

CADO: a Computer Aided Design and Optimization Tool for Turbomachinery Applications

Tom Verstraete

Von Karman Institute for Fluid Dynamics, Sint-Genesius-Rode, Belgium, tom.verstraete@vki.ac.be

Abstract

This paper presents an optimization tool used for the design of turbomachinery components. It is based on a metamodel assisted evolutionary approach combined with a Computer Aided Design (CAD) library for the shape generation. The analysis of the components performance is performed by Computational Solid Mechanics (CSM) and Computational Fluid Dynamics (CFD). Typical optimization problems include the design of axial and radial compressors/turbines. One important aspect in shape optimization of turbomachinery components is their parameterization. A detailed description of the parameterization and choice of parameters is discussed. A robust geometry generation tool is presented with special care for detailed geometrical features such as fillet radii. The success of automated designs depends as well on the robustness of the grid generation process, which is also discussed. Several examples illustrate the capabilities of the presented approach. The use of optimization techniques allows to speed up the design process and introduces innovative designs. **Keywords:** Turbomachinery, Computer Aided Design, Computational Solid Mechanics, Computational Fluid Mechanics.

1. Introduction

A current trend in turbomachinery component design is to rely more and more on predictions from numerical tools in the design process. Where in the past components were designed by using rules of thumb and simple correlations coming from theoretical considerations and experimental experience, nowadays computational tools such as Computational Fluid Dynamics (CFD) and Computational Structural Mechanics (CSM) are well integrated into the design process.

In order to reduce the time-to-market while improving the product quality, optimization techniques are currently very attractive. They tend to replace the time-consuming trial and error design method by an automated design procedure. However, many component designs require a multidisciplinary approach, which is often achieved by an iterative exchange between different departments. For example, in a traditional turbine blade design the blade is first designed and optimized by the aerodynamic department before being passed to the structural department. If from a structural point of view the blade does not meet the requested targets, it is sent back to the aero design with some restrictions on the design space. Many iterations between the different departments may be needed before a compromise is found.

On the other hand, in an optimization strategy using the concurrent approach, all disciplines are evaluated at the same time, and modifications to the design are made based on a global view of its performance in the different fields. This approach allows eliminating the time-consuming iterations between different departments and reduces the design cost. Moreover, the direct interaction between the different disciplines results often in innovative designs that would not have been obtained by traditional design approaches.

This paper presents a Computer Aided Design and Optimization tool (CADO) for turbomachinery applications, developed at the von Karman Institute for Fluid Dynamics. The main components of the tool are a Computer Aided Graphical Design (CAGD) library for the automated generation of blades, an automatic mesh generation tool both for the fluid domain (finite volume mesh) and the solid domain (finite element mesh), a Computational Solid Mechanics (CSM) and Computational Fluid Dynamics (CFD) code and a metamodel assisted evolutionary algorithm as optimization method, driving the entire design process.

The parameterization of a design is a very crucial step in the set-up of an optimization process. This paper will describe the parameterization used for an axial turbine and a radial compressor. The parameterization for other components can be derived from these two examples. Starting from the parameterization, the CAGD process describing the geometry will be presented. The fluid and solid grid generation process is described, starting from which a CFD and/or a CSM computation can be performed.

2. Optimization Method

Evolutionary Algorithms (EA) have been developed in the late sixties by J. Holland [5] and I. Rechenberg [12]. They are based on Darwinian evolution, whereby populations of individuals evolve over a search space and adapt to the environment by the use of different mechanisms such as mutation, crossover and selection. Individuals with a higher fitness have more chance to survive and/or get reproduced.

When applied to design optimization problems, EAs have certain advantages above gradient based methods. They do not require the objective function to be continuous and are noise tolerant. In the presence of local minima, they are capable of finding global optima and avoid to get trapped in a local minimum. Moreover, these methods can efficiently use distributed and parallel computing resources since multiple evaluations can be performed independently. The evaluation itself does not necessary needs to be made parallel. Disadvantages of EAs are mainly related to the large number of function evaluations needed.

2.1. Single-objective Differential Evolution

Differential Evolution (DE) is a relatively new evolutionary method developed by Price and Storn [10]. It is easily programmable, does only require a few user defined parameters and performs well for a wide variety of these parameters. A determination of optimal user defined parameters is very often unnecessary.

Differential evolution, like all EAs, is population based and requires at each iteration the evaluation of an entire population of designs. The nomenclature resembles the one of evolutionary processes. A design vector \vec{x} is called an individual; the collection of individuals at one given iteration is called a population, and the evolution of a population happens within generations, i.e. the children of the current population form the next generation.

To describe one version of the single-objective DE [10], the t -th generation containing T individuals is considered. Each individual \vec{x}_t contains n parameters.

$$\vec{x}_t = (x_1, x_2, \dots, x_n) \quad (1)$$

To evolve the parameter vector \vec{x}_t , three other parameter vectors \vec{a}_t , \vec{b}_t and \vec{c}_t are randomly picked such that $\vec{a}_t \neq \vec{b}_t \neq \vec{c}_t \neq \vec{x}_t$. A trial vector \vec{y}_t is defined as

$$y_i = a_i + F \cdot (b_i - c_i) \quad i = 1..n \quad (2)$$

where F is a user defined constant ($F \in]0, 2[$) which controls the amplification of the differential variation ($b_i - c_i$). This procedure is usually called the mutation. The candidate vector \vec{z} is obtained by a recombination operator involving the vectors \vec{x}_t and \vec{y}_t and is defined as

$$z_i = \begin{cases} y_i & \text{if } r_i \leq C \\ x_i & \text{if } r_i > C \end{cases} \quad i = 1..n \quad (3)$$

where r_i is a uniformly distributed random variable ($0 \leq r_i < 1$) and C is a user defined constant ($C \in]0, 1[$). This procedure is usually called the crossover, in analogy with Genetic Algorithms (GA). The final step in the evolution of \vec{x}_t involves the selection process and, for the minimization of the objective function $f(\vec{x}_t)$, is given by

$$\vec{x}_{t+1} = \begin{cases} \vec{z} & \text{if } f(\vec{z}) \leq f(\vec{x}_t) \\ \vec{x}_t & \text{if } f(\vec{z}) > f(\vec{x}_t) \end{cases} \quad (4)$$

The selection process involves a simple replacement of the original parameter vector with the candidate vector if the objective function decreases by such an action.

2.2. Multi-objective Differential Evolution

Many strategies exist to extend DE to multi-objective problems. Most of the algorithms are based on similar techniques used in GAs [1, 7]. Rai [11] introduces a selection of parents based on the distance in objective space. The idea is that in the final stages of the optimization, the differential variation ($b_i - c_i$) should result form individuals close to the selected individual \vec{x}_t . As such, the method behaves similar to a single-objective DE, where in the final stages the differential variations reduce in magnitude, as all

individuals locate near the optimum. In the methods of Abbas et al. [1] and Madavan [7], the distance in parameter space between the different individuals of the Pareto front can be very large, even in the final stages of the evolution. This will result in large changes of the proposed individual $\bar{\mathbf{z}}$, which is an unwanted feature. By favoring the individuals close to $\bar{\mathbf{x}}_t$ in the selection of the individuals $\bar{\mathbf{a}}_t$, $\bar{\mathbf{b}}_t$ and $\bar{\mathbf{c}}_t$, this is avoided. The later method is implemented in CADO.

2.3. Metamodel assisted Differential Evolution

The major drawback of evolutionary algorithms such as DE is the total number of evaluations needed. In general, more than thousand evaluations are commonly needed, and depending on the complexity of the optimization problem (both number of parameters and complexity of the objective function), this number can drastically increase.

One way of reducing the unrealistic number of evaluations can be obtained by replacing the expensive evaluations (involving FVM and/or FEM) by a computationally cheaper method. This could be achieved by using a low fidelity model that is still based on a physical model, for instance by reducing the number of gird points, or reducing the complexity of the model (e.g. replacing the NS Solver by an Euler solver).

A next step in reducing the total computational cost is to use an even less accurate model, not anymore based on a physical model, but on an interpolation of already analyzed individuals by the higher fidelity models [15, 14, 3]. Such techniques are in literature mostly referred to as “Metamodel Assisted Evolutionary Algorithms”.

The metamodel performs the same task as the high fidelity model, but at a very low computational cost. However, it is less accurate, especially for an evaluation far away from the already analyzed points in the design space.

The implementation of the metamodel into the optimization system depends on how the system deals with the inaccuracy of the model. The technique implemented in CADO uses the metamodel as an evaluation tool during the entire evolutionary process [9, 16]. After several generations the evolution is stopped and the best individual is analyzed by the expensive analysis tool. This technique is referred to as the “off-line trained metamodel”. The difference between the predicted value of the metamodel and the high fidelity tool is a direct measure for the accuracy of the metamodel. Usually at the start this difference is rather large. The newly evaluated individual is added to the database used for the interpolation and the metamodel will be more accurate in the region where previously the evolutionary algorithm was predicting a minimum. This feedback is the most essential part of the algorithm as it makes the system self-learning. It mimics the human designer which learns from his mistakes on previous designs.

The implementation of the previous described technique is different for a single- and multi-objective optimization problem. The algorithm for the single-objective problem is shown in Algorithm 1, and is referred to as SODE2L (Single-Objective Differential Evolution with 2 Levels).

Algorithm 1 Single-objective metamodel assisted differential evolution (SODE2L)

- 1: create an initial database
 - 2: **for** $it = 1$ to *required* **do**
 - 3: train metamodel on database
 - 4: perform single-objective DE by using the metamodel
 - 5: evaluate the best individual by expensive evaluation tool and add to the database
 - 6: **end for**
-

The database stores the relationship between design vectors and performance vectors of the designs already analyzed by the accurate evaluation. It serves as the knowledge to teach the metamodels.

The accuracy of the approximation predictions strongly depends on the information contained in the database. Several geometries are analyzed at the start of the optimization algorithm by the accurate evaluation in order to train the first metamodel. Although a creation of random samples can be used to populate the initial database, more elaborate methods exist. The Design Of Experiments (DOE) method [8] is used in present optimization algorithm [6]. This maximizes the amount of information contained in the database for a limited number of computations [8].

It is essential to note that the algorithm is self-learning as each iteration a trial design is added to the database, increasing the knowledge of the system. The algorithm contains two major loops. The outer loop, shown in Algorithm 1, is called an iteration, while the inner loop, which consists in the evolution process of DE and is not shown in Algorithm 1, is called a generation. Since the metamodel evaluation

is performed very rapidly, a large number of metamodel evaluations are possible during the DE process. Usually very large number of generations (1,000 to 10,000) can be performed on large populations (40 to 100 individuals) within seconds.

The implementation for a multi-objective optimization problem differs from the previous method as now the DE algorithm results not in one single best design but in a set of non-dominated individuals. The method is shown in Algorithm 2 and is referred to as MODE2L. From the set of non-dominated individuals, only a few individuals are selected, to reduce the computational cost of the method, and to avoid a clustering in some areas of the design space. The same method proposed by Deb for NSGA-II is used.

Algorithm 2 Multi-objective metamodel assisted differential evolution (MODE2L)

```

1: create an initial database
2: for  $it = 1$  to required do
3:   train metamodel on database
4:   perform multi-objective DE by using the metamodel
5:   select a reduced number of individuals from the Pareto front by a distance metric
6:   analyze all individuals from the reduced Pareto front by expensive evaluation tool and add to the
     database
7: end for

```

Several metamodels have been implemented within CADO. The user is free to decide for each performance variable which metamodel will be used, and can set different training techniques for the individual metamodels. Table 1 summarizes the different metamodels available. It is however also possible to avoid the use of a metamodel. In that case the FEM/FVM is directly used by the DE algorithm, resulting in a large computational cost. Usually the best geometries from a metamodel assisted optimization are used as starting point for such an optimization. These methods are called SODE1L and MODE1L for single respectively multi-objective optimization problems.

Table 1: Metamodels implemented in CADO

Abreviation	Description	Training method
PRRS	Polynomial Regression Response Surfaces	polynomial regression
ANNLAL	Artificial Neural Network	local adaptive learning
ANNDE	Artificial Neural Network	differential evolution
RBF	Radial Basis Function network	local adaptive learning
Kriging	Kriging	maximum likelihood optimization

3. Parameterisation of turbomachinery blades

The prudent selection of relevant geometry parameters is one of the most critical aspects of any shape optimization procedure. The optimal shape should be able to be represented by the parameterized model, which implies a good choice of the parameters *and* their range. However, the number of parameters need to be kept as low as possible and the range of each individual parameter needs to be limited in order to reduce the search space for the optimizer. Parameters for which the optimal value is easily determined by correlations should be excluded from the optimization. Simplifications can be made to the optimization problem by omitting parameters that are assumed to have little or no impact on the performance of the geometry.

Moreover, when using metamodel assisted optimization methods, it is crucial to have as simple as possible relations between the design parameters and performance. Metamodels typically tend to perform bad for highly non-linear relations, as they are difficult to be learned. Moreover, the high non-linearity requires as well a large design of experiments (DOE) before the start of the optimization, considering the higher order interactions between the different parameters. By limiting the DOE to a small number of experiments, metamodels might be trained with uncomplete information on the CSM/CFD response, resulting in large inaccuracies on the prediction of the performance. As a result, the optimum found by the optimization algorithm might be far from a global optimum, simply because the metamodel was not trained by information near the optimum of the CSM/CFD response.

Most designs of shapes rely on curves. In particular Bézier and Bspline curves [4] are highly suited for the parameterization of a design. They have a very simple formulation, are unique and the characteristics of the curve are strongly coupled with the underlying polygon of control points. Moreover, the degree of the curve can be easily increased. Following text includes some examples for using Bézier and Bspline curves for the parameterisation of typical turbomachinery blades.

3.1 Parameterisation of a 3D axial turbine blade

The parameterisation of a 3D axial turbine blade starts by the definition of 2D blade profiles at different radii. Typically at three different sections the blade profiles are defined (hub, mid and tip section), however, more sections are possible in case a more local control is needed (e.g. to control the secondary flows near the endwalls).

In Fig. 1 the parameterisation of a 2D profile is presented. It starts by the definition of a camberline, which is defined by an inlet blade metal angle β_{in} , an exit blade metal angle β_{out} , an axial chord length λ and a stagger angle γ . The stagger angle and axial chord length define the position of the leading edge and trailing edge. The intersection of the inlet and outlet lines respecting the inlet and outlet angle defines the point P_{Mid} shown on Fig. 1. The points P_{LE} , P_{Mid} and P_{TE} define the control points of the Bézier curve describing the camber line.

The suction and pressure side curve are also defined by Bézier curves, for which the control points are specified relative to the camber line. In Fig. 1 the construction of the suction side curve is illustrated. First a stretching law is imposed on the camberline curve, which is specified by a stretch factor and a number of points. For each point obtained on the camber line (with the exception of the first and two last points), a normal distance is specified to obtain the corresponding control point. For the first point the distance is computed by a specified radius of curvature ρ_{LE} , this to obtain geometric continuity of the second degree (G2) at the leading edge between the suction side and pressure side curve. The distance of the last point is equal to the trailing edge radius ρ_{TE} , also specified. Finally, the second last control point is computed by a specified wedge angle φ , see Fig. 1. The definition of the pressure side is similar but independent from that of the suction side. Only ρ_{LE} and ρ_{TE} are taken the same. Usually fewer control points are used on the pressure side.

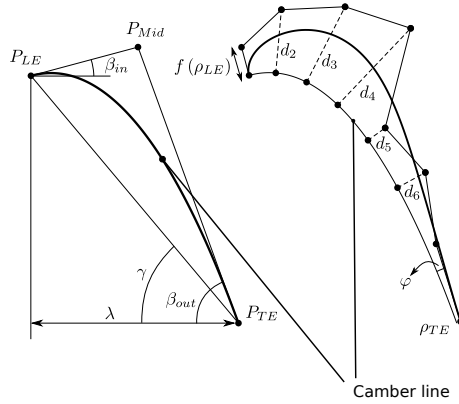


Figure 1: Parameterisation of the suction side of an axial turbine blade.

The three dimensional shape of an axial turbine is obtained by stacking the previously defined 2D blade profiles from hub to tip. In Fig. 2 the stacking process is shown for a blade defined at three different sections. Each section is laying on a cylindrical surface. The stacking of the profiles is performed through the center of gravity for rotor blades and through the trailing edge for turbine vanes, as is the case in Fig. 2. Finally, the straight stacked shape is altered by applying a lean and sweep distribution, applied to several intermediate sections between hub and shroud. The lean and sweep law are defined by Bézier curves as shown in Fig. 2, for which the control points are optimization parameters. In most cases the laws are defined by angle and distance parameters and not the cartesian coordinates of the control points, this to obtain an as straight forward relation as possible between parameters and performance values such as losses.

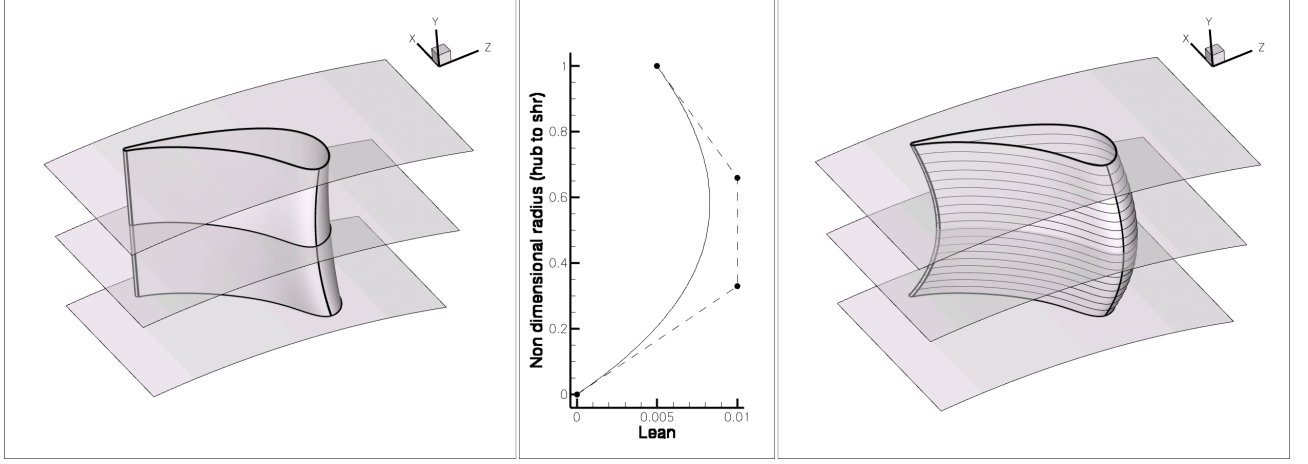


Figure 2: Parameterisation of the suction side of an axial turbine blade.

3.2 Parameterisation of a radial compressor

The 3D radial compressor is defined by:

- the meridional contour at hub and shroud (Fig. 3)
- the camber line of the main blade (Fig. 5)
- the thickness distribution normal to the camberline (Fig. 6)
- the number of blades

The meridional shape of the impeller is subdivided in several patches (7 in the figure), reaching from the inlet to the outlet. The forth patch in the figure corresponds to the position where the main blade is positioned. For each patch a curve needs to be defined at the hub and shroud. This curve can be defined by a Bézier curve or B-spline curve. The coordinates of the control points are geometrical parameters that can be changed by the optimization program and allow to modify the meridional passage. For some of the control points the coordinates are directly controlled by the optimizer, while other coordinates are controlled by other parameters, e.g. angles and distances. This parameterization has been introduced to easily respect tangency at each interface of two patches, and to have a better control on the different shapes that can be generated by the optimizer. For each parameter upper and lower limits are defined.

The blade camber lines at hub and shroud are defined by the distribution of the angle $\beta(u)$ between the meridional plane m and the streamline S (Fig. 5, with $u \in [0, 1]$ the non-dimensional meridional length, 0 at the leading edge and 1 at the trailing edge). They are usually defined by a Bézier curve with 4 control points as shown in Fig. 4. The value of β_0 and β_3 are respectively the inlet and outlet blade angle. The two remaining values β_1 and β_2 define the Bézier control points at 1/3 and 2/3 of the meridional length respectively. The camber line circumferential position θ (Fig. 5) is then defined by

$$R \cdot d\theta = dm \cdot \tan(\beta) \quad (5)$$

and allows the transformation from the β distribution to (x, y, z) coordinates of the camberline provided that the meridional shape is given. The Bézier curve is used for both hub and shroud definitions, totaling 8 design parameters for the blade camber line definition. In general, also the ordinates of the internal control points (here fixed at 1/3 and 2/3) can be changed as well as the number of control points.

The relative position of the camberline at hub and shroud needs to be defined as well. Usually the leading edge of the hub camberline is positioned at $\theta = 0$ in cylindrical coordinates. The shroud camberline can then be freely chosen with respect to the hub camberline. Two possibilities exist, depending on whether the leading edge (lean) or trailing edge (rake) position is defined with respect to the hub camber line. In most applications the rake is used as a parameter as it is defined at a higher radius than the lean, resulting in a smaller sensitivity.

Connecting the camberlines at hub and shroud results in an infinitely thin blade. Adding a thickness distribution at hub and shroud results in the final blade shape. The blade thickness distributions at hub and shroud are defined by a B-spline distribution as given in Fig. 6. All coordinates of the control points can be added as optimization parameters, except for the leading and trailing edge ones, and the second

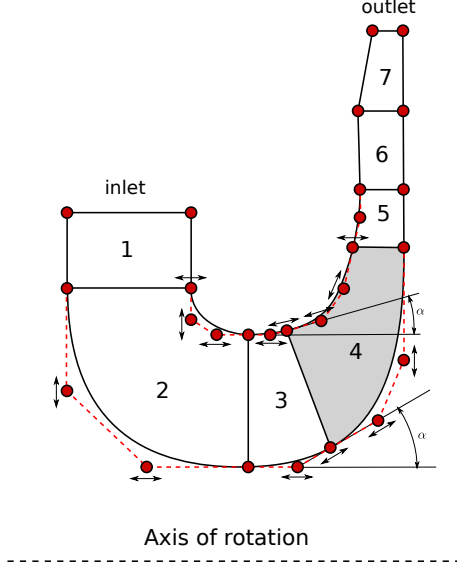


Figure 3: Definition of the meridional contour.

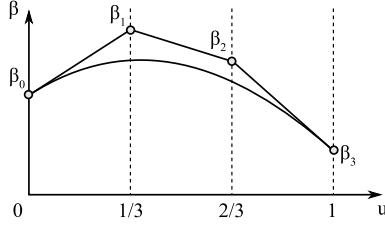


Figure 4: Beta distribution.

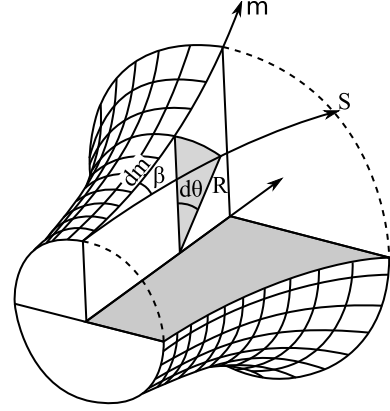


Figure 5: Camberline definition by a beta distribution.

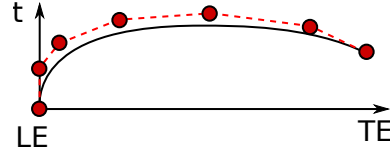


Figure 6: Blade thickness definition.

control point needs to be on the y-axis to obtain a G2 continuous leading edge. It is also possible to specify the radius of curvature at the leading edge, which defines the position of the second control point.

The number of blades is also a possible design parameter to be optimized, but in most cases it is fixed for vibration and manufacturing reasons. The total number of design parameters varies between 20 and 35 in general.

4. Geometry and mesh generation

A first step in the evaluation process consists of generating the geometry and discretizing it for the subsequent FEM and FVM analysis. The geometry is first constructed from its definition into a geometrical representation, which consists of a collection of faces, edges and vertices. This representation is used to construct the grid for the stress and NS computations.

The generation of the geometrical representation of the shape is made through the use of points, curves and surfaces. For the curves it is common in CAGD to use Bézier and B-spline curves [4], and Bézier and B-spline surfaces are also commonly used for the surfaces. Bézier curves are often used to define the geometry, as explained in the previous section, while B-splines are more used for the definition of complex shapes due to their higher flexibility. As an example, the hub curve in the meridional passage of the radial compressor is defined by several Bézier curves, while in the resulting CAGD model these curves are united into one single B-spline curve. Following section will provide some explanation on how the CAGD model of the radial compressor is constructed. Several CAGD methods are necessary for this and are implemented in CADO, including:

- interpolation of a set of points by a B-spline curve (with/without tangent directions)
- intersection of 2 curves in 2D
- intersection of a curve and surface in 3D
- intersection of two surfaces
- projecting a point to a curve (2D,3D) or surface

- offsetting surfaces and curves by a distance
- extending a curve or surface in a continuous way
- joining several curves into one single continuous curve
- creation of a surface by a loft through several curves

The generation of the surfaces for the suction and pressure side of the blade are performed in the following way. First, the camberline at hub and shroud is defined by integrating Eqn. 5 using the definition of the meridional curve and β -distribution, respectively at hub and shroud. The integration is performed by using a Legendre-Gauss quadrature of the 8th order in 50 intervals. This results in 51 points, for which a B-spline curve is constructed interpolating the set of points. In a next step a camberline surface is constructed by a loft through the camberline curves at hub and shroud. The thickness distribution, specified for hub and shroud, is placed perpendicular to this surface for both suction and pressure side. This is performed by running from leading edge to trailing edge on both camberlines (hub and shroud) in a specified number of steps. Smaller steps are used near the leading edge, since there a rapid variation of the thickness is present. A point is constructed by offsetting half the thickness perpendicular to the camber surface at that location in both directions. The resulting set of points is interpolated by a B-spline for both suction and pressure side at hub and shroud. A small correction to both curves is made near the leading edge to guarantee a G2 continuous leading edge. The second control point of both curves is modified such that the curves are tangent and have the same radius of curvature at the leading edge. Finally, a B-spline surface is lofted through the two suction side curves to generate the suction side surface, a similar procedure is used for the pressure side. Both surfaces are extended in a G3 continuous way near the hub, shroud and trailing edge, this because for the later need to intersect with other surfaces.

In Fig. 7(a) a view on the main blade together with the hub and shroud surfaces is given. As can be seen at the shroud, the blade surface has been extended to allow the generation of a fillet radius between the meridional surface and blade. A detailed discussion on the generation of the fillet radius will not be given. It involves several complicated CAGD operations, such as offsetting of a surface and intersection of two surfaces. In Fig. 7(b) a detailed view on the fillet radius is given. The mathematical representation of the G3 smooth surfaces are represented by a structured grid. However, the actual surface is much smoother than its discretized representation.

The creation of the geometry by CAGD entities such as B-spline surfaces allows to export the mathematical definition of the geometry to commercial CAD packages. In Fig. 8 the result of such an export is shown, after performing a periodic repetition of the generated part. The export was performed using the STEP format.

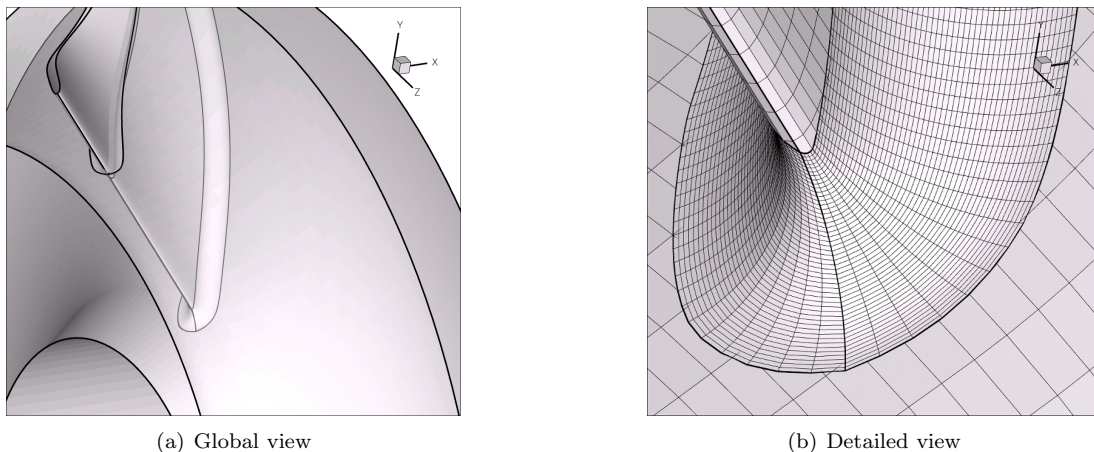


Figure 7: View on the fillet radius.

The fluid and solid domains are defined by the geometrical model. For the evaluation of the geometry in terms of aerodynamics and structural mechanics, these domains need to be discretized by a finite volume and finite element grid respectively. For the fluid domain, the fillet radius is not included and the grid used is a structured grid. First a grid in the meridional plane from hub to shroud is generated, with a stretching near the walls. For each grid line in the meridional plane, a surface of revolution is constructed. The intersection with the blade is computed, and finally a structured grid in this surface is

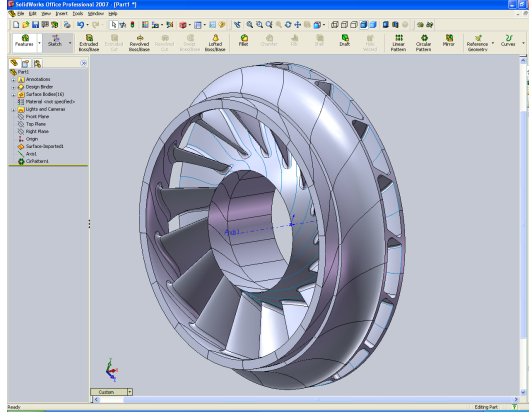


Figure 8: View on the exported geometry in a commercial CAD package.

generated by an elliptic smoother. The collection of all grids on these meridional surfaces constructs the full 3D grid.

For the solid domain, unstructured quadratic tetrahedral elements are used as a compromise between element quality and automatic meshing. First the bounding surfaces are meshed by using a Delauny mesh procedure. The interior solid grid is then constructed using a similar procedure in 3D. In Fig. 9 a view on the solid and fluid grid is shown.

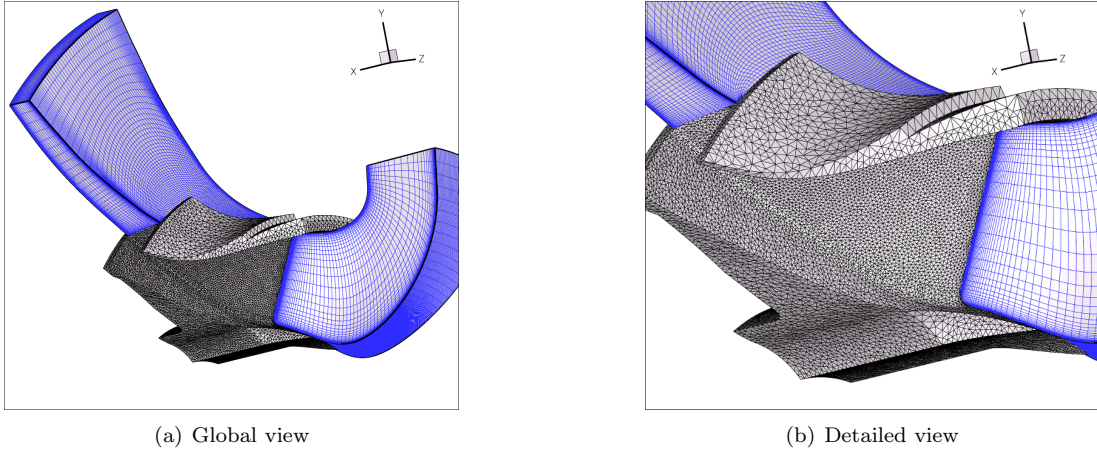


Figure 9: View on the mesh for both fluid and solid domain.

It was chosen to construct the geometry by an in-house CAGD code to have full access to the entire geometry generation process, allowing a robust and automated geometry generation process. Moreover, it allows a direct control over the parameterization of the design. The use of an in-house grid generation tool also simplifies the task of geometry exchange, and the specification of areas where the grid needs to be refined.

5. Geometry evaluation

The aerothermal and stress evaluation are made in parallel. The TRAF3D solver [2] is used to calculate the aerodynamic performance of the radial compressor. A structured H-grid with the same number of cells is used for all computations to guarantee a comparable accuracy for all the samples evaluated. This is important because erroneous information due to grid dependencies could drive the optimization process in a wrong direction. For each geometry, several CFD computations are performed in parallel, each with a different pressure ratio. As such, the entire performance map of the compressor or turbine is generated.

The commercial code SAMCEF [13] is used for the stress and/or vibration calculation. Quadratic tetrahedral elements are used as a compromise between element quality and automatic meshing. Similar

grids with typically 250 000 nodes and 150 000 elements are used for radial compressor shapes. The grid is refined in areas of stress concentrations, e.g. at the fillet connecting the blade to the hub. A periodic boundary condition is applied in the case of radial turbomachinery, such that only a limited part of the geometry needs to be analyzed. For axial turbomachinery components, the root of rotor blades is modelled and considered as clamped in the structural computations.

6. Conclusion

An optimization tool for the design of turbomachinery components has been presented. It consists of an automated design method including CAGD and mesh generation tools. A concurrent strategy is adopted for the multi-disciplinary design approach, allowing the search for a global optimum satisfying all constraints. The method presented is used on a daily basis for the design of typical turbomachinery components, reducing severely the design effort, and has lead to innovating and surprising design choices, that would not have been investigated at first sight.

References

- [1] H. A. Abbas, R. Sarker, and C. Newton. PDE: A Pareto-Frontier Differential Evolution Approach for Multi-objective Optimization Problems. In *Proceedings of the Congress on Evolutionary Computation*, volume 2, pages 971–978, Piscataway, New Jersey, 2001.
- [2] A. Arnone. Viscous analysis of Three-Dimensional Rotor Flow Using a Multigrid Method. *ASME Journal of Turbomachinery*, 116:435–445, July 1994.
- [3] J.-F. M. Barthelemy. Approximation Concepts for Optimum Structural Design - a Review. *Structural Optimization*, pages 129–144, 1993.
- [4] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design*. Academic Press, Inc., San Diego, 1993.
- [5] J. H. Holland. *Adaption in Natural and Artificial Systems*. University of Michigan Press, 1975.
- [6] K. Kostrewa, Z. Alsalihi, and R. A. Van den Braembussche. Optimization of Radial Turbines by Means of Design of Experiment. In *Tech. Rep. VKI-PR-2003-17*, 2003.
- [7] N. K. Madavan. Multiobjective Optimization Using a Pareto Differential Evolution Approach. In *Proceedings of the Congress on Evolutionary Computation*, volume 2, pages 1145–1150, Honolulu, Hawaii, 2002.
- [8] D. C. Montgomery. *Design and Analysis of Experiments*. John Wiley and Sons, New York, 1997.
- [9] S. Pierret. *Designing Turbomachinery Blades by means of the Function Approximation Concept based on Artificial Neural Network, Genetic Algorithm, and the Navier-Stokes Equations*. PhD Thesis, Faculte Polytechnique de Mons, December 1999.
- [10] K. Price and N. Storn. Differential Evolution. *Dr. Dobb's Journal*, pages 18–24, April 1997.
- [11] M. Rai. Single and multiple objective optimization with differential evolution and neural networks. In *VKI Lecture Series on Introduction to Optimization and Multidisciplinary Design in Aeronautics and Turbomachinery*, Brussels, June 2008.
- [12] I. Rechenberg. *Evolutionsstrategie — Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Fommann-Holzboog, Stuttgart, 1973.
- [13] SAMTECH S.A. SAMCEF V12.2 Manual. Technical report, 2008. See also URL <http://www.samcef.com>.
- [14] T. W. Simpson. Comparison of Response Surface and Kriging Models in the Multidisciplinary Design of an Aerospace Nozzle. 1998. ICASE Report No. 98-16.
- [15] V. Torczon and M. W. Trosset. Using approximations to accelerate engineering design optimization. August 1998. ICASE Report No. 98-33.
- [16] T. Verstraete. *Multidisciplinary Turbomachinery Component Optimization Considering Performance, Stress, and Internal Heat Transfer*. PhD Thesis, Universty of Ghent, June 2008.