [36, 35, 34]. The SF method has a more physical approach, based on reversing the internal stresses that cause springback.

To avoid many die design-related issues, the goal of the algorithms presented in this section is to find the optimal *forming shape*. This is the shape of the *product* when the tools are still closed (see Figure 3.3). The shape of the forming tools needs to be derived from this forming shape. This will be the focus of Section 3.4, where the compensation method will be expanded to work for industrial forming processes as well.

3.3.1 Displacement Adjustment

DA springback compensation is based on the previously introduced principle of *overbending*. $\vec{\mathbf{d}}$, the desired geometry and $\vec{\mathbf{s}}$, the springback geometry are topologically identical sets of points. Topologically identical means that the geometry is mathematically built-up in the same way. Another way to formulate this requirement is to require that there exists a one-to-one mapping from each point $\vec{s}_i$ on geometry $\vec{\mathbf{s}}$

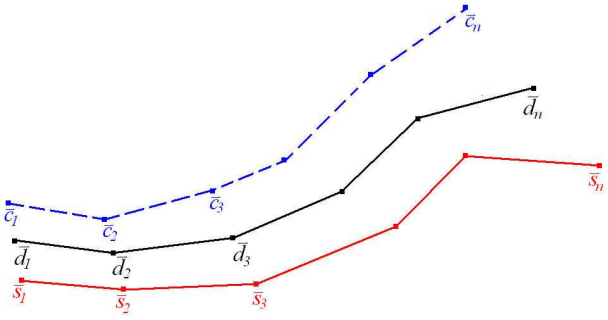


Figure 3.4: Principle of the DA method

to another point $\vec{d_i}$ on $\vec{\mathbf{d}}$.

$$\vec{\mathbf{s}} = [\vec{s_1}, \vec{s_2}, \vec{s_3}, ..., \vec{s_n}] \tag{3.1}$$

$$\vec{\mathbf{d}} = [\vec{d_1}, \vec{d_2}, \vec{d_3}, ..., \vec{d_n}] \tag{3.2}$$

When FE meshes are used, these points are represented by the nodes. The compensated forming shape $\vec{\mathbf{c}}$ is also topologically identical to $\vec{\mathbf{d}}$ and $\vec{\mathbf{s}}$.

$$\vec{\mathbf{c}} = [\vec{c_1}, \vec{c_2}, \vec{c_3}, ..., \vec{c_n}] \tag{3.3}$$

It is found in the following way:

$$\vec{\mathbf{c}} = \vec{\mathbf{d}} - 1 \cdot a(\vec{\mathbf{s}} - \vec{\mathbf{d}}) \tag{3.4}$$

When compensation factor a has a value of one, this equation means that the shape deviation due to springback is reversed and added to the desired geometry. The assumption is that a product that is formed to the compensated forming geometry $\vec{\mathbf{c}}$ will spring back to the desired geometry $\vec{\mathbf{d}}$. Analogously to the tube-bending problem, this assumption is never fulfilled exactly due to nonlinearities in the deep drawing process: after compensation the forming process and the resulting springback will be slightly different. Two solutions are suggested:

- To find an optimal value for the compensation factor
- To use an iterative procedure to find the optimal forming shape $\vec{\mathbf{c}}$

The DA method was initially proposed as an iterative procedure in [24, 23, 71]. The initial forming shape $\vec{\mathbf{c}}^{j=0}$ is generally the desired shape. The forming shape in the $(j+1)$-th iteration is found with the following recursive equation:

$$\vec{\mathbf{c}}^{j+1} = \vec{\mathbf{c}}^{j} - 1 \cdot (\vec{\mathbf{s}}^{j} - \vec{\mathbf{d}}) \tag{3.5}$$

Note that each point in $\vec{\mathbf{c}}$ is adjusted independently so that it is theoretically possible to achieve a fully optimal forming shape.

Distance field
Instead of using the shape deviation field:

$$\vec{\boldsymbol{\xi}} = \vec{\mathbf{s}} - \vec{\mathbf{d}} \Leftrightarrow \vec{\xi_i} = \vec{s_i} - \vec{d_i} \tag{3.6}$$

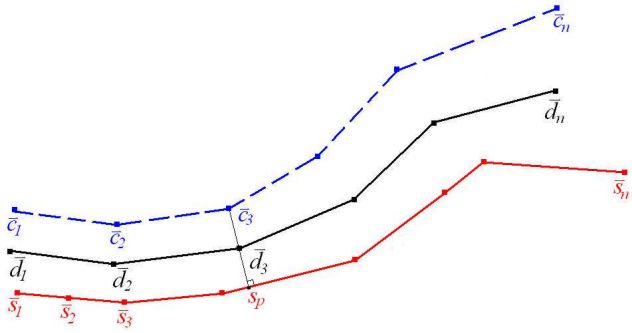


Figure 3.5: Principle of the distance based DA method

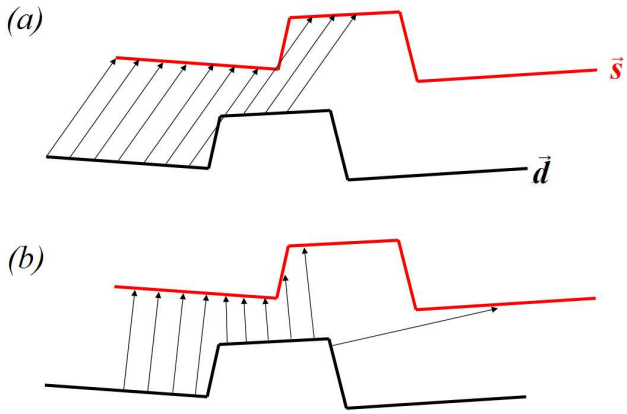


Figure 3.6: Displacement (a) vs. distance (b) field

a distance field could also be used between $\vec{\mathbf{s}}$ and $\vec{\mathbf{d}}$. For example, the normal distance field is visualized in Figure 3.5.

This approach is sometimes the only available method, for example when the compensation is based on measurements instead of FE springback predictions, or when, as in Figure 3.5, $\vec{\mathbf{d}}$ and $\vec{\mathbf{s}}$ are not topologically identical. However, the use of a displacement field is generally preferred because especially when springback is large, the displacement field will be smoother and provide more information. This is demonstrated for a very simple example in Figure 3.6. The arrows in (a) indicate the springback displacement field, the arrows in (b) the distance field.

3.3.2 Spring Forward

Instead of compensating springback based on a geometric displacement field, the springforward method uses the *cause* of springback, the internal stresses. The procedure is presented in the flowchart of Figure 3.7, taken from [36].

As presented in the papers, the algorithm is suitable only for 2-dimensional geometries that is modeled in FE using beam elements. The main idea is to capture the internal moments in the blank after forming. When the tools are released, these moments cause the blank to spring back. In the compensation procedure, these unbending moments are reversed and applied to a virgin blank in the desired geometry.

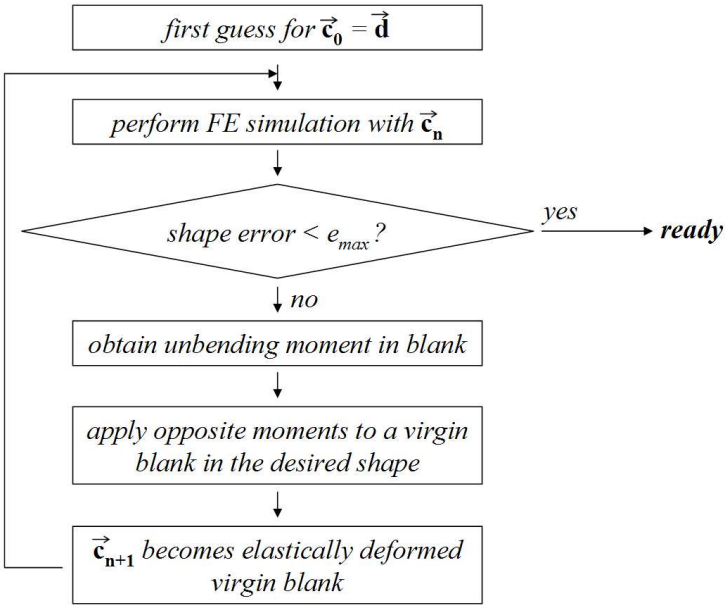


Figure 3.7: Springforward algorithm flowchart [36]

This virgin blank consists of the same material, but it does not have any internal stresses. Due to these moments, the virgin blank will 'spring forward' in the opposite direction, delivering the compensated forming shape $\vec{c}$. This is demonstrated schematically in Figure 3.8.

Similarly to the DA method, the SF method was presented as an iterative procedure. Therefore, in a consecutive iteration with the new forming shape $\vec{c}^1$, the internal moments are captured again. These are applied to the virgin blank to produce forming shape $\vec{c}^2$. In this way, the method is repeated until a stable solution is achieved. However, Wagoner points out that, compared to the (iterative) DA algorithm, the SF algorithm "converges more slowly, if at all, or may converge to incorrect die shapes" [71].

The problem is that, unlike the DA procedure, there is no fixed geometrical target for the optimization process. In fact, there is no reason for the blank to converge further to the desired shape after the first iteration. This can already be demonstrated using the elasto-plastic stretching of a bar as a model for the deep drawing process. This is a one-dimensional problem, and instead of the unbending moments, the stretching force at the end of the forming stage is used in the SF procedure.

So, in the first iteration the product is stretched from a length of 1 towards its target length 1.025. The tension stress is calculated using the stress-strain curve of Figure 3.9, and then the bar is unloaded. The tension stress is reversed and applied to a virgin bar with the desired length to produce the first compensated forming shape. Now the bar is stretched to the length of the compensated forming shape, again, the required tension stress is calculated. This way 6 iterations are carried out, and the results are listed in table 3.1
The process converges, but a shape error remains. The DA method was also applied in the bar-stretching example and converges to the correct shape very rapidly, as

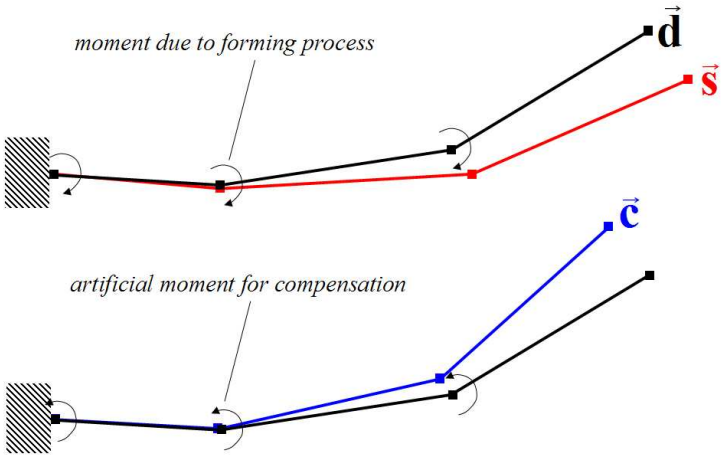


Figure 3.8: The Springforward algorithm principle

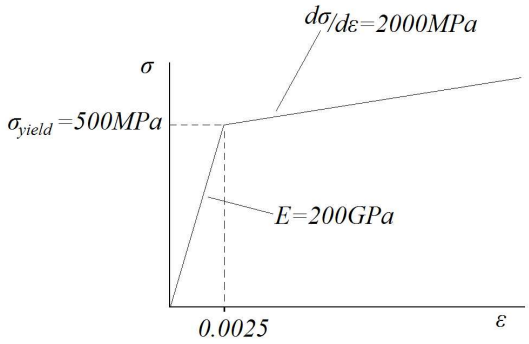


Figure 3.9: Material model for the bar stretching model

j	l_j	σ_j(MPa)	l_{sb}	shape error(%)
0	1.025	545	1.022275	100
1	1.0277931	550.58625	1.025040194	1.475
2	1.0278218	550.6435091	1.025068537	2.51511875
3	1.027822	550.644096	1.022275	2.525779967
4	1.0278221	550.644102	1.022275	2.525889245
5	1.0278221	550.644102	1.022275	2.525890365

Table 3.1: Results of the SF method for the bar-stretching model

Table 3.2 shows.

j	l_j	σ_j (MPa)	l_{sb}	shape error(%)
0	1.025	545	1.022275	100
1	1.027725	550.45	1.02497275	1.0
2	1.0277523	550.5045	1.024999728	0.01
3	1.0277525	550.505045	1.024999997	0.0001
4	1.0277525	550.5050505	1.025000000	$9.99999E-07$
5	1.0277525	550.5050505	1.025000000	$9.99999E-07$

Table 3.2: Results of the DA method for the bar stretching model

Another problem is that in the original description, the unbending moments were used. This is only possible in the case of a one or two-dimensional FE model, for a three-dimensional problem springback is caused by the complex internal stresses in the blank. There is no obvious way to 'reverse' these internal stresses. However, it is proposed here to use a nodal force field. This field is calculated by constraining all blank nodes in all directions after forming. The reaction forces then form a springback force field, which can be reversed and applied to a virgin blank-mesh to cause the spring forward deformation.

The push-back step
To solve the faulty convergence of the SF method, another step is introduced in the SF iteration: the push-back step (see Figure 3.10). After a forming simulation, the blank is allowed to spring back. After this, the blank is pushed back into the desired shape, delivering the push-back force field. With this push-back force field the product is compensated. Another important change is that, in the second and following iterations, the compensation is carried out on the compensated geometry of the previous iteration. In this form the method is now called Push-back Spring-Forward or PBSF

Even when the two problems of SF, faulty convergence and three-dimensional application, are theoretically solved with this principle, the results remained unsatisfactory even for simple problems, such as the deep drawing of a nap. This happens because after forming, a large in-plane tension is found in the product. When, during the springforward calculation, this state is reversed, large compressive stresses occur. This makes the deformation unstable, in a similar way to the findings of Gan [24], and in some cases the FE calculation of the springforward step failed to converge [43]. Because of these problems, the (PB)SF method was not investigated further.

3.3.3 Analytical verification of iterative and one-step DA

The main focus of springback compensation is its use in combination with FE simulations. However, these simulations require large calculation times and comprise several complex physical and numerical phenomena. Therefore, firstly, an analytical model of a forming process was used to investigate the properties of the DA method. Besides the fast and reliable calculation, this has another advantage: since the model is straightforward and because it only has few parameters, it can also

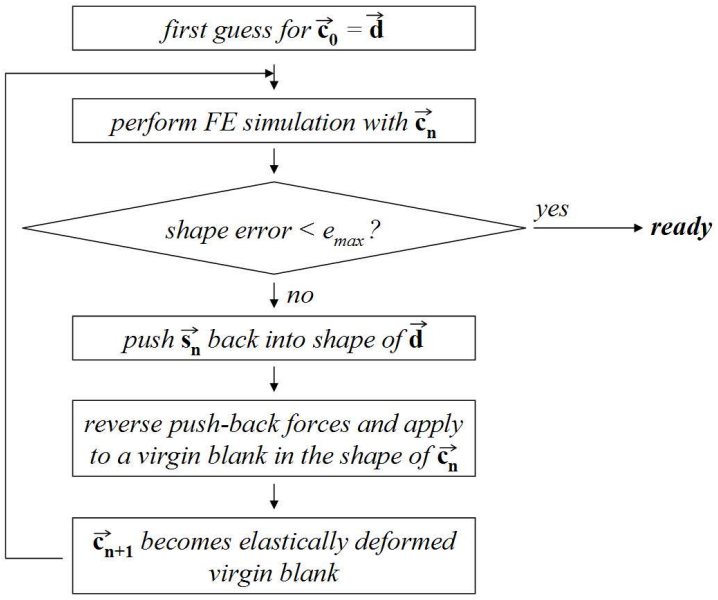


Figure 3.10: Principle of the PBSF method

provide principal insights in the compensation process. In this section, the following questions will be answered:

- How can the optimal forming shape be calculated directly?
- What is the relationship between process parameters and the compensation factor?
- What is the relationship between the desired geometry and the compensation factor?
- Does the iterative DA variant perform better than the one-step method?
- How does the convergence of the iterative variant relate to process parameters and desired geometry?

The text in this section is strongly based on the journal paper co-written by Gan and Wagoner [45].

The stretch-bending model
The analytical forming model [73, 74] represents a stretch-bending process that was developed to assess the accuracy of FE springback calculations. An initially straight bar is bent to a forming shape with radius R due to a bending moment M and a tension force T. When the loading moment is removed, the bar springs back to a radius r (Figure 3.11).

The strain ε in the direction along the beam is calculated in the following formula. The tension strain is ε_t.

$$\varepsilon_{xx} = \varepsilon = \frac{z}{R} + \varepsilon_t \tag{3.7}$$

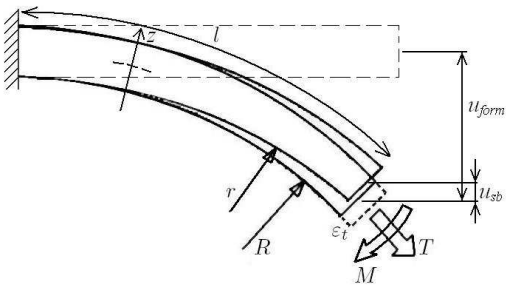


Figure 3.11: The analytical stretch-bending model and the main model parameters

By filling in $\varepsilon = 0$ the neutral line z_0 can be found:

$$\varepsilon = 0 \Leftrightarrow z_0 = -R\varepsilon_t \tag{3.8}$$

An elasto-plastic material model was chosen, using the Hollomon law:

$$\sigma = \begin{cases} \sigma_0 + K(\varepsilon - \frac{\sigma_0}{E})^n & \sigma > \sigma_0 \\ E\varepsilon & -\sigma_0 < \sigma < \sigma_0 \\ -\sigma_0 - K(|\varepsilon - \frac{\sigma_0}{E}|)^n & \sigma < -\sigma_0 \end{cases} \tag{3.9}$$

In this model, E is Young's modulus, σ_0 is the initial yield stress, K and n are parameters for the hardening behavior. At the end of the deformation stage, the bar is held in equilibrium by the tension force T and the moment M. These can be calculated analytically as follows:

$$T = \int_{-t/2}^{t/2} \sigma w \mathrm{d}z \tag{3.10}$$

$$M = \int_{-t/2}^{t/2} \sigma z w \mathrm{d}z \tag{3.11}$$

In this foprmula, w is the width of the beam. Details of the model's assumptions and the calculation of these integrals can be found in [73]. It is important to notice that only the springback caused by the unbending moment is considered for the equilibrium of the model. The tension force along the bar is only applied as a fictive strain. The radius R after springback can be calculated with the following formula [75]:

$$\frac{1}{R} - \frac{1}{r} = \frac{M}{EI} \tag{3.12}$$

Direct calculation of the optimal forming shape
In the case of the analytical model, no compensation strategy is required: For a given desired radius after springback r_{target}, the required forming radius $\bar{R}$ can be directly calculated from Equation (3.12):

$$\frac{1}{\bar{R}} - \frac{1}{r_{\text{target}}} = \frac{M}{EI} \Leftrightarrow \frac{EIr_{\text{target}}}{M(\bar{R})r_{\text{target}} + EI} - \bar{R} = 0 \tag{3.13}$$

Since $M = M(\bar{R})$, this is a nonlinear equation. The function is well-behaved so that a bisection algorithm [56] was used to find $\bar{R}$ numerically. In Figure 3.12 the results

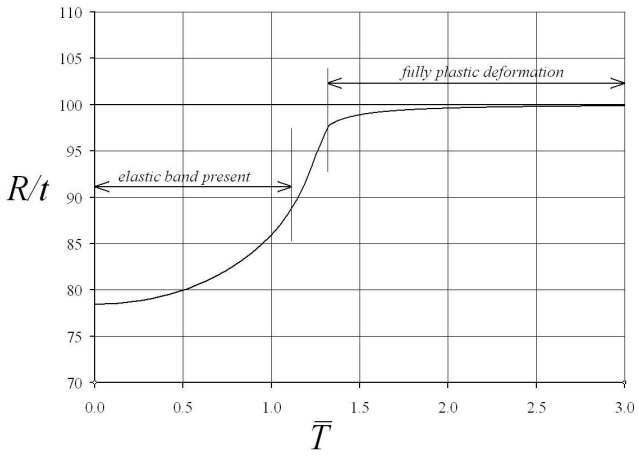


Figure 3.12: Optimal forming radius $\bar{R}$ for various normalized tension forces

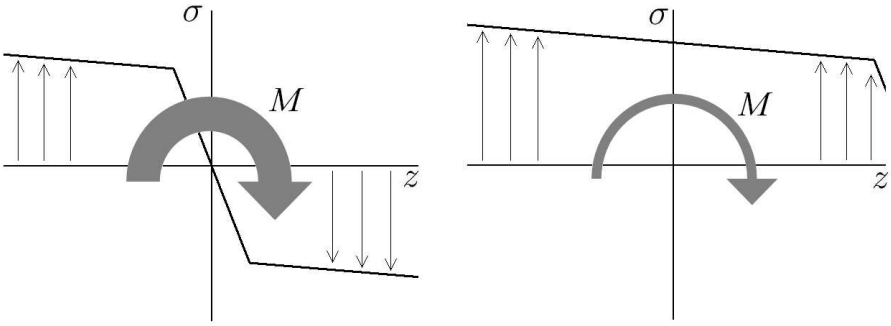


Figure 3.13: The unbending moment for pure bending (left) and a large tension load (right)

are visualized for various values of the normalized tension force $\bar{T}$. *Normalized* means in this context that the tension force is divided by the force, required to achieve plastic deformation under tension only:

$$\bar{T} = \frac{T}{\sigma_0 wt} \tag{3.14}$$

The desired radius was set at $r_{\text{target}} = 1.0$ and $r_{\text{target}}/t = 100$. This is a shallow curvature, which represents the shape of many automotive body panels. When the stretching force is increased, $\bar{R}$ converges to r_{target}.

The explanation for this phenomenon is that, as in more complex forming processes, springback decreases with increasing in-plane tension [41]. The stress-state over the thickness of the bar is compared in two situations: with no additional tension load and with a large tension load. The stress profile over the thickness of the bar is visualized schematically in Figure 3.13. When the elastic zone is no longer present anymore in the bar, the stress state gets a gentle slope and the unbending moment , and therefore springback, diminishes. In this case, the bar can be bent to the desired shape directly and no compensation is required, as Figure 3.12 shows.

One-step Displacement Adjustment

Firstly, the focus is on the one-step application of the DA method. The DA method relies on displacement data, whereas only bending radii R and r are calculated in

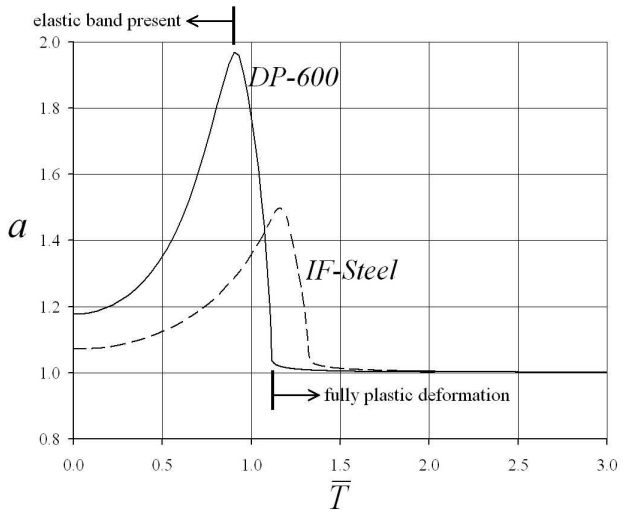


Figure 3.14: Optimal compensation factor for two different steels at varying tension force, $R/t = 100$

the analytical model. However, the springback displacement u at the end of the bar can be derived directly from these variables and the length of the bar l, as shown in Figure 3.11.

$$u(R) = \sqrt{R^2 - l^2} + R \tag{3.15}$$

With the springback displacement and the optimal forming displacement, the optimal compensation factor $\bar{a}$ can be directly derived using Equation (3.15):

$$\bar{a} = -1 \cdot \frac{u(\bar{R}) - u(r_{\text{target}})}{u(r) - u(r_{\text{target}})} \tag{3.16}$$

The results are shown in Figure 3.14. In the graph the compensation factor is drawn for a varying tension force T, and for two materials; interstitial free (IF) steel and DP-600 (see table 3.3 and Figure 3.15). When an elastic band is present inside the bar (for IF steel $\bar{T} < 1.32$) the compensation factor rises with increasing tension. When the force is so great that the bar is deformed plastically in the entire cross section, the compensation factor becomes 1.0. It can also be concluded that for higher strength materials, a higher compensation factor needs to be applied.

	Young's Modulus	Yield Strength	K	n
`IF-Steel`	210 GPa	150 MPa	425 MPa	0.40
`DP-600`	210 GPa	420 MPa	600 MPa	0.47

Table 3.3: Material data

Now, why is a compensation factor required at all? In the one-step application of DA it needs to be applied because the forming process is nonlinear: after springback compensation, the forming process has changed and springback has become different as well.

For this analytical example the only parameter that is changed during compensation is the forming radius. Changing the forming radius will change the amount of

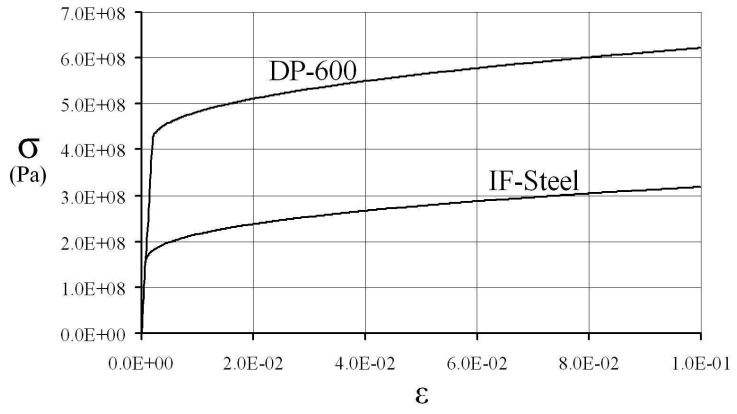


Figure 3.15: Stress-strain curves for IF-Steel and DP-600

springback. When the tension strain is considered, this change is different in three cases:

- $\varepsilon_t = 0$, pure bending
- $0 < \varepsilon_t \leq \frac{t}{2R}$, elastic band inside the bar
- $\varepsilon_t > \frac{t}{2R}$, fully plastic deformation

Therefore, different compensation factors are required, which explains the complex shape of graph 3.14. The stress profiles inside the bar are shown for all three situations in Figure 3.16 (left). The grey lines represent bending to a radius of 1.0, the black lines show the stress profiles when the bar is bent *further*, to a radius of 0.8. R/t equals 100, and the material is IF-steel. In the case of pure bending, the elastic band becomes slightly narrower as bending increases. In the case of fully plastic deformation, only a very small change occurs (due to the hardening in the material). However, in the intermediate case, the stress profile becomes different because the neutral line shifts.

The unbending moment, which causes springback, is calculated by integrating the function $\sigma(z)z$ over the thickness, as shown in Figure 3.16. This function $\sigma(z)z$ is shown in the graphs on the right, again for bending radii of 1.0 (grey) and 0.8 (black). The unbending moment, and therefore springback, only changes significantly in the intermediate case, due to the shifting of the neutral axis. The unbending moment becomes larger by bending further, so springback becomes larger as well and a compensation factor higher than 1.0 can be used to great effect. The location of the neutral axis is dependent on the tension strain in the following way(see Equation (3.8)):

$$\frac{\mathrm{d}z_0}{\mathrm{d}R} = -\varepsilon_t \tag{3.17}$$

This means that the shift of the neutral axis increases when the tension strain increases, so the change in springback increases with increasing tension strain as well. This explains the increasing compensation factor in Figure 3.14 in the intermediate case. When the tension strain is so large that the neutral axis is not present

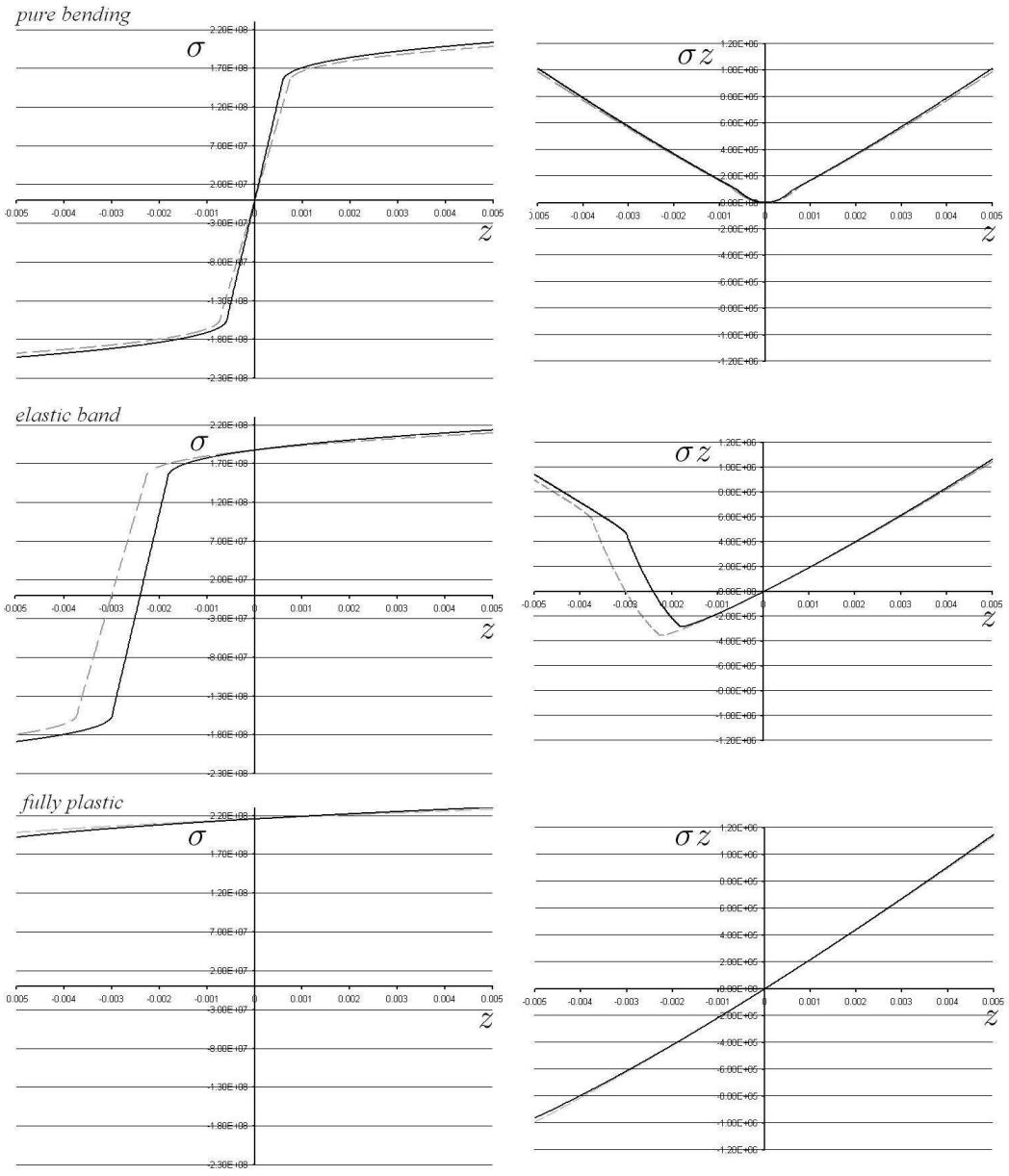


Figure 3.16: Stress profiles (left) and the springback moment integrand (right). IF-Steel, $R/t = 100$

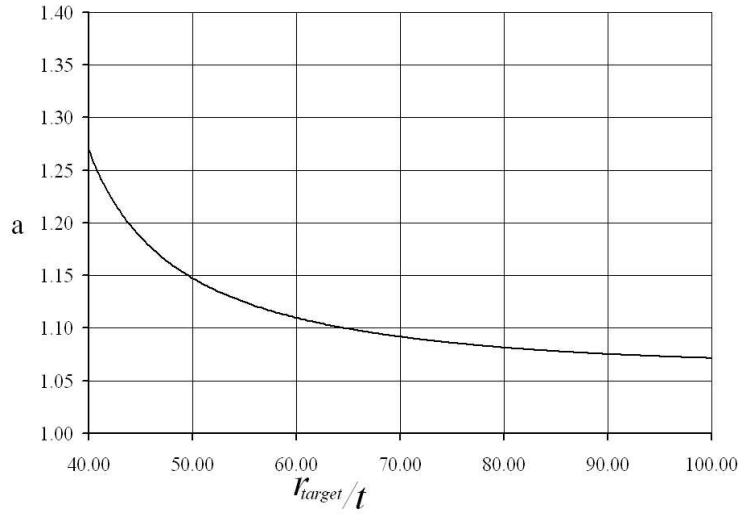


Figure 3.17: Optimal compensation factor vs. target radius). $t = 0.01$, IF-Steel

anymore, only plastic deformation takes place and the optimal compensation factor approaches 1.0.

It is also interesting to know the relationship between the tool geometry and the compensation factor $\bar{a}$. In Figure 3.17, $\bar{a}$ is shown with varying values of r_{target}/t. In industrial parts, the curvature differs strongly over the geometry. The model shows that the shallower the geometry is, the lower the compensation factor becomes.

As this simple example has shown, the compensation factor depends on the stress state in the bar, as well as the geometry that is formed. In an industrial forming process, both the geometry and stress state are very different at various places in the product, and due to hardening, even the material behavior will start to vary. Hence, a different compensation is required at different locations in the part. This can only be achieved with the iterative variant of DA. Still, the one-step DA variant should not be dismissed. Even though the single compensation factor makes the results less accurate in industrial applications (as shown in [47] and the following section), it is sometimes the only way to carry out springback compensation, for example when geometrical measurements of prototype products [52] instead of FE simulations provide the springback field.

Iterative application of DA
The iterative variant will now be demonstrated for the stretch-bending process. Figure 3.18 shows the shape error during the subsequent iterations with the iterative DA process for various tension forces. Note that the shape error was normalized by dividing it by the shape error using the original tools (this is done because springback is different for different tension loads). It can be concluded that in all cases, the iterative implementation of DA will quickly converge to the correct forming-radius, irrespective of the tension force T, however T does influence the convergence rate of the method.

As the tension force T increases, the convergence becomes slower, until the load is so high that the bar is fully plastically deformed (for IF-steel: $\bar{T} > 1.32$), then

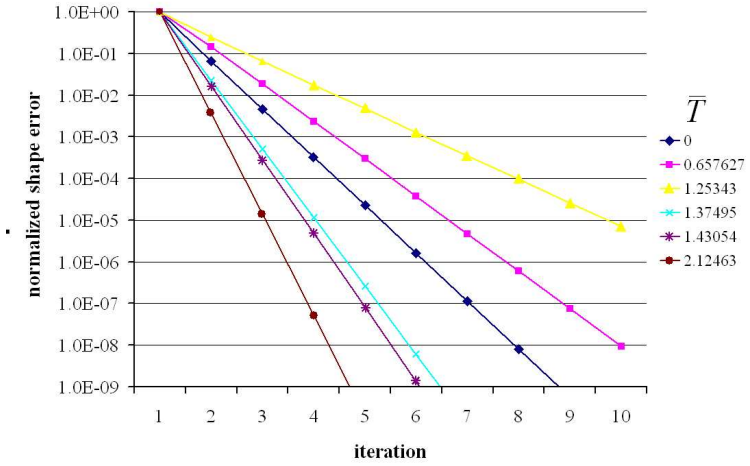


Figure 3.18: Convergence of iterative DA with various tension strains, $R/t = 100$, IF-Steel

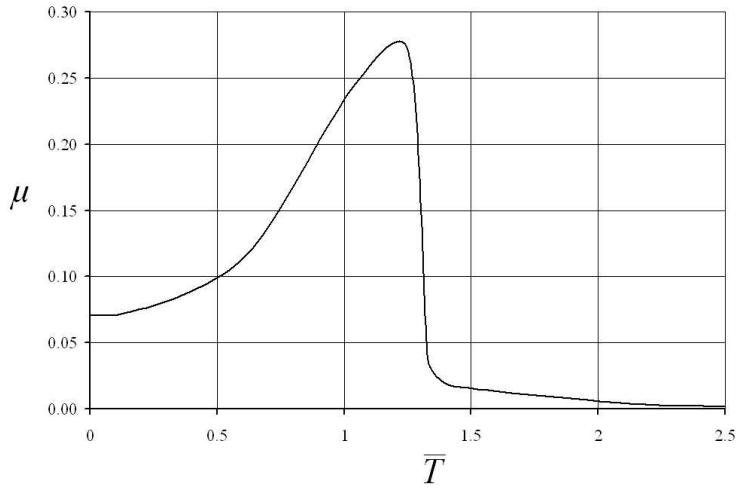


Figure 3.19: Rate of convergence for varying tension force, IF-Steel, $R/t = 100$

convergence is very fast again. This is shown more clearly in Figure 3.19, which shows the rate of convergence in the i-th iteration:

$$\mu = \frac{r_{i+1} - r_{\text{target}}}{r_i - r_{\text{target}}} \tag{3.18}$$

When the DA process is applied to the analytical model, μ is approximately constant in each iteration. For large values of T, μ approaches zero, which means that the speed of convergence is very high. Note that the graph is similar to graph 3.14. In the iterative DA process, the tools are compensated with a factor of 1.0 in each iteration, so when the optimal 'one-step' compensation factor is close to one, very fast convergence is obtained. When an elastic band is present in the bar the compensation becomes slower as the process is more non-linear.

The conclusion is that when the bar is fully plastically deformed: Not only does the compensation become very fast, springback is also reduced drastically. From an industrial standpoint, this is important for shallowly curved objects such as a car

roof, where there is relatively little plastic deformation in the product area. The amount of springback increases, and it becomes harder to compensate.

Conclusions for the analytical model
The analytical stretch-bending model provides insight in springback behavior and in how to compensate most effectively. It has been shown that the compensation depends on geometry, material and process parameters:

- The compensation factor is higher for strongly curved geometries.
- The compensation factor depends strongly on the in-plane tension in the bar.
- The compensation factor is higher for higher strength steels.
- No single correct global compensation factor can be provided for complex blank geometries.
- The convergence speed of the DA method also depends on material, process and geometrical parameters, however, quick convergence is reached for all cases.
- The basic assumption for iterative application of SF is not correct.
- The iterative DA method leads to better tool shapes than the SF method.
- The SF method is sensitive to the position of fixation points due to high compressive stresses in the blank, and due to buckling effects, calculation of the compensated geometry is not possible in most cases.

3.4 Springback compensation for industrial processes

The application of the DA algorithm in three dimensions and in the context of real industrial forming processes, rather than academic problems, requires many additional operations, which are all related to tool design requirements. The DA principle applies to the forming shape of the product only, and it needs to be transferred to the tools with great care. As pointed out in chapter 1, the deep drawing process can be very sensitive to small changes in the tool geometry. When drastic changes are applied to the tools, the behavior of the process may change dramatically after compensation, and the expected improvement in geometrical accuracy will not be achieved.

The transfer of the compensation of the forming shape to the forming tools is the subject of the following section. The smoothness of the tool surfaces needs to be maintained. This is not the only requirement, the gap-width between the tools has a great influence on the draw-in of the blank, as was discussed in the previous chapter. It is also beneficial to keep the blankholder area unchanged to retain the properties of the forming process. When the product features steep-angled walls, undercuts are likely to occur in the compensated tools.

3.4.1 Retaining tool surface quality

The main DA principle was defined in Eq. (3.5) as:

$$\vec{\mathbf{c}}^{j+1} = \vec{\mathbf{c}}^{j} - 1 \cdot (\vec{\mathbf{s}}^{j} - \vec{\mathbf{d}}) \tag{3.19}$$

The compensation field $\vec{\mathbf{\Phi}}^{j} = -1 \cdot (\vec{\mathbf{s}}^{j} - \vec{\mathbf{d}})$ is only known on the nodes $\vec{d_i}$ of the desired mesh $\vec{\mathbf{d}}$. The topology of the tool meshes $\vec{\mathbf{c}}^{j}$ is different from the product mesh and the tool geometries are larger, therefore interpolation as well as extrapolation of the compensation are required. Therefore, $\vec{\mathbf{\Phi}}^{j}$ is approximated by a continuous function $\Psi^{j}(\vec{x})$. This has an additional advantage: because $\Psi^{j}(\vec{x})$ is a smooth function, no abrupt changes are applied to the tool geometry. For this reason this method is now referred to as the *Smooth Displacement Adjustment* (SDA) method [44].

For a geometrical point $\vec{c}_i$ on the tool surface the shape modification is defined by:

$$\vec{c}_i^{\,j+1} = \vec{c}_i^{\,j} + \mathbf{\Psi}^{j}(\vec{c}_i^{\,j}) \tag{3.20}$$

This geometrical point can be an FE tool-node, but the function can also be applied in a CAD context, as will be shown in the next chapter. For clarity, the iteration variable j will be omitted in the following equations, unless necessary.

The approximation function $\Psi(\vec{x})$ is defined as a summation of $k_{\max}$ basis functions $\theta_k(\vec{x})$

$$\Psi(\vec{x}) = \sum_{k=1}^{k_max} \vec{a}_k \theta_k(\vec{x}) = \mathbf{a}\boldsymbol{\theta}(\vec{x}) \tag{3.21}$$

Different bases can be chosen for $\theta_k(\vec{x})$. A set of polynomials is the most straightforward choice.

$$\theta_k(\vec{x}) = x^{f_k} y^{g_k} z^{h_k} \tag{3.22}$$

Each θ_k can be defined by setting the factors f_k, g_k and h_k, an example set could be:

$$\boldsymbol{\theta}(\vec{x}) = \begin{bmatrix} 1 \\ x \\ y \\ z \\ xy \\ xz \\ yz \\ x^2 y \\ \vdots \end{bmatrix} \tag{3.23}$$

The distance of Ψ to the data points of $\mathbf{\Phi}$ is minimized in an L2-sense.

$$\min_{\vec{\mathbf{a}}} \; \|\Psi(\vec{x}) - \mathbf{\Phi}\| \tag{3.24}$$

The vector $\mathbf{a}$ is the solution to the minimization problem. Because the solution of the x, y and z-components is independent, the problem is split into its components

$$\Psi(\vec{x}) = \begin{pmatrix} \Psi^{x}(\vec{x}) \\ \Psi^{y}(\vec{x}) \\ \Psi^{z}(\vec{x}) \end{pmatrix} \tag{3.25}$$

Note that x, y and z are used as indices only. The calculation of the $\Psi^x(\vec{x})$ is shown here.

$$\Psi^x(\vec{x}) = \sum_k a_k^x \theta_k(\vec{x}) \tag{3.26}$$

A potential for the distance of the function to the data-cloud consisting of n points is defined as

$$\Pi = \frac{1}{2} \sum_{i=0}^{n} (\Psi^x(d_i^x) - \Phi^x(d_i^x))^2 \Delta A_i \tag{3.27}$$

This potential Π becomes minimal when the parameter set $a_l^x,\ 0 < l < k_{\max}$ is optimal:

$$\frac{\mathrm{d}\Pi}{\mathrm{d}a_l^x} = \sum_{i=0}^{n} (\Psi^x(d_i^x) - \Phi^x(d_i^x)) \frac{\mathrm{d}\Psi^x(d_i^x)}{\mathrm{d}a_l^x} \Delta A_i = 0 \tag{3.28}$$

This can be rewritten as:

$$\frac{\mathrm{d}\Pi}{\mathrm{d}a_l^x} = \sum_{i=0}^{n} \left(\left(\sum_k a_k^x \theta_k(d_i^x) \right) - \Phi^x(d_i^x) \right) \theta_l(d_i^x) \Delta A_i = 0 \tag{3.29}$$

This is a set of equations for each $l \in 0 < l < k_{max}$ and it can be written as a matrix-equation, and solved by inversion:

$$\mathbf{M}\mathbf{a}^x = \mathbf{c}^x \Leftrightarrow \mathbf{a}^x = \mathbf{M}^{-1}\mathbf{c}^x \tag{3.30}$$

with

$$\mathbf{a}^x = [a_0^x, a_1^x, \ldots, a_{k_{max}}^x]^T \tag{3.31}$$

and the components m_{kl} of matrix $\mathbf{M}$

$$m_{kl} = \sum_{i=1}^{n} \theta_k(\vec{d_i}) \theta_l(\vec{d_i}) \Delta A_i \tag{3.32}$$

and the components c_l^x of vector $\mathbf{c}^x$

$$c_l^x = \sum_{i=1}^{n} \Phi^x(\vec{d_i}) \theta_l(\vec{d_i}) \Delta A_i \tag{3.33}$$

The fitting procedure is weighted, using the Voronoi surface ΔA_i, as shown in Figure 3.20.

A more flexible, yet smooth, set of basis functions is the B-spline volume. The approximation function is then defined as:

$$\Psi(\vec{x}) = \sum_i \sum_j \sum_k N_{i,p}(x) N_{j,p}(y) N_{k,p}(z) \mathbf{a}_{ijk} \tag{3.34}$$

with $\mathbf{a}_{ijk}$ a three dimensional array with so-called 'control points', that define the shape of the volume. $N_{i,p}$ is the i-th B-spline basis function of degree p. The recurrence formula is most widely used to define the basis function:

$$N_{i,0}(u) = \begin{cases} 1 & \text{if} \quad u_i \leq u < u_{i+1} \\ 0 & \text{if} \quad \text{otherwise} \end{cases} \tag{3.35}$$

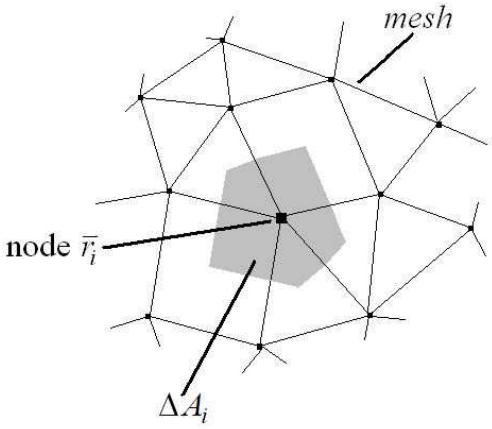


Figure 3.20: The Voronoi surface around a node

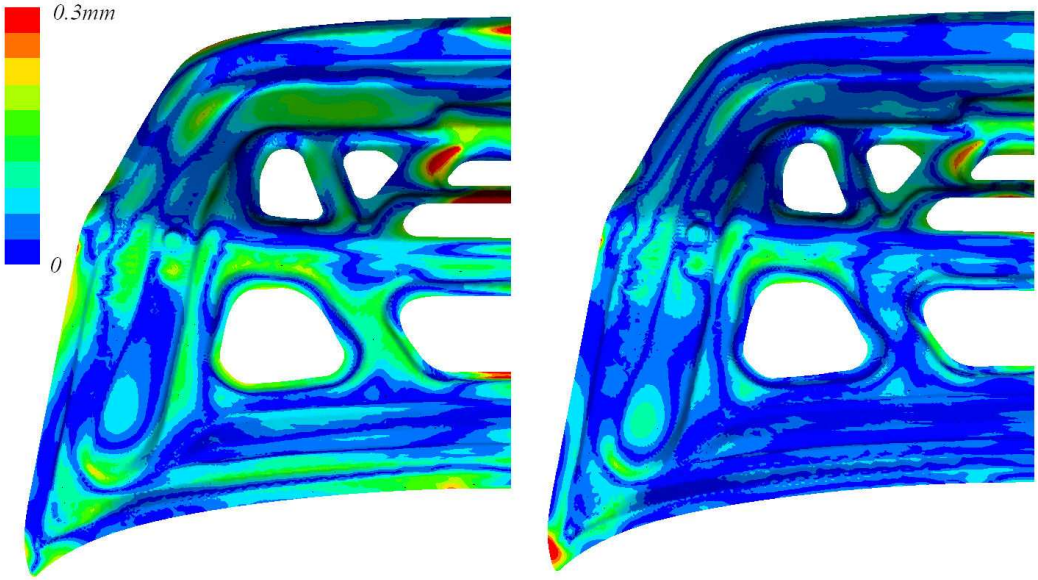


Figure 3.21: Approximation error for a quadratic (left) and cubic (right) spline function

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \tag{3.36}$$

Generally, the more control points are being used, the more flexible the function becomes, but its behavior might also become unpredictable. The least squares fitting of this function is principally identical but requires a more complex implementation. For more in-depth information on spline mathematics the reader is referred to the standard works [53] and [22], as well as the next chapter.

Both polynomial and spline approximation functions were tested on springback data of industrial parts. Due to increased flexibility, easily adaptable parameters and strict linear independence of its basis-functions, the spline function is much preferred and will be used in all industrial examples. The results are shown in Figure 3.21 for the trunk lid panel. For the left figure, a B-Spline volume with 3x3x3 quadratic sections is used, in the right figure 4x4x4 cubic sections. Shown is the

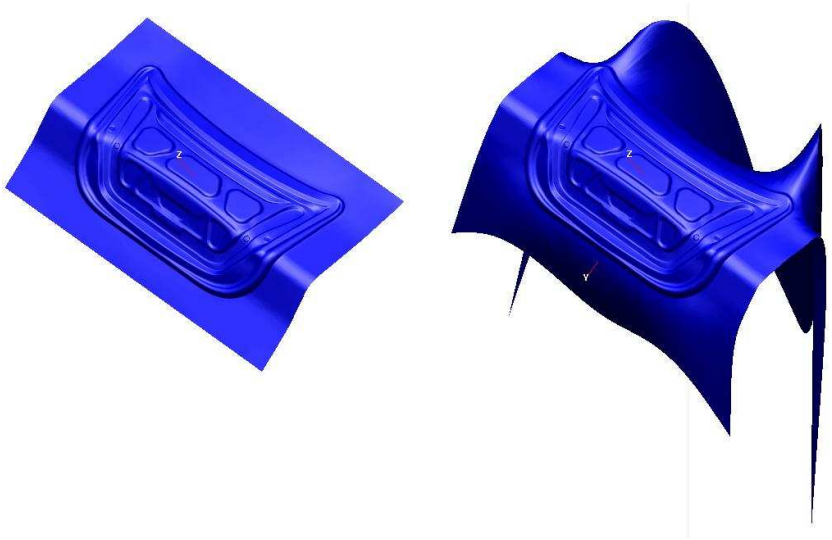


Figure 3.22: Extrapolation problem for the B-spline approximation function

distance between the springback geometry and the approximated springback geometry. Note that even with the cubic function, the error is considerable, at 10% of the maximum springback displacement. It will be shown in the next section that approximation accuracy is always a trade-off against the stability of the function in extrapolation.

3.4.2 Retaining the blankholder surface

The added flexibility of the B-spline volume function comes at a cost. The fully 3-dimensional function is fitted to a data set, defined on the surface of the desired geometry $\vec{\mathbf{d}}$ only, and could therefore be regarded as under-defined. In the deep drawing tools, the so-called die-addendum geometry is added, as well as the blankholder surface. These geometries are not part of the product geometry, they are designed by the process design engineer to achieve a beneficial blank draw-in and stretching of the blank. Because the approximation function is not supported by data points in this area, its behavior is not predictable. Figure 3.22 shows that this is a serious problem. The left die is the original tool, the right die is compensated with an compensation factor of 1.3 and a 3x125-parameter approximation function. Heavy changes are applied to these areas, which cause problems in the tool production and invoke undesirable changes in the forming process.

Two solutions were considered:

- Additional constraints on the approximation function
- Multiplication the approximation function with a cutoff function

Additional boundary conditions can help to tame the behavior of the approximation function in the extrapolation area. A generally applied strategy is to limit the gradient of the function. The disadvantage of this method is that the approximation properties of the function are impaired over the entire geometry.

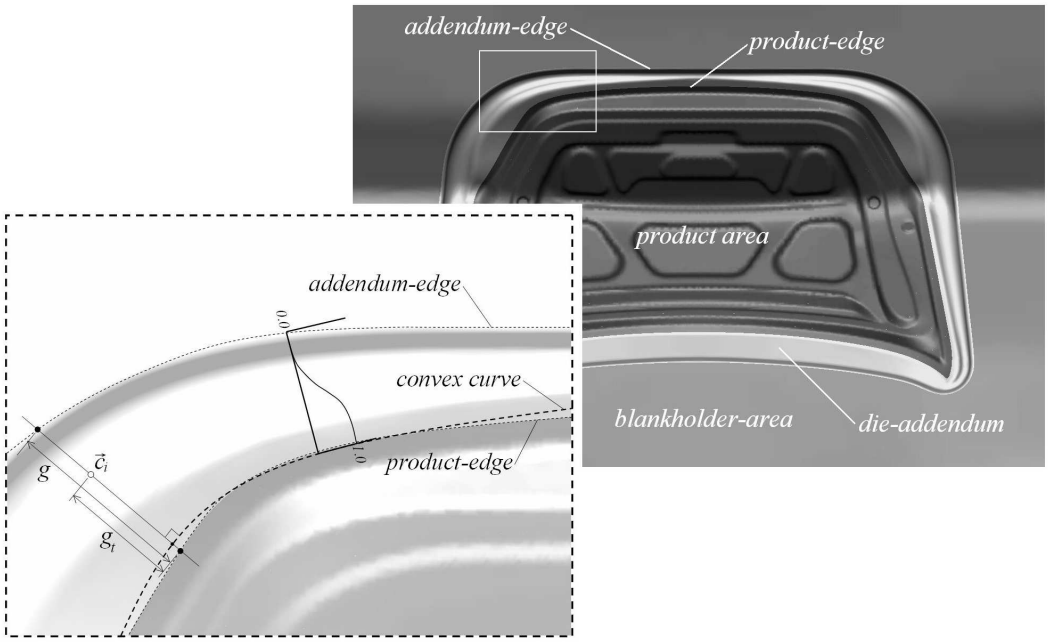


Figure 3.23: Principle of the cutoff function

A cutoff function is a function that smoothly changes from one to zero outside the product area of the blank. When the approximation function is multiplied by the cutoff function, it will also smoothly change to zero. It is quite cumbersome to define such a function, however, with the implementation that is proposed here, the approximation properties remain completely unchanged in the product area. Therefore, this method was chosen and implemented.

The cutoff function is based on two user defined (mesh) polygons, the product-edge and the addendum-edge. If the tool-node $\vec{c}_i$ is located inside the product edge, the cutoff value is 1.0. If it is outside the addendum-edge, the value becomes 0.0. Between the two curves the function value drops smoothly.

The implementation is clarified in Figure 3.23. All procedures are carried out in the xy-plane, with the z-axis as drawing direction. Firstly, the product-edge is approximated with a convex B-spline curve. The tool-point $\vec{c}_i$, for which a cutoff-value $\rho(\vec{c}_i)$ is to be calculated, is projected onto the convex curve. The projection line intersects the product-edge and the addendum-edge. g is the distance between the edges, and g_t the distance from $\vec{c}_i$ to the product edge. The cutoff value is now calculated using the polynomial that is shown in Figure 3.23:

$$\rho = 2\left(\frac{g_t}{g}\right)^3 - 3\left(\frac{g_t}{g}\right)^2 + 1 \tag{3.37}$$

The convex approximation curve is required to make sure that the resulting cutoff function does not self-intersect: no curve normal will intersect another curve normal. Figure 3.24 shows the cutoff function for a trunk-lid die. It remains smooth even when the product edge is not convex (as indicated by the circle). The final result is that now the compensation smoothly transforms to zero in the die-addendum area, as shown in Figure 3.25.

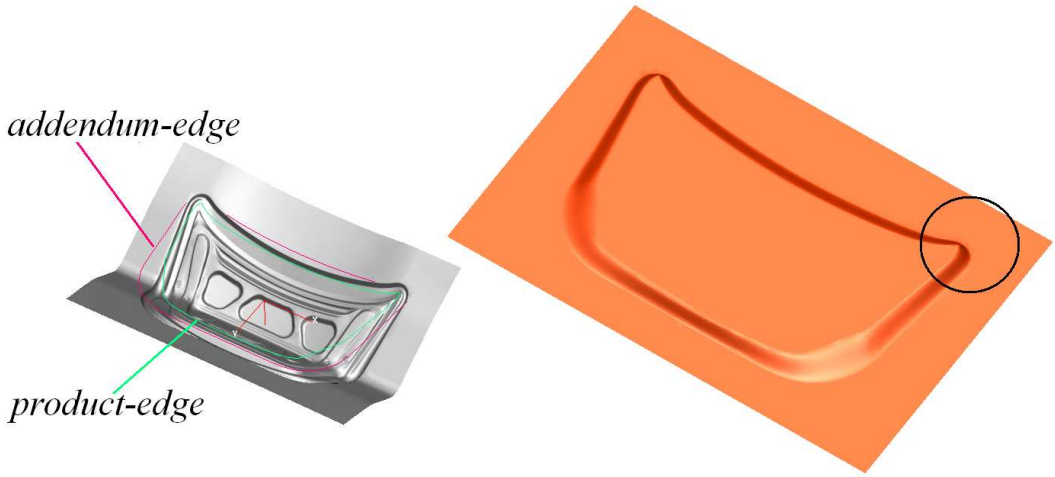


Figure 3.24: The resulting cutoff function

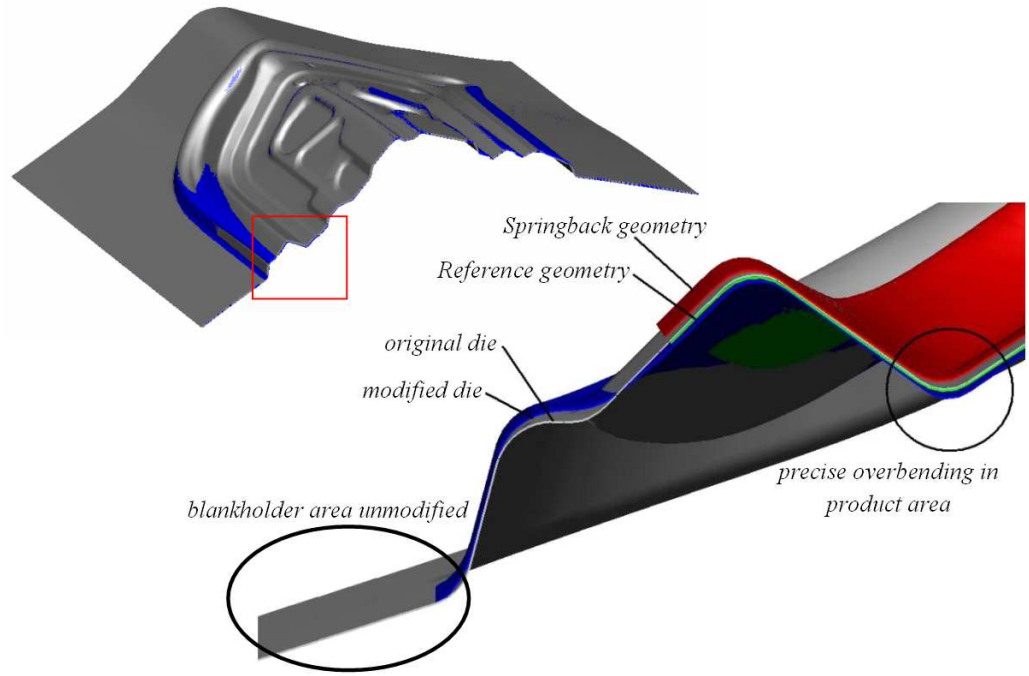


Figure 3.25: Smoothly compensated die

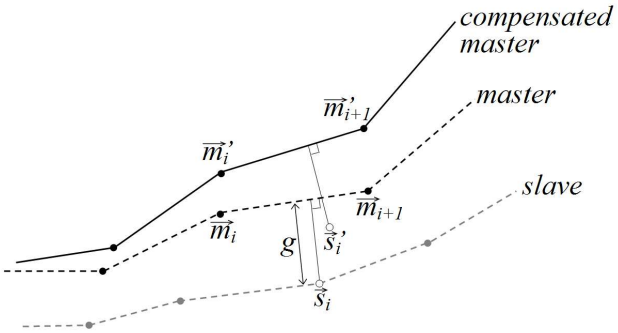


Figure 3.26: Master-slave compensation

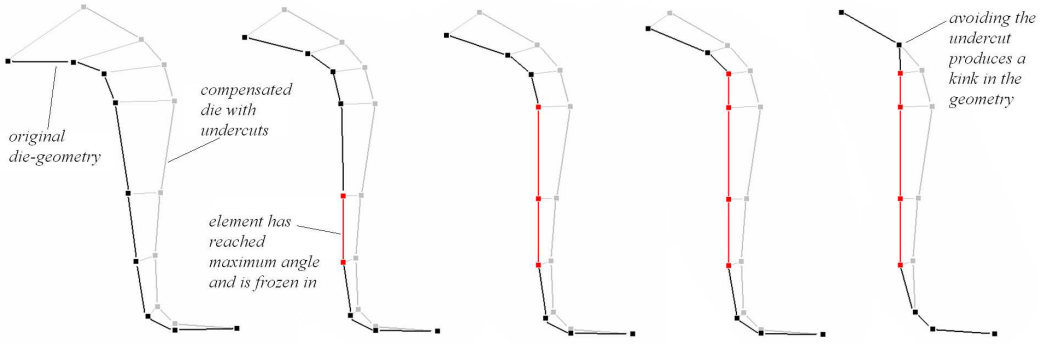


Figure 3.27: Principle of the undercut-avoiding algorithm

3.4.3 Gap-width preservation and undercut avoidance

The approximation function is fitted to the data of the desired geometry, which is a surface. However, the function is defined in the entire 3D space. Because the approximation function is so flexible, the gradient in the direction normal to the blank becomes considerable. This means that the shape modification for the die and the blankholder varies and the gap-width between those tools is changed after compensation. This change in gap-width can amount more than 30% of the original gap-width so it has a significant influence on the blank draw-in, so it should be avoided.

A so-called *master-slave* compensation strategy was developed for tool meshes. The idea is clarified schematically in Figure 3.26. One of the tools is declared as the master-tool, and it is compensated as previously described. The nodes of the slave-tool follow the shape change of the master-tool. A slave node $\vec{s}_i$ is projected onto an element of the master-tool. The barycentric coordinates of the projection point are calculated using the (unmodified) master-nodes $\vec{m}_i$ and $\vec{m}_{i+1}$. The projection point is now retrieved on the compensated master-element and the compensated slave-node is constructed by using the gap-width g.

Another issue is the possibility of undercuts in the compensated geometry. In some cases the angle of the tool walls exceeds 90 degrees after compensation, making the compensated tool unusable. The undercut-avoidance algorithm carries out the compensation gradually until an element reaches the maximum admissible angle, as shown in Figure 3.27. The element is then 'frozen in', after which the compensation

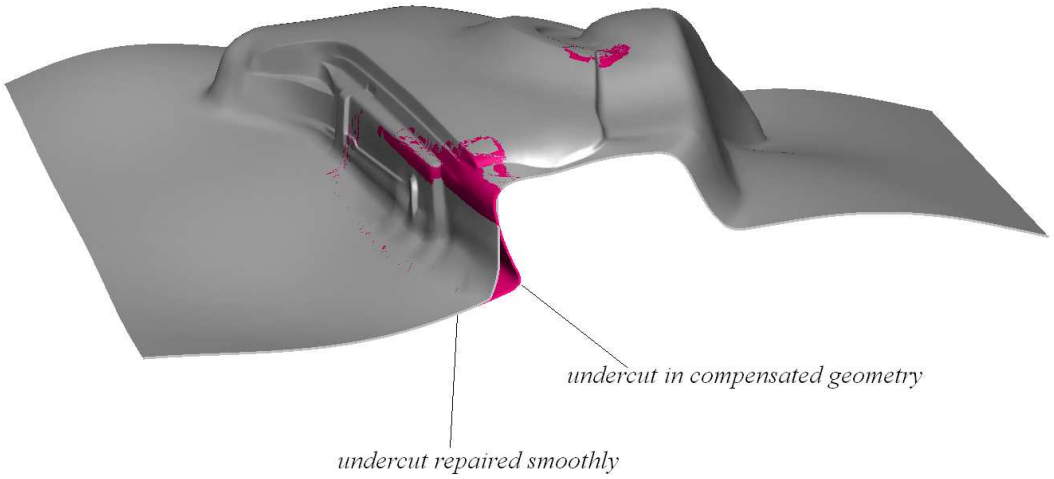


Figure 3.28: Avoiding undercuts: Example.

continues for the other elements until another element reaches the maximum angle. This element is also frozen in. This way, the algorithm continues step by step until the compensation has been completed. The figure also shows that this algorithm might produce kinks in the tool geometry. For this reason, a complex smoothing algorithm is included to retain the tool-surface quality. Figure 3.28 below shows the results of a heavy die-compensation without and with the undercut-avoidance algorithm.

3.4.4 Implementation

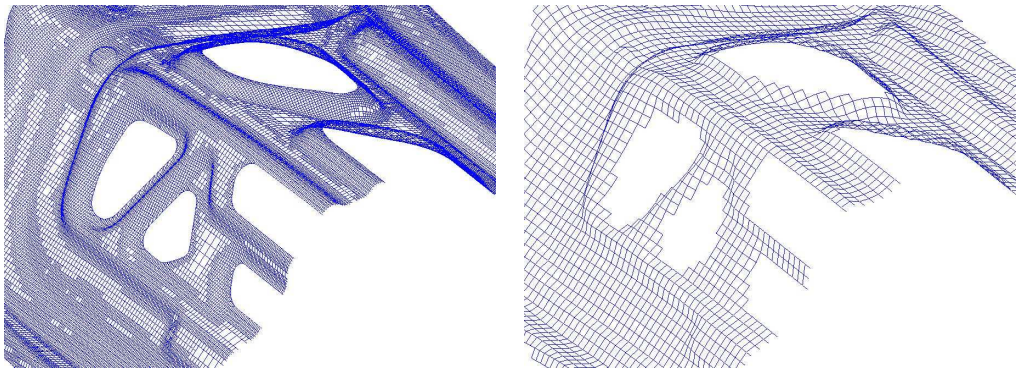
The SDA algorithm was implemented in C++ to work in combination with the commercial PAM-STAMP software. The SDA method and all previously mentioned additional algorithms can be used for any FE forming simulation software, however, some script-programming is required for the SDA software to automatically start, monitor and evaluate simulations.

When blank mesh refinement is used in the forming simulation, a problem occurs. In each compensation iteration, the forming tools are changed. This causes slight differences in the mesh refinement and as a result, the actual springback mesh $\vec{\mathbf{s}}^j$ is not topologically identical to the desired geometry $\vec{\mathbf{d}}$ anymore. While refinement could be switched off, the same problem occurs during a trimming simulation, where new nodes are added during the splitting of border elements. Therefore, a *derefinement* algorithm was implemented to remove the refined nodes to solve this incompatibility, as shown in Figure 3.29.

3.5 Examples

3.5.1 Process 1: Free forming

The first process is the free forming of a thick strip of metal. Free forming means that no blankholder is involved during forming. The product, made by Volkswagen AG, forms the backbone of a wheel suspension part. The blank material is DP-500 steel,



(a) original mesh (b) derefined mesh

Figure 3.29: Derefining the blank mesh

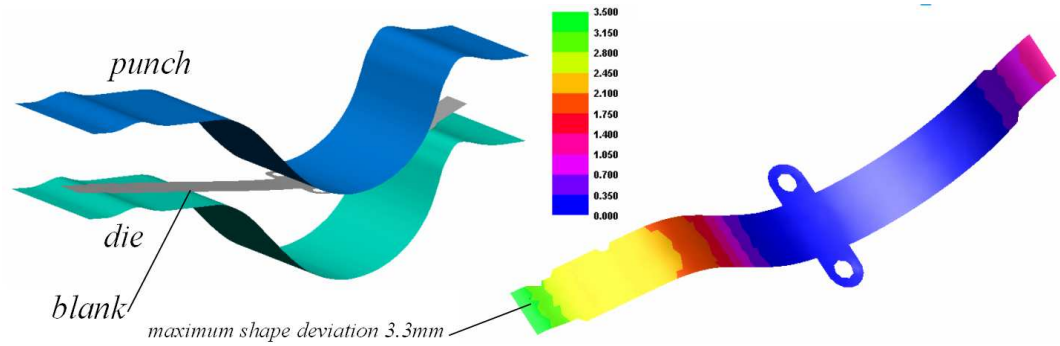


Figure 3.30: The strip forming process in PAM-STAMP (left) and springback (right)

with a sheet thickness of 3.5mm. The process has been modeled in PAM-STAMP, using an explicit solver during forming and an implicit solver for springback. The forming tool geometries and the springback deformation are shown in Figure 3.30. The maximum shape deviation amounts 3.3mm.

The free forming process was compensated with both the one-step and iterative methods. The result of the one-step compensation can be found in Figure 3.31 for various compensation factors. The mean shape deviation for all nodes is plotted, as well as the maximum shape deviation. The black arrows indicate the best compensation factor is between 0.65 and 0.69 with a reduction of 72 and 80% for the mean and maximum shape deviation. Realistically, at least 5 FE simulations are required to find this optimum compensation factor. Iterative compensation outperforms the one-step method after the third iteration already, as Figure 3.32 shows. The mean and maximum shape error are reduced by 94 and 96%.

The reason for the improved performance is that the compensation starts to vary over the part during iterative compensation. In one-step compensation the compensation factor a is the factor between the springback displacement and the compensation displacement. This can be generalized to a 'local compensation factor' a_i in iteration j, comparing the actual amount of compensation with the initial amount of

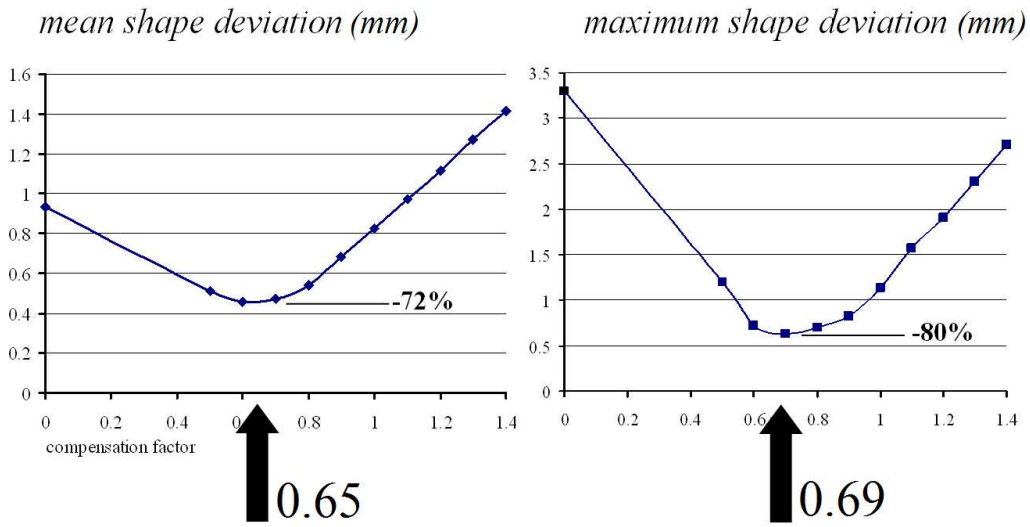


Figure 3.31: Results of one-step springback compensation for process 1

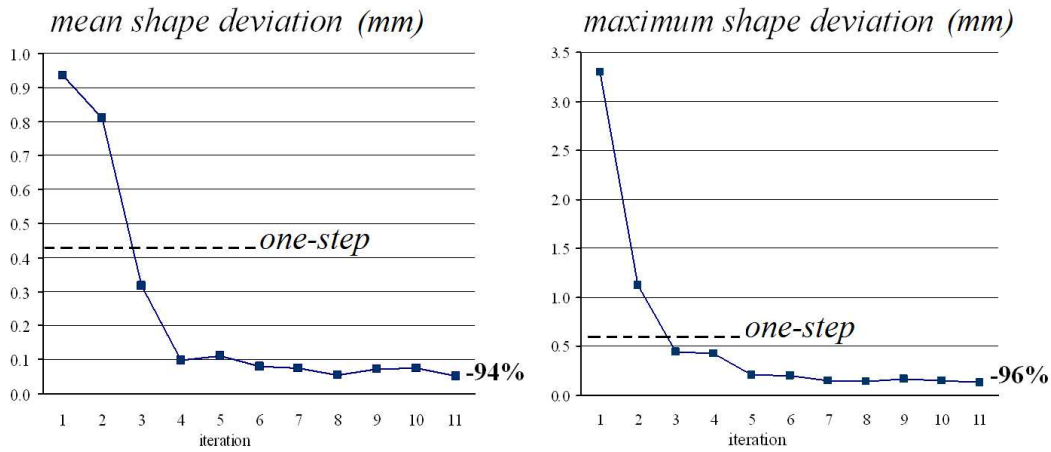


Figure 3.32: Results of iterative springback compensation for process 1

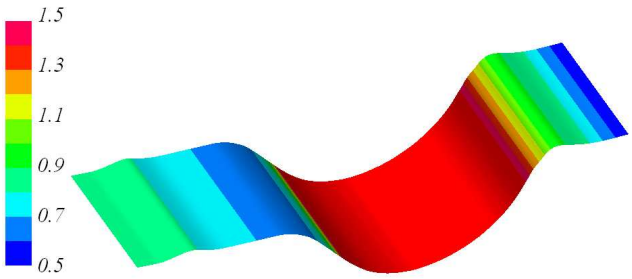


Figure 3.33: Local compensation factor on the strip-punch

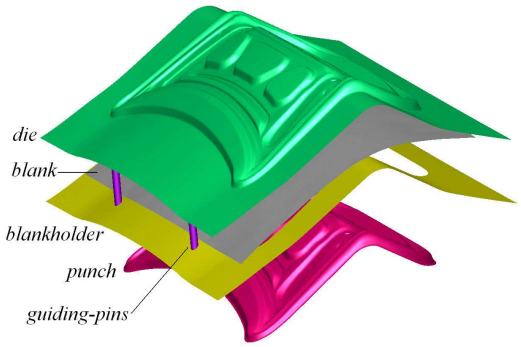


Figure 3.34: The NUMISHEET 2005 benchmark #1 process

springback:

$$a_i = \frac{\text{actual compensation}}{\text{initial springback}} = \frac{|\vec{c}_i^{\,j} - \vec{d}_i|}{|\vec{s}_i^{\,0} - \vec{d}_i|} \tag{3.38}$$

This is visualized in Figure 3.33. Looking at the image, it becomes clear that when the local compensation factor varies between 0.5 and 1.5, no one-step compensation factor can be defined that achieves the same accuracy.

3.5.2 Process 2: Inner panel drawing

This product, an inner trunk lid panel, was used as a benchmark process at the 2005 Numisheet Conference. It is formed in two stages; a deep drawing stage and a trimming stage. The FE meshes of the tools and the pre-bent blank as well as the trimming operation are shown in Figures 3.34 and 3.35. The blank is made of ZStE180 steel and has a thickness of 0.8 mm.

After trimming, the FE node at the location of the red point in Figure 3.35 is fixed in all three directions and rotations, and the blank is allowed to spring back freely. The resulting deformation is shown in Figure 3.36.

As pointed out previously: tool smoothness is of vital importance for the compensation. Surface smoothness can be verified by inspecting the light reflection on the geometry. This can be carried out on real geometries, but also on FE meshes or CAD data by using specialized software. Figure 3.37 shows the results of such an analysis. As far as the accuracy of FE meshes can reveal, the compensation did not

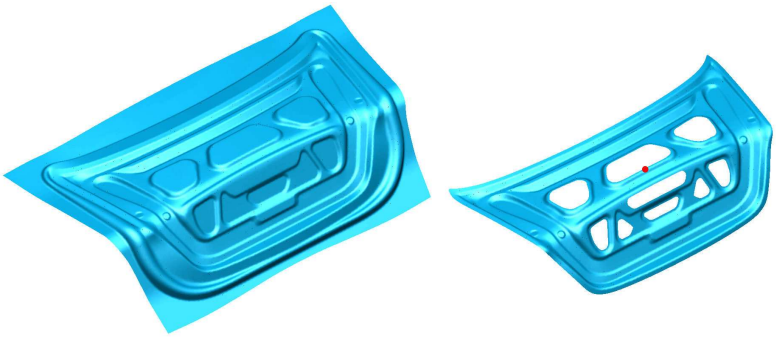


Figure 3.35: Trimming the NUMISHEET panel

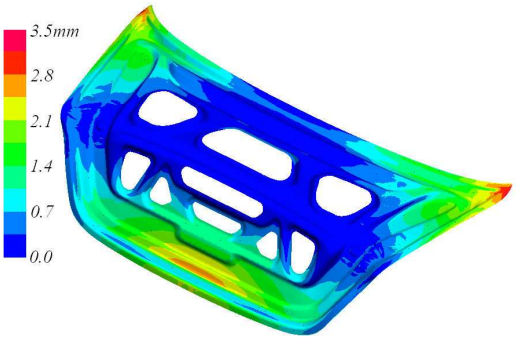


Figure 3.36: Springback displacement

impair the surface smoothness of the tool.

Figure 3.38 shows the shape error for the trunk lid panel during each iteration j. Interestingly, the best result is already achieved in the third or fourth iteration, a further iteration decreases the accuracy. Figure 3.39 again shows the shape deviation for the original process and the compensated tools ($j = 7$), but now the visualization range is set in the order of magnitude of the sheet thickness. This reveals that the shape deviation has decreased significantly below sheet thickness in the center of the product. However, the flanges of the product show an unsatisfactory result. Three possible explanations are:

- Compressive stresses in the flange result in long wavelength wrinkles that cannot be compensated effectively, because their exact location is physically unstable: the wrinkle 'wanders' along the flange when the process is changed by the compensation. The finite accuracy of the compensation is due to process changes
- Due to the derefining process, the shape deviation data in the flange is lost.
- The error of the approximation function becomes too large

Hypothesis 1
The springback displacement in the flange changes less than 10% during the consecutive iterations. Therefore, the cause of the problems cannot be the instable nature

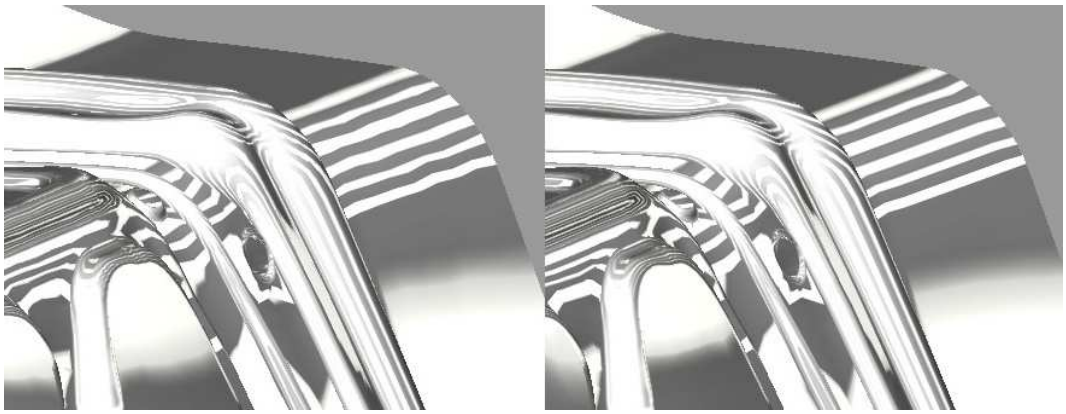


Figure 3.37: Tool light reflection before (left) and after compensation (right). Image courtesy of ICEM software

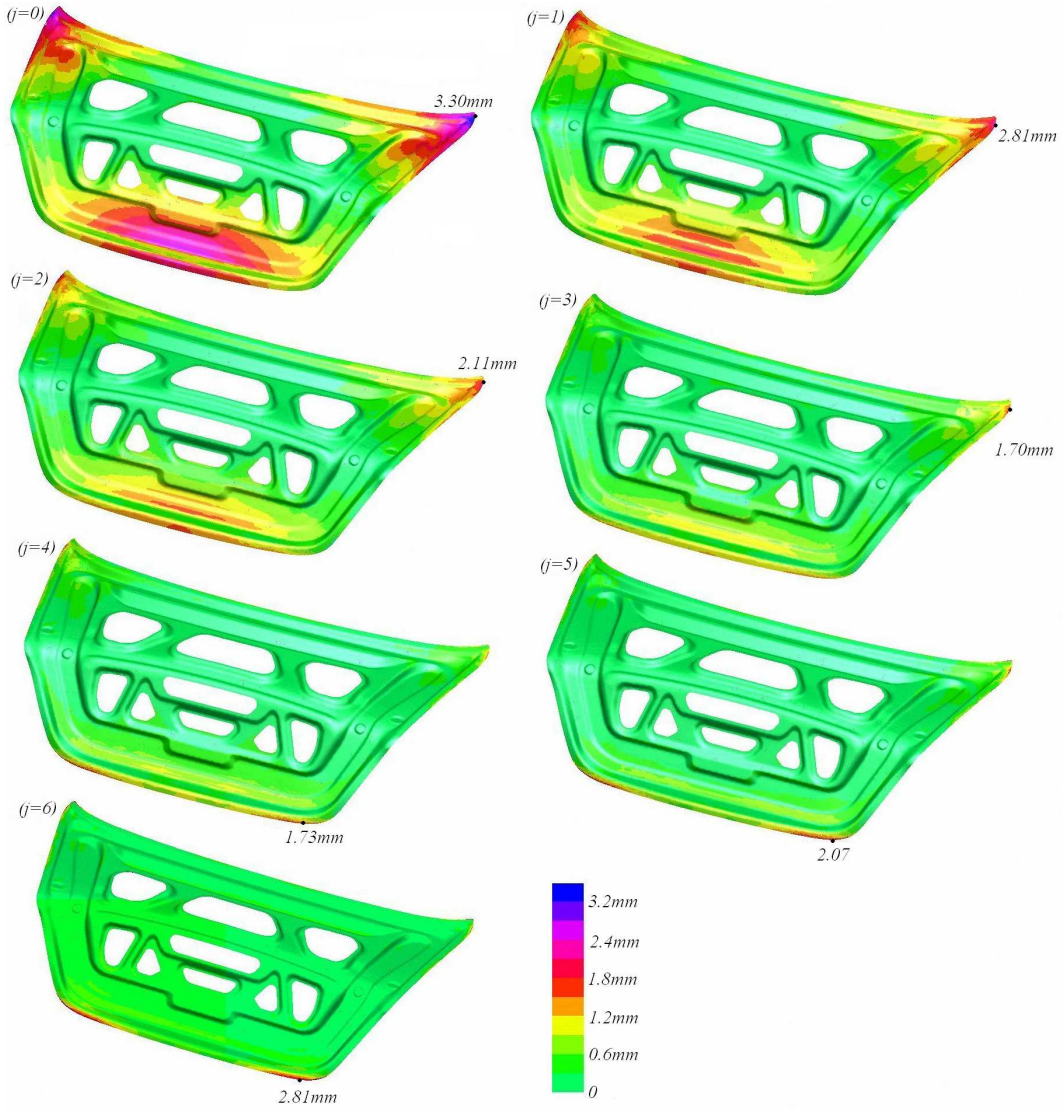


Figure 3.38: Results of the springback compensation for process 2

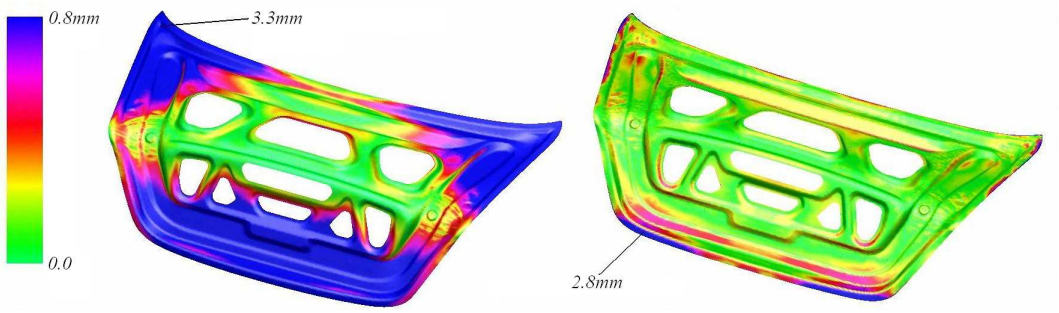


Figure 3.39: Shape deviation with original tools (left) and with optimized tools (right)

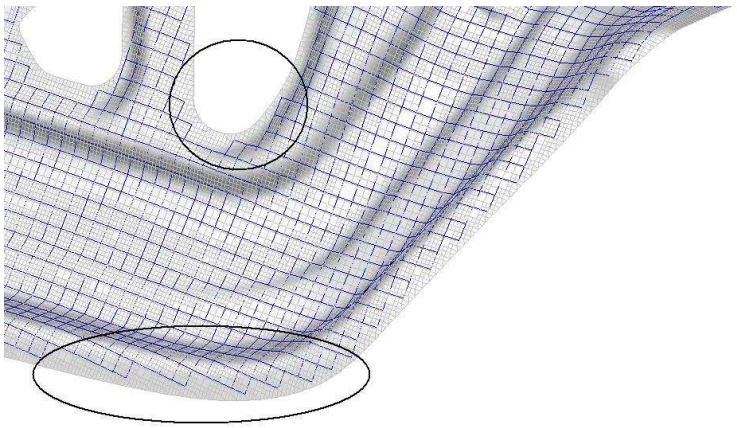


Figure 3.40: Loss of edge details due to derefining

of large wrinkles.

Hypothesis 2
The second hypothesis is confirmed by Figure 3.40. Almost all nodes in the flange are removed during derefining, leaving the SDA algorithm too little data for accurate compensation. This can easily be resolved by using a distance field instead of the shape deviation field and derefining algorithm after the first 2 or 3 iterations.

Hypothesis 3
To verify the third hypothesis, the error of the approximation function needs to be compared to the geometrical error that remains in the compensated product. The mean shape error is a more objective measure for the quality of the compensation. Here, a surface-weighted quadratic mean error is calculated in the current iteration and for the original forming process:

$$\varepsilon_{mean} = \frac{\text{actual shape deviation}}{\text{initial shape deviation}} = \frac{\sum_i |\vec{d_i} - \vec{s}_i^{\,j}|^2 \Delta A_i}{\sum_i |\vec{d_i} - \vec{s}_i^{\,0}|^2 \Delta A_i} \tag{3.39}$$

The line 'Shape Deviation-1' in Figure 3.41 also reveals that the best results are obtained in the fourth iteration. The reason is that the approximation error, shown in Figure 3.42 becomes large with respect to the actual shape deviation. The ap-

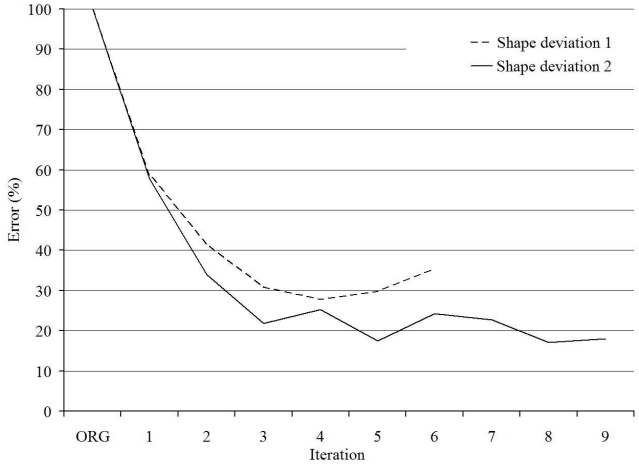


Figure 3.41: Shape error results for the NUMISHEET benchmark part

proximation error of the SDA method can be calculated as follows:

$$\varepsilon_{app} = \frac{\text{actual approximation error}}{\text{actual shape deviation}} = \frac{\sum_i |(\vec{d_i} - \vec{s_i^j}) - \psi_i^j|^2 \Delta A_i}{\sum_i |\vec{d_i} - \vec{s_i^j}|^2 \Delta A_i} \tag{3.40}$$

With decreasing shape error magnitude, the remaining error becomes more local and the global approximation function will have difficulties capturing such detailed shape errors. Oscillations will start to occur and the results will get worse. The limitations for the compensation accuracy are therefore a combination of hypothesis one and three.

Increasing the number of parameters to 7x7x7x3 (1029) improves the results, as the lines 'Shape Deviation-2' and 'Approximation Error-2' show. A reduction in shape error of more than 80% is achieved. However, a further increase in the number of parameters in the approximation function leads to numerical problems in solving the fitting problem and to extrapolation problems in the compensated die-addendum, even when the cutoff function is used.

Iterative versus one-step compensation
Just like the free-bending problem of process 1, the compensation also varies over the geometry in the iterative procedure, as Figure 3.43 shows. In the left picture the norm of the (initial) springback displacement $\|\vec{s_i^0} - \vec{d_i}\|$ is shown, and it differs from the norm of the compensation $\|\vec{c_i^6} - \vec{d_i}\|$, in the right picture.

In order to verify whether the iterative procedure leads to better results than one-step compensation, the methods are compared again. An optimal compensation factor needs to be found first. Using the distribution of local compensation factors a_i from the iterative results a weighted mean value is calculated as follows:

$$a_{mean} = \frac{\sum_i a_i \cdot \Delta A_i |\vec{s_i} - \vec{d_i}|}{\sum_i \Delta A_i |\vec{s_i} - \vec{d_i}|} \tag{3.41}$$

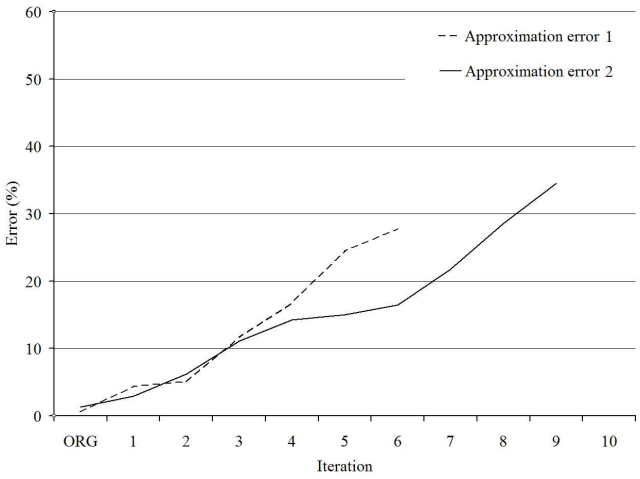


Figure 3.42: Approximation error for the NUMISHEET benchmark part

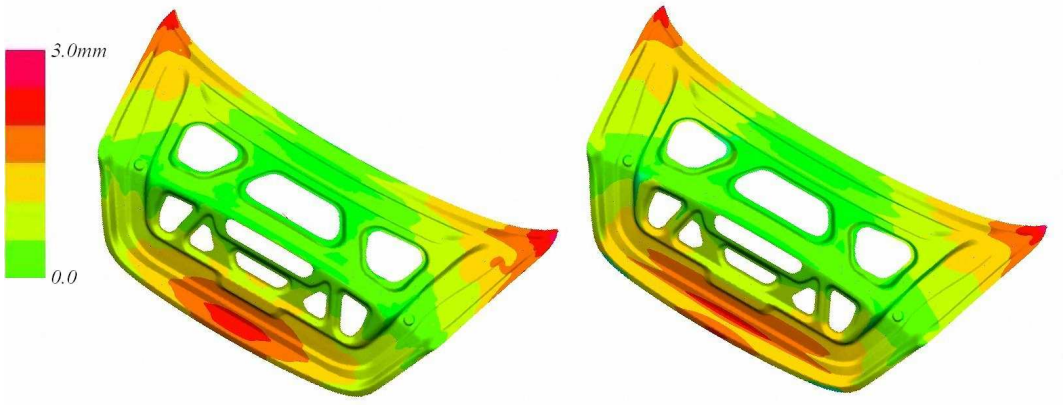


Figure 3.43: Springback (left) versus compensation displacement (right) after 6 iterations

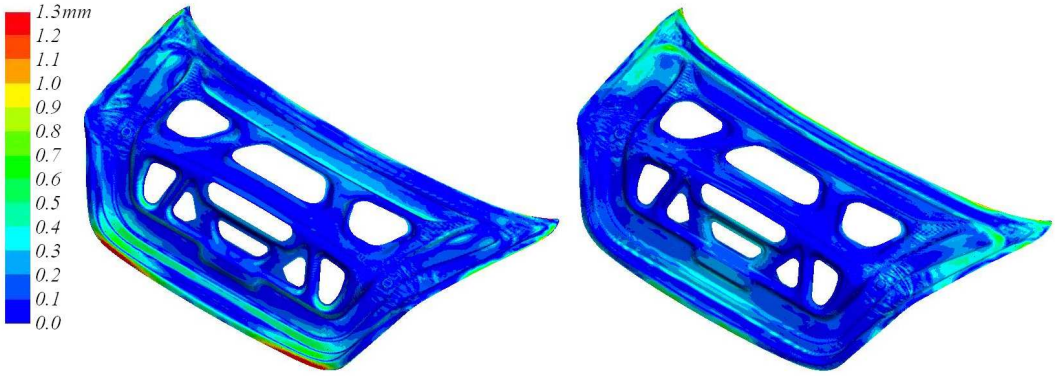


Figure 3.44: Shape deviation after iterative (left) and one-step SDA (right) compensation

The local compensation factor a_i is weighted with the nodal Voronoi area ΔA_i, as introduced in Figure 3.20, and the norm of the springback displacement. The use of this is to filter out the nodes with small or zero springback displacement: Due to the approximation error, the compensation displacement can become very large compared to the springback displacement, leading to erroneous local compensation factor values.

Equation 3.41 resulted in a mean compensation factor of 1.12 or 112%. Surprisingly, the result of the compensation is slightly better than for the iterative process, as can be concluded from Figure 3.44. The reason is mainly the aforementioned oscillations of the approximation function in the flange area. Note that an optimal compensation factor is generally not known, and some form of optimization process has to be carried out to obtain the best value. Again, the industrial rule of thumb of 130% is not of much use. For the iterative procedure this is not necessary.

3.5.3 Process 3: Outer panel drawing

Finally, the forming process of a flexible outer body panel was investigated. The product was an outer front fender part by Volkswagen. The forming tools are shown in Figure 3.45. Again, the forming process consisted of a forming stage and a trimming stage. Note that the process settings and material are fictive. In the simulation, ST14 with a thickness of 1mm was used as blank material. The springback deformation was shown at the beginning of the chapter in Figure 3.1.

For process 3, the front fender, the advantage of the iterative procedure is evident, as can be seen in Figure 3.46. The optimal compensation factor was determined by a standard optimization technique and amounted 0.78, an unexpectedly low value. The reason for this is the unusual twisting of the part in springback. Therefore, the compensated tools have a smaller drawing-depth than the original tools. As a consequence, the springback of the part has decreased after compensation and a compensation factor below one has to be chosen. In the iterative variant, the optimal solution is found automatically. The accuracy of the one-step tools is surpassed in the third iteration already and the optimal shape is found in the 9th iteration. This tool-shape leads to a significantly lower shape error, reinstating the recommendation

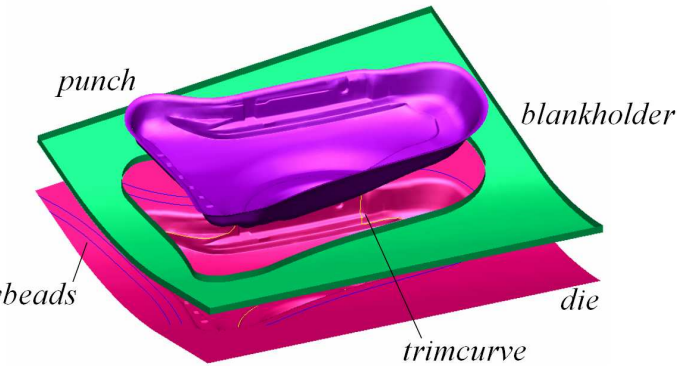


Figure 3.45: The forming process for the front fender

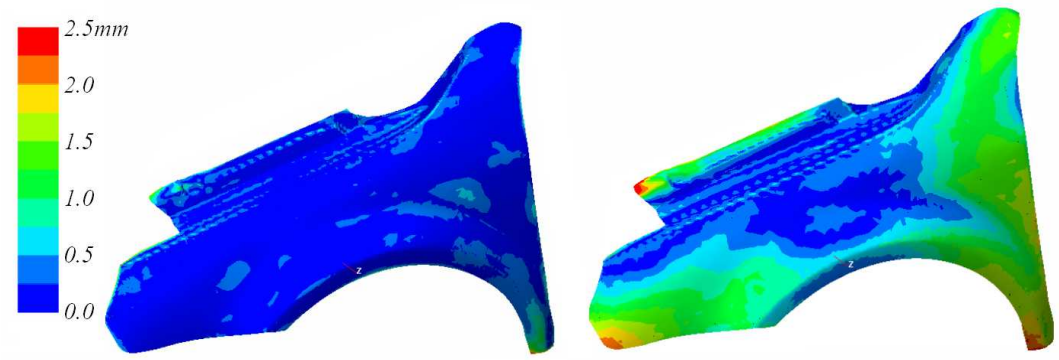


Figure 3.46: Shape deviation after iterative (left) and one-step compensation (right)

for the iterative procedure.

Unexpectedly large values (2.5-5.0) were found for the local compensation factor. The explanation is that during the iterations the blank had shifted considerably in tangential direction due to changes in the draw-in. Therefore, the compensation vectors were also almost tangential to the blank surface [47]. This error cannot be compensated effectively. Therefore, in the implementation of the SDA algorithm, a normal distance based compensation is applied in the final iterations.

The front fender is a relatively flexible part, and therefore the forces required to push the product back into the desired shape, were calculated as well. For this calculation, the product was clamped in the middle and pushed back at three significant locations. Note that these points were not taken from the official measurement plan, which is much more complex. The push-back forces before and after compensation are shown in Figure 3.47. It becomes clear that, because the forces exceed 30N, compensation is required. After compensation, all push-back forces are within the tolerance.

3.6 Conclusion

For all industrial processes, numerical springback compensation has brought a significant improvement in geometrical accuracy, while retaining the usability of the tools. Even when the quality of the results may not lead to an exact *first-time-*

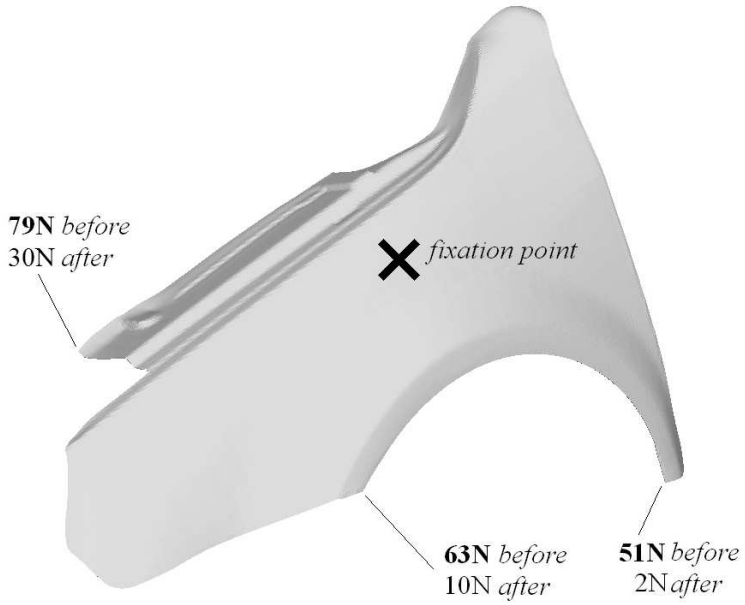


Figure 3.47: Push-back forces on the fender panel

right process on the real press, the improvement will help saving time in the tool development process.

- In all cases the use of the iterative variant of the Smooth Displacement Adjustment algorithm is preferred over the one-step procedure. Not only is the amount of required simulations lower, the effectiveness is also higher in most cases.
- The use of the displacement field is preferable for the first iterations, in consecutive iterations a distance calculation leads to better results.
- The approximation function limits the accuracy of the compensation. However, this limitation is unavoidable when the tool smoothness and usability have to be maintained.

Glossary

ε strain
$\boldsymbol{\theta}$ vector of shape functions
μ rate of convergence (inverse)
$\vec{\boldsymbol{\xi}}$ distance field
Π potential
ρ cutoff function
σ stress
Φ approximation function
a compensation factor
a_i local compensation factor
ΔA Voronoi area
$\vec{\mathbf{a}}$ shape function parameters
$\vec{\mathbf{c}}$ compensated forming shape
$\mathbf{c}$ fitting RHS
$\vec{\mathbf{d}}$ desired shape
E Young's modulus
EI bending stiffness (beam theory)
g distance to edge (cutoff function)
j iteration
K Holomon material parameter
l length
M bending moment
$\mathbf{M}$ fitting matrix
n Holomon material parameter
$N_{i,p}$ B-spline basis function
r radius of bar after springback
R forming radius
$\bar{R}$ desired radius
$\vec{\mathbf{s}}$ springback shape
t thickness
T tension force
u displacement at end of bar
w width
z coordinate in bar thickness direction

Chapter 4

Modification of tool CAD geometries

4.1 Introduction

Computer-Aided Design (CAD) forms the backbone of today's product development processes. Forming tools are designed with a combination of 3D solid and surface modeling. The product area of the forming tools is directly derived from the part's CAD geometry. In order to ensure the geometrical quality and smoothness of the stamped part, the geometrical tolerances of the initial design are inherited by the tool design. For outer body panels, even the change in curvature, the second geometrical derivative, needs to be taken into account during the design.

When the tools need to be modified by the tooling CAD department for springback compensation, it is a challenge to maintain the tight geometrical tolerances. Even when the compensation is subtle, the behavior of the surfaces might be unpredictable during modification, and there is no established way to retain smoothness constraints between surfaces in today's CAD systems. Manual modification of the tool surfaces for springback compensation is therefore a demanding and time-consuming task. In some cases, the accuracy of the compensation is compromised when subtle or detailed shape changes cannot be applied to the tool geometry correctly.

In the previous chapter, a compensation algorithm was developed for tool-meshes, using a continuous compensation function. The goal of the current chapter is to find a way to automatically apply this function to CAD geometries as well. Firstly, the different types of CAD geometries are discussed in section 4.2. Depending on the application of the product, the geometry is modeled as a so-called *Class A* or *B* geometry and different compensation strategies are required. Such a CAD compensation strategy has been developed and it is introduced in section 4.3. The basic principle will be explained, as well as the addition of simple and complex boundary conditions. The method is demonstrated for various academic problems. As the method is treated mainly from a mathematical point of view, the final section shows the possibilities for further exploration and implementation in an industrial context.

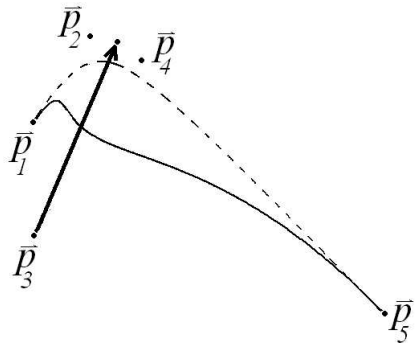


Figure 4.1: Non-local control over a Bézier curve

4.2 Surface qualification and quality

Before the modification strategy can be introduced, two questions need to be answered:

- What is a high-quality geometry?
- How are CAD geometries created?

The two main criteria for a CAD geometry are the correct representation of the object and the editability of the geometry. One would expect the representation of the geometry to be always completely exact since it exists in a 'perfect' mathematical world. In practice, the shape of the surface and the transition from one surface into another do not *exactly* satisfy desired conditions: They are fulfilled within a certain tolerance only.

The reason for this is the second requirement: editability. To exactly apply to all boundary conditions, extremely flexible surfaces would be required. With increasing flexibility, the surface functions tend to become unstable, similar to the B-spline volume used in the SDA algorithm. This means that while the boundary conditions are satisfied, the surface may exhibit unwanted bulges. In manual modifications, the surface also becomes very sensitive to small parameter changes, making it hard to manipulate the shape of the surface. As a simple example, a parametric curve, based on the same (Bézier) principle is modified in Figure 4.1. Even though a parameter is changed on the left side of the curve, the shape of the entire curve changes: The effect of parameter changes is non-local.

For various types of products the tolerances and editability requirements are different as well, leading to specific challenges during shape modification. In the automotive industry, product geometries are therefore differentiated in two categories: *Class A* and *Class B* geometries. Note that in this chapter, the focus is on surface modeling, as this is the most frequently used method in die-design.

4.2.1 Class A surfaces

Products that have aesthetic requirements, for example body and interior panels, are modeled with class A geometries. The term Class A means that unwanted changes in curvature are not allowed: They cause erroneous light reflection lines that distort

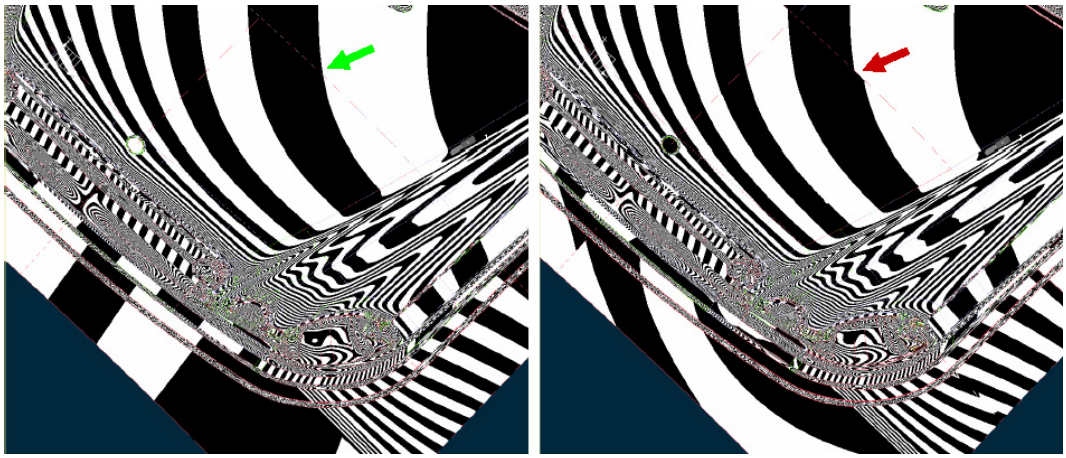


Figure 4.2: Unwanted kink in reflection line after surface modifications

the visual perception of the painted product. These reflection lines, and therefore the C^0, and especially C^1 and C^2 boundary conditions, are the main creative objects for a car-designer. If the geometry is C^2 continuous everywhere, it is regarded as smooth. A C^2 discontinuity will produce a so-called 'feature line', a C^1 discontinuity a sharp kink. The human eye is extremely sensitive to surface smoothness changes: Measurement systems must have an accuracy in the order of 10 μm to detect visual defects [5]. In Figure 4.2, a small error was made in the C^1 continuity between the surfaces. The arrows clearly show the discontinuity in the (computer generated) light reflection pattern [28, 4].

In order to develop a CAD model for the outer body panels, a clay model is manually sculpted by the designers first. When the desired shape has been obtained, the clay model is measured by making a 3D scan, resulting in a point cloud. Then so-called *free-form* surfaces are fitted manually onto the point-cloud. Automatic surface fitting algorithms do exist, however, due to the complexity of the geometry and the instability of the surface functions the results are generally not satisfactory. Even manually improving the resulting surfaces can be problematic because of the large amount and suboptimal parametrization of the fitted surfaces [50]. Instead, a much coarser fitting is done. The resulting surfaces are sufficiently manageable to be manually reshaped by specialist designers. During that process, these specialists increase surface complexity and the topology of the geometry when required. As will be shown in the following sections, the shape of freeform surfaces is defined by manipulating the so-called *control-points* in cartesian space.

Compensating tool geometries based on Class A surfaces
Class A surfaces form the basis for the design of the forming tools. This means that the strict tolerances for the part are transferred to the tool, and that these tolerances need to be held when the tool is compensated. As the tool surface follows changes in the control points, an intuitive solution would be to simply compensate the location of these points with the SDA method. The problem here is, that the control points are generally not located on the surface itself (as shown for the curve in Figure 4.1). Depending on the surface basis-functions, they can be far away. In the previous chapter, it was shown that the compensation function becomes inaccurate at a larger distance from the tool surface. More importantly, the surface behavior

can be unpredictable with respect to control-point changes, leading to inaccurate results and undesirable transition errors between surfaces. Therefore another compensation principle has been developed, taking into account these transitions. This will be explained in section 4.3.

4.2.2 Class B surfaces

Contrary to aesthetic parts, the creation and modification of functional products is mainly carried out by using *feature-based* CAD. Instead of manually 'sculpting' the free-form surfaces with control-points, geometries are created by using simple sketches, which are then applied in procedures that resemble real production techniques, like cutting or extruding. This is a powerful way of modeling and it allows easy changes in constructive parameters, such as the dimensions of a drilled hole. The relationships between the parameters of the geometry remain and the surfaces are automatically regenerated after the change. Therefore, the geometry is characterized as 'intelligent' [48].

Compensating tool geometries based on Class B surfaces
For class B geometries, curvature boundary conditions do not play an important role and the shape of the geometry is in most cases more straightforward. This makes the compensation of these geometries easier. Two strategies can be applied. The simplest way is to convert all features into free-form surfaces. These surfaces can then be modified in the same way as class A geometries. The major disadvantage of this method is that the geometry's intelligence is lost. Also, handling the converted geometry is not a trivial problem. Firstly, continuity boundary conditions between the generated surfaces have to be defined or deducted. Secondly, todays converter algorithms tend to produce many trimmed surfaces. These are surface partitions that are mathematically easy to create, but very hard to handle afterwards. This will be discussed in greater detail in the following sections.

Another way is to try and compensate the structure while retaining the feature definition. This limits the accuracy of the compensation because only a limited set of geometries can be generated with the model-parameters. For example: a circular hole in the part may become oval due to the compensation. However, when the hole was modeled as a circular *feature*, it will be restricted to remain circular. The geometrical limitations and the calculation difficulties are so large and unlikely to be solved, that this system was not investigated further.

Therefore, intelligent CAD geometries will be converted into standard free-form descriptions, as proposed earlier. To cater for the difficult structure of these converted geometries a strong focus will be on the compensation of the 'trimmed surfaces' as well.

4.3 Global surface modification algorithms

The springback compensation of tool surfaces is an application with specific requirements. The shape modifications are global and not very big. However, the accuracy needs to be very high and the transitions between the surfaces must be taken into

consideration carefully. In addition, the compensation must be carried out automatically. In contrast, most global modification algorithms, such as the Free Form Deformation (FFD) method [65, 16, 66] are intended for manual modification. Another interesting way to modify freeform surfaces is presented in [50]. Here, complex surface geometries are defined by a set of curves and boundary conditions which can be changed easily, but again, this was developed for manual modifications only.

Surface fitting techniques, also known as Reverse Engineering (RE) are the basis of the surface compensation algorithm that was developed. Instead of using the method for initial surface fitting [63], the surfaces are modified using surface *re*-fitting. Because a (correct) reference tool-geometry is already present as a starting point, curvature constraints between surfaces that are not usually applied in fitting procedures, can be taken into account.

4.3.1 Surface compensation principle

A new method for springback compensation has been developed for FE meshes, as shown in the previous chapter. The method is to apply a 3D vector function $\Psi(\vec{x})$ to the location of each node $\vec{c}_i$ in the mesh. As a result, the node is moved to a new location $\vec{c}_i'$.

$$\vec{c}_i' = \vec{c}_i + \Psi(\vec{c}_i) \quad 0 < i < n \tag{4.1}$$

Here, n is the number of nodes. A point on a surface can be described with a continuous function $\vec{S}$ with two location parameters. In CAD mathematics $u \in [0,1]$ and $v \in [0,1]$ are generally used. The shape of the surface is controlled by a set of shape parameters $\vec{\mathbf{p}} = [\vec{p}_1, \vec{p}_2, ..., \vec{p}_q]^T$. The shape parameters are defined as points in Cartesian space and are generally called *control points*.

$$S(u,v) = S(\vec{p}_1, \vec{p}_2, ..., \vec{p}_q, u, v) = S(\vec{\mathbf{p}}, u, v) \tag{4.2}$$

The challenge is to apply the compensation function to the entire surface. As a first approach, only one so-called sampling point $\vec{d}_i$ with parameters u_i and v_i on the surface is taken into consideration.

$$\vec{d}_i(u_i, v_i) = S(\vec{\mathbf{p}}, u_i, v_i) \tag{4.3}$$

Two initial requirements are:

1. The mathematical structure of the initial and compensated surfaces is identical. This means that the surface basis functions remain unchanged.
2. The compensation function maps the initial sampling point $\vec{d}_i(u_i, v_i)$ to the point with identical parameters u_i and v_i on the modified surface.

The main idea is to find an optimal set of control points $\vec{\mathbf{p}}'$ for the compensated surface, so that the point $\vec{d}_i$ is compensated accurately.

$$\begin{gathered} S(\vec{\mathbf{p}}') = \vec{d}_i + \Psi(\vec{d}_i) = \vec{d'}_i \\ \Leftrightarrow \\ S(\vec{\mathbf{p}}', u_i, v_i) \simeq S(\vec{\mathbf{p}}, u_i, v_i) + \Psi(S(\vec{\mathbf{p}}, u_i, v_i)) \ \forall u_i, v_i \end{gathered} \tag{4.4}$$

The equation is presented visually in Figure 4.3. It is important to note its ap-

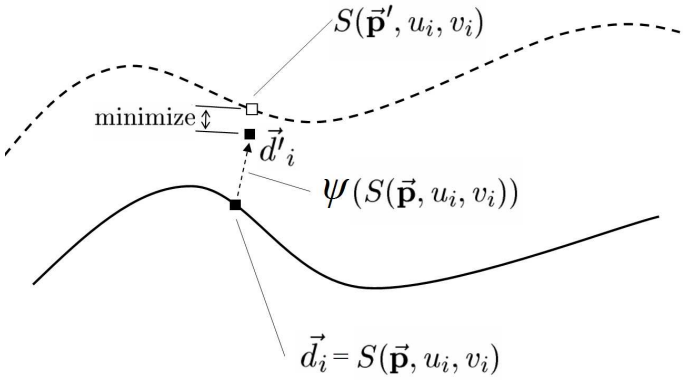


Figure 4.3: Re-fitting principle

proximative nature: It is possible to find an optimal solution for $\vec{\mathbf{p}}'$ but this is not necessarily exact because of the limited flexibility of the surface function $\vec{S}$. Similarly to the fitting of the compensation function, the control points can by found in the following way:

$$\min_{\vec{\mathbf{p}}'} \quad \left(\frac{1}{2}|S(\vec{\mathbf{p}}, u_i, v_i) + \Psi(S(\vec{\mathbf{p}}, u_i, v_i)) - S(\vec{\mathbf{p}}', u_i, v_i)|^2\right) \tag{4.5}$$

The surface function S can be formulated as a linear combination of j basis functions $N_i(u, v)$, using the vector with control points $\vec{\mathbf{p}}$.

$$S(\vec{\mathbf{p}}, u_i, v_i) = \sum_j N_j(u_i, v_i)\vec{p}_j = \mathbf{n}^T\vec{\mathbf{p}} \tag{4.6}$$

So, Equation (4.5) can be rewritten as

$$\min_{\vec{\mathbf{p}}'} \quad \frac{1}{2}(\mathbf{n}^T\vec{\mathbf{p}} + \Psi(\vec{d_i}) - \mathbf{n}^T\vec{\mathbf{p}}')^2 \tag{4.7}$$

The problem can be simplified as:

$$\min_{\vec{\mathbf{p}}'} \quad \frac{1}{2}(\vec{d'} - \mathbf{n}^T\vec{\mathbf{p}}')^2 \tag{4.8}$$

The problem can be solved by differentiating to $\vec{\mathbf{p}}'$, leading to:

$$(\vec{d'} - \mathbf{n}^T\vec{\mathbf{p}}')^T(-\mathbf{n}^T) = \vec{0} \tag{4.9}$$

$$-\mathbf{n}\vec{d'} + \mathbf{n}\mathbf{n}^T\vec{\mathbf{p}}' = \vec{0} \tag{4.10}$$

with $\mathbf{K} = \mathbf{n}\mathbf{n}^T$ and $\vec{\mathbf{f}} = \mathbf{n}\vec{d'}$ this can be written more conveniently as:

$$\mathbf{K}\vec{\mathbf{p}}' = \vec{\mathbf{f}} \tag{4.11}$$

Note that, as the arrow indicates, $\vec{\mathbf{p}}'$ and $\vec{\mathbf{q}}$ are vectors of (3D) points so Equation (4.11) represents 3 (independent) equations. These equations can be solved if the matrix $\mathbf{K}$ is positive definite. When only one sampling point on the surface is taken into consideration, this will not be the case. Instead, the surface parameters need

to be determined so that the geometrical error over the *entire* compensated surface becomes minimal. This can be achieved by using a surface integral:

$$\min_{\vec{\mathbf{p}}'} \quad \frac{1}{2}\left(\int_A \left|S(\vec{\mathbf{p}},u,v)+\Psi(S(\vec{\mathbf{p}},u,v))-S(\vec{\mathbf{p}}',u,v)\right| \mathrm{d}A\right)^2 \tag{4.12}$$

Before this equation can be solved, the proposed surface function (4.6) needs to be defined. The Bezier surface [64] is the best-known parametric surface.

$$S(u,v)=\sum_{i=0}^{m}\sum_{j=0}^{n}\vec{p}_{ij}B_{i,m}(u)B_{j,n}(v)=\mathbf{b}^T(u)\vec{\mathbf{P}}\mathbf{b}(v) \tag{4.13}$$

Unlike the previously presented vector equation, $\vec{\mathbf{P}}$ is a matrix with control points in this equation and $B_{i,m}$ is the i-th Bernstein Polynomial of degree m. The Bernstein polynomials are defined as follows:

$$B_{i,n}(u)=\frac{n!}{i!(n-i)!}u^i(1-u)^{n-i} \tag{4.14}$$

Instead of the Bernstein-polynomials, also the more complex B-spline or Non-Uniform Rational B-Spline (NURBS) basis functions can be used. They allow more complex shapes, but since the basic mathematical structure is identical to the Bezier-surface, they are not taken into consideration in this chapter. In Equation (4.13), the control points are organized in a matrix $\vec{\mathbf{P}}$ and the basis functions in vectors $\mathbf{b}(u)$ and $\mathbf{b}(v)$. For example, a quadratic by quadratic surface is defined by the following equation:

$$S(u,v)=\begin{bmatrix}(1-u)^2\\ 2u(1-u)\\ u^2\end{bmatrix}^T\begin{bmatrix}\vec{p}_{00} & \cdots & \vec{p}_{0n}\\ \vdots & \ddots & \vdots\\ \vec{p}_{m0} & \cdots & \vec{p}_{mn}\end{bmatrix}\begin{bmatrix}(1-v)^2\\ 2v(1-v)\\ v^2\end{bmatrix} \tag{4.15}$$

Even though this matrix equation is the clearest formulation, it is sometimes more convenient to write the surface function as a vector dot-product, as proposed in Equation (4.6). The following system of functions is equivalent:

$$S(u,v)=\sum_{i=0}^{m}\sum_{j=0}^{n}\vec{p}_{ij}B_{i,m}(u)B_{j,n}(v)=\sum_k N_k(u,v)\bar{p}_k=\mathbf{n}(u,v)\vec{\mathbf{p}} \tag{4.16}$$

Vector $\vec{\mathbf{p}}$ contains the control points and $\mathbf{n}(u,v)$ the basis-functions, each consisting of two Bernstein polynomials.

$$N\begin{cases} N_k(u,v)=B_{i,m}(u)B_{j,n}(v)\\ k=i\cdot n+j\\ k,n,m\in\mathbb{N}\\ 0<k<n\cdot m\\ 0<j<n\end{cases} \tag{4.17}$$

As an example, this change of notation is shown for a linear by linear surface, shown in Figure 4.4.

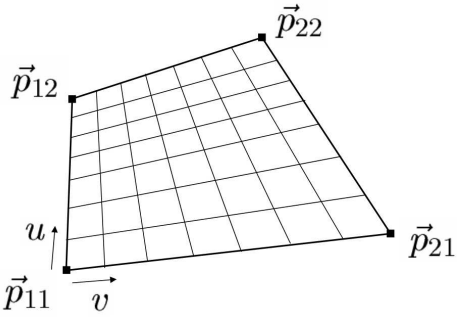


Figure 4.4: A linear by linear Bezier surface

$$
\begin{aligned}
S(u,v) &= \begin{bmatrix} B_{1,2}(u) \\ B_{2,2}(u) \end{bmatrix}^T \begin{bmatrix} \vec{p}_{11} & \vec{p}_{12} \\ \vec{p}_{21} & \vec{p}_{22} \end{bmatrix} \begin{bmatrix} B_{1,2}(v) \\ B_{2,2}(v) \end{bmatrix} \\
&= B_{1,2}(u)\vec{p}_{11}B_{1,2}(v) + B_{1,2}(u)\vec{p}_{12}B_{2,2}(v) \\
&\quad + B_{2,2}(u)\vec{p}_{21}B_{1,2}(v) + B_{2,2}(u)\vec{p}_{22}B_{2,2}(v) \\
&= \begin{bmatrix} B_{1,2}(u)B_{1,2}(v) \\ B_{1,2}(u)B_{2,2}(v) \\ B_{2,2}(u)B_{1,2}(v) \\ B_{2,2}(u)B_{2,2}(v) \end{bmatrix} \begin{bmatrix} \vec{p}_{11} \\ \vec{p}_{12} \\ \vec{p}_{21} \\ \vec{p}_{22} \end{bmatrix} \\
&= \mathbf{n}^T(u,v)\vec{\mathbf{p}}
\end{aligned}
\tag{4.18}
$$

In the following part, Equation (4.12) is solved numerically, because a closed-form solution cannot be found in the general case. Instead of just one sampling point $\vec{\mathbf{d}}'$ a matrix or grid of so-called sampling points $\vec{\mathbf{D}}$ is chosen. The coordinates of the sampling points are then modified with the compensation function Ψ, resulting in a matrix $\vec{\mathbf{D}}'$, and the surface is refitted to the modified sampling points. It is most convenient to choose a rectangular $r \times s$ grid of sampling points:

$$
\vec{\mathbf{D}} = \begin{bmatrix} \vec{d}_{11} & \dots & \vec{d}_{1s} \\ \vdots & \ddots & \vdots \\ \vec{d}_{r1} & \dots & \vec{d}_{rs} \end{bmatrix} \tag{4.19}
$$

An equispaced grid was chosen for u and v:

$$
\vec{\mathbf{D}} = \begin{bmatrix} S(u=0,v=0) & \dots & S(1,0) \\ \vdots & \ddots & \vdots \\ S(0,1) & \dots & S(1,1) \end{bmatrix} \tag{4.20}
$$

It is more convenient to order the sampling-points $\vec{\mathbf{D}}$ and modified sampling points $\vec{\mathbf{D}}'$ in the vectors $\vec{\mathbf{d}}$ and $\vec{\mathbf{d}}'$ with a length of $t = r \cdot s$ instead of the $r \times s$ matrices.

Equation (4.12) is now discretized for all t sampling points in the following way: A vector $\vec{\mathbf{p}}'$ is sought that minimizes the function

$$
\frac{1}{2}\left(\sum_{k=1}^{t}\left|\vec{d}'_k - S(\vec{\mathbf{p}}', u_k, v_k)\right|\Delta A_k\right)^2 \tag{4.21}
$$

with u_k and v_k as surface parameters for the k-th sampling point. This function is a weighted summation of Equation (4.8). It is not easy to provide a straightforward calculation for the weight ΔA_k, which should describe the area around each sampling point $\vec{d}_k$, similar to the Voronoi-area around the nodes in the previous chapter. Instead, it is assumed that the grid is equispaced, and ΔA_k can be set at the value of 1 for all sampling points. Equation (4.21) is again rewritten in matrix-form:

$$\mathbf{K}\vec{\mathbf{p}}' = \vec{\mathbf{f}} \tag{4.22}$$

The (ij) K-matrix value is calculated as follows:

$$K_{ij} = \sum_{k=0}^{t} N_i(u_k, v_k) N_j(u_k, v_k) \tag{4.23}$$

The (i)-th $\mathbf{f}$-vector value is calculated as follows:

$$\vec{f_i} = \sum_{k=0}^{t} N_i(u_k, v_k) \vec{d'}_k \Delta A_k \tag{4.24}$$

For a more detailed introduction in fitting and RE problems, the reader is referred to the standard work [42].

4.3.2 Transitions between surfaces

As discussed in the introduction, the continuity tolerances are important parameters for forming tool surfaces. In this section it is shown how these transitions can be defined mathematically, so that they can be included in the previously discussed framework. Consider two Bezier surfaces $S = \mathbf{b}^T(u_S)\vec{\mathbf{P}}\mathbf{b}(v_S)$ and $R = \mathbf{b}^T(u_R)\vec{\mathbf{Q}}\mathbf{b}(v_R)$. Note that, for convenience, the matrix-equation is used here. In the simplest case, R and S are connected C^0 continuously. If the number of control points in v-direction is identical the following equation can be solved:

$$\begin{aligned} S(1, v_S) &= R(0, v_R) \\ \mathbf{b}^T(1)\vec{\mathbf{P}}\mathbf{b}(v) &= \mathbf{b}^T(0)\vec{\mathbf{Q}}\mathbf{b}(v) \ \forall v \in [0, 1] \end{aligned} \tag{4.25}$$

With Equation (4.14) it can easily be shown that

$$B_{i,m}(0) = \begin{cases} 1 & \text{if} \quad i = 0 \\ 0 & \text{if} \quad i > 0 \end{cases} \tag{4.26}$$

and

$$B_{i,m}(1) = \begin{cases} 1 & \text{if} \quad i = m \\ 0 & \text{if} \quad i < m \end{cases} \tag{4.27}$$

Therefore, Equation (4.25) can be rewritten as follows (note that the Bernstein polynomials in v-direction are identical when both surfaces have the same degree in v direction and $v_S = v_R$ on the edge):

$$\begin{bmatrix} 0 \\ \vdots \\ 1 \end{bmatrix}^T \begin{bmatrix} \vec{p}_{00} & \dots & \vec{p}_{0n} \\ \vdots & \ddots & \vdots \\ \vec{p}_{m0} & \dots & \vec{p}_{mn} \end{bmatrix} = \begin{bmatrix} 1 \\ \vdots \\ 0 \end{bmatrix}^T \begin{bmatrix} \vec{q}_{00} & \dots & \vec{q}_{0n} \\ \vdots & \ddots & \vdots \\ \vec{q}_{m0} & \dots & \vec{q}_{mn} \end{bmatrix} \tag{4.28}$$

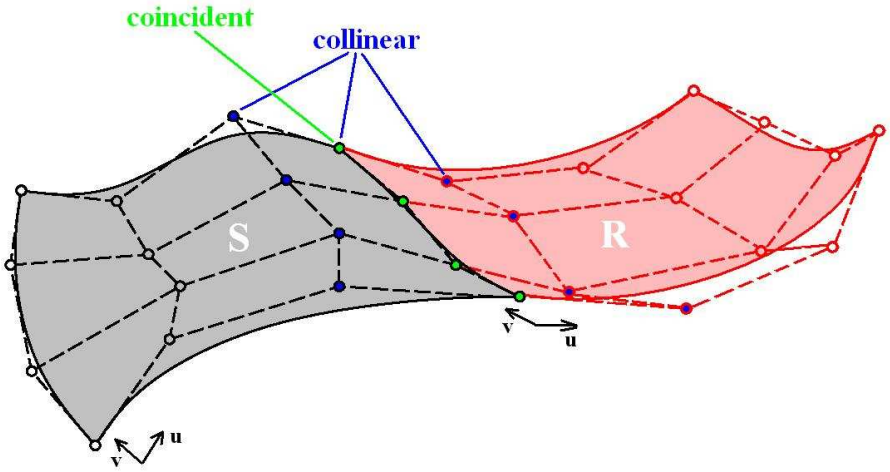


Figure 4.5: A C^1 connection between 2 Bezier surfaces

leading to

$$\begin{bmatrix} \vec{p}_{m0} \\ \vdots \\ \vec{p}_{mn} \end{bmatrix} = \begin{bmatrix} \vec{q}_{00} \\ \vdots \\ \vec{q}_{0n} \end{bmatrix} \tag{4.29}$$

This simply means that the control points need to be identical on the shared edge of the surfaces. These control points are shown in green in Figure 4.5. These control points lie on the 'zeroth' row from the edge. For C^1 continuity, the following equation needs to be solved:

$$\frac{\partial}{\partial u} S(u_S, v_S)\Big|_{u=1} = \frac{\partial}{\partial u} R(u_R, v_R)\Big|_{u=0} \Leftrightarrow \mathbf{b'}^T(1)\vec{\mathbf{P}}\mathbf{b}(v_S) = \mathbf{b'}^T(0)\vec{\mathbf{Q}}\mathbf{b}(v_R) \tag{4.30}$$

This laborious calculation can be found in [22]. It can be shown that the so-called r-th edge derivative depends only on the r-th row of control points. The surfaces in Figure 4.5 are connected C^1 continuously. This means that the control points on the zeroth (pictured in green) and first (pictured in blue) rows of both surfaces are dependent. It can be proven that each pair of control points, as shown in the picture, needs to be collinear and equally spaced to ensure C^1 continuity [53].

4.3.3 Multiple surfaces with simple boundary conditions

The modification of a set of Bezier surfaces is carried out in principally the same way as the modification of a single surface. However, instead of solving Equation (4.22) for each surface independently, all surface control-points and basis functions are added to one large matrix equation. A similar principle was applied in [58]. Each control point is assigned a unique number k, $k \in \mathbb{N}$ and $k \in [1, k_{tot}]$. k_{tot} is the total amount of parameters(and size of the K matrix). A point on one of the surfaces in the geometry S can be calculated with the following vector equation:

$$S(u^1, v^1, u^2, v^2, ...) = \mathbf{n}^T\vec{\mathbf{p}} = \begin{bmatrix} \mathbf{n}^1(u^1, v^1) \\ \mathbf{n}^2(u^2, v^2) \\ \cdots \\ \cdots \end{bmatrix}^T \begin{bmatrix} \vec{\mathbf{p}}^1 \\ \vec{\mathbf{p}}^2 \\ \cdots \\ \cdots \end{bmatrix} \tag{4.31}$$

In this function, $\mathbf{n}^1(u^1, v^1)$ represents the basis functions of the first surface. $\vec{\mathbf{p}}^1$ is a vector with the control points of this surface. It is now possible to imply boundary conditions between the control points of different surfaces. It has been shown in the previous section that two surfaces are connected C^0 continuously when the control points on the shared edge of the surfaces are identical. With a *Lagrange multiplier* or the *penalty method*, an optimal set of surface parameters (for all surfaces) can be calculated while maintaining the boundary condition that the shared control points have the same location.

To the author's knowledge, permanent continuity boundary conditions are not available in (free-form) surface modeling software such as ICEM-Surf and CATIA. The reason for this is that during modeling, large shape modifications are carried out, and each added boundary condition makes the surface model harder to handle:

- The geometry's reaction to changes of control points may become counter-intuitive.
- The system of equations that holds the boundary conditions may even become unsolvable for large modifications.

The first problem is not really important for the compensation method: User interaction is not required, the control point locations are calculated by the algorithm. The modifications that are carried out during compensation are generally small and smooth, however, to ensure that a solution will always be found, the approximative penalty method is used to avoid the second problem.

In the following part, the penalty method is applied in the context of the surface refitting problem. When one boundary condition is applied, the α-th control point (in the system's vector of control points) has the same location as the β-th control point, the following system must be solved (note that the prime from Equation (4.22) is omitted in the formulas for convenience):

$$\begin{cases} \mathbf{K}\vec{\mathbf{p}} = \vec{\mathbf{f}} \\ \vec{p}_\alpha = \vec{p}_\beta \end{cases} \tag{4.32}$$

The following analysis is a variation on the calculation in [33], pages 194-197. The reader is referred to this book for a more extensive explanation.

Firstly, the boundary condition is written in vector-form

$$\mathbf{l}^T\vec{\mathbf{p}} = 0 \tag{4.33}$$

In this example the $\mathbf{l}$-vector looks like this:

$$\mathbf{l} = [0, \ldots, 0, l_\alpha = -1, 0, \ldots, 0, l_\beta = 1, 0, \ldots, 0]^T \tag{4.34}$$

Then a potential Π is defined:

$$\Pi(\mathbf{p}) = \frac{1}{2}\vec{\mathbf{p}}^T\mathbf{K}\vec{\mathbf{p}} - \vec{\mathbf{p}}^T\vec{\mathbf{f}} + \frac{k}{2}(\mathbf{l}^T\vec{\mathbf{p}})^2 \tag{4.35}$$

The parameter k is the penalty constant, a scalar with a large value. The vector $\vec{\mathbf{p}}$ that minimizes this function is an approximate solution of the constrained problem

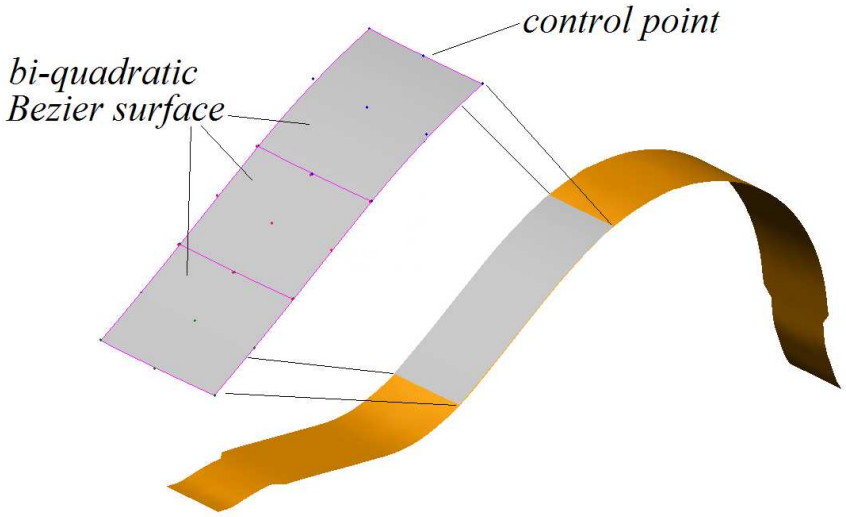


Figure 4.6: Compensation of component #1

[33].

$$\begin{aligned} 0 = & \quad \frac{\mathrm{d}\Pi}{\mathrm{d}\mathbf{p}} \\ 0 = & \quad \mathbf{K}\mathbf{p} - \mathbf{f} + k\mathbf{l}\mathbf{l}^T\mathbf{u} \end{aligned} \tag{4.36}$$

with $\mathbf{M} = k\mathbf{l}\mathbf{l}^T$ and $\bar{\mathbf{f}} = \mathbf{F} + kd\mathbf{l}$ this can be written as:

$$(\mathbf{K} + k\mathbf{M})\vec{\mathbf{p}} = \vec{\mathbf{f}} \tag{4.37}$$

The same analysis is also possible for a set of n boundary conditions. In this case $\mathbf{l}$ becomes a $k_{tot} \times n$ matrix.

The mathematics were implemented in a C++ program called *SurfaceSDA*. Component #1, the VW suspension part shown in the previous chapter, was used to test the algorithm. Three bi-quadratic surfaces from the part's CAD file, shown in Figure 4.6 are compensated with SurfaceSDA.

To check the accuracy of the surface compensation, the three surfaces were also finely discretized into a mesh and compensated with the regular SDA algorithm. When the mesh of the surface geometry becomes very fine, the exact solution of the surface modification is approached, so this can be used as a reference compensation. The distance between both geometries is visualized in Figure 4.7. The error for the SurfaceSDA algorithm is not large, it is less than 5% of the maximum amount of compensation (3mm). This small error is due to the fact that the quadratic by quadratic surfaces were not able to accurately 'follow' the compensation function. The solution for this is to elevate the polynomial degrees of the surfaces [53], or to use the piecewise polynomial (B-spline) surfaces to increase surface flexibility.

In Figure 4.8 the effect of constraints on the control points can be seen. As an experiment, the coordinates of one control point were changed in the original part #1 surfaces, so the smooth transition was lost (a). Then, a boundary condition was set between that control point and the corresponding control point on the next surface,

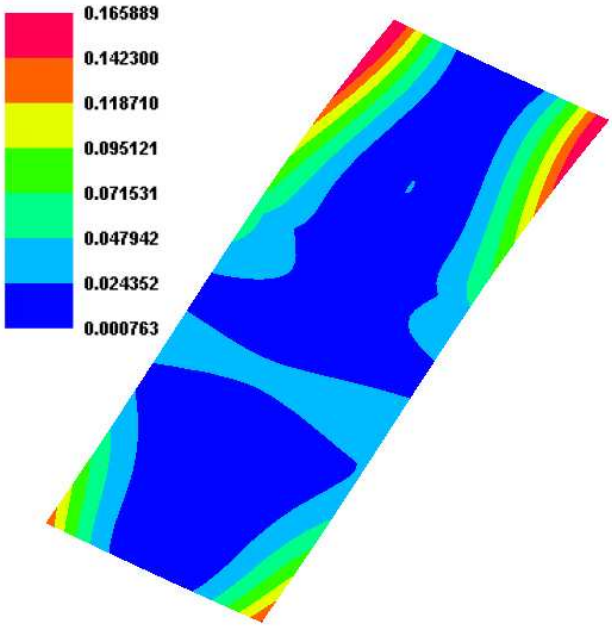


Figure 4.7: Distance between the compensated set of surfaces and a mesh of the surfaces (mm)

and the surfaces were compensated (b). As can be seen in the second picture, the surfaces are now coincident again at this point. However, because no boundary conditions were set for the other control points at the boundary, a gap appears. In (c) the middle control point was also constrained, delivering a smaller gap, and finally all three (d). Then, the transition is C^0 over the entire boundary again.

By repairing initial surface continuity errors, the compensation method actually improves the surface continuity instead of harming it, which is the case when the compensation function would simply be applied to the control points. However, since C^1 continuity was not constrained in the example, it is not maintained and a kink appears in the surface. Higher-order constraints will be treated in detail in the next sections. Even for more complex boundary conditions, the surface compensation strategy can handle minor defects in the initial geometry and resolve them.

4.3.4 Trimmed surfaces

NURBS surfaces are principally four-sided. In some cases it is inconvenient or impossible to model a geometry with these surfaces. Such a geometry is visualized in Figure 4.9. The flat bottom of the cup is created as a four sided surface first, and then the unwanted parts are trimmed off to make the surface fit to the sidewalls. The four sided surface is known as the basis patch P. The trim curve T is a Bezier or B-spline curve, a function of parameter w and a set of control points $\vec{\mathbf{p}}_T$ in the (u, v) space of P:

$$T(w) = \mathbf{n}^T(w)\vec{\mathbf{p}}_T \tag{4.38}$$

Therefore, $T(w)$ lies on the surface P by definition. A point t_i at location w_i on the trim-curve can be calculated by inserting T in P.

$$\vec{t_i} = P(T(w_i)) = \mathbf{n}^T(T(w_i))\vec{\mathbf{p}}_t \tag{4.39}$$

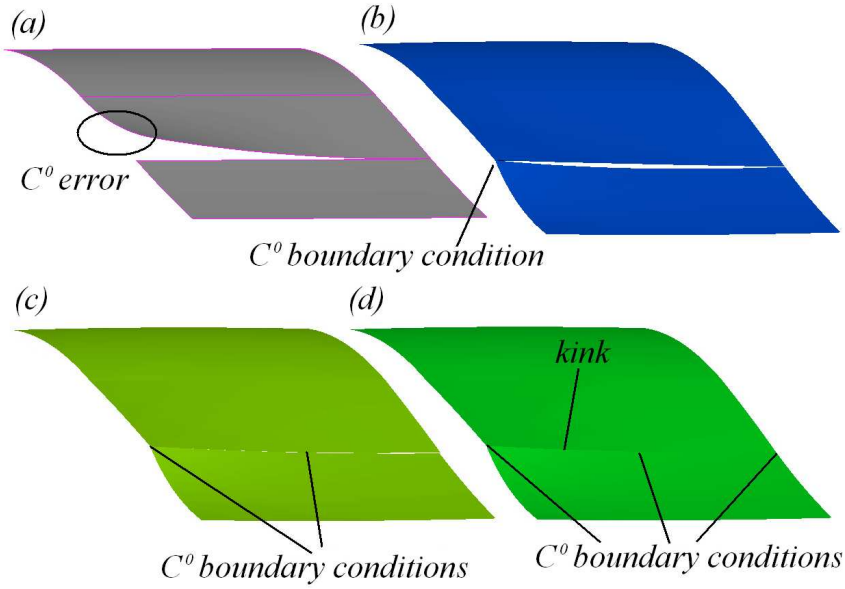


Figure 4.8: Surface continuity with different boundary conditions

Creating a trimmed surface is straightforward. Suppose that the trimmed surface is generated from the wall surfaces in the geometry shown in Figure 4.9. The intersection curve between the wall surface and the basis patch is calculated. This curve can be used directly as the trim curve for the trimmed surface. Note that finding the surface-surface intersection curve is a job in itself [17] and in most cases an approximate solution is used.

Trimmed surfaces commonly occur in surface models that were converted from a feature-based model. It is problematic to include these surfaces in the surface compensation algorithm, since the control point rules do not apply to hold boundary conditions between such a surface and another (trimmed or regular) surface. C^0 continuity is already generally hard to impose, and according to [22](p. 244) it is impossible to exactly impose C^1 boundary conditions. Another common boundary condition problem is when the transition between two surfaces does not include the entire surface edge. Finally, the control point rules become complex for higher-order continuity requirements. In the following section a general, more abstract, way of imposing boundary conditions will be discussed to cater for these situations.

4.3.5 General continuity boundary conditions

In Figure 4.10 a set of surfaces is shown. Surfaces P, Q, R and S are 'regular' four-sided Bezier or B-spline surfaces, surface T is a trimmed surface on the basis patch B.

Three types of surface-surface transitions are identified:

- **Type-a** The transition p_4 between regular (i.e. not trimmed) surfaces P and R is a regular transition. The two surfaces share an entire edge, for which one

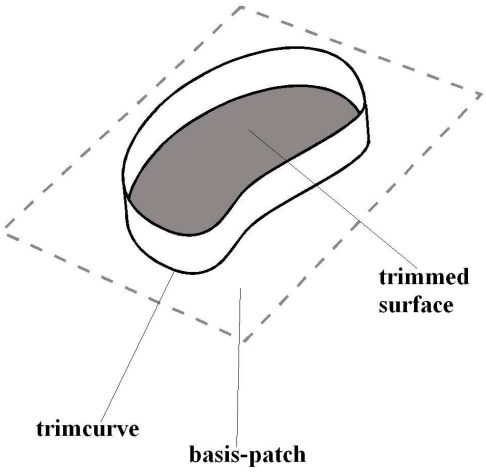


Figure 4.9: A trimmed surface

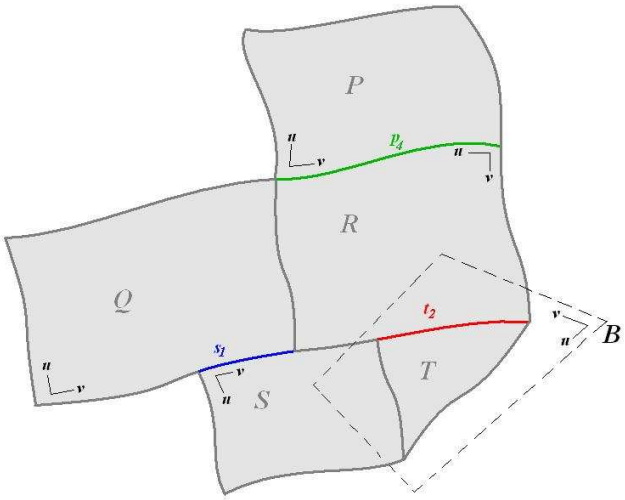


Figure 4.10: A set of surfaces and their boundary conditions

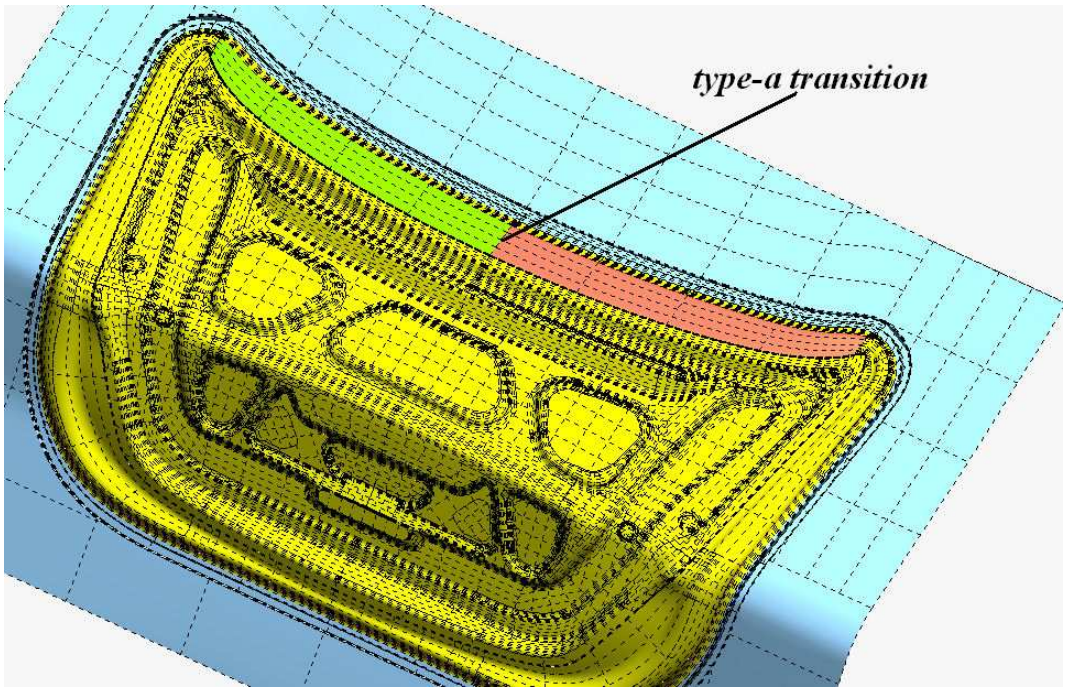


Figure 4.11: A type-a transition in component #2

parameter has a fixed value (in the example $u_p = 0$ and $v_r = 0$).

- **Type-b** The transition s_1 between the surfaces Q and S uses the surface boundaries only partially.

- **Type-c** The transition t_2 uses the edge of the regular surface R partially, and a part of the trimcurve of surface T. This trimcurve has no fixed u or v parameters.

In Figures 4.11, 4.12 and 4.13, the different transitions are shown on component #2, the trunk-lid inner panel.

In the general case, two surfaces K and L with control point vectors $\vec{\mathbf{p}}_\mathbf{K}$ and $\vec{\mathbf{p}}_\mathbf{L}$ coincide on a border curve, either on the inside of the surface, like in a trimmed surface, or on (a part of) the surface border. This curve has a definition in the parameter space of both surfaces. Therefore, the boundary curves are defined separately as $\gamma_K(w)$ and $\gamma_L(w)$, however, they are identical in cartesian space. Additionally the parameter w, along the curves, must map to points on each curve that also coincide in cartesian space.

A general boundary condition formula can be defined as follows: A C^{gh}-continuous connection between the surfaces R and S is achieved when the following equation is satisfied for the entire range of w:

$$\mathbf{n}_K^{(g)(h)T}(\gamma_K(w))\vec{\mathbf{p}}_K = \mathbf{n}_L^{(g)(h)T}(\gamma_L(w))\vec{\mathbf{p}}_L \;\forall\; w \tag{4.40}$$

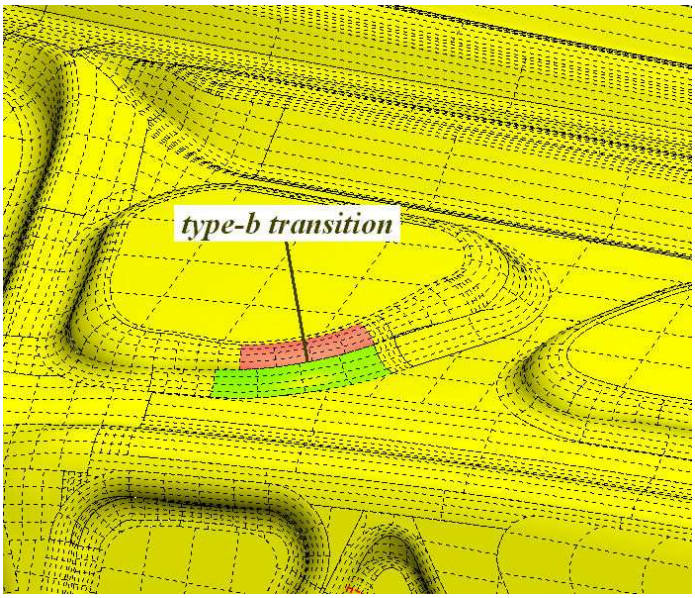


Figure 4.12: A type-b transition in component #2

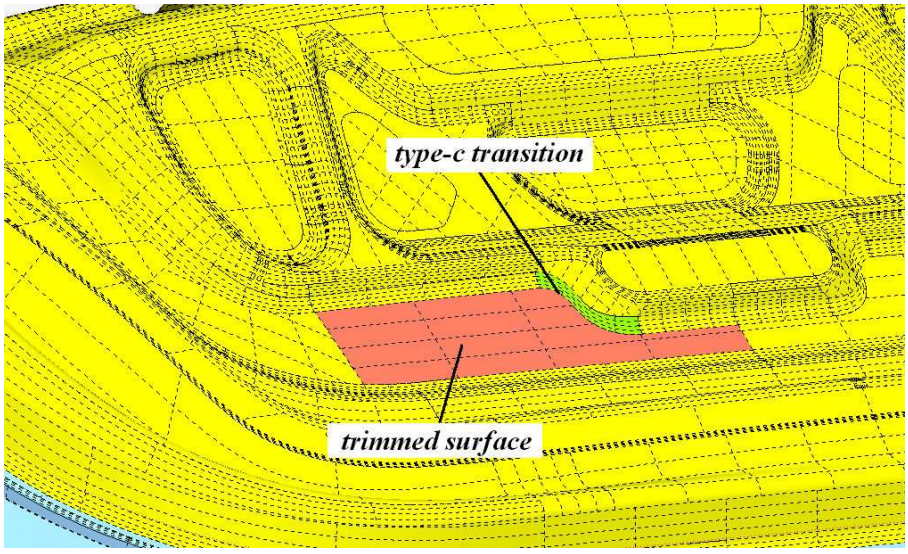


Figure 4.13: A type-c transition in component #2

The expression $\mathbf{n}^{(g)(h)}$ is a vector with the g-th derivative with respect to the u-parameter and the h-th derivative with respect to the v-parameter of the surface basis functions. In the following section, such a boundary condition is implemented in the surface compensation strategy.

Implementation of general boundary conditions in the framework
The boundary condition is added in the following steps

1. Identify boundary curves γ in the (u,v) space of both surfaces.
2. Establish a $w-$mapping between the two boundary curves.
3. Calculate and add the penalty condition to the main equation.

Identifying boundary curves
The compensation algorithm reads a set of surfaces, which initially need to be approximately C-0 connected. Even though it would be possible to automatically detect the type of transition and the degree of continuity between each pair of neighboring surfaces, at the moment this is required as user input.

In the following sections, the transition between two surfaces only is regarded. Therefore, two curves in the (u,v) space of each surface are required. The transition between surface P and R in Figure 4.10 is a type-a transition and the two curves are

$$\gamma_P(w) = \begin{bmatrix} 0 \\ v_P \end{bmatrix} \tag{4.41}$$

$$\gamma_R(w) = \begin{bmatrix} u_R \\ 0 \end{bmatrix} \tag{4.42}$$

Mapping boundary curves
Mapping the boundary curves means that both curves are written as a function of the parameter w in such a way that for each $w \in [0,1]$ both curves point to the same coordinate in cartesian space. In the example, the first curve is defined as:

$$\gamma_P(w) = \begin{bmatrix} 0 \\ w \end{bmatrix} \tag{4.43}$$

The second curve is also defined as a linear function with unknown a and b

$$\gamma_R(w) = \begin{bmatrix} aw + b \\ 0 \end{bmatrix} \tag{4.44}$$

These parameters can be found by solving

$$\mathbf{n}_P^T(\gamma_P(w))\vec{\mathbf{p_P}} = \mathbf{n}_R^T(\gamma_R(w))\vec{\mathbf{p_R}} \ \forall \ w \tag{4.45}$$

Suppose P and R have the same degree in u-direction. Then the solution is simple, namely $a = -1$ and $b = 1$ in this case.

In general, the solution to Equation (4.45) is not trivial and a closed form solution cannot be found. A generally applicable mapping solution will now be presented

and demonstrated for the type-c transition between surface T and R. The boundary curves for a trimmed parametric surface are generally defined as a parametric curve in (u, v) space. The following equation shows a Bezier boundary curve on surface T. Note that each control-point in the vector $\mathbf{p}_T$ is given in (u_T, v_T) parameters.

$$\gamma_T(w) = \mathbf{n}(w)\vec{\mathbf{p}}_T \tag{4.46}$$

Again, the goal is to reparameterize one of the curves so that Equation (4.45) is satisfied. In other words: the parameter w needs to map to the same point in (x, y, z)-space on both boundary curves. It is arbitrary which curve is reparameterized, in this case γ_R will be taken, and its parameter w will be replaced with function $s(w)$:

$$S_T(\gamma_T(w)) = S_R(\gamma_R(s(w))) \; \forall \; w \tag{4.47}$$

$s(w)$ is a scalar function that can be chosen freely, but it has to be monotonically increasing [53], p.241:

$$\frac{ds(w)}{dw} > 0 \tag{4.48}$$

In SurfaceSDA $s(w)$ is found in the following way:

1. define a set of n sampling points for w, for example
   $w_d = 0, 0.1, 0.2, 0.37, 0.52....1 \quad 0 < d < n$

2. calculate the spatial coordinates for each of the sampling points on the γ_T edge curve, $T(\gamma_T(w_d))$

3. find the same location on the R-edge curve. This means: find s_d numerically from the calculation

   $$\min_{s_d} \; \|\gamma_R(s_d) - T(\gamma_T(w_d))\|$$

4. the data points w_d, s_d form a relationship between the parameter of the edge curve T and the edge curve R. Fit or interpolate a function $s(w)$ through the (w^d, s^d) point set, that satisfies (4.48).

The process is visualized in Figure 4.14. In the C++ implementation a 'Golden Section' algorithm [56] is used in step 3. In step 4, the data points are simply used as a piecewise linear function. For the simple test cases this has turned out to be accurate enough. Note that in the piecewise function $s(w)$ still satisfies Equation (4.48).

Adding the boundary condition
With global control point numbering, as proposed in section 4.3.3, Equation (4.33) can be written for the entire system of surfaces as:

$$\mathbf{L}_R(w)\vec{\mathbf{p}} = \mathbf{L}_S(w)\vec{\mathbf{p}} \Leftrightarrow \mathbf{L}^T(w)\vec{\mathbf{p}} = 0 \tag{4.49}$$

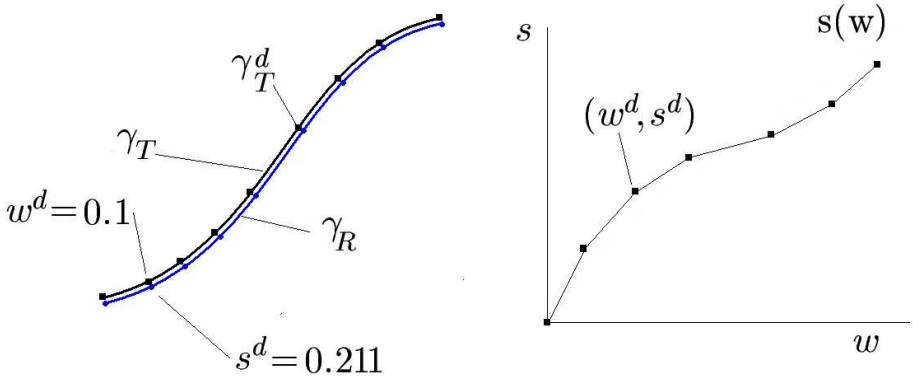


Figure 4.14: Discrete reparametrization

$$\mathbf{L}(w) = \begin{bmatrix} 0 \\ \vdots \\ \mathbf{n}_R^{(g)(h)}(\gamma_R(s(w))) \\ \vdots \\ -\mathbf{n}_T^{(g)(h)}(\gamma_T(w)) \\ \vdots \\ 0 \end{bmatrix} \tag{4.50}$$

with $\mathbf{L}^T(w) = \mathbf{L}_R(w) - \mathbf{L}_S(w)$. With the following integral, a matrix $\mathbf{M}$ is calculated so that the boundary condition is imposed on the entire edge-curve with penalty factor k:

$$\mathbf{M} = k \int_0^1 \mathbf{L}(w)\mathbf{L}^T(w)\mathrm{d}w \tag{4.51}$$

The integral for the matrix $\mathbf{M}$ (Equation (4.51)) is concocted component-wise. It can generally not be calculated in closed form, so numerical integration has to be carried out. In the compensation strategy, a simple trapezoidal integration scheme from [56] was used.

In general more than one surface-to-surface transition needs to be taken into account. To achieve this the different boundary condition matrices can simply be added:

$$\mathbf{M}_{\text{total}} = \mathbf{M}_1 + \mathbf{M}_2 + \dots \tag{4.52}$$

This matrix $\mathbf{M}_{\text{total}}$ is applied in the main system of equations, defined in Equation (4.37).

4.3.6 Results

The following list summarizes the surface compensation strategy that has been developed:

- The parametric surfaces that make up a geometry are represented by vector equations
- On the surfaces, a grid of sampling points is defined. These sampling points are compensated by the springback compensation function

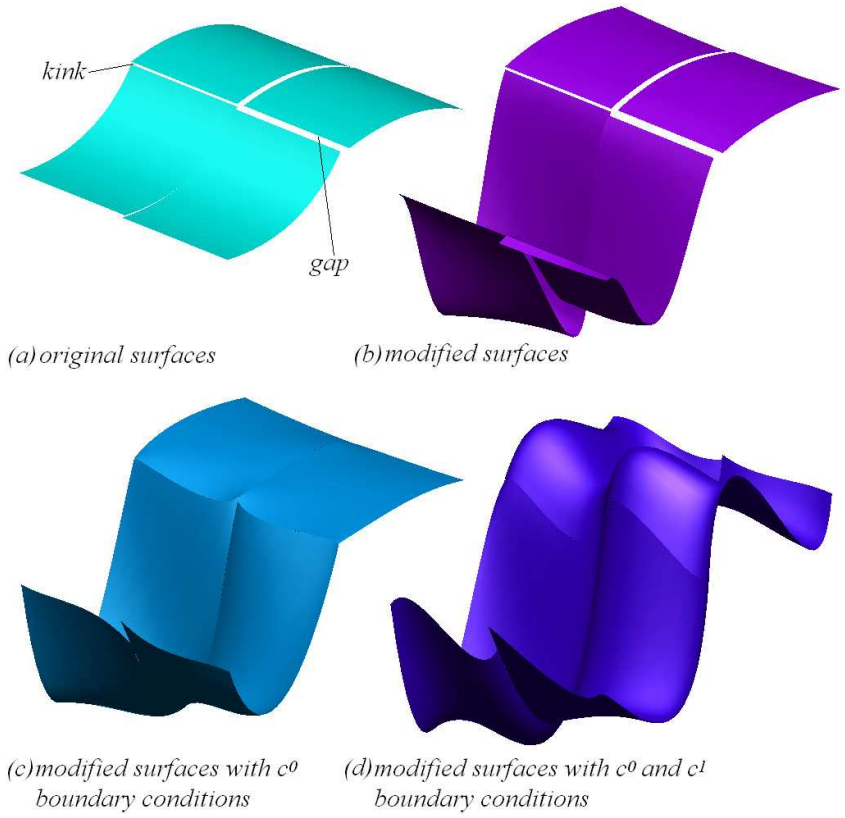


Figure 4.15: Modifying a set of four surfaces

- A system of equations is generated to re-fit the surfaces to the compensated sampling points
- Regular boundary conditions between surface control points are added to the system by using the penalty method
- General boundary conditions are added to the system, using boundary curve mapping, integration and the penalty method

The algorithm has been tested on various academic problems. In Figure 4.15 four cubic Bezier surfaces have been modified with various boundary conditions. As pointed out earlier, in real CAD geometries the transitions between various surfaces are generally not 100% accurate. Therefore, some kinks and gaps were included in the geometry, shown in Figure (a). A heavy compensation function was applied to the set of surfaces. When the surfaces are changed without boundary conditions, it can clearly be seen that the geometrical errors become larger (b).

When C^0 boundary conditions were included (c), the gaps disappeared after compensation. Note that the shape of the modified geometry is still very close to the unconstrained modified geometry (b). Finally, C^1 boundary conditions were also implied between the surfaces. It is clear that the boundary conditions are only satisfied approximately and the resulting geometry (d) deviates considerably from the unconstrained geometry. In this particular case, the effect is very large as the shape modification is very drastic and because the cubic Bezier surfaces have a very small number of control points, which limits the geometrical flexibility of the surface.

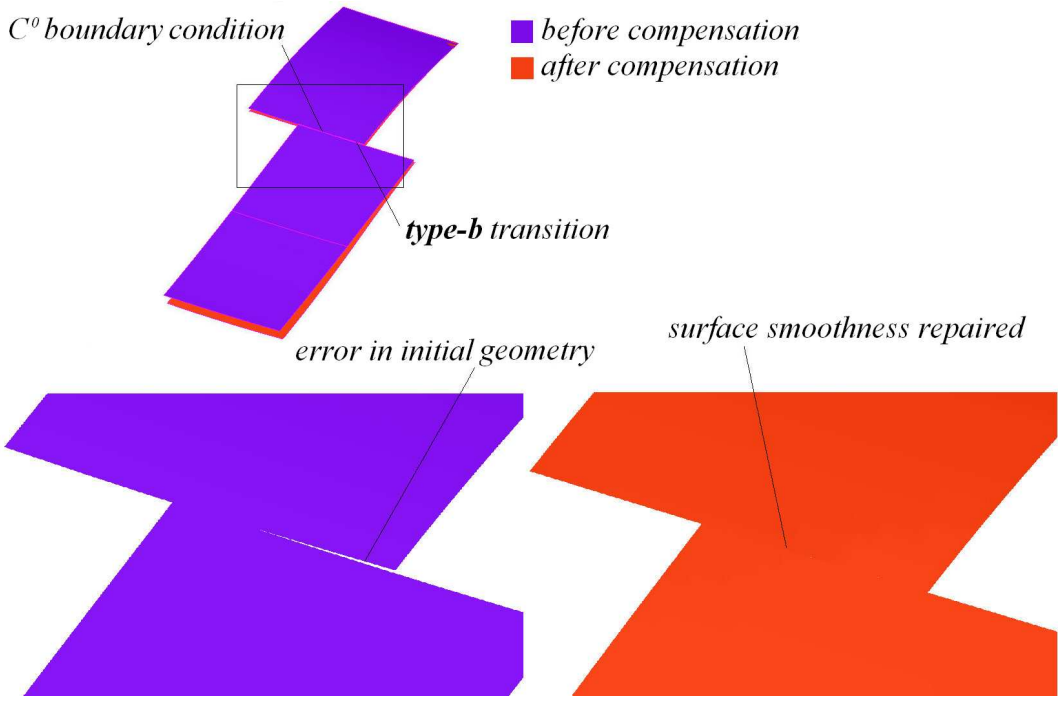


Figure 4.16: Modifying with a type-b transition

In any case, this example strongly demonstrates the advantage of using the penalty method. The boundary conditions are so severe that the geometry would be 'locked': It would be impossible to find a solution to the compensation problem. Here, the modified geometry can still be calculated and the user can decide to improve the results:

- by changing the penalty factor of each boundary condition
- by modifying the surface's definition, adding more control-points or surface sections

The surfaces become wavy in geometry (d), which is problematic for class A car body panels. Additionally, extra boundary conditions could be added to reduce this problem. It should be clear that in real cases, the surfaces have a much higher number of DOFs and the shape modification is not so drastic, so these problems will not be so severe for industrial geometries.

The boundary conditions are not imposed anymore with the control point rules, mentioned in Section 4.3.2. A type-b transition is shown in Figure 4.16. As an example problem, one of the surfaces of component #1 was shifted to create a partial transition. Coincidentally, this introduced C^0 and C^1 errors in the initial geometry. After compensation of the surfaces, this error was resolved.

4.4 Conclusion and future work

The third research hypothesis can be answered positively:

Hypothesis 3
Mesh-based shape modifications can automatically be transferred back to the tool CAD geometry.

In this chapter a framework has been developed for modifying tool surfaces while retaining the surface quality. Simply modifying the surface control points leads to bad results, therefore the surface was modified by refitting: A set of sampling points is defined on the original geometry. The points were then compensated and the surfaces were fitted back to these points while maintaining their continuity boundary conditions. When the penalty-factors were set correctly and the surfaces are sufficiently flexible, an accurate compensation was achieved for various academic examples, as shown in the previous section.

As pointed out, CAD geometries are never a perfect rendition of the geometry. While delivering a successful compensation, the surface compensation algorithm was able to cope even with large initial errors and improve on the surface quality instead of decreasing it.

However, to be able to affirm the hypothesis, the framework must be verified in more detail. The following issues can be considered for future research:

- Defining correct penalty-factors in relationship with the desired tolerances.
- Type-c transitions between trimmed surfaces need to be tested extensively and documented with numerical results.
- The method must also be integrated in a (commercial) CAD program. CAD software evolves at a very high speed, a high-performance database structure and visualization module are essential for successful application.

Glossary

γ boundary condition
Π potential
Ψ compensation function
A surface area
$B_{i,n}$ Bernstein function
$\vec{c}_i$ tool node coordinates
$\mathbf{d}$ vector of sampling points
$\mathbf{D}$ matrix of sampling points
$\mathbf{f}$ fitting RHS
k penalty factor
$\mathbf{K}$ fitting matrix
$\mathbf{l}$ boundary condition vector
$\mathbf{L}$ set of boundary conditions (matrix)
$\mathbf{M}$ boundary condition matrix
$\mathbf{n}$ vector of basis functions
N basis function
$\vec{\mathbf{p}}$ control point vector
$\vec{\mathbf{P}}$ control point matrix
$s(w)$ mapping function
S surface function
u surface parameter
v surface parameter
w (trim)curve parameter
$\vec{x}$ cartesian coordinate

Conclusion and recommendations

In this thesis, methods for virtual tool reworking have been explored. Whereas Finite Element simulations are already applied in industry for process design and feasibility checks, the focus of this research project was to use these simulations proactively in order to reduce the amount of tool reworking in the press workshop, or to avoid it altogether. This is a desirable or even necessary improvement, since in the car industry, product development time has to be reduced in order to remain competitive.

Three tool-reworking bottlenecks have been identified that are particularly time-consuming and costly:

- 1. Tool touching-in in order to adjust the blank-flow into the tool cavity.
- 2. Tool compensation for press and tool deflection.
- 3. Tool compensation for springback.

Tool and press deformations
For the first two items it was necessary to make an assessment of the magnitude of press and tool deformations and their influence on the forming process. The first research hypothesis, *The deflection of the press and forming tools influences the quality of the deep-drawn product*, was confirmed: Even the smallest tool-surface deformations strongly influence the contact pressure distribution on the blank and consequently the blank draw-in. Also, global shape deviations occur due to deformation of the forming tools and the press components, particularly when large panels are produced or when high-strength steels are used.

In the case of the so-called cross-die process, a deep drawing process that is used as a material test, publications have shown that the problems related to tool and press deformations are severe. In this thesis, an experimental setup was transferred to a Finite Elements simulation. In contrast to regular forming simulations, the tools were modeled as deformable bodies. Analysis of the results showed the same phenomena that were reported by the experimenters and revealed that taking press and tool deformations into account during the forming simulation is no luxury: Due to the deformable tool models the blank draw-in changed significantly, leading to considerable changes in the product's quality measures such as rupture risk and wrinkling.

In this particular process, the tools were relatively simple structures which could be discretized efficiently. Still, the numerical cost of the calculation was immense. This means that simulations for industrial-scale forming processes are not feasible. Regular techniques to reduce the size of the FE system of equations, such as static condensation, did not improve the efficiency of the simulation. In contrast, Deformable Rigid Bodies (DRBs) do provide a way to calculate the deformation of tools approximately. Contrary to previously published methods, the DRB approach was based on the static equilibrium equations in this thesis. The method has been improved to reduce the error caused by the discretization of the tools, and a method has been developed to assess the approximation error during the simulation. A DRB module was included in forming simulation software *DiekA* and it was tested successfully for the cross-die process. Comparing the results with the regular FE model, the change in the contact pressure distribution was captured correctly, yet the numerical cost only increased by 8%. Therefore, hypothesis 1b, *Press and tool elasticity can be included in the forming simulation at acceptable cost*, is confirmed.

Tool compensation algorithms
Accurate FE simulations, like the previously mentioned analysis, but also the strongly improved springback prediction calculations, make it possible to perform simulation-based tool compensation. In this thesis an algorithm called Smooth Displacement Adjustment (SDA) has been successfully developed, confirming hypothesis 2, *Using FE simulation results, the surface of the forming tools can be compensated automatically for springback and tool deflection to produce a geometrically accurate product.* The SDA algorithm uses FE simulation results to reduce the product's shape deviation in an iterative procedure. The main conclusions are:

- The product's shape deviation can be reduced over 80% in a small number of compensation iterations, as demonstrated for several industrial cases.
- The tool quality is maintained, including the blankholder surface, the gap-width between tools and undercuts are avoided.
- The algorithm does not require human interaction and is robust.

Contrary to the iterative variant, one-step SDA still requires trial and error and it generally leads to less accurate results. However, sometimes it is the only available option, for example when experimental measurements are used instead of simulation results. An extensive study on a stretch-bending process has shown how the compensation can be set up to provide the best results, taking into account different process, geometrical and material parameters. This study also revealed principal flaws in an alternative compensation strategy, called Spring-Forward.

Compensation of CAD geometries
In today's computer aided product development systems, design and simulation are two different worlds. Discretizing CAD geometries into FE meshes is a well-developed area, however, transferring results from FE-based algorithms back into a CAD model remains problematic.

- 'Reworking' tool geometries in a CAD context is also a process planning bottleneck and algorithmic solutions are required.

In the case of tool compensation, the modifications are generally subtle, but very stringent geometrical tolerances must be kept in order to preserve the appearance of the body panel after painting. In this thesis an algorithm has been developed that takes the smoothness constraints on surfaces into account. While applying the compensation to the geometry, it is able to maintain the surface quality and to actually repair geometrical defects in the initial CAD description. This was demonstrated on various academic geometries. This validates the last research hypothesis, *Mesh-based shape modifications can be automatically transferred back to the tool CAD geometry.*

Recommendations

In all three fields of research, algorithmic tools have been developed that can replace, or in any case assist, existing manual procedures. The feasibility and potential of the methods has been proven clearly in this thesis, however the true use will only become clear when the methods are used in industry on a daily basis. To achieve this, the following list of recommendations is given for each algorithmic tool:

Deformable Rigid Bodies

- A full-scale process needs to be modeled using the DRB approach. Only with an industrially realistic simulation, the advantage in efficiency can be proved fully.
- As tool and press deformations cannot be treated separately, the entire press structure needs to be taken into account. This is possible with the DRB approach due to the high efficiency, although modeling the press will require careful analysis.
- The interaction of DRBs under contact needs to be explored in more detail.
- It is highly recommended to apply DRB modeling in the context of flexible blankholder technology. Here, the method is essential in modeling the process correctly, and the results can be verified more easily, both in the case of an FE simulation and in experiments.

The SDA compensation algorithm

- Springback compensation was applied to single-stage deep drawing processes only. It needs to be investigated how compensation is most effective when the production process consists of many different stages
- The use of compensation in the case of global tool deformation needs to be demonstrated in an industrial context

Modification of CAD geometries

- The method needs to be integrated in a (commercial) CAD software-framework. Only then, the potential of the method can be shown for realistic tool geometries
- In this thesis, the feasibility of CAD-geometry compensation was demonstrated. Practical use requires a more careful study of the numerical parameters, used in the algorithm

Final remark

Let us return to the step-wise integration of CAD systems and FE simulations, as introduced in Figure 1.1: Broadening and improving this integration is the way forward for the quick development of new products. The tools and algorithms that have been developed in this thesis strongly support this integration. However, this will only be a successful approach, when a forming process is carefully analyzed and regarded as part of a larger chain of consecutive production processes, not just as a single operation. More importantly, even for the most complex of algorithms, human ingenuity, flexibility and experience must remain involved. Computer modeling, virtual testing and numerical optimization cannot replace these human characteristics. They, however, are a help in saving time and money in production planning.

Acknowledgements

This thesis concludes an exciting period in my life. When I tell people about the work I do, I always tell them I work on car bonnets, door panels and roofs (*bathtubs* would be possible too) and make jokes about us, 'boring' engineers. However, it is surprising to learn how many different people, design engineers, simulation engineers, managers, tool technicians and scientists are involved in forming technology and how many different disciplines come together. I have met an incredible number of inspiring, intelligent and fun people. Definitely *not* boring! Even with today's computer aided tools, each person in the chain brings in their craftsmanship: From the tool technician polishing a micrometer off a die, to a CAD surface designer fine-tuning the shape of a door-panel, to a simulation expert setting numerical parameters. Because of the network of companies involved with INPRO and NIMR, I was able to look over the shoulders of almost everybody. I would like to thank all at INPRO, Twente University, NIMR, Daimler, Corus, Volkswagen and ICEM for taking out time for me and helping me forward.

During my PhD I worked most closely with Timo Meinders from the University of Twente. Timo, thank you very much! Besides reading every single letter in all my publications, you always directly said what you thought, made me laugh and motivated me. I always left our meetings with a great feeling and a lot of new ideas. Thanks to you, this thesis is (as an inside joke) really *finished* now - I hope you agree! The support from the university went beyond my expectations and I would like to dearly thank my promotors, Han Huétink and Fred van Houten for this. They were also involved in the reading committee, as were my INPRO supervisor Bert Rietman and Ton van den Boogaart and that was a great help. My gratitude also extends to Gerben Bossenbroek for correcting the thousands of *howevers* and *therefores* in the thesis.

Most of this work was done at INPRO GmbH in Berlin and, from all the people that I had to honor to work with, Stephan Ohnimus deserves a lot of gratitude. We had countless discussions and they were not always simple - in fact, they were *never* simple. But Stephan, you were always open to share wild new ideas and they pushed me in directions that I would never have discovered on my own. I owe a lot to you. I would also like to mention Maria Goretti-Doig, Martin Petzoldt, Verenice Haase and in particular Jochen Weiher. Thank you all for your help, the great cooperation and particularly for opening new opportunities for me.

The trip to Ohio State University was an event never to forget. Rob Wagoner, and Wei Gan, thank you for hosting me and taking so much time for discussions and the journal paper. It was a pleasure working together. From the industrial side, I have received great support and new ideas from a lot of people, in particular Eisso

Atzema from Corus, Alexander Back from ICEM, Karl Roll from Daimler and Gert-jan Kloosterman at ABAQUS.

However, of all people, my parents must have been the most important sources of inspiration and motivation. At the end of my master's assignment, I was *very* sure I was not the kind of guy for science (even now I am not yet sure!). However, they ignored my doubts and complaints and said "why don't you simply try it out?". In case of trouble, they helped me on the phone, back home in The Netherlands with a piece of apple-pie or by coming to Berlin. I enjoyed biking around the city together, strolling over the Kurfürstendamm, and the breakfast at hotel Schweizer-hof. I promise I will buy a sofa bed now! My sister Merel was also a great source of help, backing me up with her personal PhD experiences. Merel, I wish you (and also Jeroen, thanks for also discussing the application aspects of all those sheet metal parts in leased cars) good luck with your new career in your happy *family enterprise*!

The end of my PhD also means that I will be moving away from Berlin. Thank you, Berlin, for the rollercoaster-like years, the good and bad (but never *boring*) times, for the great friends and fantastic musicians that I have met, for the late-night jam sessions, the barbecues in the park, the endless numbers of Christmas fairs, break-fast in a second-hand shop and my cute little apartment. I will miss you and I hope we will stay in touch.

Roald Lingbeek, Berlin, January 2008

About the author

Roald Arnoud Lingbeek was born in Vilvoorde (Belgium) on March 19th, 1980. Primary schools were attended in Appingedam, the Netherlands and in Apeldoorn. After obtaining his secondary school diploma at the Gymnasium Apeldoorn in 1998, he started his study in mechanical engineering at the University of Twente in Enschede. He joined the chair in Applied Mechanics. Courses in this field were combined with other subjects such as polymer technology and philosophy. Extracurricular activities included the organization of a symposium on biomedical engineering, a 'technology and art' project, several graphical design jobs and various cultural activities, such as playing in the university Bigband. In 2002, a work-assignment on the simulation of welding was carried out at INPRO GmbH in Berlin. The Master's assignment was also carried out at INPRO, but now on the subject of springback compensation for metal forming. The Master's degree was obtained in January 2004. From february 2004, research on the subject of springback compensation was continued, now as a PhD project for the Netherlands Institute for Metals Research (NIMR). As INPRO is a partner of the NIMR, the project was carried out in Berlin. From april 2008, Roald is employed at AutoForm Engineering GmbH in Zürich, Switzerland.

Bibliography

[1] Company norm. Internal Document Daimler AG.

[2] Gusswanddicken. Internal Document Volkswagen AG.

[3] Dieka homepage. *http://www.dieka.org*, 2007.

[4] *ICEM-surf user's manual*. 2007.

[5] A. Andersson. *Macro-Geometric Defects - a numerical and experimental study of springback and surface defects*. PhD thesis, Lund University, 2004.

[6] E.H. Atzema, C.H.L.J. ten Horn, and H. Vegter. Influence of tooling layout on sheet forming process analysis. In P. Neittaanmäki et. al. (eds.) Jyväskylä, editor, *Proceedings ECCOMAS*, 2004.

[7] S. Axler. *Linear Algebra Done Right*. Springer, Berlin, 1997.

[8] F. Barlat. Constitutive relations for metal forming simulations. In J. Cesar de Sá A.D. Santos, editor, *Proceedings NUMIFORM conference*, pages 3–24, 2007.

[9] T. Belytschko, W. Liu, and J. Moran. *Nonlinear Finite Element Analysis for Continua and Structures*. John Wiley & Sons, 2000.

[10] J. Bitzenbauer and K. Schweizerhof. Deformable rigid bodies in ls-dyna with applications - merits and limits. In *5th European LS-DYNA Users Conference*, 2006.

[11] P. Bogon. Werkzeugtechnik zur magnesium-blechumformung, ein beispiel. *www.utfscience.de*, 2005.

[12] M.H.A. Bonte. *Optimisation Strategies for Metal Forming Processes*. PhD thesis, University of Twente, 2007.

[13] I.A. Burchitz, T. Meinders, and J.Huétink. Adaptive through-thickness integration strategy for shell elements. In J. Cesar de Sá A.D. Santos, editor, *Proceedings NUMIFORM conference*, pages 699–704, 2007.

[14] I.A. Burchitz and V.T. Meinders. Do's and don'ts for accurate springback simulations. Internal Document Netherlands Institute for Metals Research.

[15] B. Chirita and G. Brabie. Experimental investigation of different influences on springback of parts formed by bending. In V. Brucato, editor, *proceedings ESAFORM*, pages 251–254, 2003.

[16] S. Coquillart. Extended free-form deformation: A sculpturing tool for 3d geometric modeling. In *proceedings SIGGRAPH '90*, 1990.

[17] L.H. de Figueiredo. Surface intersection using affine arithmetic. *http://www.cs.uwaterloo.ca/cs-archive/CS-1995/47/CS-95-47.pdf*, 1996.

[18] M.E. Dingle, P.D. Hodgson, and M. J. Cardew-Hall. Effect of global and local stiffness on blankholder pressure in draw die forming. In *SAE World Congress*, 2001.

[19] E. Doege and L.E. Elend. Design and application of pliable blank holder systems for the optimization of process conditions in sheet metal forming. *Journal of materials processing technology*, 111:182–187, 2001.

[20] Chu E., L. Zhang, S. Wang, X. Zhu, and B. Maker. Validation of springback predictability with experimental measurements and die compensation for automotive panels. In D. Yang et al., editor, *proceedings NUMISHEET*, pages 313–318, 2002.

[21] K. Adam et.al. *Schuler Metal Forming Handbook*. Springer, Berlin, 1998.

[22] G. Farin. *Kurven und Flächen im Computer Aided Geometric Design, eine praktische einführung*. Vieweg, Braunschweig, 1994.

[23] W. Gan, R. Wagoner, K. Mao, S. Price, and F. Rasouli. Practical design methods using fe modeling applied to sheet parts and dies. *Journal of Material Processing and Technology*, 126:360–367, 2004.

[24] W. Gan and R. H. Wagoner. Die design method for sheet springback. *International Journal of Mechanical Science*, 46:1097–1113, 2004.

[25] O. Ghouati, D. Joannic, and J. Gelin. Optimisation of process parameters for the control of springback in deep drawing. In J. Huétink and F.P.T. Baaijens, editors, *proceedings NUMIFORM*, pages 819–824, 1998.

[26] G. H. Golub and C. F. van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, 1989.

[27] P. Groche and T. Hofmann. Einfluss des dynamischen Übertragungsverhaltens von stößelführungen auf die arbeitsgenauigkeit von umformpressen. *EFB Hannover*, 2005.

[28] C. Hartung. *Beurteilung des optischen Erscheinungsbildes von Ziehteilen mit Hilfe numerischer Verfahren*. PhD thesis, Technische Universität München, 2001.

[29] M. Häussermann. *Zur Gestaltung von Tiefziehwerkzeugen hinsichtlich des Einsatzes auf hydraulischen Vielpunktzieheinrichtungen*. PhD thesis, Institut für Umformtechnik, Universität Stuttgart, 2002.

[30] M. Häußinger. *Eigenschaftsvergleich von Ziehteilen aus Aluminium- und Magnesiumblech*. PhD thesis, Technische Universität München, 2006.

[31] H. Hayashi. Elastic deformation of tools in stamping of large-scale autobody panels. In M. Tisza et al., editor, *proceedings IDDRG*, pages 437–444, 2007.

[32] R. Hill. *The Mathematical Theory of Plasticity*. Oxford University Press, Oxford, 1998.

[33] T.J.R. Hughes. *The Finite Element Method*. Prentice-Hall, New Jersey, 1987.

[34] A.P. Karafillis and M.C. Boyce. Tooling design accommodating springback errors. *Journal of Materials Processing Technology*, 32:499–508, 1992.

[35] A.P. Karafillis and M.C. Boyce. Tooling design in sheet metal forming using springback calculations. *International Journal for Machine Tools and Manufacture*, 34:113, 1992.

[36] A.P. Karafillis and M.C. Boyce. Tooling and binder for sheet metal forming processes compensating springback error. *International Journal for Machine Tools and Manufacture*, 36:503, 1996.

[37] Y.T. Keum, I.H.Ahn, I.K.Lee, M.H. Song, S.O. Kwon, and J.S. Park. Simulation of stamping process of automotive panel considering die deformation. In L.M. Smith et. al., editor, *Proceedings NUMISHEET 2005*, pages 90–95, 2005.

[38] E. Kim and J.S. Hwang. Digital die manufacturing in automotive stamping. In D. Yang et al., editor, *proceedings NUMISHEET*, pages 279–286, 2002.

[39] G. Kloosterman. *Contact Methods in Finite Element Simulations*. PhD thesis, University of Twente, 2002.

[40] K. Knothe and H. Wessels. *Finite Elemente*. Springer, Berlin, 1999.

[41] T. Kuwabara. Advances of plasticity experiments on metal sheets and tubes and their applications to constitutive modelling. In L. Smith et al., editor, *Proceedings 6th NUMISHEET conference*, pages 20–37, 2005.

[42] C.L. Lawson and R.J. Hanson. *Solving Least Squares Problems*. Prentice Hall, New Jersey, 1974.

[43] R. Lingbeek. Iterative springback compensation of numisheet benchmark #1. Technical report, 2005.

[44] R. Lingbeek, J. Huétink, S. Ohnimus, M. Petzoldt, and J. Weiher. The development of a finite elements based springback compensation tool for sheet metal products. *Journal of Materials Processing Technology*, 169.

[45] R.A. Lingbeek, W. Gan, R.H. Wagoner, T. Meinders, and J. Weiher. Theoretical verification of the displacement adjustment and springforward algorithms for springback compensation. *Submitted to: International Journal of Material Forming*, 2008.

[46] R.A. Lingbeek and T. Meinders. Towards efficient modelling of macro and micro tool deformations in sheet metal forming. In J. Cesar de Sá A.D. Santos, editor, *Proceedings NUMIFORM conference*, pages 723–727, 2007.

[47] R.A. Lingbeek, T. Meinders, S. Ohnimus, M. Petzoldt, and J.Weiher. Springback compensation: Fundamental topics and practical application. In N. Juster and A. Rosochowski, editors, *Proceedings 9th ESAFORM conference*, pages 403–406, 2006.

[48] C. McMahon and J. Browne. *CAD/CAM - Principles, Practice and Manufacturing Management*. Addison Wesley, 1998.

[49] T. Meinders, A.W.A. Konter, S.E. Meijers, E.H. Atzema, and H. Kappert. A sensitivity analysis on the springback behavior of the unconstrained bending problem. In L. Smith et al., editor, *Proceedings 6th NUMISHEET conference*, pages 272–277, 2005.

[50] P. Michalik. *Methods for Constraint-based Conceptual Free-form Surface Design*. PhD thesis, Technische Universität Ilmenau, 1993.

[51] A. Nilsson and F. Birath. Topology optimization of a stamping die. In J. Cesar de Sá A.D. Santos, editor, *Proceedings NUMIFORM conference*, pages 449–454, 2007.

[52] S. Ohnimus, M. Petzoldt, B. Rietman, and J. Weiher. Compensating springback in the automotive practice using mashal. In L. Smith et al., editor, *Proceedings 6th NUMISHEET conference*, pages 322–327, 2005.

[53] L. Piegl and W. Tiller. *The NURBS book*. Springer, Berlin, 1997.

[54] F. Placidi, X. Lemoine, E. Till, and H. Georgi. Assessment and prediction of spring-back for high-strength steels. Technical report, European Comission Steel Research, 2005.

[55] F. Pourboghrat and E. Chu. Prediction of springback and sidewall curl in 2d-draw bending. In *Proceedings NUMISHEET*, pages 337–351, 1993.

[56] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 1992.

[57] P. Renteln and A. Dundes. Foolproof: A sampling of mathematical folk humor. *http://www.ams.org/notices/200501/fea-dundes.pdf*, 2005.

[58] T. Reuding and M. Sreckovic. Automatic local updating of cad surface models. In *proceedings Third SIAM Conference on Geometric Design*, 1993.

[59] K. Roll and J. Hoffmann. Eine möglichkeit zur berücksichtigung der elastischen werkzeug- eigenschaften bei der blechumformsimulation. In *LS-DYNA Anwenderforum*, 2005.

[60] C. Ropers. *Berücksichtigung der Temperatur und Werkzeugnachgiebigkeit in der Simulation von Blechumformprozessen*. PhD thesis, Hannover University, 2001.

[61] E. Schelkle. Cae simulations what are the challenges in the future? ˙*www.autosim.org/downloads/AUTOSIM*, 2006.

[62] H. Schmidt, T.C. Vu, and L. Recke. Sheet forming simulation the breakthrough of a new technology. In Lee, Kinzel, and Wagoner, editors, *proceedings NUMISHEET*, pages 158–164, 1996.

[63] S. Schöne. Einsatz vom reverse engineering am melkus rs1000 - rs2000 ˙*http://tu-dresden.de*, 2007.

[64] T. Sederberg. Chapter 2 - bezier curves *http://www.tsplines.com/resources/class_notes/Bezier_curves.pdf*, 2007.

[65] T. Sederberg and S. Perry. Freeform deformation of solid geometric models. *Computer Graphics*, 20:151–160, 1986.

[66] W. Song and X. Yang. Free-form deformation with weighted t-spline. *http://www.cad.zju.edu.cn/home/yinxuehui/swh/research/2004/TFFD/TFFD.htm*, 2004.

[67] T. Tan. Informationsintegration in der virtuellen produktentstehung - vom virtuellen fahrzeug zur digitalen fabrik. In *proceedings Produktionstechnisches Kolloquium*, pages 93–97, Berlin, 2004.

[68] Z. Tekiner. An experimental study on the examination of springback of sheet metals with several thicknesses and properties in bending dies. *Journal of Materials Processing Technology.*

[69] A.H. Boogaard van den, H.H. Wisselink, and J. Huétink. Do advanced materials contribute to accuracy in industrial sheet forming simulations? In M. Geiger et al., editor, *She Met conference proceedings*, pages 71–80, Erlangen - Nuremberg, 2005.

[70] R. Wagoner. Fundamental aspects of springback in sheet metal forming. In D. Yang et al., editor, *proceedings NUMISHEET*, pages 13–19, 2002.

[71] R. Wagoner. Design of sheet forming dies for springback compensation. In V. Brucato, editor, *proceedings ESAFORM*, pages 7–14, 2003.

[72] R.H. Wagoner and J.L. Chenot. *Metal Forming Analysis.* Cambridge University Press, 2001.

[73] R.H. Wagoner and M. Li. Advances in springback. In L. Smith et al., editor, *Proceedings NUMISHEET*, pages 209–214, 2005.

[74] R.H. Wagoner and M. Li. Simulation of springback: Through thickness integration. *International Journal of Plasticity*, 2006.

[75] R.H. Wagoner, J.F. Wang, and M. Li. Springback. *ASM Metals Handbook on Forming and Forging (vol.14), ASM*, 2006.

[76] X.J. Wang, D. Tang, X.A. Tang, and J. Liu. Development of a new universal friction test for sheet forming. In J.K. Lee, G.L Kinzel, and R.H. Wagoner, editors, *proceedings NUMISHEET*, pages 55–59, 1996.

[77] M.L. Wenner. Overview - simulation of sheet metal forming. In L. Smith et al., editor, *Proceedings NUMISHEET*, pages 3–7, 2005.

[78] H. Wiemer. *Stand und Möglichkeiten der Systemsimulation von mechanischen Pressmaschinen.* PhD thesis, Technische Universität Dresden, 2004.

[79] P. Wriggers. *Computational Contact Mechanics.* John Wiley and Sons, 2002.

[80] X. Zhou. *Numerical Prediction of Springback in U-Channel Forming of Aluminum Tailor Welded Blanks.* PhD thesis, Carleton University, Ottawa, 1999.

[81] T. Zwickl, B. Carleer, and W. Kubli. Visualization of the invisible, explanation of the unknown, ruggedization of the unstable. In L. Smith et al., editor, *Proceedings NUMISHEET*, pages 145–151, 2005.