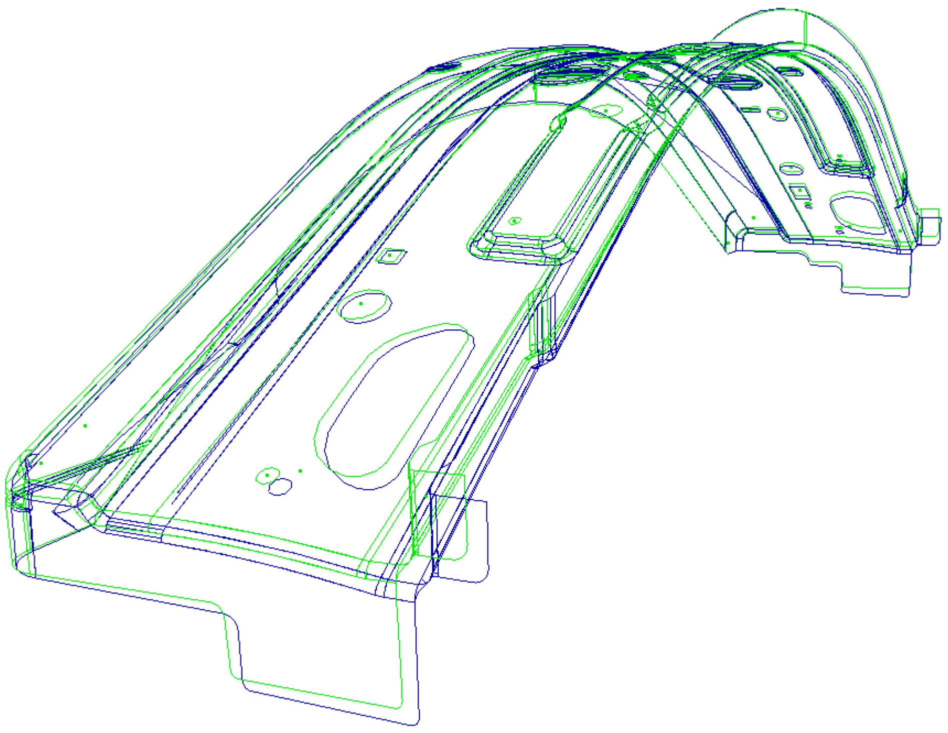
aspects of a **designtool** **for** **springback** **compensation**

Master's Thesis
R.A. Lingbeek
University of Twente / **INPRO**



University of Twente

Aspects of a design tool for springback compensation

Preface

This report is the result of my master's thesis at INPRO in Berlin. The project started at the first of April 2003 and was finished at the end of the year. Those 9 months were a very interesting and rewarding experience. INPRO provided me with a lot of freedom to develop the project as I intended. This was important, since springback compensation is something almost completely new, and we had to start 'from scratch'. At the same time, my dear colleagues Stephan Ohnimus and Martin Petzoldt developed an algorithm for the same problem, but based on a different principle. The discussions we had were a good source of inspiration, I would like to thank them for that, and for the fun time we had.

From the University of Twente, I would like to thank Timo Meinders for his helpful and detailed support during the project. This was and "rather, very, really and extremely" appreciated! Thanks especially for reviewing my report over and over again. I would also like to thank Prof. Huétink and Prof. van Houten for providing ideas and support.

The support of Alexander Back of ICEM-surf has been of vital importance for the development of CAD functionality of the algorithm, many thanks for this, and Tim Lemke of DaimlerChrysler is thanked for receiving Martin and me at the Sindelfingen plant, and for testing the CAD functionality.

I would like to thank Bert Rietman for providing me the assignment and support at INPRO and of course for reviewing my report. Finally I would like to thank all of my colleagues at INPRO-VPT. I always felt like I was considered a team member and colleague, more than simply a student and I am looking forward to my future PhD project at INPRO.

Contact

R.A.Lingbeek (Studentennummer 9807616)
Email: roald.lingbeek@inpro.de, r.a.lingbeek@student.utwente.nl
Telephone: +49 (0)30 399 97-274
Mobile Phone: +49 (0)163 2784512

INPRO Innovationsgesellschaft für
fortgeschrittene Produktionssysteme
in der Fahrzeugindustrie mbH
Hallerstraße 1
D-10587 Berlin
Germany

Private address
Gneisenaustrasse 17
D-10961 Berlin
Germany

Summary

Many products in the automotive industry are produced with the deep drawing process. When the tools are released after the forming stage, the product springs back due to the action of internal stresses. Because the geometric tolerances can be tight for sheet metal products, this shape deviation can be unacceptable. In many cases springback compensation is needed: the tools of the deep drawing process are changed so, that the product becomes geometrically accurate. In the industry, this is currently a costly and time consuming process of producing prototype products and redesigning the tools manually. The goal of this project was to develop a software tool that can automatically perform this optimisation process, using the results of finite elements (FE) deep drawing simulations.

To evaluate springback problems, the main factors are not only the geometrical accuracy but also the assembly forces; In some cases, the product can be bent back into the right shape during assembly. For some products, these forces are too high, or the shape of the assembled product may be unacceptable. Then, the first goal is to reduce springback. The deep drawing process can be optimised in various ways, mainly by influencing the material flow into the die cavity. Redesigning the structure of the product can be effective as well.

Even when the product design has been optimised, and the deep drawing process has been set up carefully, springback compensation has to be carried out to improve the geometrical accuracy of many products. To speed up the manual springback compensation process, the use of finite elements calculations instead of real prototype tools is currently tested in the industry. Several completely automatic springback compensation algorithms have been reported and tested in scientific literature. The idea of the Displacement Adjustment (DA) method is to use the shape deviation between the deep drawn product and the desired shape, multiplied with a negative factor, as a compensation function for the geometry of the tools. In literature the DA method has proven to be the most reliable and fast.

In this project, the control surface (CS) algorithm has been developed. Here a surface with a limited set of parameters is used to compare and evaluate the desired product shape and the deep drawn product, and to modify the deep drawing tools. With the control surfaces, the DA principle is again used for compensation. The control surface allows only a limited set of shape modifications such as bending, torsion and camber. The advantages of this method are that the modification of the geometry can be carried out with a CAD system as well, and that it is possible to control the algorithm manually. The algorithm is demonstrated first with a cylindrical control surface.

To make the algorithm usable in the industry, a version using a flexible parametric description for the control surface has been implemented. With this type of control surface, the mathematics behind the algorithm become more complex. The details of each step in the algorithm are discussed in detail. Then the algorithm is tested out with two products. The first product is a structural part provided by DaimlerChrysler, which is compensated in one iteration only. The resulting reduction in shape deviation amounts 64%. The results of the algorithm can be strongly improved by applying more iterations. This is demonstrated with the second example, a fuel tank cap. With this product it is also shown how local compensation can be used to raise the algorithm’s effectivity. For this, some user interaction is needed. Here, 66% reduction in shape deviation is reached. The CS algorithm has been compared to the DA method. For the structural part, the DA method performed slightly better, as expected. For the fuel tank cap, no comparison could be carried out due to problems with the tool modification in the DA method and due to practical limitations of the finite elements program.

The smooth and continuous description of geometry in CAD files is required for the generation of NC code for the milling robot that produces the deep drawing tools. So, to make the algorithm useable, the same geometry modification that is applied to the tool meshes needs to be applied to the CAD data of the tools. This is problematic since CAD files and meshes have an incompatible description of geometry. In the program ICEM-surf the an identical control surface principle is included and with this function, the CAD geometry has been modified in the same way as the meshes.

The algorithm is not yet industrially applicable in this form. Both FE simulation and the springback compensation algorithm need to become more accurate. However, the project has shown how the

complete process from FE simulation to modification of CAD files can be performed, and the results look promising. With FE deep drawing simulations and the springback compensation algorithm, the process setup of a deep drawing process will become significantly faster in future.

Introduction

Deep drawing is one of the most common manufacturing processes in the automotive industry. Most deep drawn products are structural parts of the car body, such as door panels, engine hoods and side impact protection bars. For these products, the geometrical tolerances are tight, and the tools are expensive. Therefore, accurate process planning is essential.

There have been major improvements in deep drawing simulations, and we are now able to predict the shape of the final product, its internal stresses and process forces. Upon unloading after the forming stage, the product springs back due to internal stresses. For car body panels, these springback deformations can be large, up to several millimetres. High strength steels and aluminium, used for lightweight products, are known to have particularly large springback deformations. As shown in picture 0.1, these high strength steels are already used for more than 60% of the body parts of modern cars, like the new Audi A3.

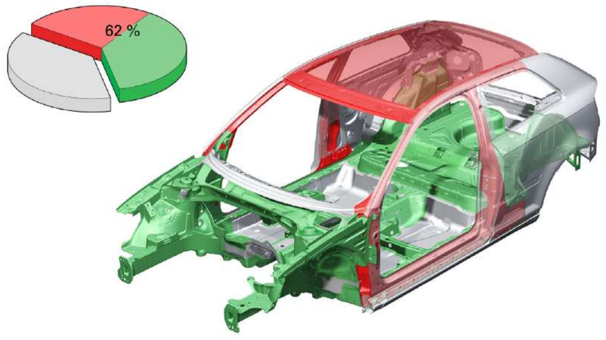


Fig. 0.1 Use of high strength steels (red and green) in the Audi A3

When the product does not meet the shape requirements, the deep drawing tools are manually redesigned so, that the shape deviations due to springback are compensated. This is a complex and costly operation, because the springback can be quite large. The springback problem is also different for every product. Now, it is a trial-and-error process of manufacturing tools, making a prototype product, measuring it, altering CAD data and reworking the stamping tools. For this process, a lot of development time and engineering experience are required.

Springback is essentially, but not exclusively, an elastic deformation. Of course, elastic deformation can be calculated relatively easily, but the calculation of springback is problematic because it is highly dependent on the internal stresses after the deep drawing stage. In finite elements calculations, the predictability of these internal stresses is poor. This is partly caused by the lack of realistic friction and contact algorithms. The calculation of springback effects in deep drawing processes has been implemented for several commercial finite elements software packages such as PAM-stamp, Autoform and INDEED. It is now possible to use these results to directly change the CAD files of the tools, and do a new simulation to check whether the shape alteration was successful. In other words: the FEM calculation is now taken into the optimisation loop. The first industrial tests have been carried out recently [Chu03].

The goal of this project is to design and implement a software tool that automatically alters the deep drawing tools so, that springback is compensated. The optimised geometrical data are transferred into a new CAD file, which are needed as a basis for the NC code for tool production right away. This way, expensive prototype tests can be avoided, and the design optimisation phase will be more effective, faster and more cost-efficient.

In the first chapter, the deep drawing process is introduced. It is shown how springback can be evaluated and how the process can be optimised to *reduce* springback. In chapter two, the two basic strategies for compensating springback are introduced and compared. The basic ideas behind the control surface algorithm are discussed in chapter three. The algorithm structure is made clear and it is demonstrated in its most simple form. A more advanced 3D algorithm is developed in chapter 4, using flexible parametric surfaces. The algorithm is tested on real industrial parts in chapter 5. The surface strategy can be used to transform CAD data as well, as shown in chapter 6. The results of the project and recommendations for future research are discussed in the conclusion. Please note that a list of the most important keywords, introduced in this report, can be found in the glossary.

1. Springback in the deep drawing process

In this chapter, the deep drawing process is discussed, and the springback problem is introduced. In paragraph 1, an overview of the deep drawing process is given. Paragraph 2 is about deep drawing computer simulations with the FE code PAM-stamp. It is shown which tasks need to be carried out to set up a FE deep drawing simulation. All deep drawn products suffer from springback, so springback reduction and compensation need to be carried out in many cases. For this, the springback has to be analysed and quantified correctly, which is discussed in paragraph 3. Before springback compensation is carried out, the design of the product and the setup of the forming process can be optimised to reduce springback. Some springback reduction strategies are discussed in paragraph 4.

1.1 The deep drawing process

In the deep drawing process, shown in picture 1.1, a product is formed from a flat sheet, the blank, by pressing it into a die. The punch reflects the desired shape of the product, the die cavity shape is produced by ‘offsetting’ the punch surface. The sheet pressed onto the die by the blankholder.

This blankholder is essential for controlling the manufacturing process. The force on the blankholder affects the way the blank slides into the die, and consequently, how the product is stretched. When the blankholder is pressed too hard, the blank will not flow into the cavity and the metal is stretched only. That can cause the blank to tear apart. When the blankholder-force is too small, the product will be formed mainly by bending. As a result, springback effects will be larger and the product could even be wrinkled. Lubrication and drawbeads in the blankholder are also used to control the material flow into the die. Modelling the friction between the sheet, die, punch and blankholder is a vital part of a simulation. When the product is finally taken out of the die, it will spring back because of internal stresses.

Normally, a deep drawing process consists of more pressing, trimming and flanging stages. For example, a car door panel has to be stamped to roughly halfway the desired shape first, is transferred to another tool set, and then stamped a second time to obtain the final shape. Then the excess material is cut away in a trimming step, followed by a flanging step. For each of those stages, a calculation (including springback) needs to be made.

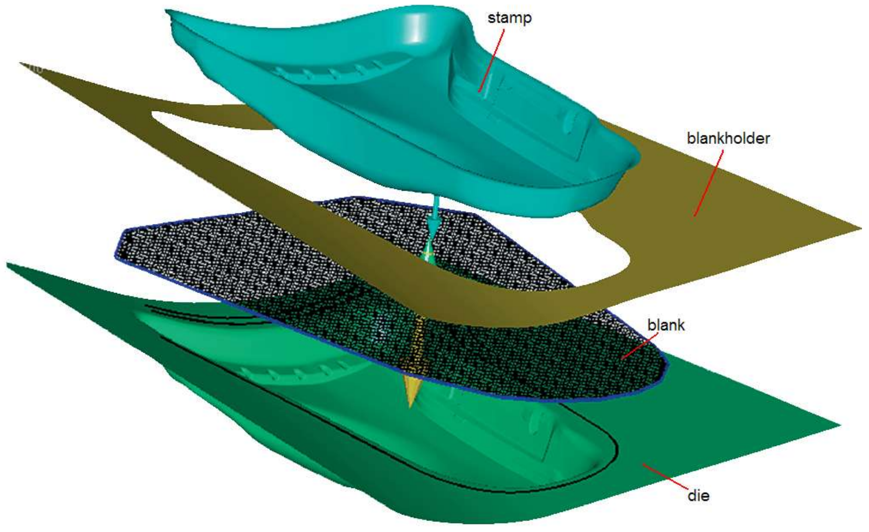


Fig. 1.1 The deep drawing process

1.2 The PAM-stamp 2G finite elements deep drawing simulation

PAM-stamp is one of the leading programs for deep drawing simulations today. Other programs, frequently applied in the car industry, are Autoform and INDEED. PAM-stamp is used for this project because it has some sophisticated automated tool creation functionalities.

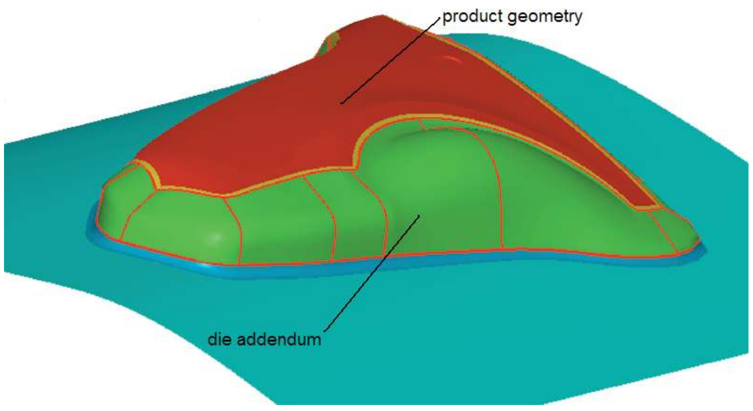
A deep drawing simulation consists of 4 basic steps:

- Conversion of CAD data into a FE mesh and creation of punch, die, blankholder and blank meshes

- Setting up the stamping process
- Performing the nonlinear FE calculation
- Evaluation

1.2.1 Generation of the tool meshes

The geometry of the product is described in a CAD file. Most sheet-formed products are modelled with surface representation. The geometry is then represented as a set of connected complex surfaces. For a FE calculation, the surfaces need to be approximated by a set of (shell) elements. The *Deltamesh*-module can automatically generate a product mesh. To construct a punch, the geometry needs to be extended, as shown in picture 1.2 below. The product is fixed to a surface with curves. An algorithm calculates a surface in between those curves, the 'die-addendum'



The die-addendum mesh and product mesh are combined to form the punch mesh. This mesh is copied to serve as a die-mesh. This die mesh is offset a bit, to create a gap between the punch and the die. A blankholder mesh is generated automatically. Finally, a mesh needs to be created for the blank. These tool meshes form the basis for the FE calculation, and they are of vital importance for the compensation algorithm.

Fig. 1.2 Construction of a punch

1.2.2 Setting up the stamping process

Here, the definition is given of the interaction between the tools and the blank. Normally the stamping process is split up in phases. First the blank is positioned on the die: the blankholder moves down, pressing the blank onto the die, as shown below (die in green, blankholder in blue and blank translucent white) In the second phase the punch moves down and forms the product. Finally the blankholder, punch and die are taken away, allowing the product to spring back (phase 3). For each of these steps, the different meshes can be set to move, or to be fixed in space. For contact between the tools and the blank, a friction and contact algorithm is needed. The physical behaviour for each mesh needs to be selected. For example, the die can be modelled as a rigid material, or as a more realistic material.

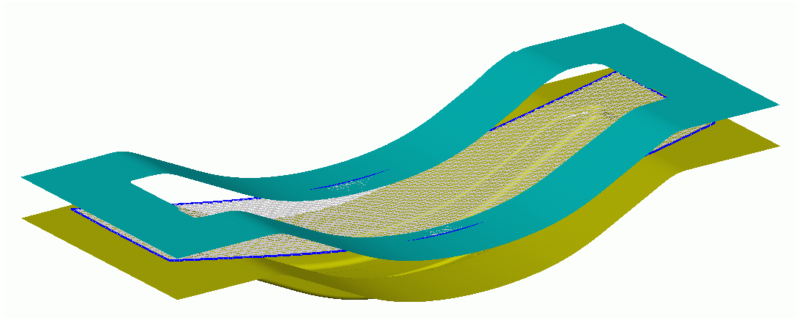


Fig. 1.3 First process step: closing of the blankholder

1.2.3 Performing the calculation

The deep drawing process is simulated using a nonlinear FE solver. PAM-stamp has an explicit solver for the blankholder closing and deep drawing phases, and switches to implicit for the springback calculation. The calculation is generally very expensive, so advanced algorithms are used to speed up the process.

For explicit FE calculations, the size of the time step (and thus the speed of calculation) depends on the size of the elements, and material parameters such as the density and the elastic modulus. Elements need to be as large as possible to keep the calculation time within acceptable limits. But, large elements are not capable to model fine product details accurately. Therefore, adaptive mesh refinement is used. The blank mesh can be coarse at the start of the calculation, and is refined automatically when needed. Therefore, the initial mesh of the blank and the mesh that is the end result of the deep drawing calculation are *not* topologically identical. Finally, the springback calculation is carried out. By releasing the tools, the blank is allowed to spring back. Still, the blank needs to be fixed in space to avoid rigid body modes.

1.2.4 Evaluation

The results of the FE calculation can be stored in any time step during the process, but for the springback compensation, the last two steps are most important: the deformed blank with the tools closed, and the blank after the springback calculation. The two blanks are shown for an example product below. The green mesh is the deep drawn blank (reference) and the red mesh is the deep drawn blank after springback. The optimisation algorithm will be developed to work for one deep drawing stage at a time, because different springback problems occur after each stage. As in each FE program, a variety of post-processing variables can be shown, such as internal stresses, strains and plate thickness

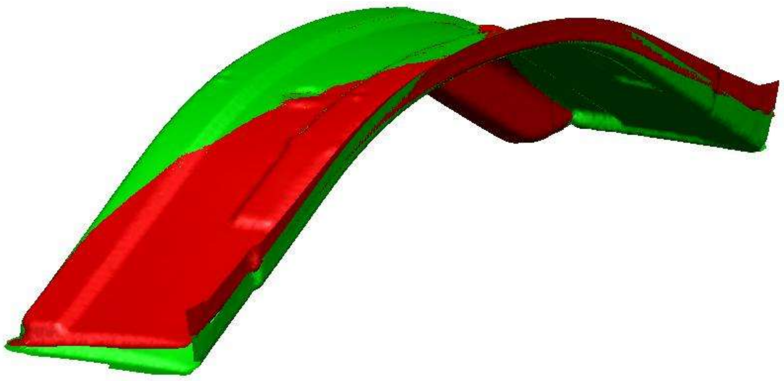


Fig. 1.4 The blank mesh before (green) and after (red) springback (deformations x5)

1.3 Springback quantification

Comparing the meshes of the deformed blank, and the deformed blank after springback is not a trivial task. In the real process, the product is not held in position anymore by fixed mechanical constraints, when the tools are released. After the forming process has been completed, the product is fastened in a larger assembly, such as a car body. During the springback calculation, boundary conditions need to be set. These boundary conditions also influence the springback shape. There are basically two ways to set the boundary conditions:

- Free springback. Only the minimal amount of boundary conditions is set to constrain the rigid body modes. The blank is completely free to springback as if it were lying on needles.

- Assembly springback. The boundary conditions are set exactly so as if the product were fastened in place on the endproduct.

The second mode is the most important, because based on this calculation a decision can be made whether the product needs to be compensated or not. Firstly, the product shape can be checked visually for large shape distortions. For car body panels in particular, the geometrical tolerances can be tight. When the product bulges out irregularly, the light reflection across the product might get distorted and compensation or reduction of springback is needed. The forces that act on the constrained nodes are also an important factor. When the force to push the product in the right shape exceeds 30N this is already unacceptable for car body panels, and compensation or reduction is required. It should be clear that after springback compensation, 100% accurate product geometry cannot be reached in practise. So, if the fastening forces are relatively small it is in many cases preferable *not* to compensate, and to check whether the product already meets its geometrical requirements after assembly. Most structural products are generally too rigid and cannot be bent back into shape during assembly because high internal stresses would be introduced in the assembly.

When geometrical compensation is required, the springback calculation should be carried out with the 'free springback' boundary conditions. The compensation algorithm will then try to find the toolset to produce a product that is geometrically accurate before assembly. The shape deviation will also be smoother, when the 'free springback' boundary conditions are used.

1.4 Springback reduction

Before springback compensation is carried out, the deep drawing process should be optimised to reduce springback first. There are numerous methods to decrease the amount of springback, and the most important ones are discussed shortly in this paragraph. A large amount of information on springback reduction can be found in literature, and it is not part of this project.

Springback reduction starts in the design phase already. A relatively flat structure will be more prone to springback problems than a cup-shaped product. Adding reinforcement ribs can reduce springback problems drastically. Also, altering sheet thickness, or radii in the structure can improve the dimensional stability. Computer optimisation of structural design features has been successfully implemented in [Gho98]. Here, a simple structure (hat-profile) was optimised, with a limited set of defining parameters. A realistic product may contain thousands of geometrical parameters, so we consider this method as highly impractical.

We think that adding or changing structural design features is a task for the designer, because a computer cannot completely oversee the functional requirements for the structure. Therefore, we consider *redesign* outside the scope of this report. It is, however, the most effective way of reducing springback.

The deep drawing process itself needs to be set-up carefully and can be optimised as well. As explained in the first paragraph, the material flow into the cavity has a big influence on the springback behaviour of a product. The material flow defines the stretching of the material in the direction of the sheet and can be controlled by

- the blankholder force
- drawbeads
- lubrication
- the design of the die-addendum

Material flow can also be directly controlled by making cuts or slots in the blank. When the mechanical requirements of the material are less stringent, and a longer cycle time is acceptable, the warm pressing process can be used.

2 Springback compensation strategies

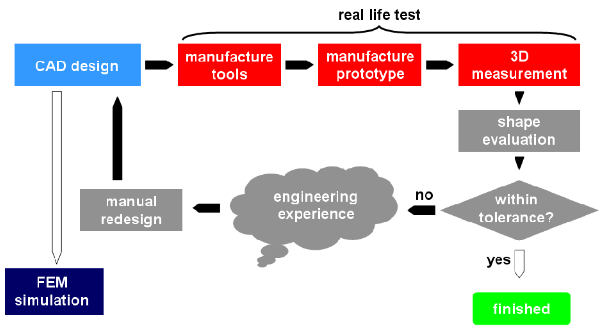
Even when the product design has been optimised, and the deep drawing process has been set up carefully, springback compensation has to be carried out to improve the geometrical accuracy of many products. In the first part of the chapter, the framework for the compensation algorithm is set up. It is investigated how springback compensation is carried out in the industry already. Right now, the most optimisations are based on tests with real toolsets. Now FE springback simulations have become available, the first tests are carried out to use these simulations in the optimisation loop, instead of manufacturing real prototype tools.

However, the goal of this project was to develop an algorithm that can carry out tool shape optimisation completely automatically. The second part of the chapter is focussed on the two most important methods that have been developed and tested in literature [Wag03]. The first method is based on the direct reduction of shape deviation between the deep drawn product and the desired shape, and is called the 'displacement adjustment' method. The second method is based on the forces from the punch onto the blank and is called the 'spring forward' method. The main principle behind the two algorithms is explained, and both methods are compared in an example.

2.1 Springback compensation

When the product design is finished, and optimized to reduce springback, the final step is geometrical optimisation of the tool geometry, which is often referred to as 'overbending'. Overbending means that the geometry of the tools is changed so, that after springback, the product reflects the desired shape better. At this moment defining the overbent shape, from which the tools are derived, is a manual job. The process is visualized in picture 2.1 (left). Note that in all flowcharts, physical tests are visualized in red, FE simulations and related operations in dark blue, and user actions in grey.

The process is started with a feasibility check, by carrying out a FE deep drawing simulation. When the FE simulation shows that the product can be produced, a toolset is manufactured, and a prototype product is produced on a real press. The product is then measured in three dimensions, and compared to the desired shape, defined in the CAD data. A process engineer then decides how to change the shape of the tools. The design department changes the CAD data, used for the machining of the toolset. The toolset is modified, and another prototype product is made. When the product shape is still outside the tolerances, more changes will be made until the tolerances are met.



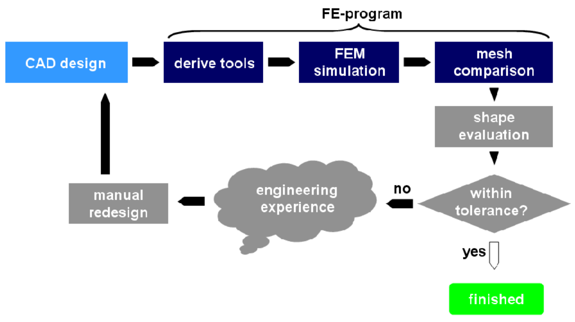


Fig. 2.1 Manual springback compensation (left), manual springback compensation with FEM (right)

Now that springback calculations have become faster and more accurate, computer simulations can be used instead of real toolsets. This speeds up the process and reduces cost because prototype products are not needed anymore. At the moment, springback calculations are not yet fully reliable, so care has to be taken when those calculations are used as a basis for springback compensation. Industrial application of this method has been described in [Chu03].

The goal of this project is to automate the process completely, as shown in picture 2.2. The computer can propose more complicated and more accurate compensation measures, and can perform several optimisation steps automatically. This way, the number of simulations can be reduced, while the end product will meet much tighter tolerances. This optimisation needs to be done separately for every single step in the deep drawing process, including the trimming phases. These phases can be particularly problematic, because the shape of the product often changes drastically during trimming.

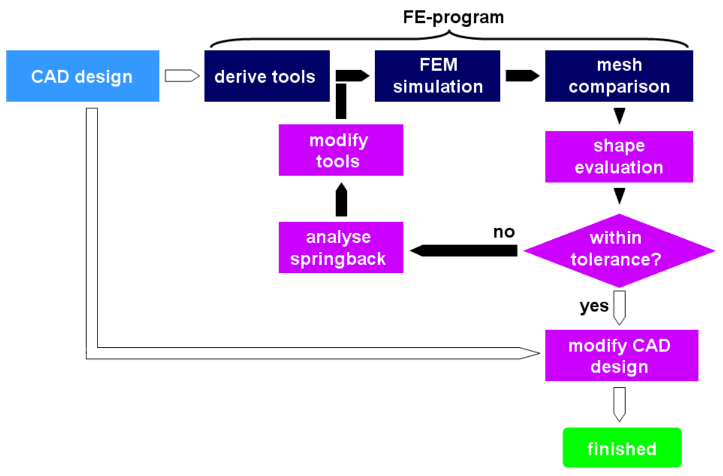


Fig. 2.2 Fully automatic springback compensation

The basic program structure is built up around the block diagram 2.2. The operations that are carried out by the algorithm are visualized in purple. The complete procedure is as follows: CAD data is imported in PAM-stamp, and then transformed into tool meshes. A mesh is created to represent the blank. The deep drawing simulation is carried out, and results in two meshes. The *reference mesh* represents the blank directly after deep drawing, with the tools still closed. When the tools are removed, the blank will spring back, resulting in the *springback mesh.* The differences between these meshes are evaluated. When the dimensions of the springback mesh are outside a specified geometrical tolerance, a *tool modification function* is calculated, using an accurate overbending strategy.

Why is the springback mesh compared to a processed mesh, and not to the (unprocessed) product shape?

- The reference mesh is the ‘best obtainable geometry’. The mesh formed from the CAD geometry may contain sharp edges, which will always be slightly curved in the deep drawn product. Therefore, the CAD and reference meshes are not fully identical, and the small shape deviations might introduce unwanted ‘noise’ during mesh comparison.

- A deformed blank is the actual result of one stamping stage. It includes not only the product, but also the die addendum, and possibly blank-cuts that have a major impact on springback. The springback compensation algorithm optimises the complete blank, not only the product area.

With the tool modification function, the tool meshes are modified. There are two ways to obtain a new tool-set: The first way is to change the tools directly with the transformation field. The second way is to modify the product only, and create a new toolset in PAM-stamp. It is possible to fit the existing die addendum to the (slightly) modified product geometry. This method is less robust: the algorithm uses the full blank, including die addendum, for the optimisation, and then optimises the product shape only. Changing the die addendum separately will affect and possibly worsen the results of the optimisation. With the new tools, a new simulation can be carried out. From the new simulation *only the new springback mesh* is used, and compared to the *original* reference mesh. When the springback product is still outside the shape tolerances, more iterations are carried out until the tolerances are met.

In each iteration, a new springback mesh is calculated with the FE simulation. The reference mesh defines the desired geometry, and is not changed during optimisation.

The target of the optimisation is to reduce the *difference in shape* between the reference mesh and the springback mesh. During the optimisation the springback itself is *not* reduced. Actually, in most cases the springback increases when the tools are optimised.

When a satisfactory geometrical accuracy is reached after a number of iterations, the process is stopped and the shape modification that has been applied to the tool meshes, is applied to the CAD files of the tools as well. With these CAD files, the NC code for machining tools can be developed directly.

2.2 The displacement adjustment method

The first iteration of the DA method is visualized in picture 2.3. A forming simulation is carried out in step 1. The mesh of the blank at the end of the forming stage forms the reference mesh (green) in step 2. The blank is allowed the spring back, delivering the springback mesh (red) in step 3. In step 4, the actual compensation is carried out. Both meshes are compared. The displacement of the nodes in the blank during springback can be calculated directly. The nodal displacements provide the 'springback vector field', a discrete field, defined on the nodes of the reference mesh only. A proposition for a compensated *product geometry* is now calculated by displacing the nodes with a *shape modification field*. In the DA method a (negative) multiplication of the springback vector field is used. As an industry rule of thumb this multiplication factor, the *overbending factor*, is around -1.3. However, in our experience, the value of this factor depends on the geometry and material of the product and can vary between -1.0 and -2.5. In the final step, the tools are derived from the modified product shape.

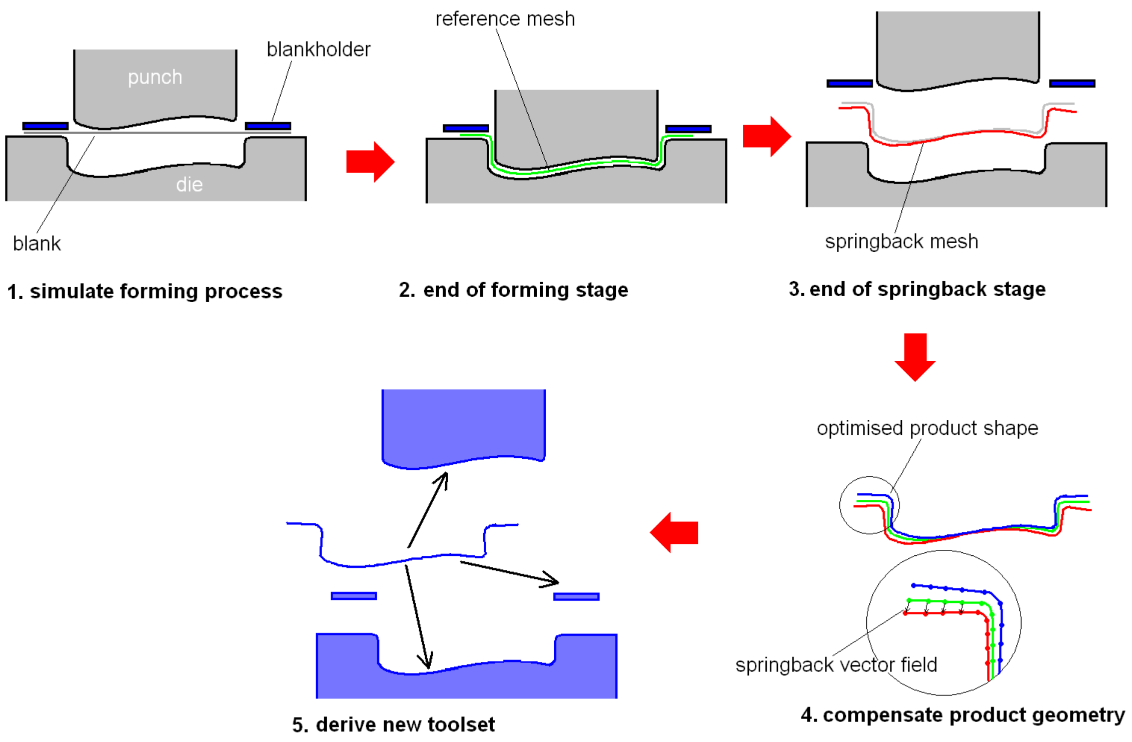


Fig. 2.3 A first iteration with the DA method

When the DA method is applied sequentially, the compensation result will become significantly better, by improving the tool shape step by step. A new FE simulation is carried out with the optimised tools from the first iteration. The result, a new springback mesh, is compared to the original reference geometry. As discussed in the previous paragraph, it is not possible anymore to construct a springback deformation field, but only a *shape deviation field*, because the reference mesh from the original simulation is compared to a springback mesh from the new simulation.

In the same way as in the first iteration, the shape deviation field is multiplied with a negative factor and applied to compensate the product shape again. This cannot be done directly, because the product geometry has already been modified once, and the field is defined on the nodes in the reference mesh only. There are two possibilities to apply the shape deviation field:

- Nodal modification: the shape modification field can be linked to the nodes of the compensated product directly. A (small) error is introduced by ‘recklessly’ moving the shape modification field onto the modified product geometry.

- Continuous modification: The displacement field can be interpolated and extrapolated to a continuous 3D shape modification function, so basically any mesh can be transformed. An error is introduced because of limitations in the interpolation function.

Both methods are demonstrated in picture 2.4. As a model for a forming process, a horizontal strip is bent downwards plastically. The first iteration is straightforward. The strip is deformed into the reference geometry (green) and springs back to the springback shape (red). The springback deformation field, pictured with the black arrows, is multiplied with a negative factor, providing the shape modification field (blue arrows). The shape modification field is applied directly to the reference product, producing the first compensated geometry (comp1). In the second FE simulation, the strip is bent downwards to this ‘comp1’ geometry, and springs back to the ‘Sb2’ shape. The ‘Sb2’ shape is already much closer to the reference geometry. To compensate the difference in shape in this second iteration, a shape deviation field is calculated between the reference geometry to the Sb2 geometry, and again multiplied by the overbending factor. Now, the resulting shape modification field needs to be applied to the ‘comp1’ geometry, delivering the ‘comp2’ geometry.

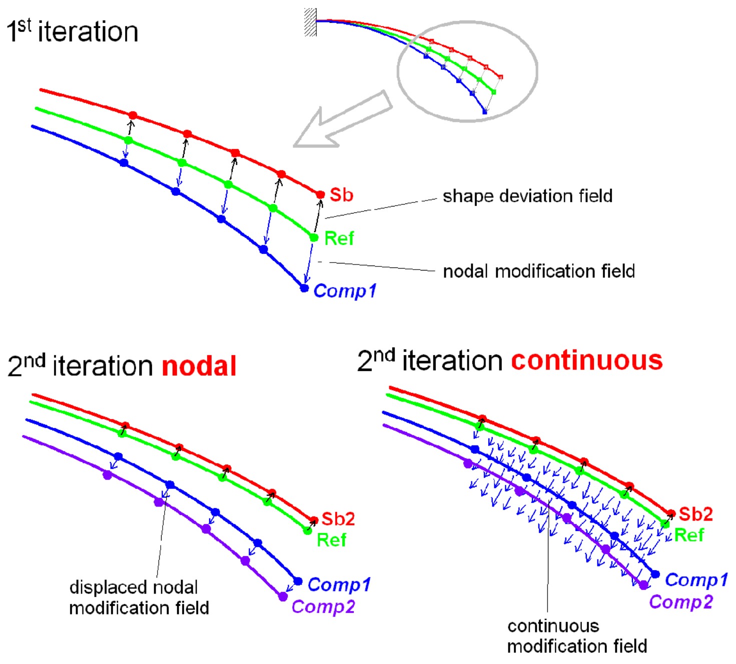


Fig. 2.4 nodal and continuous geometry modification

Now the two different options are demonstrated. In the left picture, the shape modification field is simply applied to the nodes of the comp1 geometry, even though the nodes are on a different location. In the right example, the shape modification field is not moved, but interpolated and extrapolated into a continuous 3D function. This can be applied to the comp1 geometry, delivering the comp2 geometry. The shape modification that is now carried out is principally better, provided that the approximation function is sufficiently detailed. Finding a good definition for the approximation function is not straightforward, and this is turns out to be more of a problem than an advantage. Still, the continuous transformation field has a major advantage: it can be used to directly modify the tool meshes, which are topologically different and generally larger than the product mesh. When the die, punch and blankholder are all modified directly, accurate modification is of vital importance, since the gap width between the tools needs to remain unchanged.

A practical problem for sequential application of the algorithm is that the reference and springback meshes will not be topologically identical anymore after the first iteration because of adaptive mesh refinement. Generally, the adaptive mesh refinement will be slightly different in the second iteration, causing the springback mesh from the second iteration to have a different set of nodes. However, for all simulations, the initial blank is the same. So, to make the reference and springback meshes topologically identical, the refined nodes need to be removed from the meshes. This is not a complicated task, since in most mesh files, the original nodes are always listed first.

In [Wag03], excellent results are achieved for 2D profiles. Because in this article, the 2D profiles are deformed using a flexible (rubber) die and without blankholder, the profiles are not stretched much and the springback in those profiles can be characterized by (combinations of) pure bending. As a result, the shape of the product can therefore be optimised without artificially 'stretching' the geometry. In 3D cases, membrane strains will always occur and artificial strains will be introduced by the compensation algorithm. This can make the compensation process significantly slower.

2.3 The spring forward method

Another method is introduced by Karafillis and Boyce in [Kar92]. In their 'spring forward' (SF) method, process forces are used to optimise the tool shape. The first iteration of the SF method is visualized in picture 2.5. In step 1, the forming process is simulated. After forming of the product, the contact forces (tractions) acting on the punch are 'measured' from the FE result files (step 2). As a compensation measure, the resulting force-field (*f*)is applied to the product geometry in a separate FE calculation in step 3. The idea behind this is that when the tools are closed the blank retains its shape due to action of the force field *f*. When the tools are removed, it is assumed that the blank springs back under the action of a force-field *–f*. So, by applying the force-field *f* to the reference geometry to produce the compensated geometry, it is assumed that the deformation due to springback is compensated. Note that the overbending is carried out as a separate (elastic) calculation with the reference geometry and not as an extra step after the deep drawing simulation. It is assumed that residual stresses do not influence the elastic behaviour. Finally, the obtained product shape is used to create new tools in step 4 and a new forming simulation is carried out. Note that during the iterations, always the original product geometry is compensated with the force field.

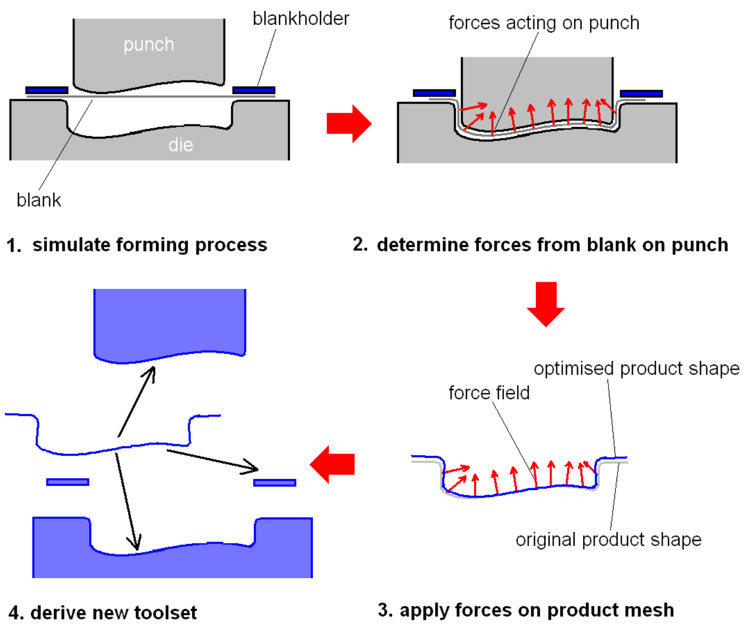


Fig. 2.5 A first iteration with the SF method

The advantage of this strategy is that the problem is compensated in physically more or less the same way as it originates. Letting the structure deform 'naturally' might provide better results than simply imposing a deformation field, and introducing 'artificial' strains. There are also some major disadvantages: When the principle is used in and optimisation loop, it "converges more slowly, if at all, or may converge to incorrect die shapes" [Wag03]. It is also very sensitive to the definition of boundary conditions during the springback calculation.

The most important problem is that there is no fixed geometrical reference, the geometry is likely to converge to an unwanted shape. This can already be demonstrated using the plastic stretching of a bar as a model for the deep drawing process. The bar has a length of 1, and the target is to stretch it to a length of 1.025. So, the displacement field for the desired shape *u** amounts 0.025. The analysis is built up in the same way as in [Kar92, p.121-122]

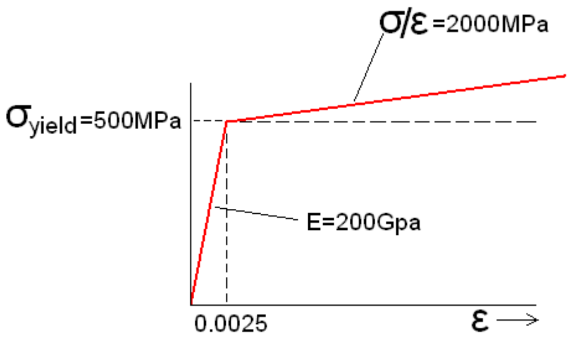


Fig 2.6 a linear elastic linear plastic material curve

So, in the first iteration the product is stretched towards its target length *l=1.025*, and unloaded. From the ideal-elastic-linear-plastic curve shown in 2.6, the deformation load, modelling the tool forces, and the springback displacement can be derived:

$$_{deformation} \quad 545 \tag{2.1}$$

$$_{sb1} \quad _{deformation} / E \quad 0.002725 \qquad l \quad u_{sb1} \quad 0.002725 \tag{2.2}$$

so, the displacement after unloading can be calculated:

$$u_{ul1} \quad u^* \quad u_{sb1} \quad 0.025 \quad 0.002725 \quad 0.022275 \tag{2.3}$$

However, it is desired that

$$u_{ul1} \quad u^* \tag{2.4}$$

Now, as a compensation measure, the bar is loaded with the deformation load from the first step, in its desired shape (i.e. with a length of 1.05), to form the new geometry for the next forming step. Note that the yield strength, and therefore the elastic region, has now increased to 545MPa.

$$_{compensation}(\quad _{deformation1}) \quad 0.002725 \tag{2.5}$$

$$l_2 \quad l_1(1 \quad _{compensation}) \quad 1.0278 \tag{2.6}$$

so, the displacement u_{l2} used for loading the bar in step 2 is 0.0278. So, in this new deformation, the bar is now stretched from 1.0 to 1.0278. The results are calculated:

$$_{deformation2} \quad 551 \tag{2.7}$$

$$_{sb2} \quad 0.002752 \tag{2.8}$$

The displacement after unloading is now closer to the target: $u_{ul2} \quad 0.025040$

The calculation has been carried out for 6 iterations, and as a comparison, an optimisation with the DA method is carried out as well. The results are listed in table 2.7

i	l_i	?$_{deformation}$	?$_{springback}$	l_{ul_i}	shape deviation (%)
0	1.025	545	0.002725	1.022275	100
1	1.0277931	550.58625	0.0027529	1.025040194	1.475
2	1.0278218	550.6435091	0.0027532	1.025068537	2.51511875
3	1.027822	550.644096	0.0027532	1.025068828	2.525779967
4	1.0278221	550.644102	0.0027532	1.02506883	2.525889245

5	1.0278221	550.644102	0.0027532	1.025068831	2.525890365
6	1.0278221	550.644102	0.0027532	1.025068831	2.525890376

i	l_i	$?_{deformation}$	$?_{springback}$	l_{ul_i}	shape deviation (%)
0	1.025	545	0.002725	1.022275	100
1	1.027725	550.45	0.0027523	1.02497275	1
2	1.0277523	550.5045	0.0027525	1.024999728	0.01
3	1.0277525	550.505045	0.0027525	1.024999997	0.0001
4	1.0277525	550.5050505	0.0027525	1.025	9.99999E-07
5	1.0277525	550.5050505	0.0027525	1.025	9.99812E-09
6	1.0277525	550.5050505	0.0027525	1.025	9.77811E-11

Table 2.7 Results of the SF optimisation (top) and the DA method (bottom)

Both algorithms converge, but the length after unloading (l_{ul}) shows that a shape deviation remains when the SF method is applied. The DA method converges faster and the shape deviation is already negligible after 3 iterations. In [Wag03] the same analysis is carried out, but now with a realistic 2D deep drawing simulation. The same convergence problems are found: the SF method is slow, and does not necessarily converge to the right geometry, as picture 2.8 shows.

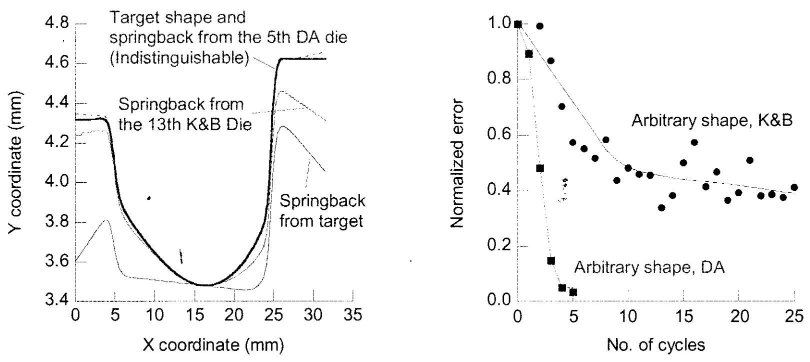


Fig. 2.8 Optimisation of a 2D profile. Picture taken from [Wag03]
Note: A 'normalized error' of 1.0 means a shape deviation of 100%

Another problem is, that it is also is complicated to feed the compensation back into CAD data. The product may bulge out irregularly, its basic shape is not maintained. This means that there is no straightforward way to define a continuous modification function. We therefore left the SF method.

3 The control surface method

The DA-method has proven to be effective and robust, but for practical application of the algorithm, some major problems need to be solved. One of those problems is feeding back the modified geometry back into the CAD system. Another problem is controlling the algorithm to make, for example, local compensation possible. Finally, the algorithm needs to be able to directly transform the larger and topologically different tool geometries as well. The control surface (CS) algorithm, which is discussed in this chapter, solves these problems.

The general idea behind the control surface algorithm is that a flexible surface, called *control surface*, is used as an approximation for the product geometry before and after springback. The principle of the DA method is applied to these control surfaces, called reference and springback surface, instead of to the meshes. The result of this analysis is a so called *transformation surface*. This surface is used to modify the shape of the tools directly. In the first paragraph, these basic principles are explained and visualized.

In paragraph 3.2 the steps of the CS algorithm are discussed in more detail. These basic steps are:

1. Definition of the type of control surface: which kind of surface needs to be used for which springback compensation problem?
2. Approximation of the reference and springback meshes with control surfaces: how can the control surface be fitted through the mesh to approximate the geometry correctly?
3. Calculation of the shape of the transformation surface: how can the DA method be applied to the surfaces to produce a useful springback compensation?
4. Modification of the tools: how can the transformation surface be used to modify the meshes of the deep drawing tools?

In paragraph 3.3, a CS algorithm is implemented using a cylindrical surface as the control surface.The shape optimisation possibilities are too limited to make the algorithm useful, but because the mathematics behind this type of surface are straightforward, the four steps can be made clear more easily. In the following chapter, an applicable but more complicated algorithm is developed, using a more flexible control surface definition.

3.1 Structure of the control surface algorithm

All deep-drawn products have a shape that is defined in a CAD file as a collection of surfaces. The global shape of the product can be represented and approximated by a simpler surface with a small set of shape-parameters. In picture 3.1 a product and the control surface that approximates it, are shown.

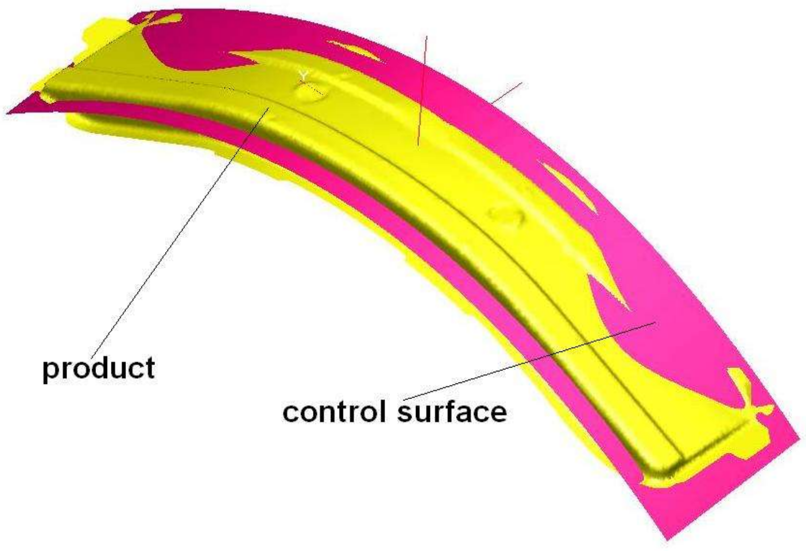


Fig. 3.1 product and control surface

The idea of the DA method was to compare the reference and the springback meshes and derive a shape deviation field. This shape deviation field is then multiplied with an overbending factor, and the resulting shape modification field is applied to the product geometry. In the CS method the reference and springback meshes are approximated by the reference and springback surfaces first. Then the same DA principle is applied to these surfaces, instead the meshes. This is shown in picture 3.2. The reference surface is pictured in green, the springback surface in red and the transformation surface in blue. In this picture, the springback is exaggerated to make the show the principle more clearly: The transformation surface, which can be seen as an 'approximation for the compensated geometry' is created by taking the shape deviation and multiplying it with an (always negative) overbending factor.

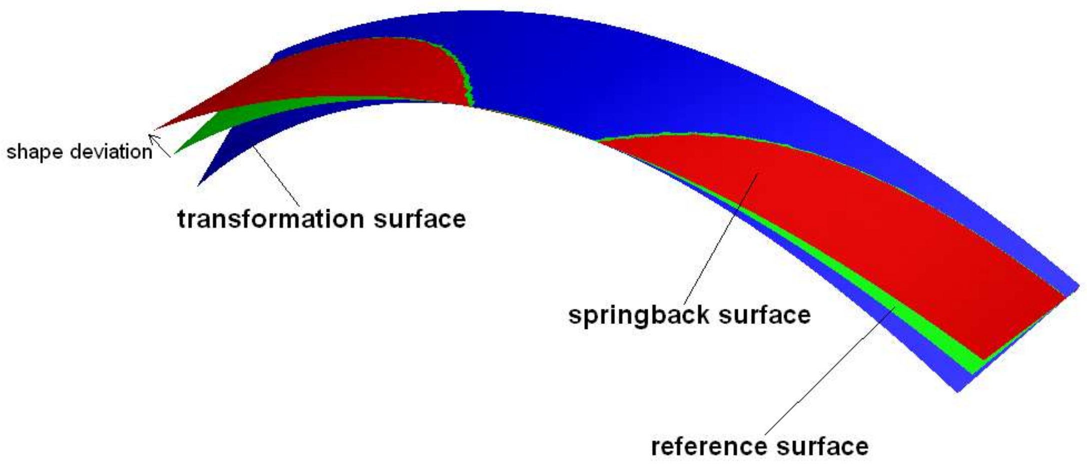


Fig. 3.2 control surfaces and the DA principle

The next step is to use the transformation surface for the modification of the product or tool geometry. How this works is shown in picture 3.3. A product is shown in grey, and the control surface in blue. In the left picture the product is 'linked' to the control surface. Then the shape of the control surface is changed. Because the product is linked to the control surface, it follows the change in shape. So, for this process, two surfaces are needed, and the mesh that is to be modified. In the CS algorithm, these two surfaces are the reference and the transformation surface. First the product is linked to the reference surface. Then the shape of the control surface is changed into the transformation surface, and the product follows this change in shape. Because this method is based on analytical surface descriptions, the modification field is continuous and any mesh can be modified in this way. When the control surface is a smooth surface, which is generally the case, the modification will be smooth as well. So, with the CS method, the tools are modified directly. As with all algorithms, the modified toolset is used for a new FE simulation. The algorithm can also be used iteratively, until the required geometrical accuracy is reached.

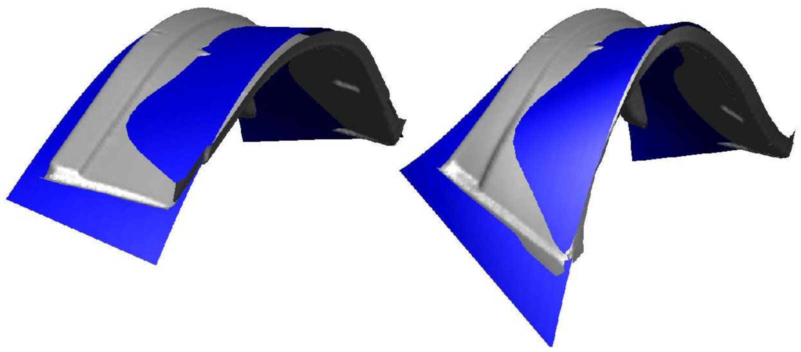


Fig. 3.3 Surface controlled shape modification

For clarity, a first iteration of the CS algorithm is visualized in picture 3.4. Again, the steps in the algorithm are:

1. A deep drawing simulation is carried out. In this (imaginary) case, a torsional springback problem is found.

2. Control surfaces are fitted to the reference and springback meshes, delivering the reference surface and the springback surface.

3. The two surfaces are used to construct a transformation surface, using the DA principle, and a certain overbending factor

4. The tool meshes are modified, based on the reference and transformation surfaces

5. A new FE simulation is carried out. The resulting springback mesh is compared to the reference (desired) geometry and, if needed, another iteration is carried out. The process can be repeated until the dimensional accuracy is satisfactory.

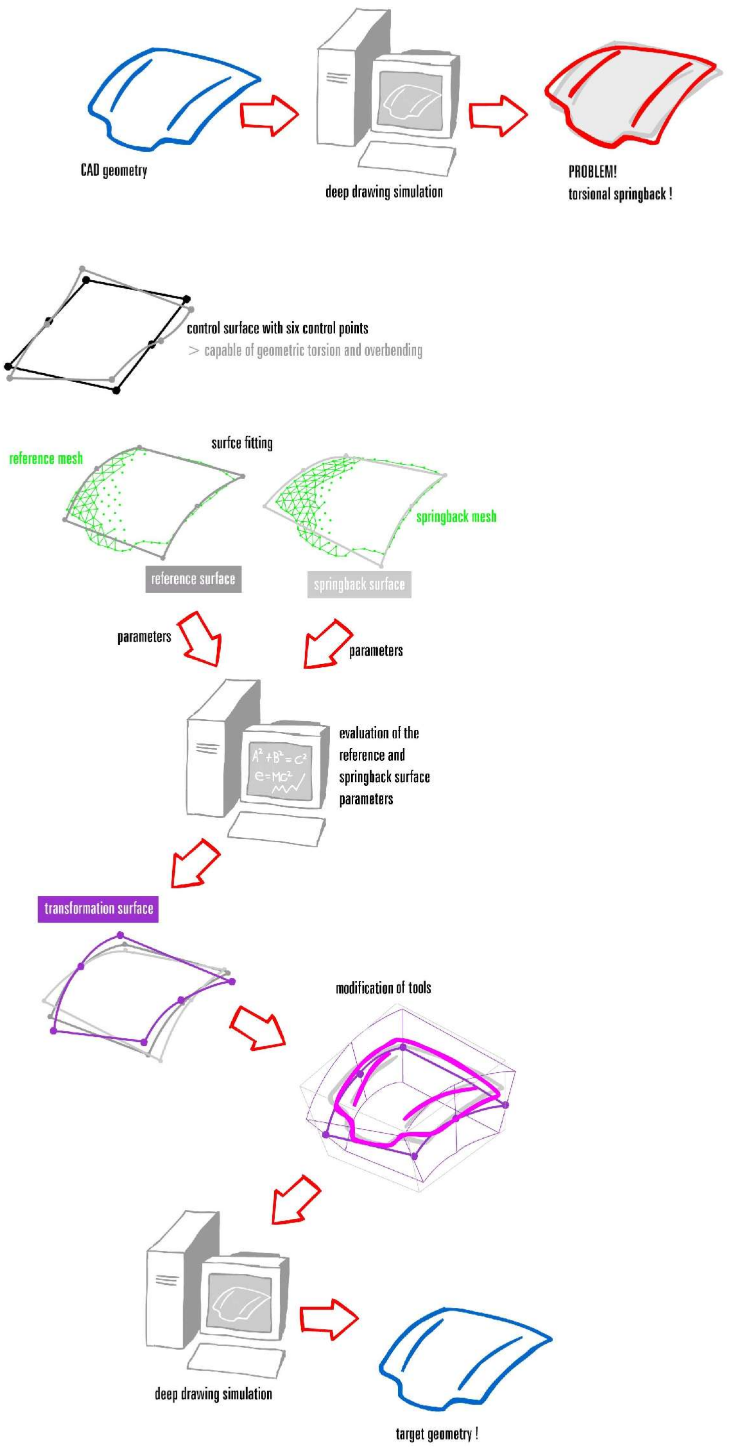


Fig 3.4 The free surface controlled DA algorithm

3.2 The control surface algorithm step by step

In this paragraph, the mathematical details of each of the steps in the algorithm are explained. The limitations and problems of each step are discussed in more detail.

3.2.1 Definition of a control surface

Basically any continuous and smooth surface representation can be used for the control surface. Because the control surface is used to approximate the reference and springback meshes, it is important that the surface can show the difference in shape accurately.
The approximation itself can be very rough. The mathematical flexibility is also important for the second function of the control surface: the modification of the tool meshes. The shape and parameters of the surface define the (im)possibilities of this modification.

It is more important that the definition of this surface reflects the type of springback problem, than that it is able to approximate the product's shape accurately!

Because the shape modification is limited by the ‘mathematical flexibility’ of the control surface, the definition of the control surface influences the accuracy of the optimisation. For example: if a product with a torsional springback phenomenon is optimized using a cylindrical (single curved) surface, the optimisation algorithm will try find an optimum tool shape by ‘bending’ the product geometry only, because more complex shape modifications cannot be carried out. Unfortunately, bending the product will not compensate the springback problem properly and an accurate tool geometry will not be found.

In order to approximate the reference and springback meshes, the control surface is fitted through these meshes. To make this fitting process robust, the number of parameters in the surface needs to remain low. When the surface parameters are (relatively) independent, the fitting process becomes more stable as well. Therefore, finding the right control surface means finding a compromise between springback compensation accuracy and robustness of the algorithm.

Two useful surface definitions are the aforementioned cylindrical surface, used for simple overbending only, and the Bezier/B-spline surface for much more complex shape optimisations. The cylindrical surface will not bring detailed springback compensation, but because the mathematics behind it are straightforward, it is used in paragraph 3.3 to demonstrate the algorithm in its most simple form.

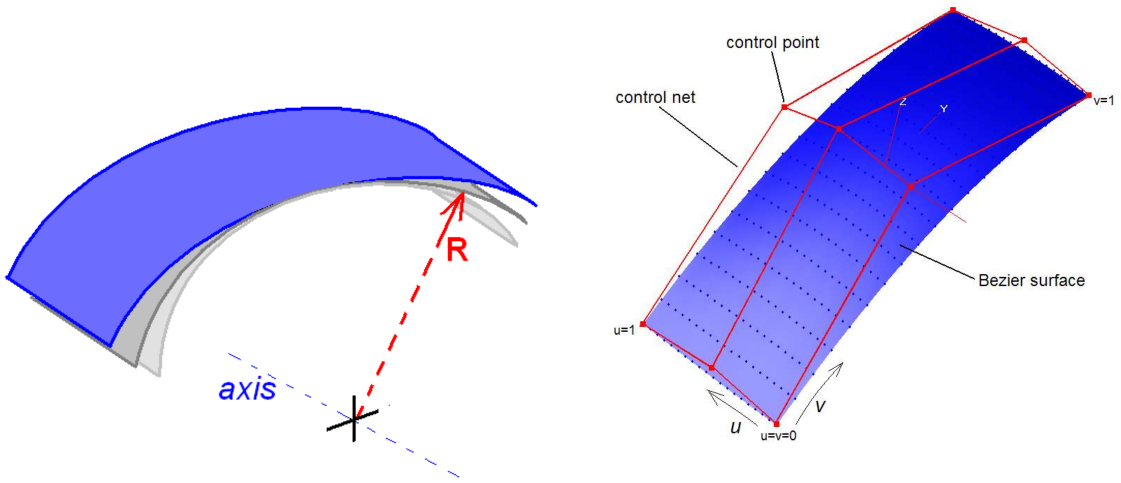


Fig. 3.5 Cylindrical surface (left), Bezier surface (right)

3.2.2 Surface fitting

When the appropriate surface (and its number of parameters) is chosen, it can be fitted onto the point cloud of nodes in the reference and springback meshes. In any case, fitting the surface means the basic optimisation problem of minimizing the sum of the distances from each node to the surface. The least squares method is applied here. For a certain set of parameters, the distances from each node to the surface are squared and summed. The result is the objective function Q, dependant on the set of node coordinates and the surface parameters. The optimum parameter set can be found at the spot where Q has its global minimum.

There are numerous multivariate function minimisation algorithms, such as the Downhill Simplex Method, Powell’s Method and even evolutionary algorithms. All strategies have one major problem: they will find a minimum, but that may not be the global minimum (i.e. the parameter set for the best fitting surface). Therefore, the user has to provide a reasonable first guess for the algorithm to find the right minimum. This is straightforward with a cylindrical surface, but it can be very complicated with a ‘wobbly’ Bezier surface. The surface fitting process can be made more stable by slowly increasing the number of parameters: The fitting process is started with a surface defined by a very small amount of parameters. This surface is fitted, and the number of parameters is increased. With this surface a new fitting procedure is started using the old surface as a starting guess. In the same way, the number of parameters is increased iteratively until the surface is fitted with the desired accuracy

The definition of an initial guess is important for the fitting of the reference surface only. Because the reference and springback meshes are only slightly different, the resulting reference surface can used as a starting guess for the fitting of the springback surface. This makes the fitting process substantially faster and more robust. When during the fitting of the reference surface a local minimum is found instead of the global minimum, a ‘wrong’ surface is fitted. Still, the solution for the springback surface will probably converge to an ‘identically wrong’ shape and the compensation algorithm might even come up with a good compensation (but with more geometrical strain, see §3.2.3)

3.2.3 Calculation of the transformation surface and coordinate transformation

With the parameter sets for the reference and springback surfaces, the next step is to find a good transformation surface. As explained in paragraph 3.1, the transformation surface is constructed in the same way as the standard DA method is applied to meshes. The springback deformation, modelled by the difference between the springback and reference surfaces, is multiplied with the (negative) overbending factor, rendering the transformation surface. For clarity, picture 3.2 is repeated here:

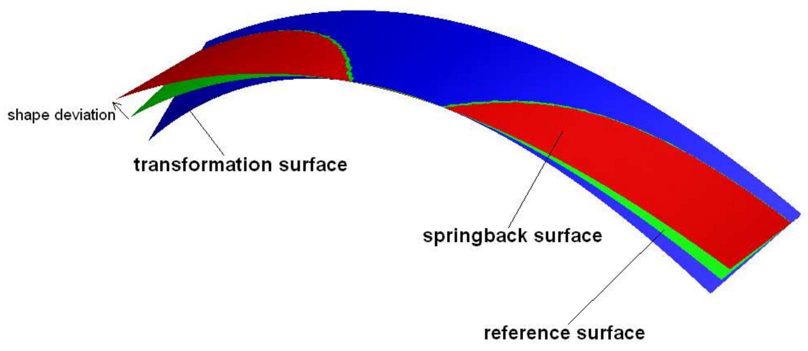


Fig. 3.6 calculation of the transformation surface

The challenge is to define the ‘right’ overbending factor. Generally, the algorithm will perform several iterations to achieve the desired product geometry. If the overbending factor is taken too large, the shape will be overcompensated in each iteration, and the optimisation process may become unstable. If the overbending factor is taken too low, many iterations will be needed to obtain the optimum die shape. This will make the process very slow, because for each iteration, an expensive FE simulation is carried out. As pointed out earlier, the industry rule of thumb is to overbend with a factor of -130%. This means, a dimensional deviation of 10 mm between the reference and springback surface means a compensation measure (in the opposite direction) of 13mm. This is really a very global number; the overbending factor can be very different for each product. It is possible to change the overbending factor adaptively during the optimisation process, but this is only effective with a large number of iterations.

With the reference surface and the transformation surface the shape modification can be carried out. The general idea behind the shape modification is explained most clearly in two dimensions. In picture 3.7 the process is visualized. Node *p* is a node in the tool mesh that is modified. First, the vector that is normal to the reference surface and points at node *p* is sought. The length of this so called *offset-*vector is called the *offset.* Note that the offset variable has a negative value when the node p is located ‘underneath’ the reference surface. The location of the offset vector on the reference surface is the second variable to define the location of the node relative to the reference surface. In this 2D case, the arc length along the surface can be used. We have now defined the location of the node in a ‘quasi-Cartesian’ coordinate system along the reference surface. To find *p”,* the location of the node *p* in the modified mesh, the arc length is now laid out along the transformation surface. At the end of this arc length a new normal vector with the length equal to the offset value is constructed. This vector points at the modified node *p”*. Note that it is irrelevant which side of the reference surface is defined as the underside, as long as both reference and transformation surface have the same definition.

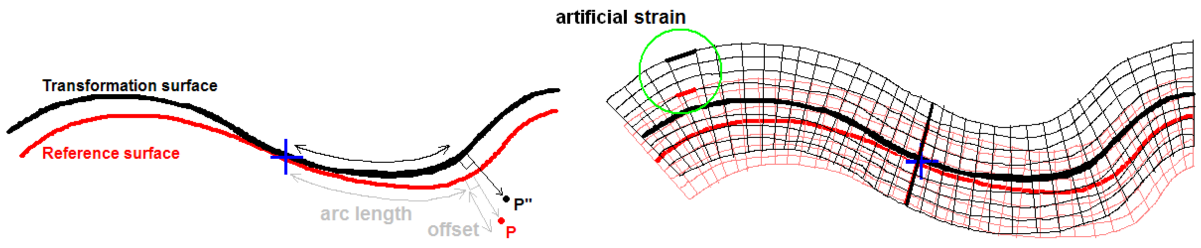


Fig. 3.7 mesh modification principle

A curve segment, located on the reference surface, will have the exact same length in the transformed coordinate system. But when the offset of this curve segment is not zero, it will be stretched or compressed during coordinate transformation, inducing *artificial strain* (the thick red and black curve segments in the green circle of picture 3.7 have different lengths). This means that the modified geometry will be artificially deformed. However, the smoothness of the geometry, as well as its details will be preserved very well, because the shape change is very even over the whole structure. Because of this, the shape modification can be performed on CAD files as well, as will be demonstrated extensively in chapter 6.

When the control surface is defined as a Bezier or B-spline surface, it is impossible to use the arc length principle to define the location of the offset vector on the reference surface. Therefore, a local coordinate on the surface is used instead. How this works is explained in chapter 4.

3.3 The 2D circular arc algorithm

In this paragraph, the CS algorithm is implemented with a cylindrical control surface. The centre of the single surface radius is located at the z-axis, which means that the surface can be fully defined by two parameters: a radius *R* and height of the centre point *h.* This is visualized in picture 3.8. the blank mesh of a structural part is visualized in pink, the control surface with thin red lines. By changing the radius R, the curvature of the surface is changed, it is 'bent' around the x-axis. The shape optimisation possibilities are too limited for practical application, but because the mathematics behind this type of surface are straightforward, the optimisation process can be made clear.

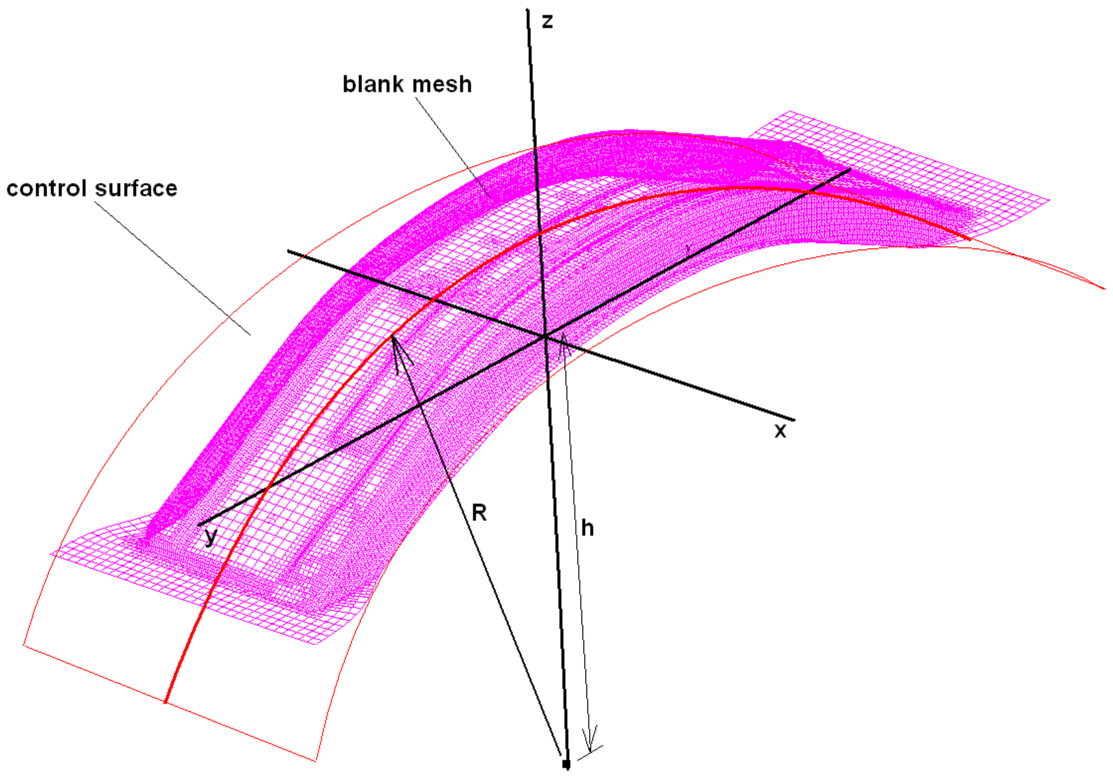


Fig. 3.8 the single bent control surface

3.3.1 Fitting the cylindrical surface with the Downhill-Simplex method

As explained in paragraph 3.2.2, for curve fitting, the least squares distance function needs to be minimized. This is a multivariate minimisation with parameters *h* and *R*. The Downhill Simplex method, developed by Nelder and Mead in 1965, is chosen. “The downhill simplex method may frequently be the best method to use if the figure of merit is ‘get something working quickly’ for a problem whose computational burden is small.”[Pre92].

The idea of the method is to take a polygon of N+1 points (vertices) for an N-dimensional minimisation problem. This polygon is called a *simplex.* A start-simplex is defined by the user, and then the simplex ‘walks’ downwards across the function until it finds a minimum. There are four basic simplex movements: reflection, reflection and expansion, contraction and multiple contraction. We will explain the most important movement, reflection, here. More detailed information about the algorithm can be found in [Pre92].

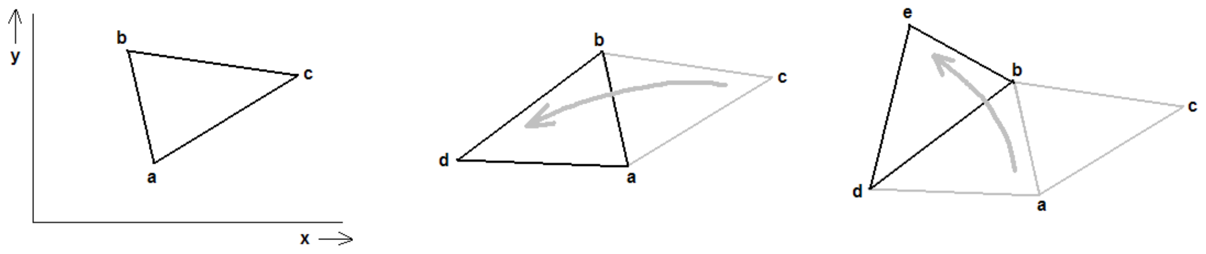


Fig. 3.9 the Simplex reflection

In figure 3.9, ABC is the start-simplex for a two dimensional problem. The function is evaluated in those three points. When point C turns out to have the highest value, the point is mirrored across line AB. Then a new function evaluation is carried out for point D. When point A now has the highest function value the triangle is flipped across line BD. This process is continued until the value of the target function does not change beyond a certain accuracy anymore.

The definition of the start-simplex is not mathematically defined. Basically any set of coordinates can be chosen, but the end result does not necessarily need to be the global minimum of the function. For a circular arc the values of *h* and *R* can be ‘guessed’ from an image of the mesh. We have chosen to take this point as the centre of the triangle, and locate the other points around it with a user defined range for *h* and *R* separately. There is no standard strategy for generating a start-simplex, so other ideas can be considered when problems might occur.

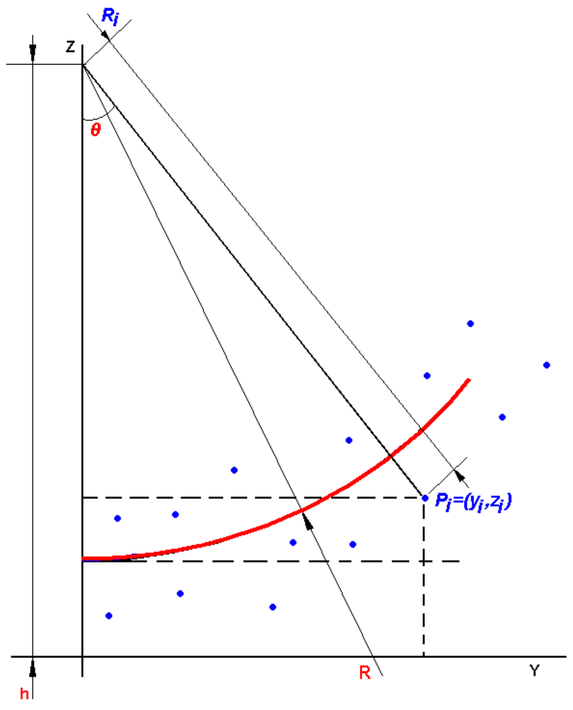


Fig.3.10 Curve fitting

A target function Q is defined, summing up the quadrate of the distances between a point and the curve as follows:

$$Q(R,h)=\sum_{i=1}^{n}(R-R_i)^2=\sum_{i=1}^{n}\left(R-\sqrt{(h-z_i)^2+y_i^2}\right)^2 \qquad (3.1)$$

with R_i as the radius of the i-th node in a pole-coordinate system around the centre point (defined with the parameter h). The variables are shown in picture 3.10.

The curve fitting turned out to be robust as fast, which was expected because only 2 parameters are to be found. When more variables need to be found, the Nelder-Mead method may become unreliable since the parameters will not necessarily converge in the vicinity of the start values.

3.3.2 Transformation proposition

With the now available parameters for the reference and springback surfaces, a proposition can be calculated for a transformation surface. The distance between the reference and springback surface is defined as in picture 3.11. The parameter L_{ref} is the largest of the nodal y-coordinates, and represents the width of the product. Now, the arc length along the reference surface is calculated. The endpoint of the springback surface is determined by moving the same arc length along the springback surface. The distance between the two endpoints is calculated. Note that the springback surface is displaced in z direction so, that the minimum of the springback surface is coincident with the minimum of the reference surface.

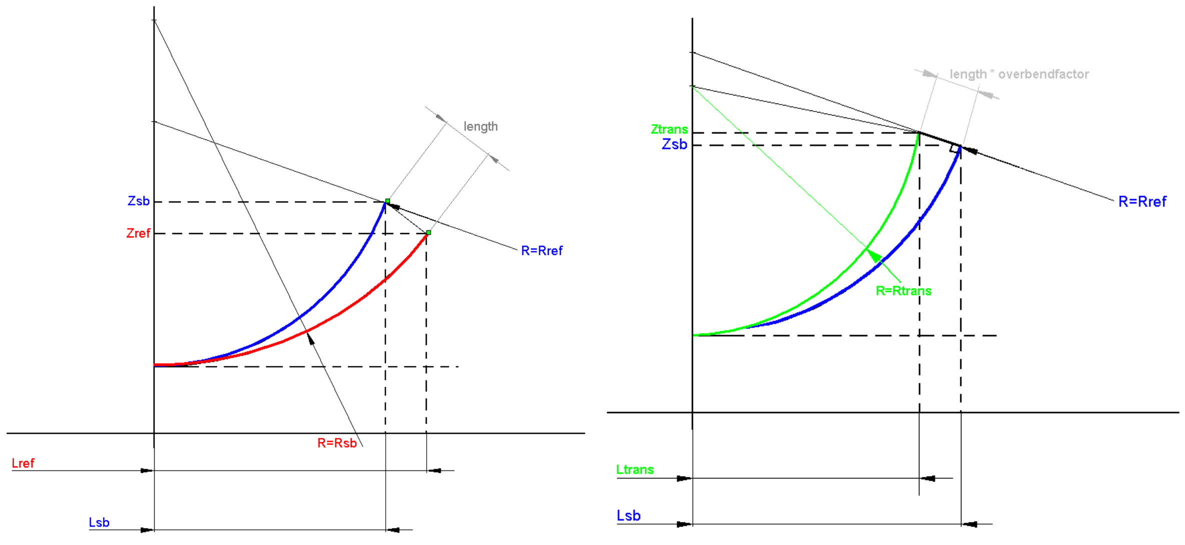


Fig.3.11 determination of the distance between the surfaces (left) and creation of a transformation surface (right)

With this length, the transformation surface can be defined as visualized in picture 3.11 (right). A normal vector with this length, multiplied by a certain overbending factor (1.0 meaning 100% overbending), is placed at the end of the reference surface. It points at the endpoint of the transformation surface. Again, the minimum of this surface is coincident with that of the reference curve. Now, the circular curve is fully defined and the transformation radius R_{trans} can be calculated.

It is possible to calculate an endpoint of the transformation curve in the same way as the calculation of the *length* value, i.e. without the normality constraint for the vector. This results in a nonlinear equation, which can only be solved numerically. We preferred a simpler approach, because in practical situations *length* is very small compared to R_{ref}.

3.3.3 Modifying the mesh

Finally a coordinate transformation is carried out, with the R_{trans} and h_{trans} values. The new location is determined by of point P can be calculated as follows; First, the two polar coordinates R and ? of the i-th node are calculated in the reference coordinate system:

$$_i \quad \arctan\frac{y_i}{h_{ref} \quad z_i} \qquad (3.2)$$

$$R_i \quad \sqrt{y_i^{\,2} \quad h_{ref} \quad z_i^{\,2}} \qquad (3.3)$$

The offset, as defined in §3.2.3 can be easily calculated:

$$offset_i \quad R_i \quad R_{ref} \qquad (3.4)$$

In the polar coordinate system around h$_{trans}$, the new location of point P is given by $\hat{P}_i(\ _i, S_i)$.

The first parameter can be found by taking the same arc length, as found along the reference curve, along the transformation curve. It can be easily derived that:

$$_i = \frac{Rref}{Rtrans} \ _i \qquad (3.5)$$

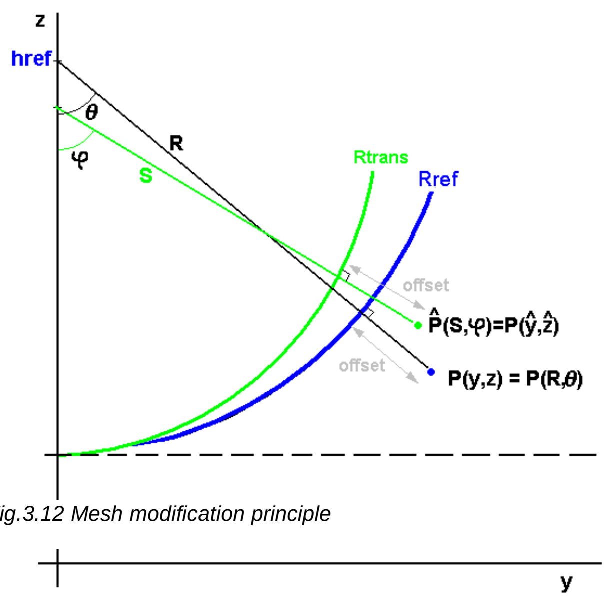


Fig.3.12 Mesh modification principle

The *S*-value can be derived by adding the offset to the transformation radius

$$S_i \quad Rtrans \quad offset_i \qquad (3.6)$$

Now the coordinates are transformed back to the Cartesian coordinate system:

$$\begin{aligned} \hat{y}_i &\quad S_i \sin\varphi_i \\ \hat{z}_i &\quad S_i \cos\varphi_i \quad href \quad Rref \quad Rtrans \end{aligned} \qquad (3.7)$$

3.3.4 Example

The algorithm is now complete. It has been tested with a reference product. In picture 3.13 below, the reference mesh is displayed, and its approximation curve in green. The orange curve represents the approximation of the springback mesh. Both approximations take less than a second. As expected, the curve for the springback mesh has a larger radius.

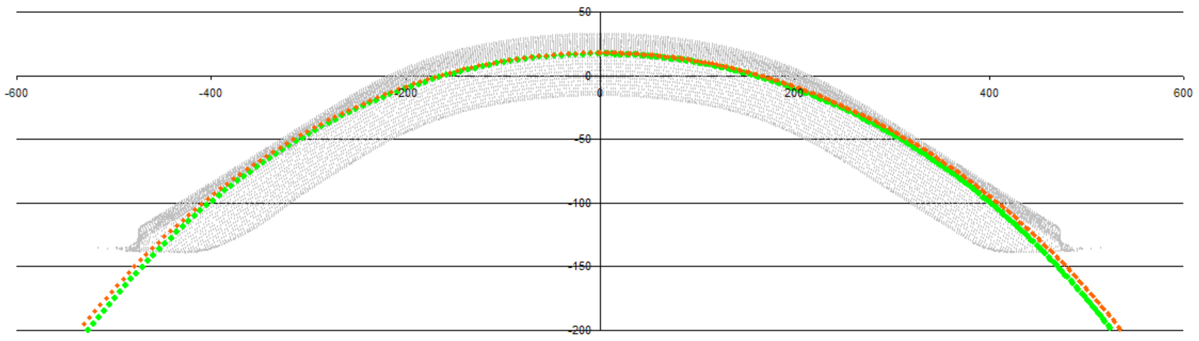


Fig.3.13 determination Approximation of a real product (courtesy of DaimlerChrysler)

From these two curves, a transformation surface is calculated, with an overbending factor of 5.0. This is not realistic, but it clears up the picture and shows that the product shape can be drastically bent without large unwanted deformations. A new simulation can now be carried out to check whether the algorithm was successful. Obviously, the cylindrical control surface is way too coarse for any springback problem, so we did not carry out a new FE calculation.

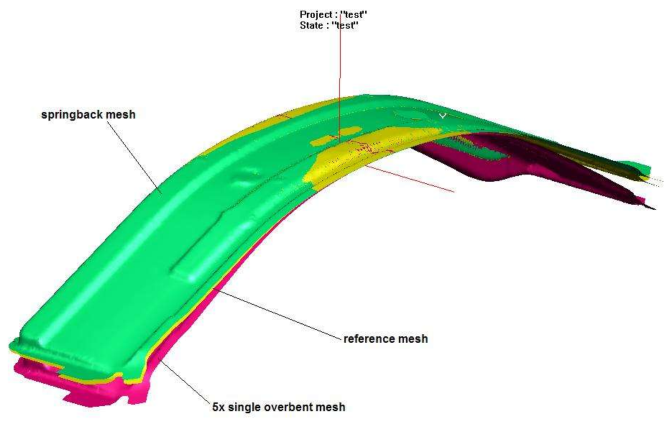


Fig.3.14 Overbending the DC structural part with the algorithm

4 The 3D Bezier surface algorithm

For industrial applications, the algorithm presented in chapter 3 will be too limited. To compensate complex springback deformations, a more flexible algorithm is developed. The circular control surface from the previous algorithm has been replaced by the more flexible Bezier and B-spline surfaces. The principle of the algorithm remains the same, but the mathematics behind it cannot be described with simple formulas anymore. In this chapter, the procedures in the algorithm, and the mathematics behind it, are explained in detail. Therefore this chapter also serves as a reference manual for the algorithm.

The first paragraph serves as an introduction to parametric geometry mathematics. Here some basic concepts behind Bezier and B-spline surfaces are introduced, which will be used frequently in the following paragraphs.

In paragraph 4.2, the procedure of the algorithm is explained in detail. The procedure is split up in separate steps, which are discussed in separate paragraphs. The first step is the definition of a suitable Bezier control surface. The user can define the complexity of the surface and find a compromise between algorithm stability and compensation accuracy. In the second step, the surfaces are fitted. The procedure that is followed is described. Because the mathematics of the surface fitting procedure are complicated, these are discussed later in paragraph 4.3. The third step is to calculate the transformation surface. How the reference, springback and transformation surface are mathematically connected is shown here. Finally, the tools are modified. The modification of geometry is now based on Bezier surfaces, and is a bit more involved.

As pointed out, the procedure for fitting a parametric surface through a point cloud (in the second step) is not a trivial task. How this is achieved, is explained in paragraph 4.3. It is also possible to perform local springback compensation in a part of the product. In some cases, only this part of the product has stringent geometrical requirements. Local springback compensation can also be used to raise the accuracy of the algorithm: after a global compensation, the remaining problem areas can be treated individually and with higher accuracy in following iterations. How this is implemented is presented in paragraph 4.4

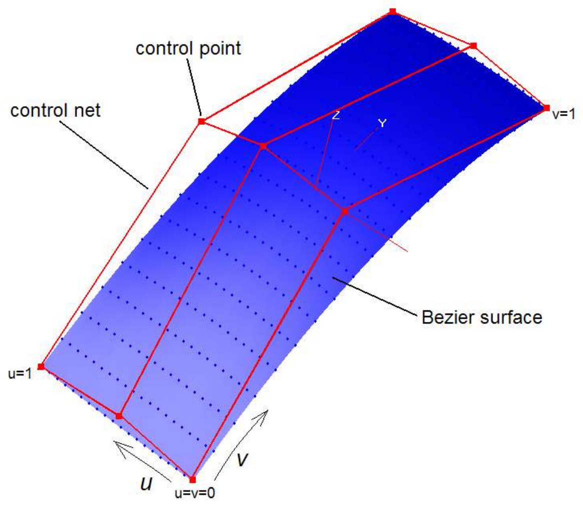


Fig. 4.1 A quadratic by quadratic Bezier surface

4.1 Basic parametric geometry mathematics

Curves and surfaces can be presented in implicit and parametric functions. For implicit equations, the geometry is defined by a (generally nonlinear) function of the coordinates (x, y, z)

$$f\ \ x,y,z\quad 0 \tag{4.1}$$

For parametric geometry, each coordinate value is represented by a separate function of one or more parameters, providing more flexible geometric modelling and manipulation possibilities:

$$C\,x,y,z\quad x(u),y(u),z(u) \tag{4.2}$$

The Bezier curve is the most straightforward parametric curve. It was developed in the 1960s, to describe the complex geometries of car bodies. A Bezier curve of the n-th degree is controlled by a *control polygon* consisting of *n+1 control points* P_i. The curve has one parameter, *u*, ranging from 0 to 1 along the curve. For the calculation of the coordinate values of the curve at a certain parameter value, *n+1* blending functions ($B_{i,n}$) are evaluated. The curve's coordinate values are found by multiplying each of the control points by their respective blending function and summing the results. Bernstein polynomials (eq. 4.4, fig 4.2) are used as blending functions

$$\vec{C}(u) = \overset{n}{\underset{i=0}{}} B_{i,n}(u)\vec{P}_i \qquad 0 \quad u \quad 1 \tag{4.3}$$

$$B_{i,n}(u) = \frac{n!}{i!(n-i)!} u^i (1-u)^{n \ \ i} \qquad 0 \quad i \quad n \tag{4.4}$$

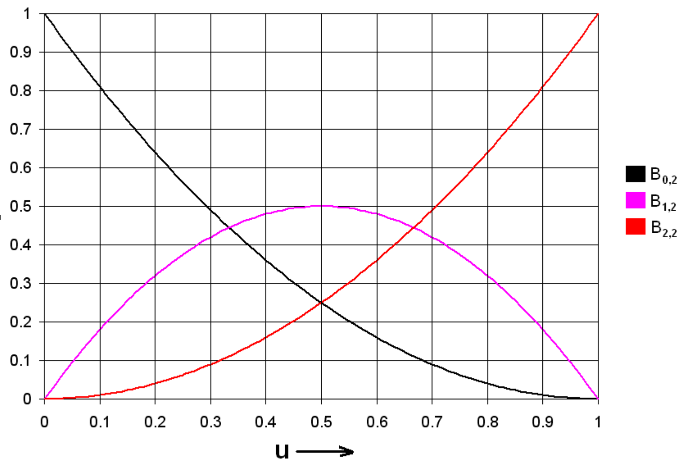


Fig. 4.2 The three Bernstein blending functions for a quadratic Bezier curve

This concept is expanded for modelling Bezier surfaces. For surfaces the tensor product scheme is most widely applied. The basis functions are now bivariate combinations of univariate basis functions. The following function is the description of an *n* by *m*-th degree surface:

$$\vec{S}(u,v) \quad \sum_{i=0}^{n}\sum_{j=0}^{m} f_i(u) g_j(v) \vec{P}_{ij} \qquad f(u)^T \left[\vec{P}_{ij}\right] \left[g(v)\right] \quad 0 \quad u \quad 1 \quad 0 \quad v \quad 1 \tag{4.5}$$

Where *P* is a *n+1* by *m+1* matrix of three dimensional control points and *f* and *g* are *n+1* and *m+1* vectors containing the Bernstein blending functions for both parameters. Note that S is a vector, containing x, y and z coordinates, the two blending functions are applied to vectors as well. The control points form a control net, as visualized in picture 4.1. A Bezier curve, or surface, is always bounded, and defined on the parameter range only.

A Bezier surface consisting of one single polynomial segment (*patch*) has its limitations. When the surface is fitted through a point cloud, and a large number of constraints need to be satisfied, a very high degree is needed. Higher degree functions are generally numerically inefficient and possibly unstable. Therefore piecewise polynomial surfaces have been developed. The parameter region is then subdivided into different regions. For example: in the parameter range of a cubic Bezier curve, 2 extra breakpoints have been defined:

$$u \quad u_0, u_1, u_2, u_3 \qquad u_0 \quad 0, u_3 \quad 1 \tag{4.6}$$

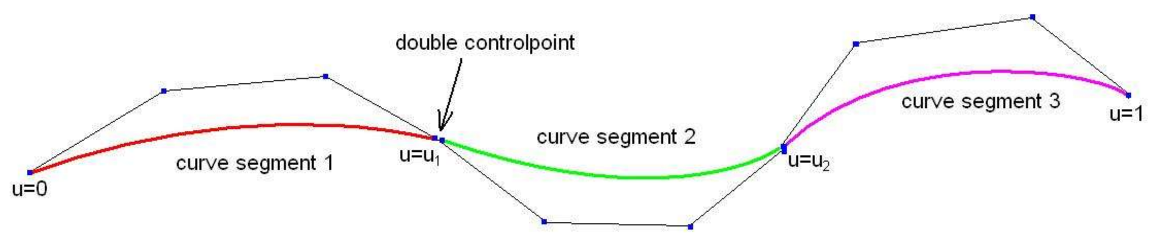


Fig. 4.3 A C-0 continuous combination of 3 Bezier curves

Now, the curve is built up with three separate cubic curve segments, each having 4 control points. It is obvious that the resulting 12 control points cannot be chosen independently, because the curve needs to be at least C0 continuous (uninterrupted). The last control point of a segment must be identical to the first of the following segment. When the transition area is required to have a higher level of continuity, even less independent control points are required. This system has two disadvantages:

- Redundant data needs to be stored for the constrained control points
- Because of the constraints on the movement of the control points, shaping the curve can become unintuitive.

To solve these problems, B-spline curves and surfaces have been developed. The blending functions have been modified to consist of more than one polynomial. The control points are defined so, that no redundant data needs to be stored, and that curve shaping remains relatively easy. The shape functions are nonzero only on a limited number of subintervals. This means that, when a control point is moved, only a limited part of the curve is changed. Instead of splitting the parameter range, a so called ‘knot vector’ *U* is used, and the basic form of the curve and surface descriptions as in equations 4.3 and 4.5 is maintained.

$$U \quad u_0,....u_m \quad u_i \quad u_{i+1}, i \quad 0,...,m \quad 1$$

The *i*-th, *p*-th degree B-spline blending function can now be calculated recursively with the following formula:

$$N_{i,0}(u) \quad \begin{array}{ll} 1 & u_i < u < u_{i\ 1} \\ 0 & otherwise \end{array} \qquad (4.7)$$

$$N_{i,p}(u) = \frac{u-u_i}{u_{i\ p}-u_i} N_{i,p\ 1}(u) + \frac{u_{i\ p\ 1}-u}{u_{i\ p\ 1}-u_{i\ 1}} N_{i\ 1,p\ 1}(u) \qquad (4.8)$$

As an example, the blending functions for a piecewise linear B-spline are shown in picture. Note that in this case the range of the *u*-parameter is taken as [0,5].

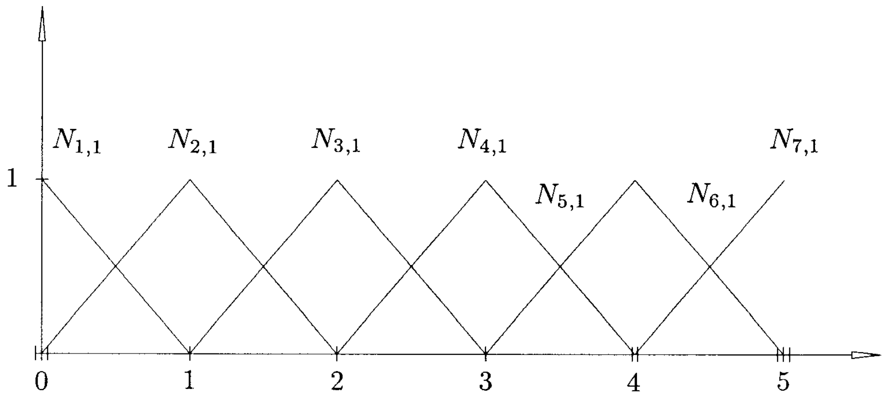


Fig. 4.4 A The nonzero linear B-spline blending functions, U={0,0,0,1,2,3,4,4,5,5,5} picture taken from [Pie97]

The B-spline surface is now defined as follows:

$$S(u,v) \qquad N_p{}^T \; \left[P\right] \; \left[N_q\right] \tag{4.9}$$

Altering values in the knot vector *U* is a powerful yet unintuitive way to control the shape of the surface. For example: one knot value can be placed more than one time in the knot vector, producing a cusp. Generally, for each added coincident knot, the curve's level of continuity is lowered by one at this location. When enough coincident knots are used, the curve or surface can even be 'cut' into two separate pieces.

The (*u,v*) parameterization can be used as some sort of local coordinate system along the surface. For Bezier surfaces, the coordinate system is uniformly spaced. This is not necessarily the case for B-spline surfaces.

For an in depth discussion on Nurbs mathematics, [Pie97] is recommended

4.2 The algorithm procedure

4.2.1 Step 1: Definition of a suitable control surface

In the algorithm setup file, the degrees of the surface in u and v direction are specified. As explained before, the control surface defines the compensation possibilities for the algorithm. In picture 4.5 a linear by quadratic surface is shown. It can be used for (asymmetric) bending and applying torsion. To make the application of camber possible, at least a quadratic by quadratic surface is needed. With these surfaces most global springback problems can be compensated.

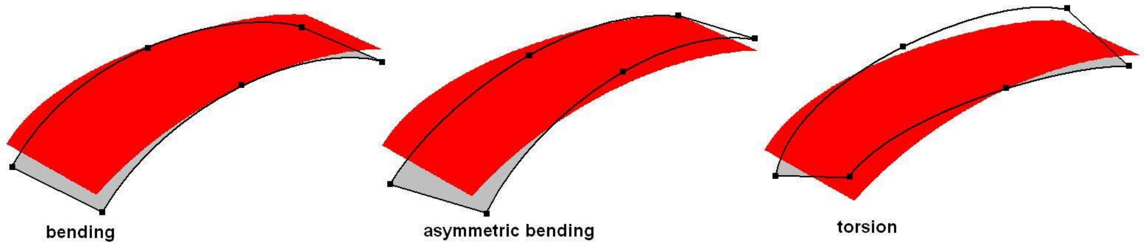


Fig. 4.5 deformation possibilities of a linear by quadratic surface

The relatively simple linear by quadratic surface is defined by 6 control points, so 18 coordinate values are needed to define the shape. This presents a major problem for the surface fitting: the control point coordinates are heavily dependant, and a number of 18 free variables means that the surface fitting may become unstable already. Therefore, to reduce the number of surface fitting variables, each control point is allowed to move along a defined (3D) line only. The location of the control point can now be described with a single coordinate, called *location*, along this line. The line is described by a *base point*, and a unity-vector called *direction.* In picture 4.6 a linear by quadratic Bezier surface is shown with its 6 basepoints and six directions. Before the surface is fitted to approximate a certain mesh, these variables are defined.

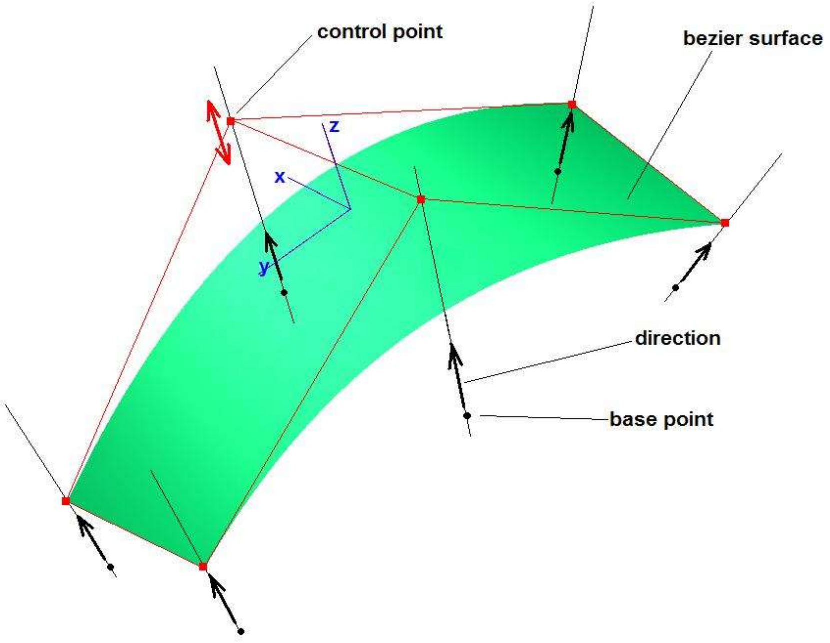


Fig. 4.6 a linear by quadratic surface, its basepoints and directions

4.2.2 Step 2: Fitting of the reference and springback surfaces

The reference and springback meshes need to be oriented so, that the deep drawing direction is (roughly) parallel to the z-axis. This is done to avoid complicated positioning problems. Around the reference mesh, a bounding box is created using the maximum and minimum node coordinate values in x, y and z directions. The base points are laid out as an equispaced grid on a plane in the bounding box, as shown in picture 4.7. The plane is parallel with the x-y plane. The direction vectors are set in parallel to the z-axis.

As an initial guess, the locations (i.e. the position of the control point on the line along which it can be moved) are set to a user-specified value. When a good graphical user interface is available, the user can drag the control point so, that the surface fits through the geometry roughly. The curve fitting algorithm will start at this point and find the 'correct' surface. This is especially important when a surface with more control points is used. Alternatively, the algorithm can be used fully automatically. Then, a B-spline *startvalue surface* consisting of linear by linear surface-segments, with the same configuration of control points is fitted first. For example: when the user has defined that a cubic by quadratic control surface with twelve control points is to be used, this will be a surface with 6 linear by linear patches. Because the piecewise linear function, defining the startvalue surface is very stable and the control points are completely independent, this fitting process is very robust. For this surface fit, the starting locations are all set at the middle of the bounding box, in z direction. After the very rough fit, the *locations* of the control points can be used as a starting guesses for the reference (Bezier) surface that is fitted.

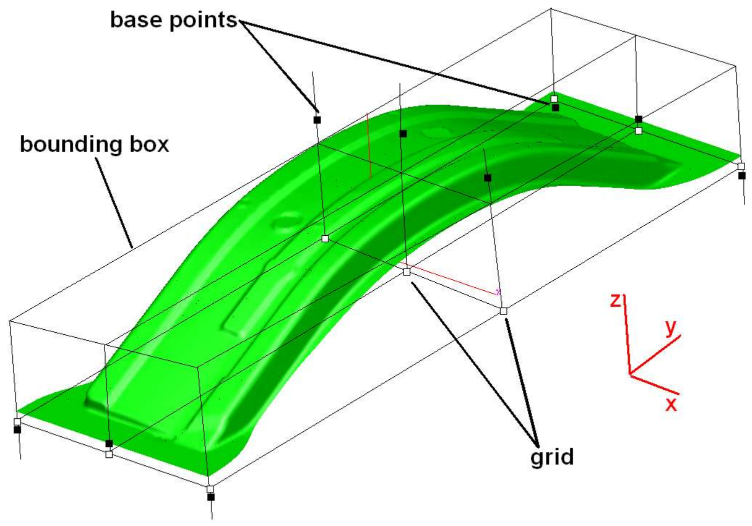


Fig. 4.7 Bounding box and grid

With the set of starting values, the reference surface is fit. It is recommended to check whether the surface fits the geometry well. After the fitting of the reference surface the springback surface needs to be fitted. The control points of the reference surface are now used as the starting values for the surface fitting. The control points of the reference surface are also set as the base points of the springback approximation. The vector normal to the reference surface is used as the *direction* vector. Basically, the springback surface is now fitted by moving the control points along a normal vector on the reference surface, as shown in picture 4.8.

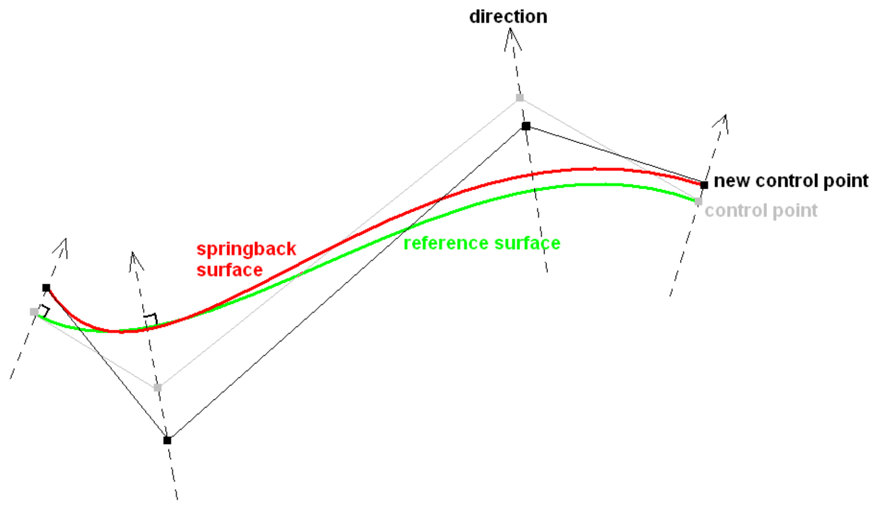


Fig. 4.8 Normal displacement for springback surface fitting

4.2.3 Step 3: Proposition for a transformation surface

When the reference and springback surface have been approximated with surfaces, a transformation surface can be proposed. During springback surface approximation, the control points were moved along a vector normal to the reference surface. The transformation surface is now constructed by taking the distance from the springback control points to the reference surface control points (along the direction vector), and placing the control points for the transformation surface in the opposite direction. In picture 4.9, the left side of the previous picture is shown in detail. The distance between the reference surface control points and the transformation surface control points is identical for 100% overbending, but is normally multiplied by the overbending factor.

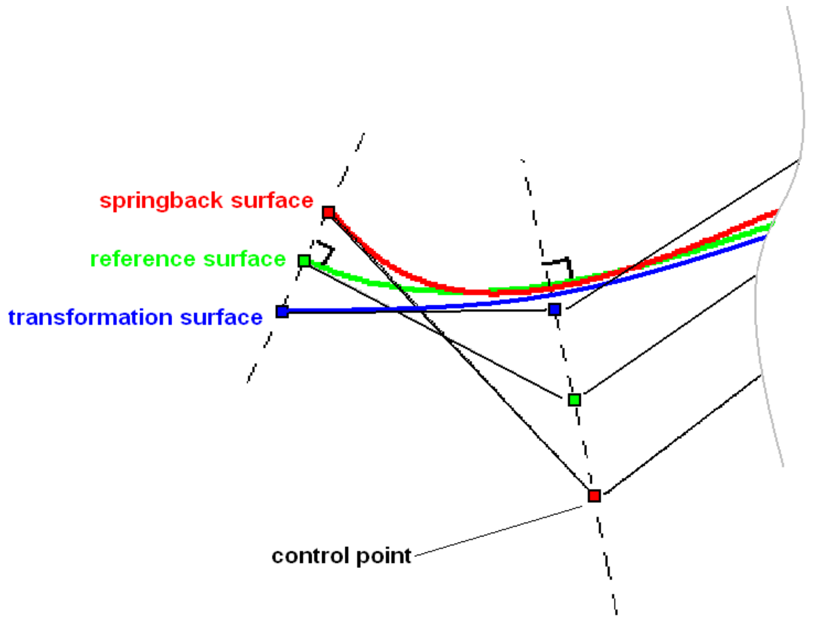


Fig. 4.9 construction of the transformation surface

When the overbending factor is set identically for each point, a global compensation is carried out. The overbending factor can also be set differently for each control point. This opens up a lot of possibilities.

The overbending factor can be set to zero for certain areas of the product geometry that need to remain unchanged. Because of the fact that a Bezier surface retains its level of continuity irrespectively of the position of the control points, the transition zone in between the area that is modified and the area that remains unchanged will remain smooth. Also, the product can be divided into several springback problem areas, with different overbending factors for each area. Using different compensation factors locally will only be of practical use when the grid of control points is sufficiently fine. This presents a major problem for the curve fitting algorithm, which may become instable. A solution is to fit a global surface first, and then refine the surface later, adding extra control points locally. This will be discussed in paragraph 4.4.

4.2.4 Step 4: Transformation of the tool geometry

Now that the transformation surface is known, the product can be transformed. In the cylindrical surface version of chapter 3, the arc length could be used to define the location of the offset-vector. Also, the control surface could be 'bent' without introducing geometrical strain because the surface is 'developable' (i.e. deformable/bendable without introducing strain). This is generally not the case for a 3D surface description. To define the location of the offset factor, now the parameterisation of the surface is used. The Bezier surface's *u* and *v* parameters can be seen as some sort of 'warped coordinate system' along the surface.

The principle is shown in picture 4.10. The normal vector on the reference surface, pointing at node P is determined. Point Q is located where the vector starts on the reference surface, and the*u* and *v* parameters of this point are calculated. The calculation of the parameter values can only performed numerically, with a Newton iteration scheme. (This is explained in detail in section 4.3.2). The point Q' is found by determining the point with the same u and v parameters as Q, but now located on the transformation surface. The offset vector is placed normally at Q and points at the transformed node, P'.

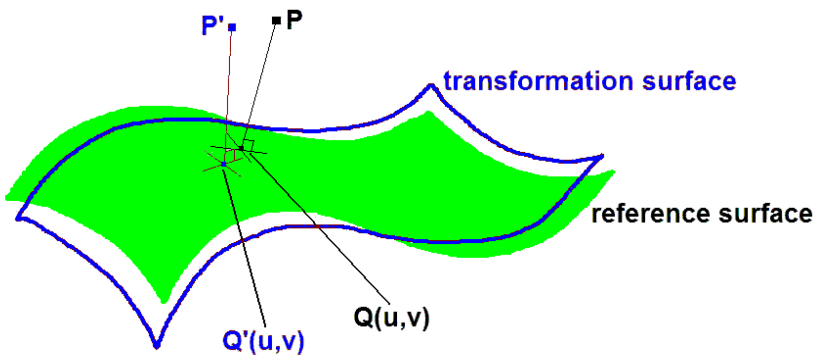


Fig. 4.10 Mesh modification with control surfaces

When geometry is modified in this way, artificial strain is introduced. This strain can be divided in two categories:

- Strain in the control surface itself. As pointed out, a parametric control surface cannot generally be modified without introducing artificial strain in the surface itself.
- Strain that is cause because the geometry is not located exactly on the surface. This effect was explained in paragraph 4.2.3

4.3 Fitting parametric surfaces

In this paragraph, the mathematics behind surface fitting are explained in detail. For all surface fitting processes, the target is to find the right location-values for each control point in the surface. As explained

4.3.1 Introduction

For Bezier or B-spline *curve* fitting, many algorithms have been developed. In most cases, the control points are chosen as the free variables in the fitting procedure. As parametric geometry has many shaping possibilities and many degrees of freedom, care has to be taken in choosing the right boundary conditions. In many cases the curve, or surface, can obtain an unwanted shape and still be 'mathematically correct'.

A data *interpolation* algorithm is most easily explained. The basic procedure is as follows:

- Sort the data cloud so, that the points are in the right sequence (i.e. along the curve)
- Choose a knot vector (for B-spline)
- Assign a 'sensible' parameter value *u* to each point in the data cloud.
- Solve the system of linear equations and find the control points

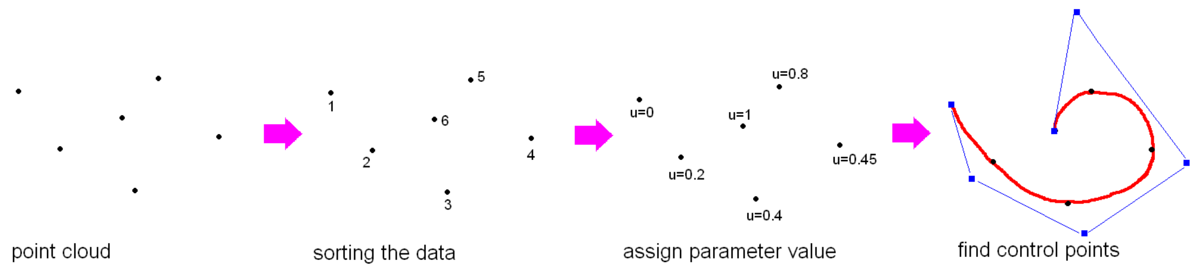


Fig. 4.11 data interpolation

When a data cloud needs to be *approximated* with a parametric curve, the problem becomes more complex. Now, much more data points are supplied than needed for a full curve definition. Because approximation is a costly process, a selection of points is made, and a curve is fitted using the 'least squares' principle. The deviation of the curve from the data is checked. If this deviation is larger than a specified tolerance, more data points are added to the set and a new curve is fitted. This procedure is repeated until the curve is fitted with the required tolerance. A fast and stable algorithm has been developed [Pie97, p.410].

When a surface needs to be fitted, there is no general principle for sorting the data cloud, and assigning a sensible parameter value to data points. To solve this problem, we have developed an algorithm that works in a more straightforward way. Again, the knot vector is fixed, and (a selection of) the locations of the control points are the free variables. The target is now to minimize the distances from each node to the surface. Two minimisation methods have been tested, they will be discussed in §4.3.3. This minimisation process is not without problems. As stated in §4.1, altering the position of a control point has effect on only a (limited) range of the surface. The influence of neighbouring control points overlaps, and therefore the free variables are not fully independent.

4.3.2 Point on surface projection

In order to calculate the target function for a certain set of variables (control point coordinates) the distance from each node to the surface has to be calculated. The basis of the solution is a*point inversion* algorithm, which calculates the *u* and *v* parameters for a given point *on* the surface.

Calculating the two parameters from a point given in three Cartesian coordinates is an overdefined problem:

$$\begin{array}{ll} x & f(u,v) \\ y & g(u,v) \\ z & h(u,v) \end{array} \qquad (4.10)$$

This means that an exact solution cannot be found because the point is never *exactly* on the surface. Therefore, a numerical tolerance has to be specified. In theory, this calculation is solvable in closed form for surfaces with lower degrees. Because the degrees of the surface are defined by the user, more flexibility is needed and a numerical algorithm has been chosen for this calculation. This algorithm searches the point on the surface that is *closest* to the input point. This point does not need to be on the surface exactly, and we therefore refer to this process as point on surface projection. With this closest point, the distance between the input point and the surface can be calculated.

For a curve, with parameter *u* only, the distance between point *P* and the curve *C(u)* is minimal when the function

$$f(u) \quad C'(u)\ (C(u) \quad P) \tag{4.11}$$

equals zero. This point is found using the Newton iteration scheme, with a good initial guess for *u*:

$$u_{i+1} \quad u_i \quad \frac{f(u_i)}{f'(u_i)} \tag{4.12}$$

Convergence is reached when the distance of the points is within a certain tolerance:

$$\left|C(u_i) - P\right| \leq \tag{4.13a}$$

or when the vector pointing at point *P* is normal to the surface:

$$\vec{C}'(u_i) \perp (\vec{C}(u_i) - \vec{P}) \Rightarrow \frac{\left|\vec{C}'(u_i) \cdot (\vec{C}(u_i) - \vec{P})\right|}{\left|\vec{C}'(u_i)\right| \cdot \left|(\vec{C}(u_i) - \vec{P})\right|} \leq \qquad \varepsilon \quad 0 \tag{4.13b}$$

For a surface, the solution is identical, but the mathematics are a bit more involved because of the two parameters; see [Pie97, p.230].

A Bezier or B-spline surface is a defined on a bounded area. The *u* variable is therefore checked and limited to the {0…1} range. If *u* does not change anymore during the process, then the closest point is on the edge of the curve and the process is halted. (picture 4.12, left) In our application this is problematic. The surface is sized so, that it fits the blank mesh best. But, the surface is also used to modify the toolset. The tools are significantly larger than the blank, and therefore, tool nodes may be outside the region where the surface is defined. In the program, the commercial NLIB NURBS-modelling library is used. The function for point projection has been changed so, that the surface is automatically expanded when a parameter is outside the [0,1] range. The extension is a ruled surface which is at least G1 continuous. G1 continuity is a less stringent boundary condition than C1: for G (eometric)1 continuity, the curve may be reparameterized to obtain C0 and C1 continuity. Care has to be taken, that the bounding box fits around the geometry well, because this expansion can become unstable, when the node is further away from the surface.

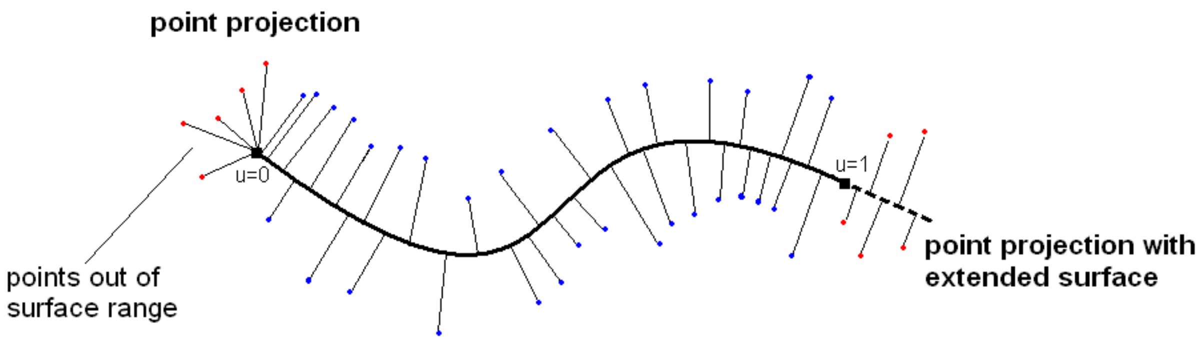

point projection
u=0
u=1
points out of surface range
point projection with extended surface

Fig. 4.12 normal (left) and extended point projection (right)

Finally, for each calculation, initial values have to be supplied for *u* and *v*. When the surface curvature is low, which will be the case with the generally rather flat control surfaces, starting guesses of 0.5 for both variables provide robust results. For fitting the reference surface (numerically most expensive), the x and y variables of the base points can be used to speed up the Newton process. How this works is explained in picture 4.13:

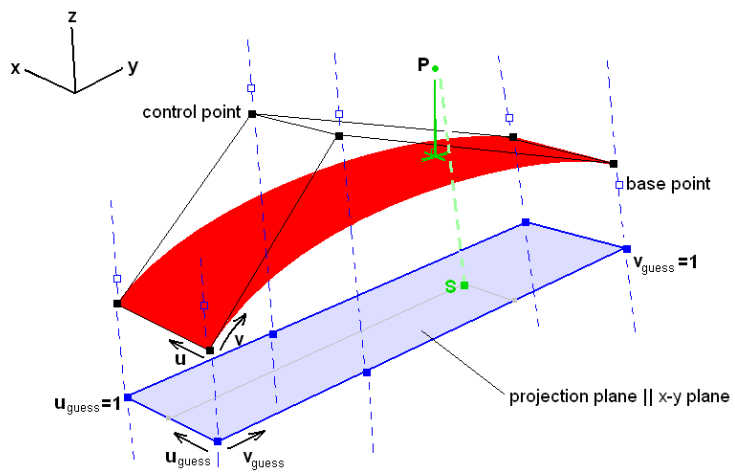


Fig. 4.13 finding a good start value for point projection

The base points are projected onto a plane, parallel to the x-y plane. The outer points of these projections span a rectangle, in which a local (u_{guess}, v_{guess}) coordinate system is constructed. As on the surface, the range for both parameters is [0..1]. The point P, to be projected on the surface, is projected on the projection plane, resulting in point S. The (u_{guess}, v_{guess}) 'coordinates' of S are calculated and used as start values for the Newton process. If the surface is relatively flat, the starting guess will be accurate. This speeds up the fitting process by around 30%.

4.3.3 Minimisation algorithms

The problem is to find the vector P, filled with the *locations* of the control points, for which the objective function

$$f(\vec{P}) = {}_{i=0}^{n} \left| N_i - Q_i \right|^2 \qquad (4.14)$$

has its minimum. In this function, N_i is the *i*-th node, Q_i the point on the surface closest to N_i, calculated with the point projection algorithm, so calculating the target function value is an expensive operation. Therefore, the minimisation algorithm needs to do the least possible amount of function calls. It also needs to remain stable with a large number of free variables. As pointed out, accurate starting points are needed to find the reference surface. For the springback surface, the initial values follow from the reference surface shape. In both cases the surface fitting algorithm needs to find a solution relatively close to the starting values. This reliability is more important than the numerical cost.

The Nelder-Mead process has already been explained in the previous chapter. For fitting the Bezier surface instead of the cylindrical surface, more parameters are needed. This makes a good starting simplex essential. [Pre92] suggests the following start simplex around P_0

$$P_i = P_0 + \quad e_i \qquad 1 \quad i \quad n \qquad (4.15)$$

with ?as the *characteristic problem length scale*. This parameter can be chosen small if the initial guess is known to be accurate. For a problem with a dimension of five (five free variables), and P_0 set as the origin, the simplex looks like this:

$$S_{start} = P_0 \quad P_1 \quad P_2 \quad P_3 \quad P_4 \quad P_5 = \begin{bmatrix} 0 & \lambda & 0 & 0 & 0 & 0 \\ 0 & 0 & \lambda & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda & 0 & 0 \\ 0 & 0 & 0 & 0 & \lambda & 0 \\ 0 & 0 & 0 & 0 & 0 & \lambda \end{bmatrix} \tag{4.16}$$

This start simplex makes the surface fitting process rather hard to control, because the exact role of the start value and ?is unclear. While testing the surface fitting, this turned out to be a problem, since the solution of the fitting algorithm converged to solutions further away from the initial point.

As a solution to this problem, the Powell multivariate optimisation algorithm has been included in the program. The input parameters of this process are a initial point *P_0* and a set of directions *N*. (Note that these directions are not identical with those used in the control surface algorithm) The function *f(P)* is then minimized one dimensionally along the first direction (red frame in picture 4.14): a 'normal' function minimisation algorithm tries to find the scalar for which the function

$$f(P + \quad n_1) \tag{4.17}$$

is minimal. P is then replaced by $P + \quad n_1$, the direction is changed to n_2 and a new minimum is sought (blue frame in picture 4.14). The algorithm cycles through the set of directions until the function stops decreasing. Normally, the set of unit vectors for the n-dimensional function space is used as the matrix with directions. This is a steady method, but not the fastest: it is a common occurrence that optimising the function in one direction spoils the results of the previous direction. The solution is to update the set of directions continuously. This is what happens in Powell's method. How this works exactly is described in [Pre92, p.414]

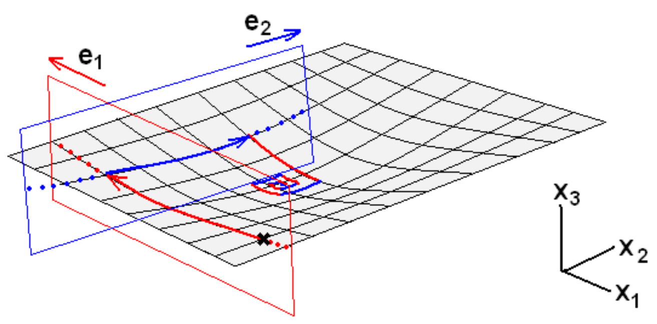


Fig. 4.14 Minimising a function with a fixed direction set

The curve fitting process is numerically expensive, because, for fitting a surface through a mesh with 30.000 nodes in 100 iterations, already $3*10^6$ Newton processes are needed. The easiest way to speed up the process is to use only a part of the nodes in the mesh. The algorithm has been tested using only 10% of the nodes, and the fitting process was still very accurate. As the node-surface distance calculations are the dominant factor in the algorithm, the calculation time is reduced by more than 80%.

Care has to be taken, when nodes are ignored in the approximation process. When a certain part of the structure contains exactly those nodes that are ignored, the shape of the mesh will get distorted and the approximation will be wrong. It is advisable to check the results of the approximation visually. A better way is to start with only a small selection of nodes, do some approximation steps, and derive a global surface shape. This solution is used as a starting point for a more accurate, but slower approximation with more nodes. The solution that is found now can, again, be used a starting point for a very accurate approximation with all nodes available.

4.4 Local overbending using surface refinement and degree elevation

As pointed out, surfaces with a large number of control points are needed for local overbending. Fitting a surface with more than 16 free parameters requires an accurate initial parameter set. It is also rather unstable, so obtaining a correct initial guess may take several attempts. However, for local overbending a much finer grid is often required for accurate springback compensation. The problem has been solved by the following procedure:

- A relatively simple Bezier reference surface is fitted
- The Bezier surface is refined (adding control points) and its degree is elevated (if required).
- The control points in a user specified region, called the *active area*, are set as free control points.
- A new fitting procedure is carried out to fit the refined reference surface
- The refined reference surface is used as a starting point for the springback surface fitting
- A springback surface is fitted, a transformation surface calculated and tool geometry is modified as previously discussed.

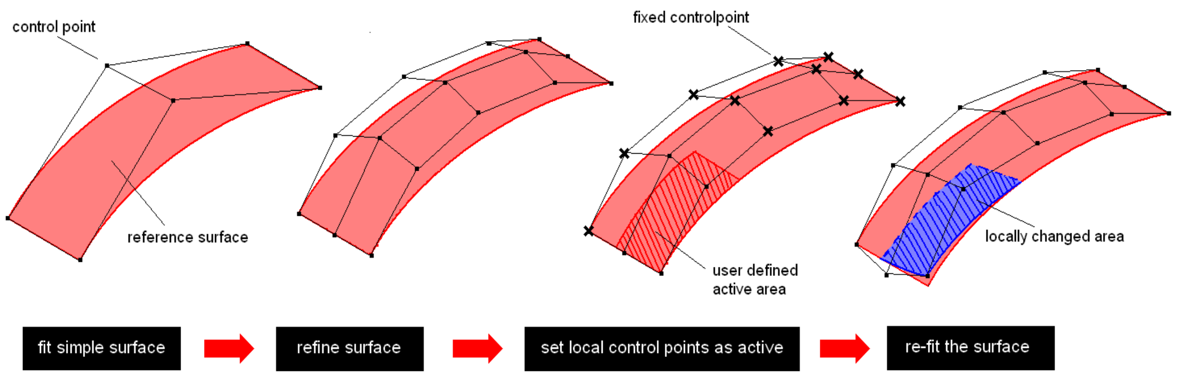


Fig. 4.15 Refinement and refitting of the reference surface

The Bezier surface that is fitted first should approximate the geometry roughly. When the refined B-spline surface is fitted, the approximation has become more accurate locally in the active area. The algorithm remains stable because the control points that are not in this area remain fixed in place. In picture 4.15, the refined surface contains 15 control points, but the active area where compensation is carried out, contains only 4 control points. Because of the smoothness of Bezier surfaces, the transition between the part of the product that is modified, and the part that is not changed, remains smooth as well.

4.4.1 Surface refinement

A Bezier surface is essentially a subset of the collection of B-spline surfaces. The knot vectors for Bezier surfaces, transformed in B-spline notation, have the following form:

$$U \quad 0,....,0,1,...1$$
$$V \quad 0,....,0,1,...1 \qquad (4.18)$$

After the first fitting stage, surface refinement is applied. The set of control points is expanded, providing more local control over the curve. In this process, new knots are inserted in the knot vectors for the u and v direction. The new, larger, set of control points is then calculated so that the shape and parameterisation of the surface remain identical.

Let $C(u)$ be a Nurbs curve with a knot vector $U \quad u_0,...,u_m$. Now, a new knot $\bar{u}_{new} \in \; u_k, u_{k\;1})$ is inserted in the knot vector, to form a new knot vector

$$\bar{U} = \; \bar{u}_0 = u_0,...,\bar{u}_{k\;1} = \bar{u}_{new}, \bar{u}_{k\;2} = u_{k\;1},.... \qquad (4.19)$$

Because U spans a vector space that is a subset of the vector space spanned by $\overline{U}$, C(u) can be represented identically on $\overline{U}$. Knot insertion is therefore the process of solving the set of linear equations:

$$\vec{C}(u) = \sum_{i=0}^{n} N_{i,p}(u)\vec{P}_i = \sum_{i=0}^{n+1} \overline{N}_{i,p}(u)\vec{Q}_i \qquad (4.20)$$

Solving this large set of equations is a costly operation. More efficient algorithms are discussed in [Pie97, p.142]. As a result of (multiple) knot insertion, the surface becomes a B-spline surface. In picture 4.16, surface refinement is demonstrated, first in u direction, then in v direction.

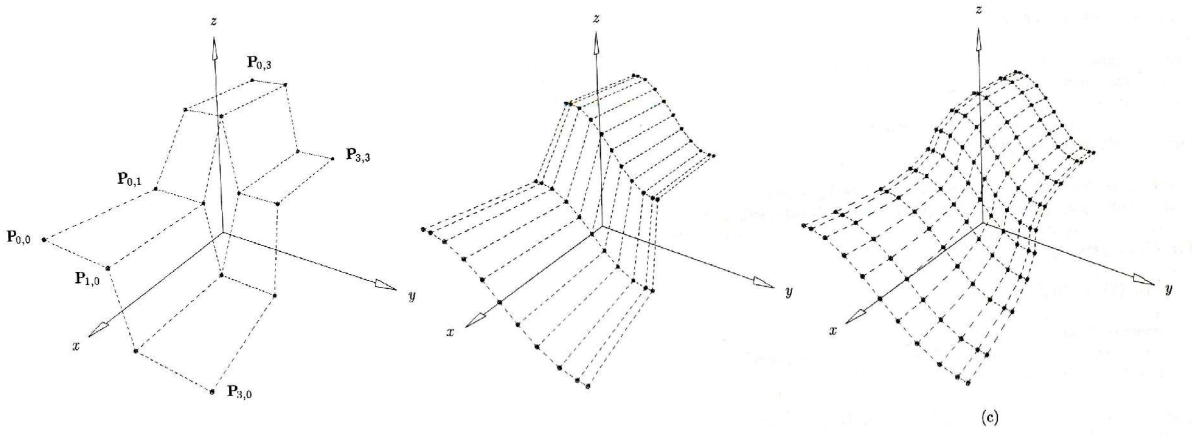


Fig. 4.16 Surface refinement. Change in control net for original surface (left), refinement in u direction (middle), and in both directions (right). Picture taken from [Pie97]

Instead of adding a knot to the knot vector and then calculating the new set of control points, it is also possible to insert a new row of control points at the desired location, and then calculate a new knot vector, while keeping the surface shape and parameterisation unaffected. This is referred to as *inverse knot insertion.* In this way it is possible to directly define the location of a row of control points. But, when another row of control points is added, the locations of all other rows will be changed automatically by the algorithm. It is therefore very hard to control the 'density' of control points in a certain area. For increased robustness, we have chosen to always refine the surface globally. The disadvantage is that the surface will contain more (fixed) control points than necessary. It is found that the small increase in calculation time was acceptable.

4.4.2 Elevation of the surface degrees

Another way to add control points is to elevate the *u* and *v* degrees of the surface. With increasing degrees, the resulting surface will be increasingly difficult to fit. In general, lower degree surfaces with more knots will fit the data better than higher degree surfaces with fewer knots: the curvature of the surface will be lower.

In picture 4.17 this is visualized: the solid line is a B-spline curve with a degree of 2, the dashed line has a degree of 3 and the dotted line a degree of 4. The quadratic curve follows the point cloud 'best', that is, the curvature of the curve is the lowest. It is clear that the difference in curvature in the cubic function makes it very flexible, and therefore also quite unstable.

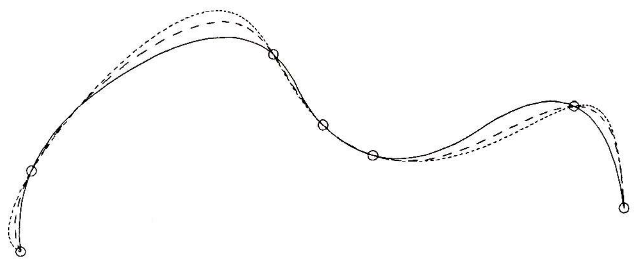


Fig. 4.17 Fitting a quadratic (solid line), cubic (dashed) and 4h order (dotted) curve. Picture taken from [Pie97]

Degree elevation has been implemented, because for some types of geometry, using a linear by quadratic surface already provides accurate global springback compensation results. When the surface is then refined to solve local springback effects, it remains linear in one direction and will develop sharp edges. So, before local springback compensation can be carried out, the control surface degree has to be raised to at least 2 in both directions, to obtain a smooth surface.

5 Optimising industrial products

In this chapter, two products are analysed and optimised with the algorithm developed in chapter 4. The first product is a structural part provided by DaimlerChrysler. The second example is a fuel tank cap. The structural part is compensated in one iteration only. It is shown how the algorithm parameters need to be set to acquire accurate results. The modification of a tool mesh is visualized. The resulting reduction in shape deviation is evaluated, using a software tool that visualizes the difference between two meshes. The fuel tank cap is optimised with more iterations. With this product it is shown how local compensation can be used to raise the algorithm's effectivity. Then, the control surface algorithm is compared to the DA method to investigate the strengths and weaknesses of each method.

FEM simulations from PAM-stamp and INDEED form the basis for the compensation. In industrial applications, care has to be taken when compensating springback using simulation data. Springback calculations are highly dependent on the stress distribution in the product after the deep drawing stage, and when the simulated stress distribution is flawed in some way, the compensation may even worsen the products geometrical accuracy. It is recommended to build up experience with springback calculations and check the results or calibrate the calculation by doing extensive measurements on real product s first. One should realise that the compensation can only be as accurate as the FEM results and that inaccuracy builds up fast. For example: when an 80% accurate FEM analysis is used, and the springback compensation reaches 70% accuracy, the final effectivity of the procedure can only reach 56%.

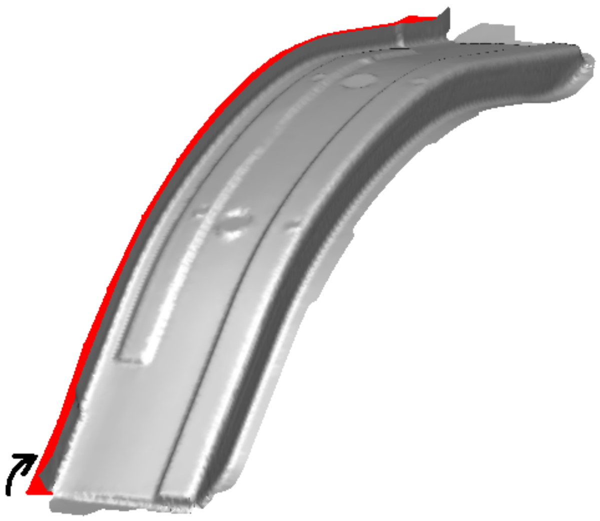


Fig. 5.1 the OP50 forming stage: bending the flange

5.1 The forming process of the DC-part

This product is a part of the body structure of the Mercedes-Benz E-class, and it is located under the luggage compartment. It is made in a complex multistage deep drawing process. In this example, only the 'OP50' step of the forming process is analysed. The previous forming stages have already been simulated and optimized separately, so the intermediate product is considered geometrically accurate. Only the left flange of the product is deformed, the rest of the product is held in place by a large blankholder.

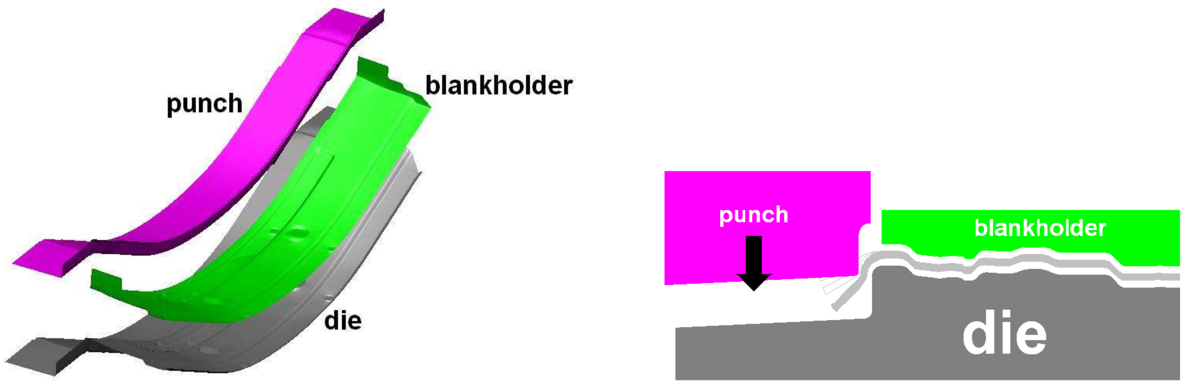


Fig. 5.2 Tools used in the OG50 forming stage

The forming simulation has been carried out in INDEED. The resulting reference and springback meshes are imported into a utility program which calculates the distance between the meshes for each node and stores the data in a file that can be read by a postprocessor. There are two ways to measure the distance between the meshes. When the meshes are topologically identical, the ‘displacement’ of the nodes can be calculated directly. When the meshes are topologically different, only the normal distance from each node in the reference mesh to the elements in the springback mesh can be calculated. Some care has to be taken when the results from the normal distance calculation are evaluated. In picture 5.3, the distance between a reference mesh and a springback mesh is evaluated. The only difference between the meshes is a translation. In the node-to-node displacement diagram, a proper distance field is shown. For the normal-method, a peak value occurs, and on the edge, no distance can be calculated. The normal method is therefore not reliable for evaluating small details in the product and the product’s edges; however, it is adequate for evaluating global springback.

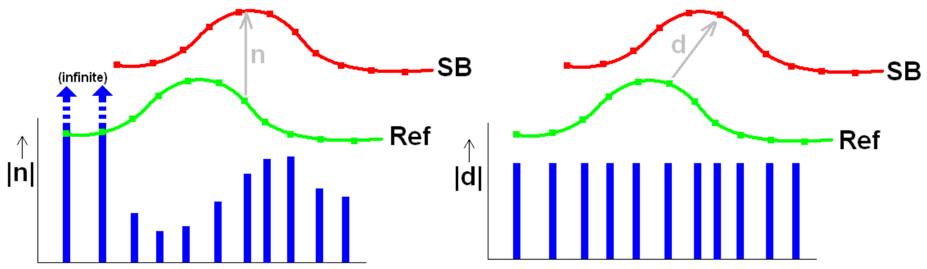


Fig. 5.3 Normal distance field (left) and nodal displacement field (right)

The result of the normal distance calculation for the reference and springback mesh of the forming simulation with the original toolset is shown in picture 5.4. The maximum shape deviation amounts 4.2mm The springback can be seen as a global mode, a large part of the product springs back. The smooth distance field also shows that the deformation is even. With this information, the right surface degrees are selected. In this example, a linear by cubic surface is used. With this surface, it is possible to fit the product shape quite accurately: in one direction the product is relatively flat, and in the other direction the product has a relatively constant curvature. The springback deformation can be seen as a combination of torsion and bending in the length direction of the product. This can also be modelled accurately with a linear by cubic surface. This surface only has 8 control points, so the algorithm will be robust.

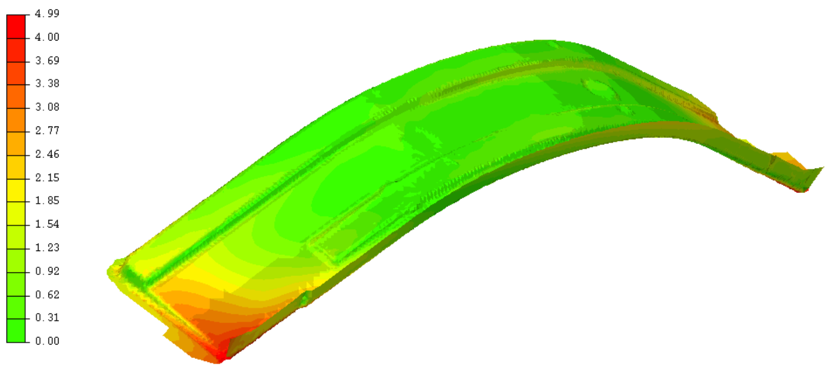


Fig. 5.4 Normal distance field between reference and springback meshes, original toolset

5.1.1 Preparing the analysis

During the FE simulation the mesh is refined locally where smaller bend radii are present. Therefore, the reference and springback meshes have element sizes that vary strongly over the product and the nodes are not evenly distributed. When a surface is fitted through this unevenly distributed ‘node cloud’, it is attracted to curved parts of the product because these parts contain more nodes. For this product, this lead to unwanted surface shapes. A function has been implemented in the program to remove the refined nodes from the meshes.

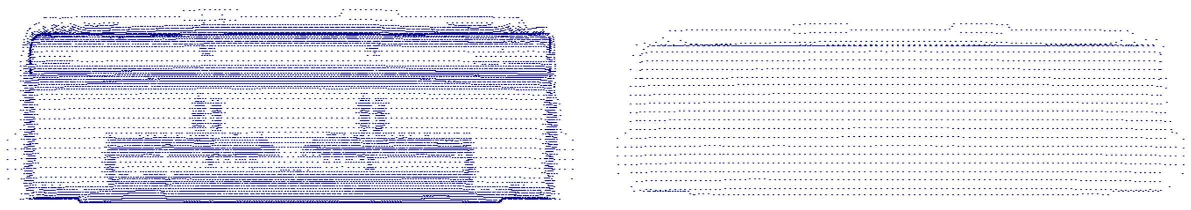


Fig. 5.5 original reference mesh (left), and with refined nodes removed (right)

The result is shown in picture 5.5. Without the refined nodes, the node distribution across the product is more even, leading to a better surface fit. This functionality is also useful when the deep drawing process has to be optimized in more than one iteration. After the first compensation is carried out, the tool data are changed and a new simulation is started. The resulting springback mesh will be topologically different to the original springback mesh because the mesh refinement will not be completely identical. By removing the added nodes, the mesh is converted back to its original topological state. Finally, the mesh size is automatically reduced from 20000 nodes to only 2000, which speeds up the surface fitting drastically.

Then, tool meshes need to be provided. The tools are exported in their fully closed configuration (at the end of the forming stage) to make sure that they are modified in the same coordinate system and still fit together after the shape modification.

Now, the accuracy parameters for the surface fitting process have to be set. The surface fitting process is halted when the objective function, in this case the least squares distance function, does not change more than this value. A maximum number of iterations needs to be specified. If no convergence is reached, the surface fitting will be halted at that point. It is highly recommended to review the input parameters then, and to restart the algorithm, but it is also possible to carry on with the analysis with the current surface. We found that the surface fitting was accurate when the fitting accuracy was set at 1e-6. The fitting of the reference surface takes the longest time, around 3 minutes on a Pentium4 2.4GHz computer. The solution is found after 15 iterations of the Powell process. The subsequent fitting of the springback surface was significantly faster. The calculation time is dependent on the accuracy of the initial shape of the reference surface. For this analysis, the parameter values for the initial shape were provided in the input file. For the next example, a fuel tank cap, the algorithm will find the initial values automatically.

5.1.2 Results of the analysis, and feedback into INDEED

The result of the reference and springback surface fitting are visualized in picture 5.6. Here it can be checked that the control surface was capable of showing the springback deformation: the colour pattern (accentuated by the dashed stripes) on the springback and reference meshes is approximately the same on the reference and springback surfaces. This is only a very rough sign of the accuracy of the compensation; finding the right compromise for the surface degrees is also a matter of experience.

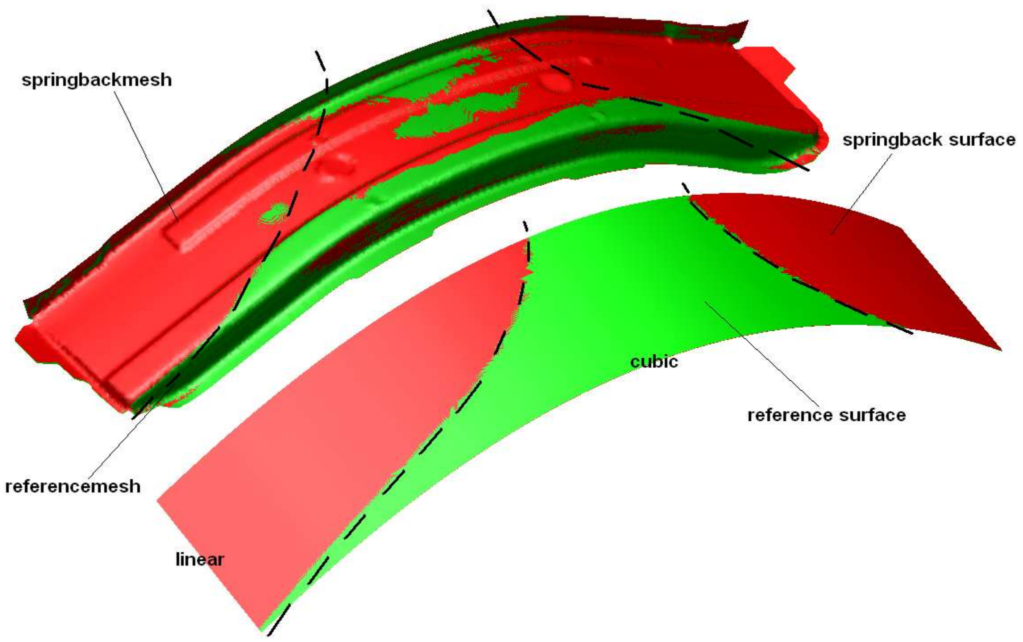


Fig. 5.6 reference and springback meshes (left), reference and springback surfaces (right)

In the next picture, the transformation procedure is visualized. The reference mesh and reference surface are visualized in green, the transformation surface and the modified tool mesh are visualized in blue. As can be concluded from the colour pattern on the tool mesh, in this case the blankholder, the tool mesh followed the change in shape of the control surface.

The transformed meshes can now be imported into the INDEED project file. Because the modified tool meshes are in the 'fully closed' position, they need to be repositioned to the starting position by translating the meshes in the drawing direction. When all process references are reset for the new meshes, and the old ones are deactivated, the simulation can be started. It is recommended to do an undercutting check first, because the shape modification may have rendered unusable tools. When undercutting occurs with the modified tools, the product cannot be optimised with the springback compensation algorithm. When the deep drawing process has already been maximally optimised to reduce springback, there remains no other option than to review the design of the product.

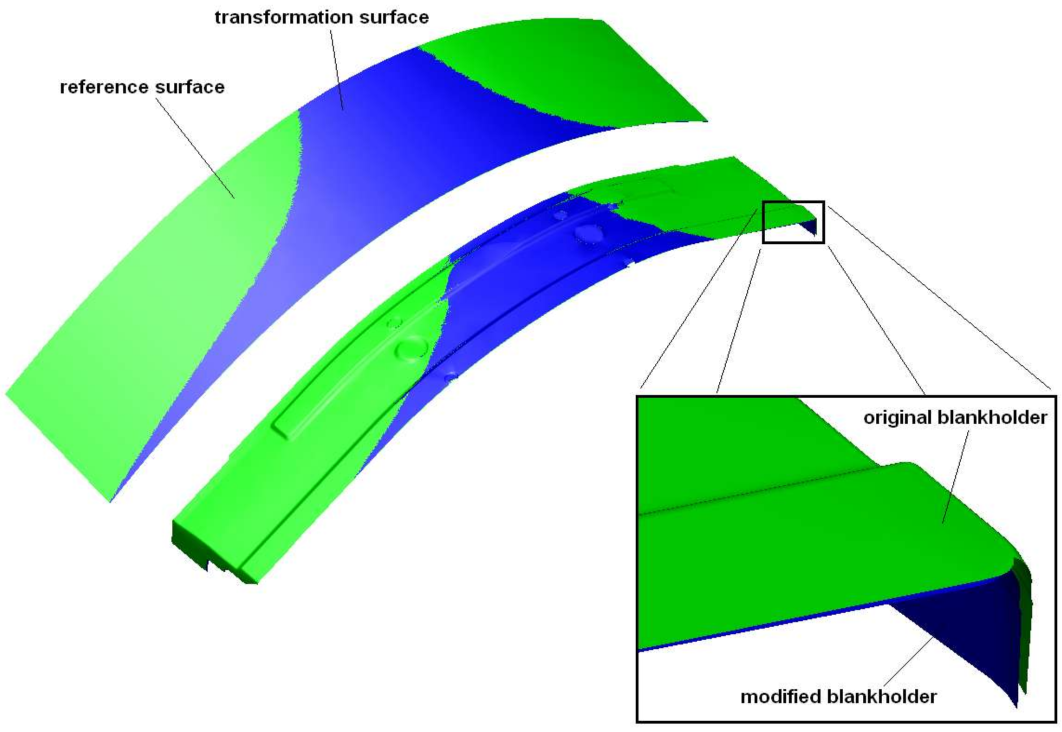


Fig. 5.7 reference, springback and transformation surfaces (left), the punch mesh and modified punch mesh (right)

5.1.3 Results of the INDEED simulation with modified tools

In picture 5.8, the normal distance from the reference mesh to the springback mesh has been visualized. The upper part is simulated with the original tools, the lower part with the optimised tools. The overbending factor is set at 250%. Note that this overbending factor is significantly larger than the industry rule of thumb of 130%.

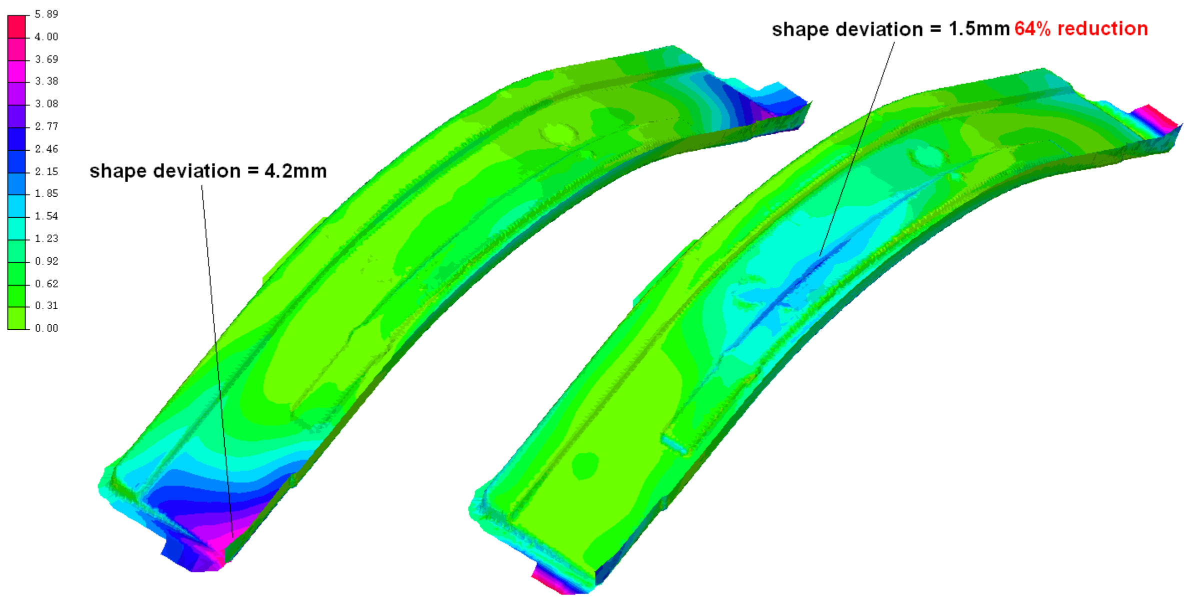


Fig. 5.8 Distance to reference mesh for the original toolset (top), and the optimized toolset (bottom)

After optimisation the global mode of the springback has been removed, a smaller deformation remains. The small flanges on the left and right of the product still show large springback. This is because they are not held in position by a (optimised) blankholder. In the middle of the product, the problem with the normal distance measuring between two meshes is observed.

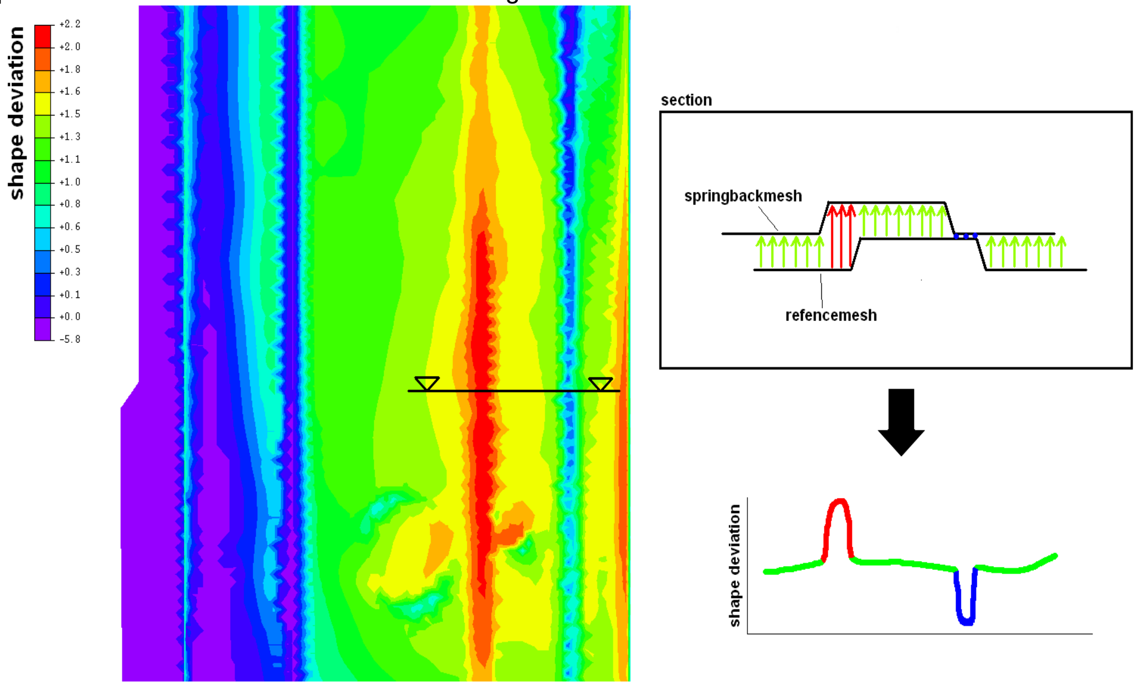


Fig. 5.9 the fuel tank cap blank (left) and the end product after trimming (right)

Where the large shape deviation is shown (in dark blue), a reinforcement rib is present. Because the two geometries are only displaced a little bit, a positive and a negative peak appear in the diagram. This is made clear in picture 5.9. The shape deviation in the middle of the product is visualized. A cross section at the black line is pictured schematically in the box on the right. From the picture follows that the peaks only appear because the meshes are displaced, not because their shape is generally different at the location. Therefore, in the schematic, the green level is the right shape deviation. For the DC-part, the maximum shape deviation after optimisation is therefore 1.5mm. This means a decrease in shape deviation of 64% is reached in one iteration.

5.2 The fuel tank cap

The next example is the fuel tank cap of picture 5.10. The product is produced with a straightforward deep drawing process. The product is pressed in one stage and then the excess material is trimmed. With this example, the advanced features of the algorithm will be demonstrated:

- optimisation with more iterations
- automatic start value finding for the surface fitting process
- local overbending

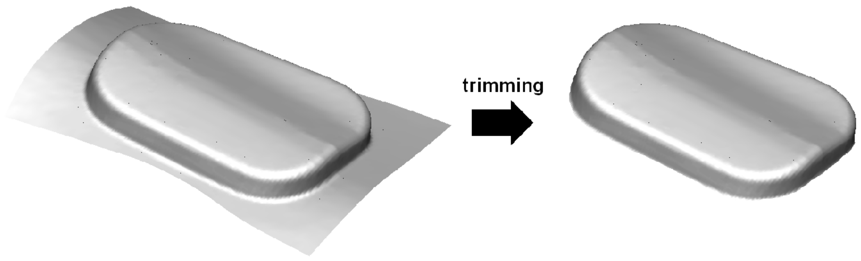
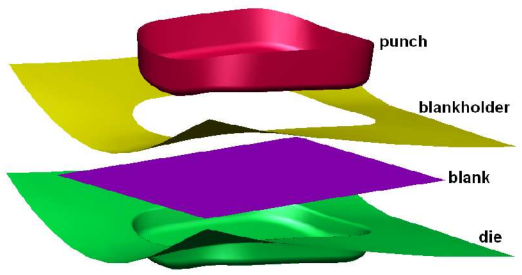


Fig. 5.10 the fuel tank cap blank (left) and the end product after trimming (right)

The deep drawing simulations have been carried out with PAM-stamp. The original plan was to include the trimming stage in the optimisation process as well, but unfortunately, the curve that defines the trimming stage in PAM-stamp cannot be modified externally. When the geometry of the tools is modified, this trimming curve needs to be modified as well, which was unfortunately not yet possible in PAM-stamp. Another problem was that after trimming, some serious wrinkling occurred, making the comparison between the reference and springback geometry unreliable. This wrinkling occurs because we did not have the original deep drawing setup data, and the material of the blank was simply guessed. The deep drawing process was apparently not set up correctly, however the effectivity of the control surface method can still be demonstrated.



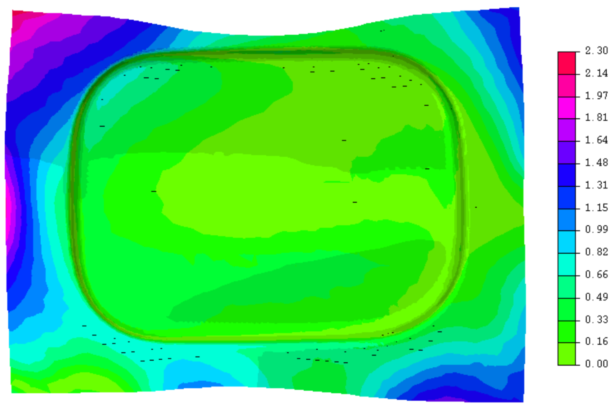


Fig. 5.11 the deep drawing process for the fuel tank cap

Fig. 5.12 Shape deviation with original tools

The results of the first simulation are shown in picture 5.12. As can be seen, only the part of the blank that is trimmed off later, exhibits springback effects. Even after the trimming phase, the product remains geometrically accurate (wrinkling aside). In practise, no compensation would be carried out, but in this academic case, we will show that it is possible to optimise the geometrical accuracy of the complete blank.

For the first iteration, a quadratic by quadratic control surface is applied. With this surface, camber can be applied, which is required in this case. The 9 start values for the fitting of the reference surface are not given by the user, but generated automatically. As discussed in §4.2.2, a B-spline startvalue surface, consisting of 4 linear by linear patches, is fitted first. The z-startvalues for this surface are set at the mean value of the minimum and maximum z-coordinates of the bounding box. This fitting process was stable and fast, as expected. In picture 5.13 the resulting surface and its control points are shown.

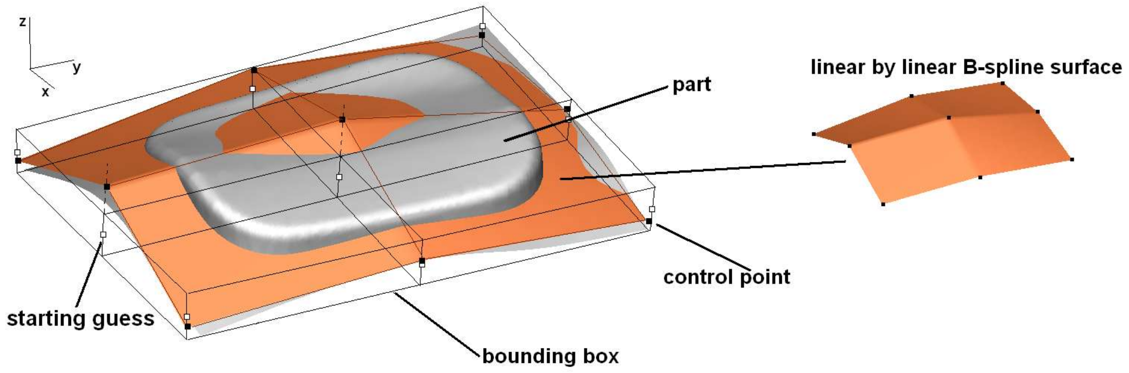


Fig. 5.13 Finding start values with a linear by linear B-spline surface

With these control points as starting points, the reference and springback surfaces can be fitted without problems, and a tool compensation is carried out. The overbending factor is set at 1.3. The tool meshes are imported into PAM-stamp and a new simulation is carried out. The resulting normal-distance plot can be found in picture 5.14. The shape deviation has been decreased drastically, though the maximum shape deviation has only been reduced by 34%. The global springback mode has been compensated, but smaller modes appear.

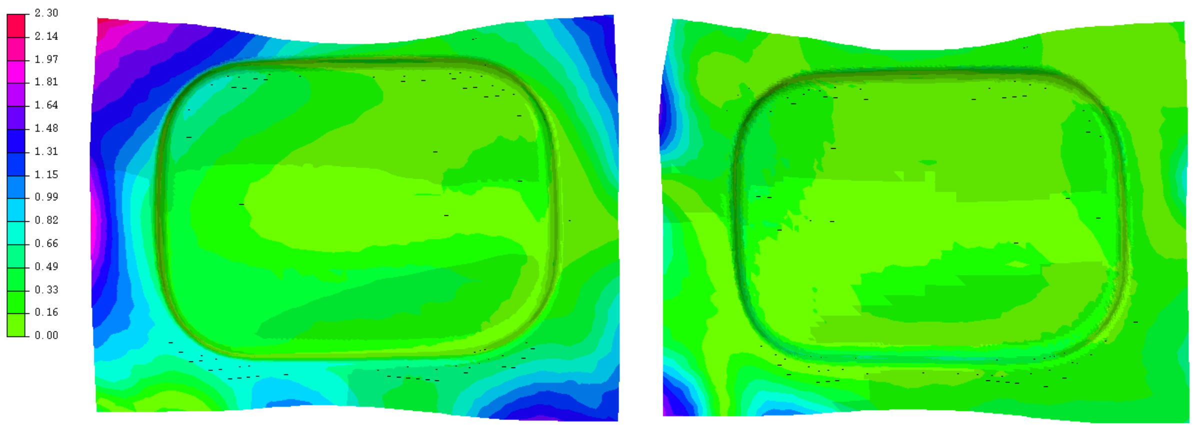


Fig. 5.14 Shape deviation with original tools (left) and after iteration 1 (right)

Another iteration is carried out to reduce the shape deviation further. Because the major shape deviation occurs on the left side of the product, we have decided to use the local overbending function. Three rows of control points have been added to the quadratic by quadratic Bezier surface (9 control points), to form a quadratic by quadratic B-spline surface with 18 control points. Now, the*active area* is defined as shown in picture 5.15. The user can define this area on the control surface that may be changed during optimisation. The control points that are not in this area are fixed, and therefore, the geometry remains unchanged there.

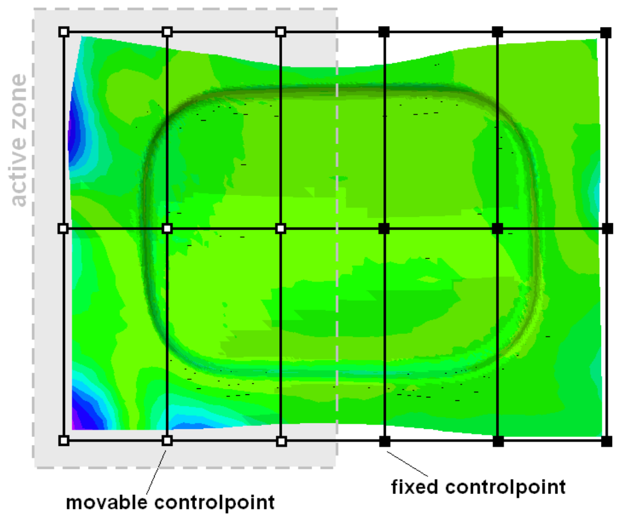


Fig. 5.15 surface setup for the second iteration

The advantage of local overbending is that now a more flexible surface can be used, allowing more detailed compensation. Because the amount of free variables is kept relatively low, the surface fitting procedures remain robust.

After the second iteration, a shape deviation reduction of 47% is reached and the maximum shape deviation amounts 1.35mm. Another iteration is carried out. Again the surface is refined; it now contains 63 control points. Two active areas have been defined; they include the two parts of the product where large shape deviation is present. Also, the springback compensation factor has been reduced to 60%. After some experiments, it turned out that the shape optimisation process is more robust when the compensation factor is slowly reduced during the optimisation. The reason for this is that the refined surfaces may suffer from ‘waviness’. After the third iteration, the maximum shape deviation amounts 0.87mm, the springback compensation is now 66%.

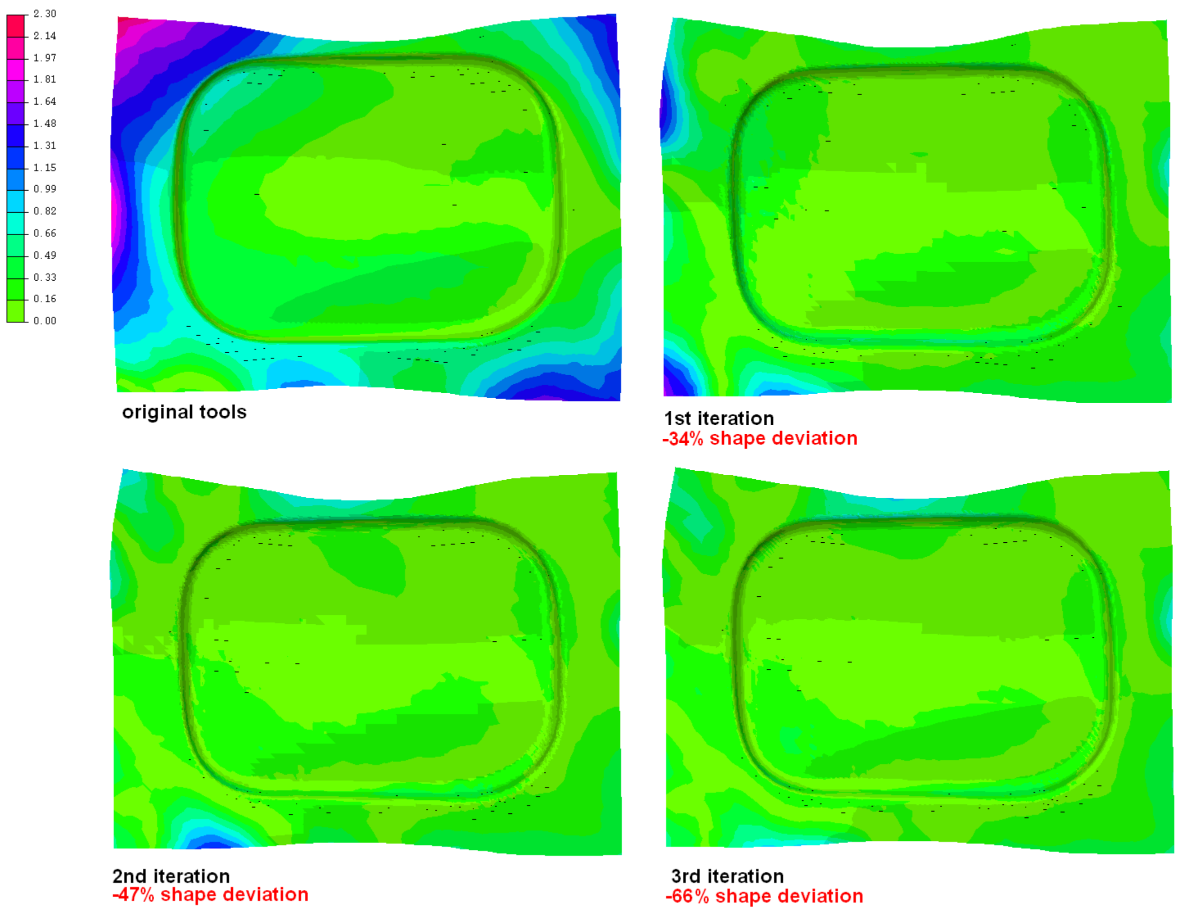


Fig. 5.16 Shape deviation with original tools (left), after iteration 1 (middle) and after the second iteration (right)

The dimensional stability in the functional part of the blank, the actual tank cap, has been improved as well, as can be seen in picture 5.17. Note that the blank has *not* been 'physically' trimmed in the FE simulation, only the functional part of the blank is shown to make the picture clearer. A shape reduction of 57% may not seem much, but it should be pointed out that the maximum shape deviation of 0.33mm is already significantly less than the blank thickness (0.88mm).

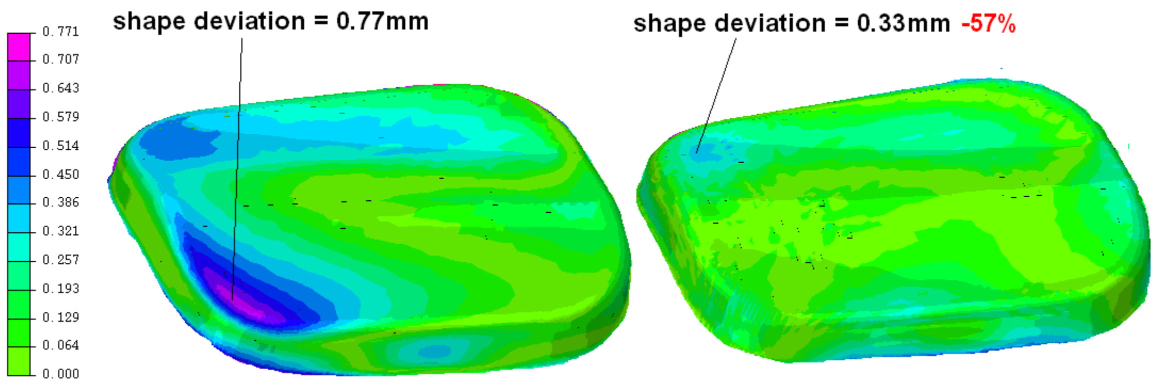


Fig. 5.17 shape deviation in the product area with original tools (left) and after 3 iterations (right)

Another way to evaluate the effectivity of the algorithm is to compare the L1 and L2 norms of the shape deviation field. As can be concluded from table 5.18, the mean shape deviation is reduced by 50% in the first iteration. In the following iteration the improvement is only 10%, and in the third iteration no improvement is made at all. This is another sign of the waviness in the control surface. When the control surface cannot exactly model detailed springback deformations, small areas with large shape deviations are changed into larger areas with smaller shape deviations.

	Original toolset	1st iteration		2nd iteration		3rd iteration	
L0 (maximum)	2.56	1.70	**-34%**	1.35	**-47%**	0.87	**-66%**
L1 (mean)	0.44	0.22	**-50%**	0.17	**-60%**	0.17	**-60%**
L2	0.63	0.28	**-56%**	0.23	**-63%**	0.21	**-66%**

Table 5.18 Shape deviation with L0, L1 and L2 norms

5.3 Comparison with the DA method

The DA method has also been implemented at INPRO as a separate program. The DA method should give significantly better results because:

- It uses a deformation vector field instead of comparing geometry. This is basically the same problem as with the visualisation of form deviation, discussed in paragraph 5.1. The node-to-node deformation field is more effective than the normal distance field, which simply compares two geometries.
- The geometry is not approximated
- The modification of the tools is not limited to the modes that are possible with the control surface
- The method does not suffer from instability when high accuracy is required. However, the algorithms accuracy is dependant on the interpolation function of the shape modification field.

First, a compensation is carried out for the structural part. For the DA method, the overbending factor has been set at 250% as well. The shape modification field is interpolated with a field function with 11 parameters:

$$C(x,y,z) \quad a \quad b\ x \quad c\ y \quad d\ z \quad e\ xy \quad f\ xz \quad g\ yz \quad h\ xyz \quad i\ x^2 \quad j\ y^2 \quad l\ z^2 \qquad (5.1)$$

The result of the analysis is shown in picture 5.19: With the control surface method, a springback reduction of 64% is obtained. As expected, the direct DA method performs better, the effectivity of the compensation amounts 71%. The advantage over the CS method is not as large as expected. This is probably due to the coarseness of the interpolation function that was available. We think that the advantage of the DA method will become clearer when the algorithms are compared in an optimisation with more iterations.

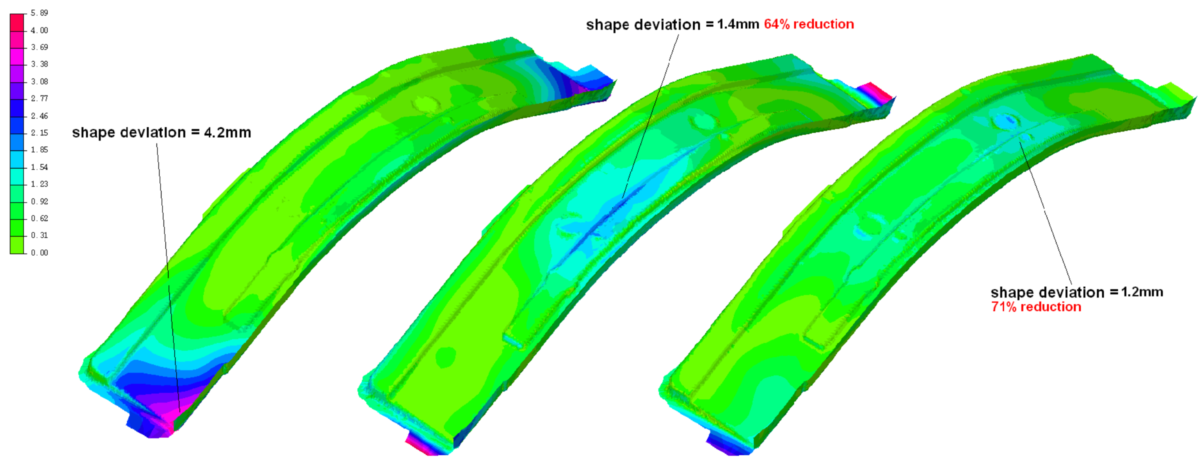


Fig. 5.19 shape deviation with the original tools (left), tools optimised with control surface (middle) and DA method (right)

Unfortunately, it was not possible to test this. When more iterations are carried out, the die needs to remain in the same place after each tool modification, because only then the resulting springback meshes can be directly compared. Unfortunately, before the INDEED calculation starts, all tool meshes are moved to make initial contact (autopositioning) and the blank mesh is taken as the fixed object. So, after the deep drawing phase, the position of the deformed blank is different for each calculation. In

PAM-stamp, the autopositioning can be defined exactly, so the die is set as a fixed reference and the problem does not occur.

Therefore, the DA versus CS comparison is carried out with the fuel tank cap, for which the calculations are carried out with PAM-stamp. Again the overbending factor is set identically at 130% for both algorithms. The optimised tools are imported back into PAM-stamp and a new simulation has been carried out. Surprisingly, the blank flow into the die during forming was completely different after the optimisation. From contact pressure plots in PAM-stamp it was found that the contact friction, caused by the blankholder during stamping, had changed after optimisation with the DA method.

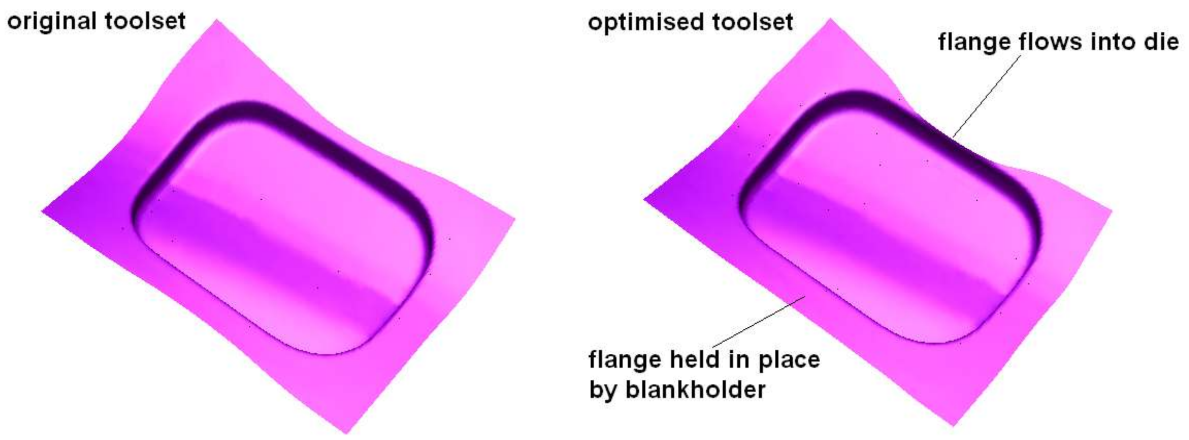


Fig. 5.20 blank shape after forming with the original tools (left) and the tools optimised with the DA method (right)

With the mesh to mesh distance tool, a plot has been made of the gap width between the die and the blankholder before and after optimisation. As picture 5.21 shows, the gap width changes drastically during compensation. Afterwards, the gap is about 5% wider at the top and around 5% narrower at the bottom. Because the tools are set as undeformable tools in most deep drawing simulations, this gap is a critical parameter. The blank is strongly held on one side of the die, but it flows away under the blankholder at the other side. The problem is probably caused by the interpolation function for the shape modification field, but the exact cause of the problem is unclear.

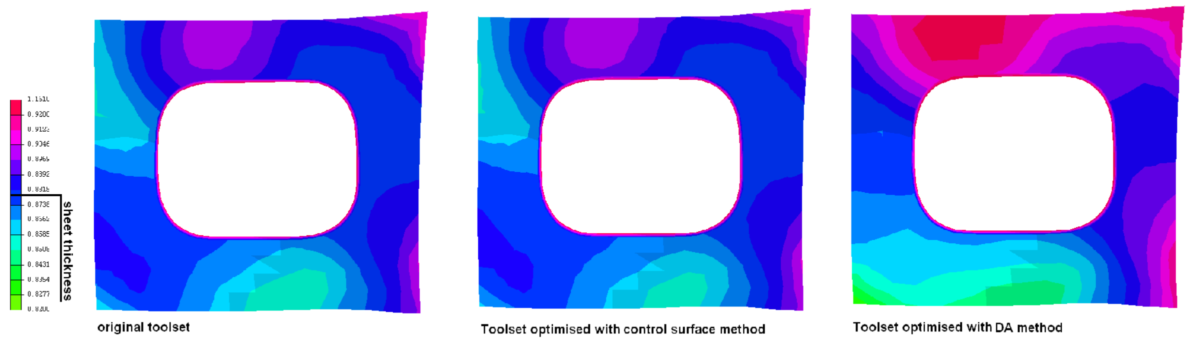


Fig. 5.21 gap width between die and blankholder before optimisation (left), with control surface (middle) and standard DA (right)

As can be concluded from picture 5.21, the control surface method does not suffer much from this problem, especially not when the shape modification is subtle, and when the control surface follows the product shape accurately. With this problem, it is impossible to compare both strategies directly. This case shows clearly how important it is that the tools fit perfectly before and after optimisation. For this product, it is clear that the blank meshes before and after optimisation cannot be compared directly. But also when the difference in gap width is more subtle, it strongly influences the shape of the product, and it becomes impossible to evaluate the effectivity of the springback compensation algorithm. This is also a problem for the real process setup at the production plant. In many cases the gap width is changed manually by the process engineers when the blank does not flow as expected. Such manual changes show that the deep drawing was simulated incorrectly, which also means that the springback calculation and compensation are then highly unreliable.

5.4 Conclusion

The reduction of the maximum shape deviation is up to 64% in the first iteration for the D.C. part, where only 34% is reached for the fuel tank cap. This is because the structure of both products is entirely different. The fuel tank cap is geometrically very stable, and does not show large springback. It turns out that the compensation is then less effective. The springback deformation for the structural part is larger, but it is also easier to compensate. It is therefore impossible to give a general effectivity percentage. With successive iterations the accuracy can be raised. Based on experience with these and some other products, we have found that a 75% reduction in shape deviation is achievable. Even after several iterations, 100% reduction cannot be reached with the algorithm. This is due to:

- limitations of the control surface to model the springback exactly: when the number of control points is too large the control surface cannot be fitted reliably anymore.
- limited shape modification possibilities of the control surface

On the basis of these and some other products that were tested, a number of 12 free variables has proved to give very reliable results. With 18 or more free control points, the surfaces may become instable and problems can occur. Of course, the surface deformation modes are relatively limited with a set of 12 control points, but as shown with the fuel tank cap, accurate springback compensation (66% reduction in shape deviation) can be achieved by compensating the global springback mode first, and then compensating the smaller springback areas by using more accurate local overbending in consecutive iterations. It is also advisable to reduce the springback compensation factor during the optimisation process, to ensure stable results.

The CS method can be applied automatically, but at the moment the results will only be sufficiently accurate *with* user interaction.

Because the process is visually interpretable, the user has exact insight in the product modification and can decide which springback deformation needs to be compensated. This is an advantage in the industry, where the demand is to provide *support* for springback compensation first, before a completely automatic algorithm is developed. The DA method is expected to be more adequate after several iterations because of it relies on a shape deviation field, and is better used completely automatically. However, due to the tool gap problems we have not yet been able to compare the performance of the two algorithms directly.

The L0-norm (maximum shape deviation) is not the only variable for checking the quality of the compensation. In the fuel tank cap, the surfaces suffer from a slight ‘waviness’. The small deformations which are introduced in the geometry may render uneven light reflections. Some experiments have to be carried out to see whether this is a problem or not. With the surface controlled overbending algorithm, the user may choose to apply a simpler surface to reduce the effect (but at the cost of dimensional accuracy).

6 Modification of CAD data

To make the algorithm useable, the same geometry modification that is applied to the tool meshes needs to be applied to the CAD data of the tools. The smooth and continuous description of geometry in CAD files is required for the generation of NC code for the milling robot. Also, the process engineer can still make manual changes to the tool design, after the algorithm has been applied. How the CAD geometry modification has been included in the algorithm is explained in this chapter.

In the first paragraph, the incompatibility between meshes and CAD files is made clear. This incompatibility makes feeding back the shape modification that was derived by the algorithm complicated. How CAD geometry can generally be created and modified is discussed in the second paragraph. Paragraph 6.3 focuses on how the CAD modification has been implemented and tested with the ICEM-surf software package. In the last paragraph, it is discussed briefly how CAD modification could be combined with the DA strategy.

6.1 FE mesh versus CAD geometry

In CAD programs, the geometry of deep drawn sheet metal products is represented mainly by parametric surfaces. Simpler geometric entities, such as flat and cylindrical surfaces, are also used extensively. To allow for free modification of geometry, the simpler geometric entities need to be converted into a parametric description.

As pointed out in chapter 4, the two most common ways to modify the shape of parametric surfaces are to move the control points, and to change the knot vector. Even the most intuitive method, changing the location of the control points, can yield unexpected results. This is caused by the fact that the control points have influence over a wide range of the surface, they are therefore not independent. Also, the surfaces are connected with certain boundary conditions, so the control points of different surfaces are coupled as well to satisfy those boundary conditions.

The surfaces can be refined to make the geometry more flexible. The number of control points is increased, providing more (local) modification possibilities. At the same time, the risk of obtaining unwanted (but mathematically correct) shapes is increased.

In the finite elements method, geometry is discretized into nodes and elements, and the nodes provide a huge amount of degrees of freedom (DOFs). To change the geometry of a mesh, the nodes can be moved. Every node is completely independent, and the shape changes are limited to the elements to which the node is connected. From table 6.1 it becomes clear that the two systems are principally incompatible.

FE mesh	CAD geometry
Many DOFs (3 for each node)	Little DOFs (3 for each control point)
Completely independent DOFs	DOFs have large and overlapping influence zones
Discrete	Smooth, analytically described

Table 6.1 FE and CAD geometry compared

6.2 Current CAD shape modification possibilities

Because (re)shaping a set of surfaces is so complex most general CAD systems are intended for *generating* geometry only. With concepts such as *sweeping* and *lofting* complex shapes can be produced. As an example, the principle of sweeping is shown in picture 6.2. Because the swept surface is defined with the profile and path curves, its shape cannot be directly modified in 3 dimensions. All other regular surface descriptions have this problem. Therefore, the first step is to convert all surfaces in the geometry to parametric surfaces, which can be modified freely. When the NURBS description is used, any surface can be converted without changing its shape. Note that this step is a one way process and that the original ‘generative’ description of the product’s geometry is lost.

[Bez78] and [Sed86] show the first practical strategies for the modification of a collection of parametric surfaces. A physical analogy for the Free Form Deformation (FFD) method [Sed86] shown in picture 6.3, is a block of flexible plastic in which the geometric objects are embedded. When the block is deformed, the embedded geometry is deformed in the same way.

Note that this is a more general version of the control surface principle: In the FFD algorithm, the control points linked to the ‘control volume’, and the shape of the product is modified by changing the control volume.

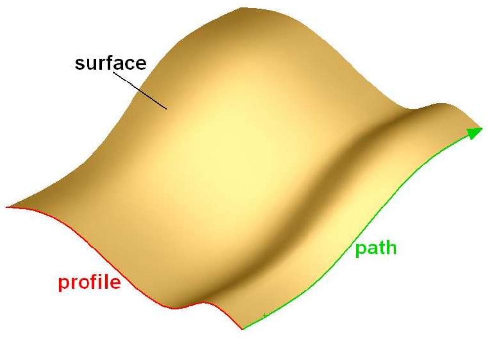


Fig. 6.2 sweeping a surface

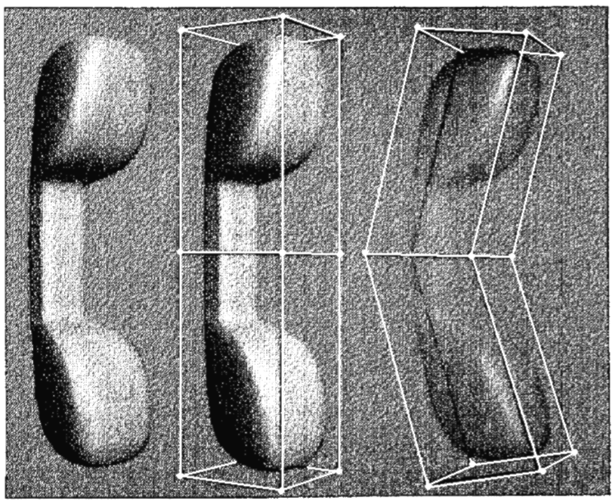


Fig. 6.3 Sederberg’s FFD (taken from [Sed86])

When the shape modification is carried out gaps may occur in between the product’s surfaces. This means that with the surfaces in the model, no shape can be calculated that satisfies all boundary conditions, and that either the surface degrees need to be elevated or surface refinement needs to be applied. The number of control points of the surfaces also define how accurate the deformation is. When, for example, the product contains a linear by linear surface, and the product needs to be bent as a springback compensation, the linear by linear surface needs to get higher surface degrees because otherwise it will remain flat.

6.3 Surface controlled modification with ICEM-surf

ICEM-surf is a software package intended for the modelling and modification of products consisting of complex surfaces. The difference between ICEM-surf and the regular CAD packages is that:

- ICEM-surf gives the user direct control over the mathematics behind the parametric surface modelling. Surface modelling and modification is carried out by moving control points directly. The surface degrees can also be specified directly to allow modelling the desired shape. While designing the product, the goal is to keep the degree of the surfaces as low as possible.

- ICEM-surf is built up around Bezier surface descriptions only. It is also possible to use the more complex NURBS surface description, but this is not recommended because the behaviour of NURBS surfaces can be unpredictable

- The use of trimmed surfaces is avoided. Many general CAD packages rely on surfaces that are trimmed (bounded) with curves, to make them fit to the surrounding geometry. These trimming operations can be seen as a local and inflexible solution to make surfaces fit together, and they cause problems when the surface are modified later on. In ICEM-surf, the surfaces are directly connected.

One should realize that when a product geometry is imported into ICEM-surf and its surfaces are transformed to Bezier description and modified they cannot be translated back. For example, when a product geometry that contains cylindrical surfaces is imported, the cylindrical surfaces are translated to a Bezier description. Because the Bezier description of a cylindrical surface cannot be exactly identical, the surface cannot be translated back. After a shape modification, the cylindrical surface is not necessarily cylindrical anymore, and it cannot be translated back into its original description anyway.

There are several ways to modify the collection of surfaces that the product contains. As pointed out, the user has direct access to the control points. Also, control surfaces can be used to modify all control points in the CAD geometry at the same time in the same way as the CS springback compensation algorithm. To carry out the same shape modification of the tools for the structural part presented in §3.2.3 on the CAD file, the reference and transformation surfaces from the CS algorithm can be exported (in IGES format) from the algorithm, and imported into ICEM-surf. The CAD geometry of the tool is linked to the reference surface. Then the shape of the control surface is changed. Because the CAD geometry is linked to the control surface, it follows the change in shape. So, the shape of the control surface is changed into the transformation surface, and the CAD geometry follows this change in shape.

Note that in this case the product's CAD file is changed, because no tool CAD files were available.

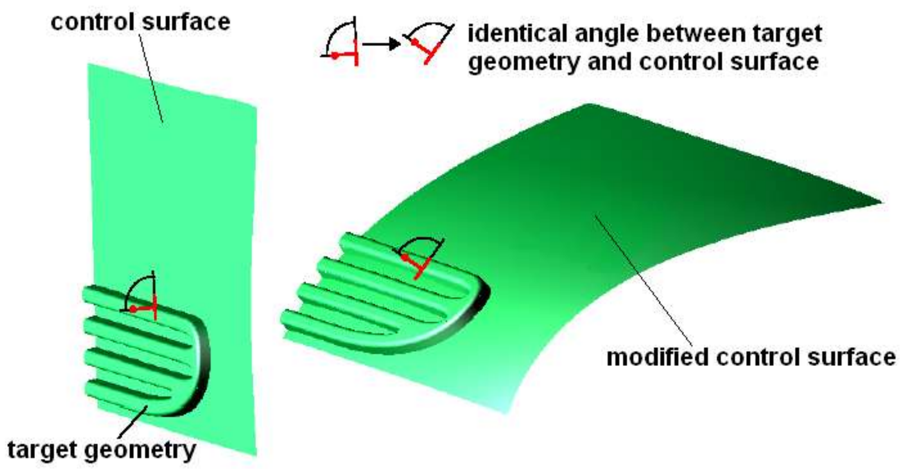


Fig. 6.4 surface controlled modification in ICEM-surf. Picture adapted from [Ice03]

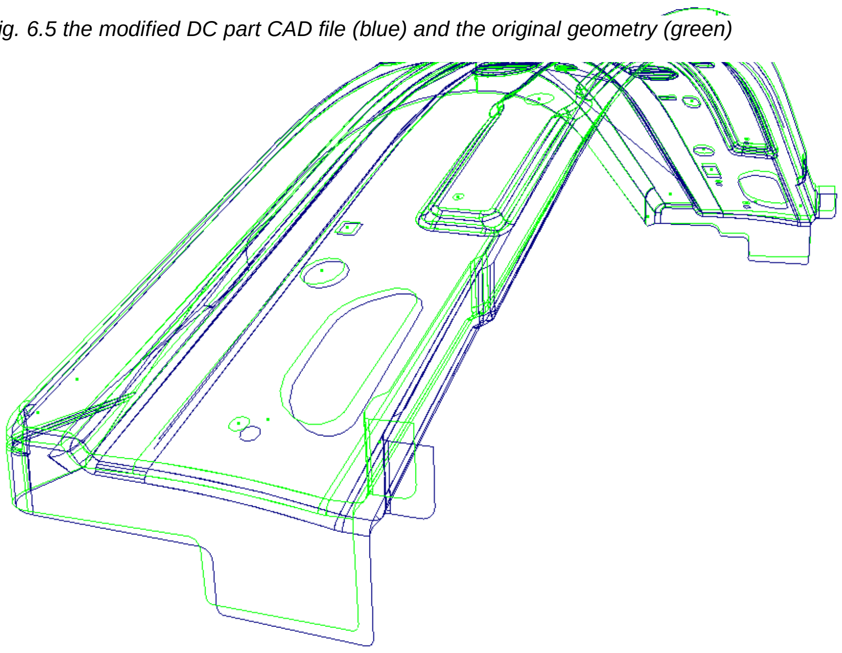
The result of the procedure is shown in picture 6.5. The surface degrees in the product's CAD file had to be increased to 8, to make the shape modification sufficiently accurate. High surface degrees do not cause problems for the control surface modification of the product, but may give rise to problems when the geometry is changed manually afterwards.

The surface degrees are always represent a compromise between accuracy and ease of shape modification.

To evaluate the difference between the modified CAD file and the modified mesh, the (normal) distance between both geometries is measured. The results are visualized in picture 6.6. This difference in shape is less than 0.2mm throughout the product, an excellent result. The question is whether this small difference in shape is due to differences in the modification algorithm, or to the subsequent meshing of the CAD file, which was required to compare the two geometries. However, the shape difference is much larger on the flanges on the far left and right of the product; it amounts up to 1.7mm there.

The explanation for this problem is that, to make the CS algorithm more stable, the control surface fits the product tightly. For the modification of the generally larger tool meshes, the control surface is extended. In the CS algorithm a 'ruled' surface is attached to the control surface. Because there are more options to extend the control surface, this surface extension is probably defined differently in ICEM-surf.

Fig. 6.5 the modified DC part CAD file (blue) and the original geometry (green)



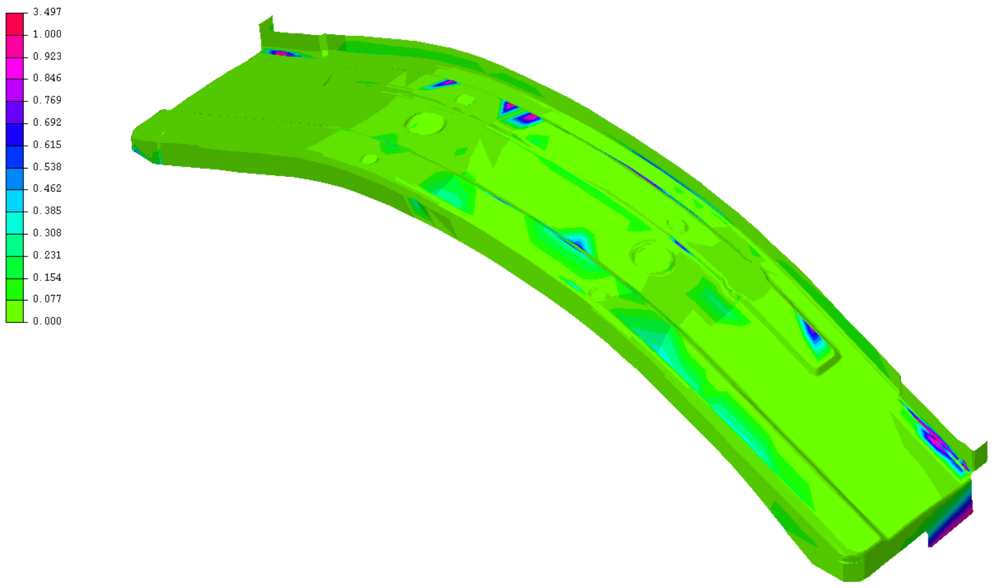


Fig. 6.6 normal distance between the modified product in SCO and ICEM-surf

6.4 CAD modification for the DA method

The result of an analysis with the DA method is a continuous shape modification function which is applied to the tool meshes. This function can also be applied to the control points of the Bezier surfaces in the CAD file directly, without using a control surface. When the surface degrees in the CAD geometry are taken rather high the surfaces will be flexible enough to follow the (possibly) complex modification function. This is mathematically straightforward. Unfortunately, it is not yet possible yet to alter the locations of the control points externally in ICEM-surf, so tests could not be carried out.

Conclusion

The goal of the project was to develop an algorithm to optimise the geometry of the deep drawing tools in order to produce a geometrically accurate product. The simulation of springback in forming processes, on which the compensation process is based, is relatively new. Therefore, the scientific literature on springback compensation is limited, and a general agreement on which springback compensation algorithm should be used has not been reached yet. The two main strategies, the displacement adjustment method and the spring forward method, have been tested, but only on academic deep drawing processes. For the spring forward method, some serious problems appear. There is no proof for the convergence of the method, and numerical problems for the required FE calculations have been reported. The more straightforward DA method is reported to be more reliable, and is used as a basis for the algorithm that has been developed.

As an alternative method, the control surface algorithm has been developed. The basic idea behind this algorithm is the use of a surface with a limited set of parameters to compare and evaluate the reference product geometry and the geometry after springback, and to modify the deep drawing tools. With the control surfaces, the DA principle is again used for compensation. The control surface allows only a limited set of shape modifications such as bending, torsion and camber. The advantages of this method are that the modification of the geometry can be carried out with a CAD system as well, and that it is possible to control the algorithm manually.

With flexible Bezier and B-spline descriptions for the control surface, accurate and detailed springback compensation has been carried out for several industry products. For real industrial products a reduction of the maximum shape deviation of around 70% between the (simulated) deep drawn product and the desired shape has been reached. It is impossible to give a general number for the effectivity of the algorithm, because it depends on the product geometry and springback. Generally, products with severe springback can be compensated effectively; the effectivity is lower for geometrically stable products that do not spring back much. After around 5 iterations, the mean shape deviation between the acquired product and the desired geometry is not reduced much anymore. Due to the limited stability and shape modification possibilities of the control surfaces, a 100% effective compensation cannot be achieved. This is caused by the ‘waviness’ of the control surface: large local shape deviations are replaced by smaller but broader shape deviations. To achieve a sufficient level of accuracy, local overbending has to be applied during the optimisation process, compensating the springback area by area. For this, some user interaction is required. The shape deviation alone is not the only number to check the quality of the compensation. Especially for products with aesthetic requirements, the product’s surface needs to be checked for waviness that might be introduced by the algorithm.

With the control surface method it is possible to transform the complete deep drawing toolset at once. The gap width between the tools, which is of vital importance for the deep drawing process, remains identical. The DA method that was evaluated to make a comparison with the control surface method, introduced deviations of the gap width. This caused a different material flow into the die cavity before and after optimisation. So, unfortunately detailed geometric comparisons between both algorithms could not be carried out. This also shows the sensitivity of the deep drawing process and its simulation as a whole; when the deep drawing simulation is modelled incorrectly or cannot be controlled precisely, the springback calculation will be unreliable and an incorrect compensation may be carried out.

The control surface strategy was also used to modify CAD data of the toolset. This is important for industrial application of the algorithm because the NC code for tool machining is based on the continuous CAD description of the product’s geometry. With the use of functionalities from the program ICEM-surf, this was achieved without problems. Here, the user will have to find a compromise between accurate springback compensation, using high degree CAD surfaces, and editability of the geometry, for which lower degree CAD surfaces are required.

At the moment it is clear that the effectivity of the springback compensation is not high enough yet. When the springback calculation has an accuracy of 80%, and a for 70% effective springback compensation is carried out, the overall accuracy amounts only 56%. Also, the goal was to develop an

automatic springback compensation tool, and at the moment user interaction is needed to adjust the settings of the algorithm during optimisation. Only then accurate results can be achieved while maintaining the algorithm’s stability. The user has to obtain experience in finding the right control surface for each springback problem.
On the positive side: the link with CAD systems proved to be very successful, which makes the CS algorithm already usable in the industry. Also, user interaction can be an advantage as well, because the process engineer has exact control over which part of the product is modified and how this is done. Possibly, the more stable and less complicated DA method can be used for automatic compensation, and combined with the control surface method for solving local springback problems afterwards. For the DA method, the link to the CAD systems has to be investigated then, and the problems with the interpolation function and tool modification have to be solved. Possibly, even parts of the spring forward method can be included in the algorithm. Most importantly, the basis of the analysis, the deep drawing simulation, still needs to become much more reliable to achieve reliable springback compensation. However, the results that were achieved with this project provide a promising outlook for the future. They show that FE deep drawing simulations can be used not only for feasibility checks, but also for the (re)design of deep drawing tools.

Literature

Books
[Gla90] Glassner A. (ed.), ’Graphics Gems’, Morgan Kaufmann, San Fransisco 1990

[Bat96] Bathe K., ‘Finite Element Procedures’, Prentice Hall, New Jersey 1996

[Far94] Farin G., ’Kurven und Flächen im Computer Aided Geometric Design, eine praktische einführung’, Vieweg Braunschweig 1994

[Hue98] Huetink J., F.P.T. Baaijens (eds.), ‘Simulation of Materials Processing: Theory, Methods and Applications’ (Numiform 1998 proceedings), Balkema Rotterdam 1998

[Ice03] ICEM-surf 4.3.1 user’s guide

[Pam02] PAM-STAMP 2G 2002 user’s guide

[Pie97] Piegl L., W. Tiller, ‘The NURBS book’, Springer, Berlin 1997

[Pre92] Press W.H., S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, ‘Numerical Recipes in C’, Cambridge University Press 1992

Articles
[Bez78] Bezier P., ‘General Distortion of an ensemble of biparametric surfaces’, Computer Aided Design 1978 vol. 10(2) page 116-120

[Chu03] Chu E., L. Zhang , S. Wang, X. Zhu,B. Maker, ’Validation of springback predictability with experimental measurements and die compensation for automotive panels’, proceedings NUMISHEET 2002, D. Yang et al. (eds.), p.313-318, 2002

[Gho98] Ghouati O., D. Joannic, J. Gelin, ‘Optimisation of process parameters for the control of springback in deep drawing’, proceedings Numiform98, J. Huetink, F. Baaijens (eds.), Balkema, Rotterdam 1998

[Kar92] Karafillis, A.P., M.C. Boyce, ‘Tooling design in sheet metal forming using springback calculations’, Int. J. Mach. Tools. Manuf., 34, p.113, 1992

[Kar96] Karafillis, A.P., M.C. Boyce, ‘Tooling and Binder for sheet metal forming processes compensating Springback Error’, Int. J. Mach. Tools. Manuf., 36, p.503, 1996

[Sed86] Sederberg T., S. Perry, ‘Freeform deformation of solid geometric models’, Computer Graphics, 20(4):151-160, 1986 SIGGRAPH proceedings

[Val03] Valente F., ‘Compensation of springback of sheet metal forming by die design’, proceedings ICAFT 2003, R. Neugebauer (ed.), p.133-147 Verlag Wissenschaftliche Scripten, Zwickau 2003

[Wag02] Wagoner R., ‘Fundamental aspects of springback in sheet metal forming’, proceedings NUMISHEET2002, , D. Yang et al. (eds.), p.13-19, 2002

[Wag03] Wagoner R., ‘Design of sheet forming dies for springback compensation’, proceedings ESAFORM2003, V. Brucato (ed.), p.7-14, 2003

Glossary

Active area	Part of the product where springback compensation is applied
Artificial strain	Strain introduced in the tool geometry by the springback compensation algorithm during modification of the tool geometry
Assembly springback	Springback with boundary conditions as if the product were fastened in place in a larger assembly
Base point	Point, defining the location of a line along which a control point can be moved
Continuous modification	Application of an interpolated shape modification field to a mesh
Control surface	Surface that is used to approximate a geometry or to modify a geometry
Control point	Point that controls the shape of a parametric curve or surface
CS method	Control Surface method
DA method	Displacement Adjustment method
Degree elevation	Increasing the degree(s) of a parametric curve or surface
Die addendum	Added geometry to construct tools from the product geometry
Direction	Vector defining the direction of the line along which a control point can be moved
Effectivity (of the algorithm)	Percentage of the shape deviation (due to springback) that has been reduced
Free springback	Unconstrained springback, without boundary conditions that influence the product shape
Location (value)	Coordinate along a 3D line, defining the position of a control point
NLIB	C++ library containing numerical subprograms
Nodal modification	Direct application of a discrete shape modification field on a mesh
Offset (value)	Distance from a node to a control surface (possibly negative)
Offset vector	Vector, normal to a control surface, pointing at a node
Overbending	Compensation of springback in the tool geometry to produce geometrically accurate products
Overbending factor	Factor between the shape deviation due to springback and the shape modification in the tool geometry
Parameter	Variable during optimisation, or local coordinate on parametric curve or surface
Quasi Cartesian	Type of coordinate system along a control surface
Reference mesh	Blank mesh after the deep drawing phase, with tools closed. Desired geometry