Direct Manipulation of Free-Form Deformation in Evolutionary Optimisation

Direkte Manipulation von Free-Form Deformation in Evolutionärer Optimierung

von

Thomas Bihrer

Diplomarbeit

im Studiengang *Informatik*

am Fachgebiet Simulation und Systemoptimierung

der Technischen Universität Darmstadt

bei Prof. Dr. Oskar von Stryk

Die Arbeit wurde erstellt in Kooperation mit
Honda Research Institute Europe GmbH
63073 Offenbach/Main

Betreuer: Dr. Stefan Menzel (HRI)

Darmstadt, Juni 2006

Abstract

The choice of an adequate representation is an important part of every evolutionary design optimisation. A flexible representation which allows a huge number of design variation while getting along with a few parameters is preferable. In the past the Free-From Deformation method was successfully applied for design optimisation. In this diploma thesis an adaptive Free-Form Deformation is combined with an Evolution Strategy. The aim is to automatically create a representation which is adapted to the problem and reduce the dependence on the initial representation build by a designer. Furthermore it should be analysed if such an adaptation can lead to a faster convergence.
In a second part a direct manipulation technique is presented to increase the influence of the object parameter on the design in order to speed up the optimisation. Both representations are tested at target matching problems.
If the optimisation is applied to a fluid dynamics optimisation problem, a computational grid is required to evaluate the designs. Since the Free-From Deformation represents deformations of an initial design, it can also be applied to a grid. Thus a costly re-meshing procedure can be omitted. To keep the structural composition of the mesh the deformations have to be restricted. The influence of these additional constraints is also analysed.

Kurzfassung

Die Wahl einer geeigneten Repräsentation spielt eine entscheidende Rolle in jeder Evolutionäre Design Optimierung. Eine flexible Repräsentation, die eine möglichst große Anzahl an Variationen ermöglicht und zugleich mit wenigen Parametern auskommt ist wünschenswert. In der Vergangenheit wurden Free-Form Deformation Methoden erfolgreich in der Design Optimierung angewendet. In dieser Diplomarbeit wird eine adaptive Free-Form Deformation mit einer Evolutionsstrategie kombiniert. Ziel ist es die Repräsentation automatisch an das Problem anzupassen und damit die Abhängigkeit von der vom Designer entworfenen Repräsentation zuverringern. Desweiteren soll untersucht werden ob die Adaption zu einer schnelleren Konvergenz der Optimierung führen kann.
In einem zweiten Teil soll durch die Verwendung einer direkten Manipulations Technik der Einfluss der Optimierungsparameter auf das Design verstärkt werden, um dadurch eine schnellere Konvergenz zu erreichen. Beide Repräsentation werden an Design Matching Problemen getestet.
Wird die Optimierung auf ein Strömungsdynamik Problem angewendet, ist ein Rechengitter notwendig um das Design zu bewerten. Da die Free-Form Deformation die Verformung von einem existierenden Design representiert, kann sie auch dazu verwendet werden das Rechengitter zu verformen und damit eine aufwendige Gitter Erzeugung vermieden werden. Damit das Analyse Gitter nach der Verformung gültig bleibt, muss diese beschränkt werden. Der Einfluss dieser zusätzlichen Nebenbedingungen wird ebenfalls analysiert.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

The efficient development of optimal shape geometries is an important part for every product design process. The integration of computational simulation methods supports designers and engineers by providing any kind of required information and reduces costly product development time. An autonomous design optimisation environment producing the optimal shape with as few as possible human interaction is preferable. To be applicable, such an environment has to fulfil different requirements.
Firstly, reliable computer software for simulating the design behaviour in the real world, e.g. computational fluid dynamic (CFD) or finite element method (FEM) simulation, is necessary. It has to model the real world accurately, because the calculated designs are primarily optimal in this simulation. These simulations are based on computational grids discretisizing the problem. Since the quality of the simulations depend on these grids, they have to be adjusted to the design.
Secondly, the optimiser, leading the optimisation process, has to improve the performance of the design to its optimum in as few optimisation steps as possible.
In the past many algorithms have been proposed to fulfil these requirements for a given optimisation problem. Beside deterministic approaches like gradient based and Newton methods, evolutionary computation becomes more and more popular. Especially for the optimisation of complex systems and problems where a derivative is not easily available. Their increased popularity can also be ascribed to the fast advancement of computer processing power and their ability for parallelisation. A more detailed description of evolutionary algorithms is given in section 1.2.
Thirdly, the representation of the shape. It specifies the search space and hence determines which solutions can be found by the optimiser. Therefore one has to ensure that the optimal design is codeable by the representation. That can be achieved by choosing a representation which enables a large variation by using a huge amount

of variables.
Controversy the number of parameter should be as low as possible to limit the search space and enable a fast convergence of the optimisation.
A representation which provides a good trade off between compactness and completeness especially for complex designs is the Free-Form Deformation (FFD). Since this representation specifies deformations of a control volume, very complex designs can be represented with a few parameters while still a large variability is provided. This kind of representation has another major advantage. The computational grid which is necessary for the evaluation of the design has not to be regenerated or repaired for the different designs produced during the optimisation. The FFD representation allows to deform an initial mesh simultaneously to the design and thus keep the quality of the computational grid. For complex designs the mesh generation of unstructured grids often requires manual interventions. These human interactions prevent an automatic optimisation process. Also automatic meshing approaches exist but they are adapted to a certain problem or can only mesh simple shapes. And even if such an automatic approach is available the meshing of complex designs which contain many edges and ridges, it requires generally a lot of time.
To guarantee the structural composition of the mesh after the deformation, an additional constraint to the Free-Form Deformation is introduced. Its influence on the optimisation is analysed in this diploma thesis. Algorithms necessary to maintain the constraint during the optimisation are developed and compared.
To improve the performance of the optimisation, the designers expend much effort to build an adequate representation. They use their problem specific knowledge about possible solutions to reduce the number of parameters without excluding promising designs from the search space. But if no roughly knowledge of the optimum is available, it is difficult or even impossible to build a proper representation. Due to the assumptions about the solution used by the designers to build a compact representation, new and unexpected solutions could be excluded from the search space.
To reduce the dependence of the representation build by designers, in this diploma thesis an adaptive Free-Form Deformation, which starts with a global representation, is developed. During the optimisation this global representation is further refined in promising areas. Such an adaptive approach should also enable to find new and unexpected solutions.
Furthermore, the optimal representation is not static over the whole optimisation process. An adaptive representation which increases the degree of freedom should enable to adapt the representation to the current optimisation state in order to speed up the convergence. Typically the optimisation begins with a macroscopic search which becomes more and more local during the process. Increasing the number of parameter during the optimisation should support this search strategy.
The deformations of the control volume in the FFD are defined by control points. The influence of these points on the design which is embedded into this volume is quit different. To achieve a high variability with few control points, their influence has to be maximised by placing the control points adequately. But due to the math-

ematical definition, it is not possible to get a high influence for all points. To achieve a larger and straighter impact on the design, a direct manipulation technique for the Free-Form Deformation is implemented to specify the deformations. The enlarged impact should speed up the optimisation process.
The interaction of this representation with an Evolution Strategy is analysed and algorithms which are necessary to use this representation in the constraint optimisation are developed.
The different FFD representations are all tested in an Evolution Strategy to optimise a design with respect to match a given target.
The arrangement of this thesis is as follows. Section two of this chapter gives an introduction to evolutionary algorithms especially the Evolutionary Strategy with covariance matrix adaptation. Afterwards the basics of Free-Form Deformation and the direct manipulation technique are discussed. Chapter two gives an overview of already used Free-Form Deformation representations in design optimisation. In chapter three the Evolution Strategy using the Free-Form Deformation as representation is explained and its advantages and limitations are exposed. Chapter four details the necessary constraint to keep the structural composition of computational grid. Different approaches to keep the constraint during the optimisation are explained and compared. The adaptive representation is introduced in chapter five and compared with the non adaptive representation. In chapter six the developed direct manipulation FFD approach is explained. The last chapter contains the conclusion of the results and different further investigations are pointed out.

1.2 Evolutionary algorithms

Evolutionary algorithms are general-purpose search procedures and belong to the group of stochastic algorithms. Mechanisms inspired by biological evolution, such as reproduction, mutation, recombination, natural selection and survival of the fittest, are used to solve an optimisation problem. A set of candidate solutions, a so called population, is adapted over several generations. In each generation new variations of solutions, the offspring population, are produced by a combinatoric and stochastic operator, the recombination, and a purely stochastic operator, the mutation. The best solutions with respect to the cost function, called fitness function in the field of Evolutionary Algorithm, are selected for the next generation. Figure 1.1 illustrates the cycle of an evolutionary algorithm.
It is supposed that evolutionary algorithms perform consistently well across all types of problems because they do not make any assumptions about the fitness landscape. It has not to be continuously and no derivatives have to be available. But the absence of information about the fitness landscape leads to a higher amount of fitness evaluations.
Popular examples of evolutionary algorithms are Evolution Strategies (ES), Genetic

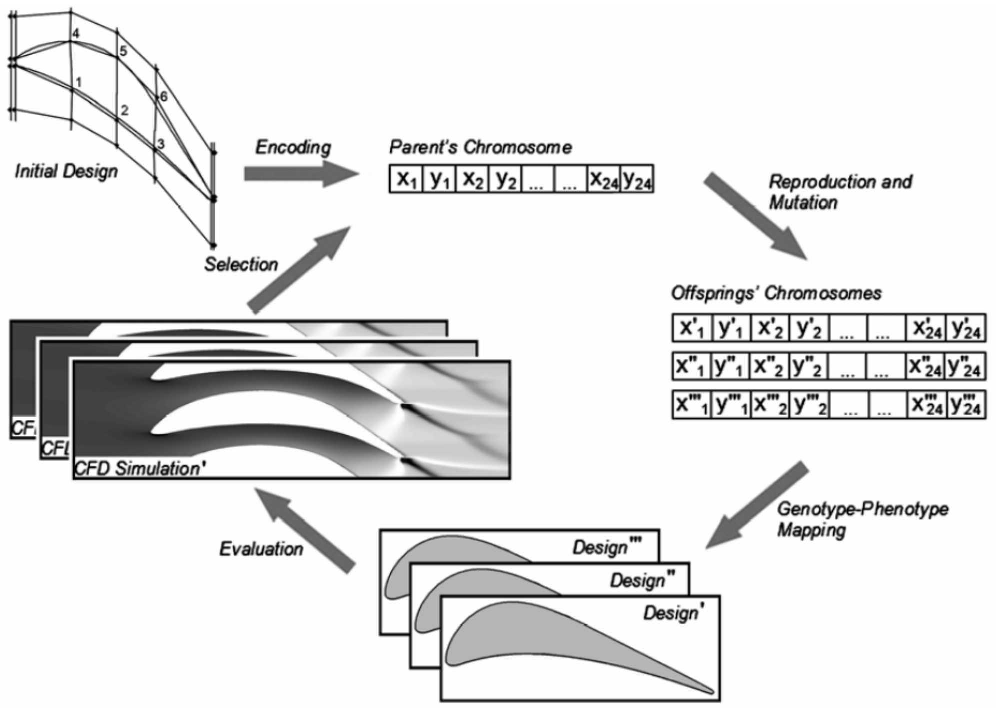

Initial Design
Encoding
Parent's Chromosome
x1 y1 x2 y2 ... ... x24 y24
Reproduction and Mutation
Offsprings' Chromosomes
x'1 y'1 x'2 y'2 ... ... x'24 y'24
x''1 y''1 x''2 y''2 ... ... x''24 y''24
x'''1 y'''1 x'''2 y'''2 ... ... x'''24 y'''24
Genotype-Phenotype Mapping
Design'''
Design''
Design'
Evaluation
CFD Simulation'
Selection

Figure 1.1: Optimisation cycle of an evolutionary algorithm [10].

Algorithm (GA) and Genetic Programming (GP). In this thesis Evolution Strategy and a special variant, the Covariance Matrix Adaptation ES (CMA-ES), are used.

1.2.1 Evolution Strategy

Evolution Strategy was introduced by Rechenberg in the sixties and later refined by himself[15] and Schwefel[19].
As shown in figure 1.1 the Evolution Strategy starts with the initialisation of the parent population. In the case of design optimisation, the initial parameterised shape is encoded into the chromosomes of the individuals. These chromosomes represent the genotype of an individual and contain the object parameter. In the case of an ES they are encoded as continuous numbers.
Next the offspring population is created by recombining the parents. In Evolution Strategy this recombination is typically a linear combination of two or more solutions, the mean of the whole population or an exact copy of one randomly chosen parent.
Then the offspring individuals are mutated by adding a gaussian distributed random vector with zero mean (figure 1.2) to the chromosomes. The variances σ^2 of the random vector, also called strategy parameters, are crucial for the ES. Evolutionary progress takes place only within a narrow band (evolution window) of this strategy parameter comparable with a step size. If the variance is too small, the optimisation

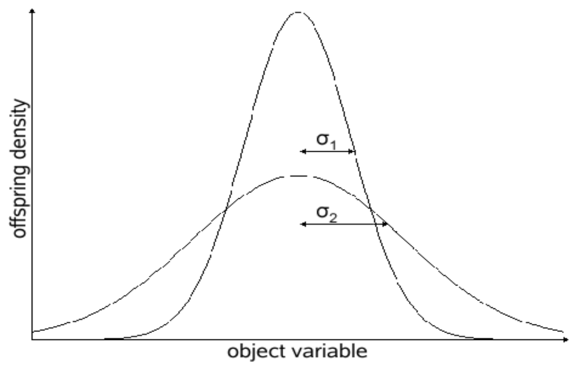


Figure 1.2: Gaussian distribution for two different variances σ_1 and σ_2.

process will stagnate whereas an oversized mutation leads to a regression. Therefore, they have to be adapted during the optimisation to the topology of the search space. The realisation of this adaption is explained in the next section.
The strategy parameter allows conclusions about the current state of the search process. Large strategy parameters indicate a global search whereas small parameters point out the convergence of the optimisation process.
Once the offsprings are mutated, the chromosomes are mapped to the phenotype of the design. The phenotype is the representation of the design used for the evaluation. It determines the behaviour of the design in the fitness evaluation. Then the fitness value for this design is evaluated and assigned to the offspring. In case of design optimisation normally a CFD simulation assesses the quality of the design.
After all offsprings are evaluated, the parents for the next generation are determined. Several selection methods exist. The best known are the (μ, λ)-selection and the $(\mu + \lambda)$-selection. In a (μ, λ)-selection the μ best individuals of the offspring population build the parents for the next generation. In contrary to the $(\mu + \lambda)$-strategy, which also considers the actual parents for the selection. Hereby, the best individuals are always kept which leads to a monotone increasing quality progress. On the other hand the $(\mu + \lambda)$-strategy makes it more difficult to overcome local optima.
After the new parents are determined the algorithm starts with a new cycle. These cycles are repeated until a stop criteria is reached. That can be specified with respect to the global strategy parameter, a fitness threshold or a maximum number of generations.

1.2.2 Self-adaptation

As already mentioned the mutation strategy parameter plays an important role in the Evolution Strategy. To adapt the variances, a self adaptation method is used. The same optimisation process as for the object variables is applied to the strategy

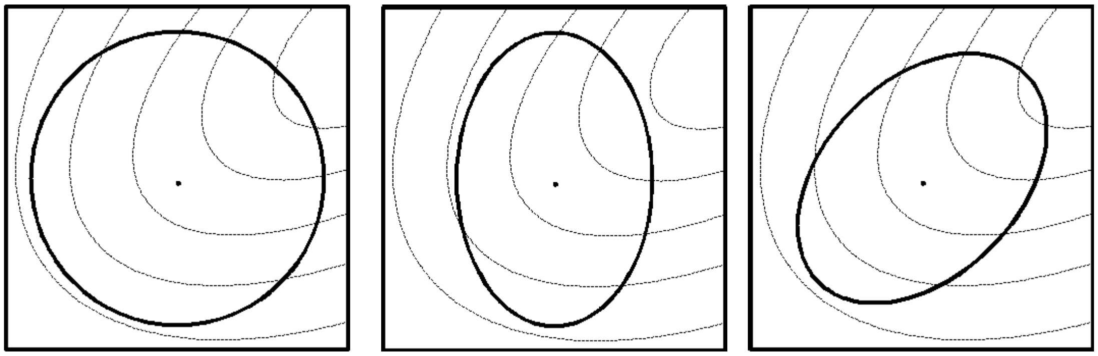

(a) $N(0,\sigma^2)$ (b) $N(0,D)$ (c) $N(0,C)$

Figure 1.3: The ellipsoids depicting one line of equal density for three different gaussian distributions, where $\sigma \in \mathbb{R}_+$, D is a diagonal matrix and C is a positive definite covariance matrix. The dotted lines depict exemplary objective function contour lines.

parameters. The different approaches vary mostly in the number of strategy parameters and the method of their adaptation. The gaussian distribution of three different self-adaptation strategies are shown in figure 1.3. The circle and the ellipses depict lines of equal probability. The probability of a random number decreases with the distance to the centre. In the left image only a single strategy parameter is used for all object variables. In this global step size adaptation (GSA) the variance for all object variables is equal. Thus the lines of equal probability are always spherical.
In the middle figure the individual step size adaptation (ISA) is applied. Every object parameter has its own step size σ_i. The normal distribution is an ellipsoid parallel too the axes of the coordinate system.
In the right figure the normal distribution of a whole covariance matrix is shown. Here, the ellipsoid can be arbitrarily oriented.
It is obvious that the adaptation to the topology of the search space for a single parameter is very limited whereas the use of the complete covariance matrix enables to adapt arbitrary directions. A detailed description of the different adaptation strategies is given in the next sections.

1.2.2.1 Mutative adaptation of the strategy parameter

The mutative adaptation like GSA and ISA are the standard approaches. In these approaches the strategy parameters are encoded additionally to the object variables into the chromosomes of each individual. Thus the same optimisation process is applied to them. The strategy parameter are passed on to the offsprings by recombination and mutation like the object parameter. Because the object and strategy parameter build one individual the selection of the best parameters is done indirectly

by the selection of the best object parameters. Individuals whose strategy parameter are better adapted to the fitness landscape have a higher probability to create an offspring with good fitness than others. As a consequence their genome has a higher chance to survive.
The mutation of the object variables and the strategy parameters are defined as follows:

$$\begin{aligned} \sigma_i(t) &= \sigma_i(t-1)e^{(\tau' z)}e^{(\tau z_i)} \\ \overrightarrow{z} &= N(\overrightarrow{0}, \overrightarrow{\sigma}(t)) \\ \overrightarrow{x}(t) &= \overrightarrow{x}(t-1) + \overrightarrow{z} \end{aligned}$$

where z, z_i are (0,1) normal distributed random numbers.
The strategy parameters σ_i are the elements of the vector $\overrightarrow{\sigma}$. Their log-normal distributed mutation consists of two parts: an overall part parametrised by τ' and an individual part with variance τ. They are both fixed during the optimisation. The strategy parameters σ_i determine the variance of the normal distributed random vector $\overrightarrow{z}$, which is added to the object parameters $\overrightarrow{x}$ of generation $t-1$ to create the offspring population. The length of the vector $\overrightarrow{\sigma}$ depends on the used approach. The GSA uses a single element whereas in the ISA their number is equal to the number of the object parameters.

1.2.2.2 Covariance matrix adaptation

Hansen and Ostermeier[7] introduced a completely derandomised self-adaptation to improve the performance of the Evolution Strategy. Their strategy consists of two parts: the adaptation of the whole covariance matrix of the mutation vector and the observation of the evolution path. The multivariate normal distribution $N(0, C)$ which is used for the mutation of the object parameters, enables to realise correlated mutations and is invariant against rotations of the search space. As illustrated in figure 1.3(c) the ellipsoids of equal density have not to be axis parallel any more.
The adaptation of the covariance matrix is based on the correlation of the new parent population and the path of evolution. This path is a sum of consecutive single steps between the individuals. Using such a cumulation of steps stabilises the adaptation and furthermore yields to more promising predictions.
The evolution path is also used to adapt the global step size. A long path indicates that the last single steps point to a similar direction. A higher step size would allow to cover the same distance in fewer steps. If the evolution path is short that the last steps have annihilated each other. Then shorter steps should be preferred. Hence the global step size is changed according to the ratio between the expected and the realised step size.
The CMA strategy furthermore reduces the stochastic influence because no random numbers are used to update the strategy parameters.

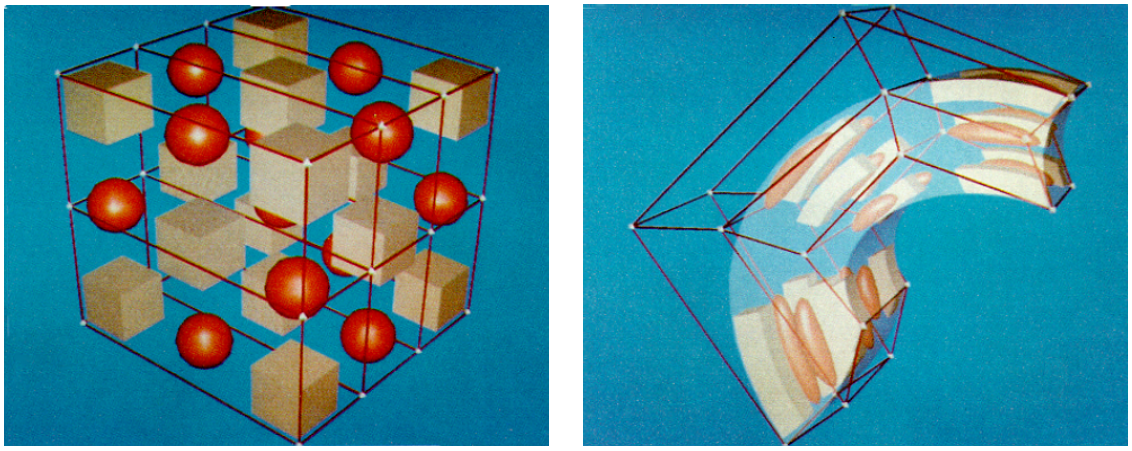

(a) Objects embedded into a plastic (b) The objects after deforming the plastic.

Figure 1.4: The control volume and embedded objects. The control lattice is also shown. [20]

1.3 Free-Form Deformation

Free-Form Deformation (FFD) has been introduced by Sederberg and Parry[20] in the field of solid and surface modelling. Instead of representing the geometry by itself only the deformation of an initial geometry is defined. This initial design can be given in any solid modelling scheme. A physical analogy for FFD is to imagine a clear, flexible plastic. The objects which shall be deformed are embedded into the plastic. Deforming the plastic forces the object to change similarly. Figure 1.4 illustrates this behaviour.
Mathematically the plastic corresponds to a control volume defined by a set of control points. The embedding of objects is implemented by transferring every object point X into a local coordinate system (1.5) with the directions U, V and W on that parallelepiped volume.

$$X = X_0 + uU + vV + wW$$

In the case of a parallelepiped volume the local coordinates (u, v, w) can be computed with linear algebra:

$$u = \frac{V \times W\,(X - X_0)}{V \times WU}, \quad v = \frac{U \times W\,(X - X_0)}{U \times WV}, \quad w = \frac{U \times V\,(X - X_0)}{U \times VW},$$

The deformation of the plastic is specified by changing the initial control points coordinates P_{ijk} and thus are used as object variables in an optimisation. The deformed position of an object point is then computed by evaluating a vector valued trivariate Bernstein polynomial:

$$X_{ffd}(u, v, w) = \sum_{i=1}^{l} \sum_{j=1}^{m} \sum_{k=1}^{n} B_i(u) B_j(v) B_k(w) P_{ijk} \tag{1.1}$$

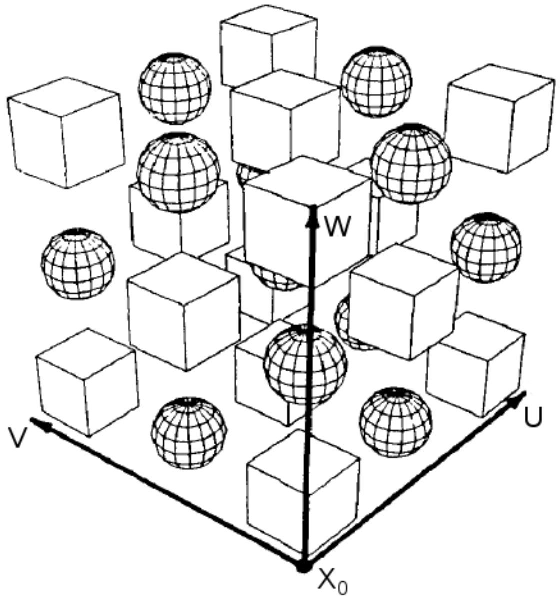


Figure 1.5: The local coordinate system [20].

where $B_i(t)$ are the Bernstein polynomials of degree l, m and n. The degree determines the number of control points in the corresponding direction. The coefficients P_{ijk} of the Bernstein polynomial are the Cartesian control point coordinates.
Free-Form Deformation has not to be applied on a whole object. It is also possible to use a control volume which only covers a part of the object. Then only the object points which lie within the volume have to be embedded. The continuity at the boundaries of the control volume can be reached by keeping the positions of the boundary control points. To achieve derivative continuity of degree C^k, the k adjacent planes of control points have to be kept fixed.
The Free-Form Deformation inherits the properties from the used Bernstein polynomial basis functions. All deformations which can be produced with FFD are smooth and the deformed points lie within the convex hull of the influenced control points.
Coquillart[3] extends the FFD by allowing arbitrary volumes and hence reduces limitations of the possible deformations. An exemplary lattice is given in 1.6. In order to have a mathematical correct defined control volume every line of the lattice has to contain the same number of control points. But several control points can be merged together and treated as a single point. This merging simply keeps the position of the different control points equal. To achieve tangent continuity at the merged points, additional their tangents have to be marked as aligned.
Furthermore Coquillart uses B-splines in the deformation function instead of Bezier curves. They consist of several Bezier segments which are linked together. The segments are specified by a knot vector. Due to the piecewise definition, the influence of one control point is local. The control point P_{ijk} of a B-spline with degree p only influences points inside the interval $[u_i, \ldots, u_{i+p+1})$ in U direction. Figure 1.7 depicts

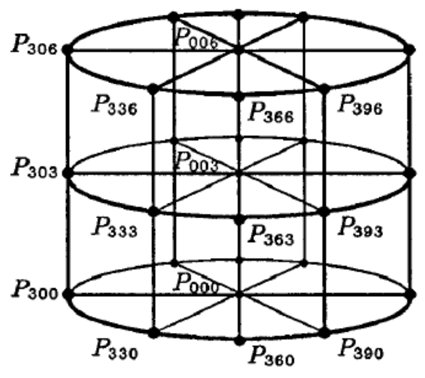
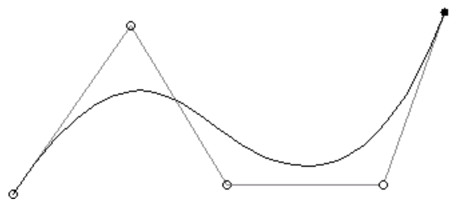


Figure 1.6: An arbitrary control volume [3]



(a) B-spline of degree three

[0, 0, 0, 0, 1, 2, 2, 2, 2]

(b) knot vector

Figure 1.7: B-spline and corresponding knot vector

a B-spline and its corresponding knot vector.
Due to the non-parallelepipedical lattice, arbitrary formed designs can be generated. The increased degree of freedom results in a higher computational effort for calculating the local coordinates. This computation, also called freezing, is decomposed in two steps and performed separately for every spline segment. To speed up the computation, first the points, which are supposed to lie within the segment, are determined. Therefore the convex hull property of the splines is used. Then the (u, v, w)-coordinates of these points inside the segment are calculated by a Newton approximation. Figure 1.8 illustrates this steps in a one dimensional example.
Another important property in regard to the desired adaptive representation is that the degree of freedom can be increased by inserting new control points without changing the shape of the embedded object.
For the Bernstein polynomials the degree elevation algorithm can be used to enlarge the variability of the deformations. In this process the degree of one of the three basis functions is increased. To accommodate the enhanced degree of freedom, a new control point is introduced into the lattice. Thereby new positions of all inner control points are computed. Figure 1.9 shows an example. The control points are distributed over the whole curve. Due to the trivariate definition, a new plane of control point is inserted into the corresponding direction.
For B-splines the knot insertion algorithm can be used. Here a new point is inserted in the knot vector of one direction to divide a spline segment. As a consequence a

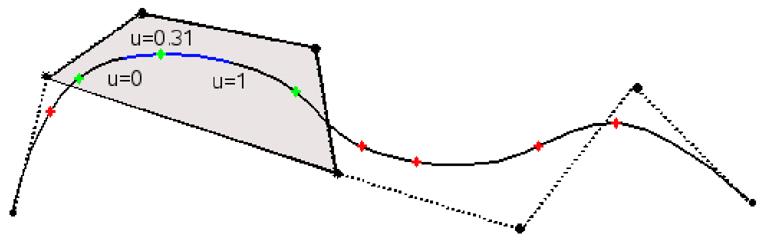


Figure 1.8: In the first step the points inside the convex hull of the control points which have influence on the current spline segment (blue segment) are determined. Afterwards the spline parameter u of these points (green) is computed by a Newton approximation. Only for the point within the current segment this is possible.

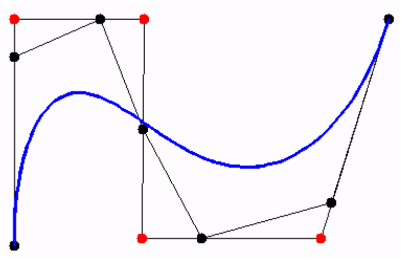
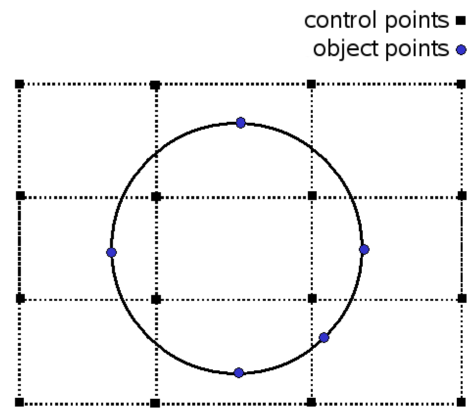
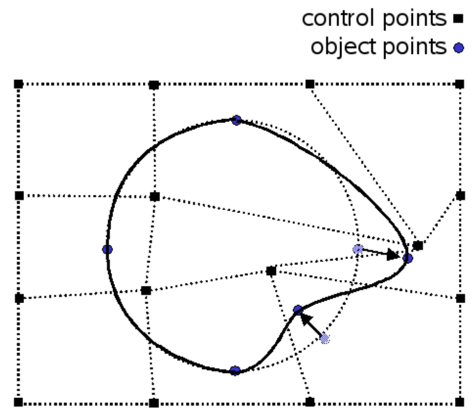


Figure 1.9: A Bezier curve before and after applying the degree elevation algorithm. The red points depict the inner points before the refinement. After the degree elevation their positions have been changed. The figure also shows that the whole curve is refined.

new plane of control points for this direction is created. In contrast to Bezier curves the positions of only a few control points have to change. When inserting a new point $u \in [u_i, u_{i+1}]$ into the knot vector only the positions of the control points $P_{(i-n+1)jk}, \ldots, P_{(i-1)jk} \forall j, k$ have to be computed. All other points are kept.
More about this refinement property is written in 5. In the following FFD will always point to the extended Free-Form Deformation from Coquillart. As basis functions B-splines of degree 3 are used. They promise a good trade off between locality and flexibility.



(a) Control lattice, object point and target deformation before the deformation



(b) Control lattice and object points after the deformation

Figure 1.10: Specification and result of an direct manipulation of Free-Form Deformation.

1.4 Direct manipulation of Free-Form Deformation

The direct manipulation technique of FFD (DMFFD) has been introduced by Hsu, Hughes and Kaufman[9] in the context of object modelling to get a more intuitive interface and to increase the control of the deformations. The approach is based on the Free-Form Deformation method, but the deformations of a shape are specified directly by a set of object points and their target positions instead of the control point displacements.
First the design to modify has to be embedded in a Free-Form Deformation control volume by computing the local coordinates. To specify the modifications of a shape, the user selects some object points and determines their desired target position. As object points arbitrary points inside of the control volume can be used. Their positions have to be given in the U, V, W coordinate system. Typically some points of the embedded design are chosen. Then the information is used to compute new coordinates for the control points to hit these deformations. An example is given in figure 1.10.
To match the desired object point displacements, the following system of equations has to be solved:

$$\Delta X = B \Delta P$$

where ΔX contains the displacement of the object points specified by the user in the Cartesian coordinate system, B is the matrix of the B-spline tensor product at the object points and ΔP contains the shift of the control points. When the object point displacement ΔX is given, Hsu uses the pseudoinverse B^+ of B to compute the new positions of the control points.

$$\Delta P = B^+ \Delta X \tag{1.2}$$

The solvability of the problem depends on the number of object points and the lattice of control points. When the number of object points is less than the number of control points which influence them, the problem is under-determined and a set of solutions exists. If their numbers are equal, only one solution exists. If more object points than control points, which can influence them, are selected, no exact solution exists. To get an exact solution, a finer lattice of control points is needed.
The use of the pseudoinverse allows to calculates the solution with the minimal error independent of the solvability. If no exact solution exists, the distance between the attained and desired object points positions is minimised. If several exact solutions exist, the approach additionally minimises the changes of the control point positions ΔP, because the pseudoinverse gives the least-squares solution.
To represent a design in an optimisation with the direct manipulation, some points, which lie on the design, are selected as object points. The target positions of these object points specify the deformation and allegorise the object variables which are subject of the recombination and mutation in every optimisation step.

Chapter 2

State of the art

Many different techniques have been developed for design representation. A good overview is given by Samareh in [17]. A method easy to implement is the discrete representation, where the design is represented by points of its boundary (figure 2.1). However, in this representation it is difficult to maintain the smoothness of the shape. Therefore additional constraints are required. Furthermore for complex designs a huge amount of object parameters is necessary to represent all boundaries exactly. That leads to a costly and difficult optimisation problem.
One of the most popular approaches are the polynomial and spline based representations. The boundaries of the shape are represented by a polynomial or spline curve specified by a set of control points. Different polynomial and spline definitions are possible as e.g. Bezier curves, B-splines or nonuniform rational B-spline (NURBS). Dependent on the used basis function an arbitrary smoothness can be achieved implicitly and thus the smoothness problem of the discrete approach is solved. Figure 2.2 shows an example of an airfoil. Besides the number of parameters is reduced drastically in comparison to the discrete approach. However, for complex shapes containing many edges and ridges still a large set of control points is required.
The Free-Form Deformation is used by Samareh [16] to represent a design in an optimisation. He wants to optimise a complete aeroplane configuration with low and high-fidelity analysis tools. The models for the analysis are based on computational grids which represent some or all components of an aeroplane. Two different models are used: computational fluid dynamics (CFD) grids used for the aerodynamic analysis and the computational structural mechanics (CSM) to represent the all

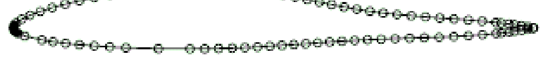


Figure 2.1: Discrete representation of an airfoil [17].

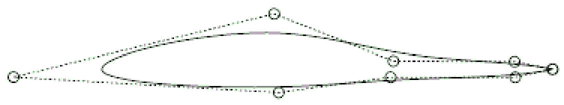


Figure 2.2: Spline representation of an airfoil [17].

aeroplane components. Building these grids from a computer-aided design (CAD) model is very time consuming and needs a huge amount of problem specific knowledge. Furthermore this generation needs human interactions. Due to this manual grid generation, an automatic optimisation is not possible. In every optimisation step these grids have to be build to evaluate the quality of the design.
To avoid such a re-meshing procedure he uses the FFD to represent a wing planform. He combines the FFD with NURBS as basis function with further soft object animation methods to represent deformations of an initial design. For this initial design the computational grids have to be created. Since the FFD and the soft animation methods represent the deformation rather than the geometry itself, the deformation can be applied to the computational grids too and thus avoid the costly re-meshing in every optimisation step. Furthermore these methods are independent of the grid topology and thus both grids (CFD and CSM) can be treated equally.
The described representation is especially adapted on the wing optimisation problem. A huge amount of problem specific knowledge was used to build the control volume of the Free-Form Deformation to reduce the number of object parameters.
In [18] Samareh uses a bivariate FFD for aerodynamic shape optimisation based on a gradient approach. He shrinks the trivariate volume along one coordinate direction and thus reduces the number of design variables by an order of magnitude, while retaining the required flexibility.
Ernest C. Perry and Steven E. Benzley [14] introduced arbitrary shape deformation (ASD) for the optimisation of three dimensional fluid flow systems. They use the FFD as representation to get a general methodology for the parameterisation of three-dimensional shapes for fluid flow systems.
Moreover similar to Samareh they want to avoid the re-creation of the required CFD mesh for every modified design during the optimisation. Therefore they use the control lattice to deform the computational grid simultaneously to the shape.
Their test problem is an optimisation of a tee fitting with a gradient-based Sequential Quadratic Programming method. They achieve a solution which provides entirely new, non intuitive features and reduces the static pressure significantly.
A multilevel Free-Form Deformation representation is used in [1][4] by Ales Janka and Jean-Antoine Désidéri to speed up the convergence. They apply the degree elevation algorithm to increase the degree of freedom of the used Bezier FFD control volume. The degree is enlarged three times after a certain number of optimisation steps. To compare their multilevel approach with fixed representations of different degree they optimise the planform of an aeroplane wing to reduce the drag. For the optimisation they use a Nelder and Mead simplex method and a Genetic Algorithm.

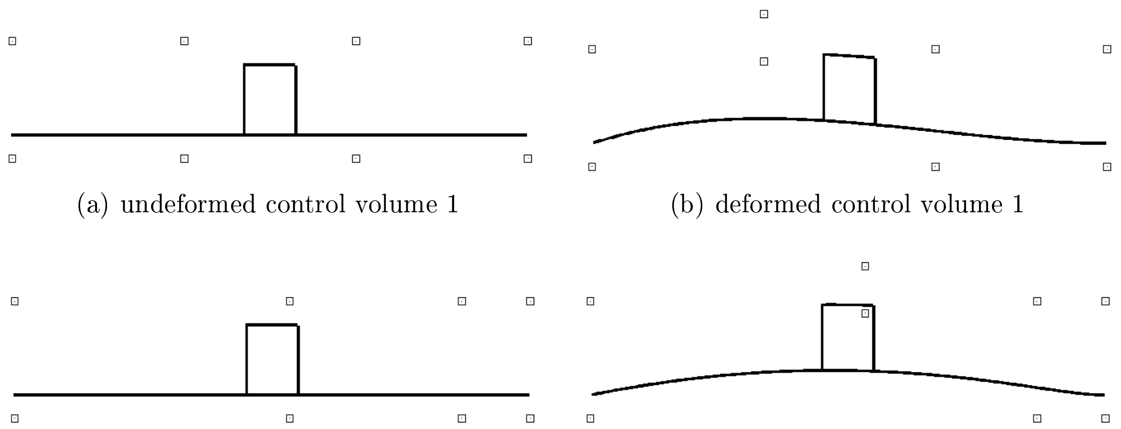
In the case of the simplex method the use of their multilevel representation could speed up the optimisation in relation to the fixed representation whose degree is equal to the one used finally in the multilevel approach. This speed up can be explained by the faster convergence at the beginning of the multilevel process. The quality of the solution achieved by the multilevel approach is similar to the one achieved with the fixed representation using the high degree. In the Genetic Algorithm the adaptive representation also speeds up the optimisation, but the representation with a fixed and high degree leads to a better solution. They suppose that this can be ascribed to their elevation strategy used for the Genetic Algorithm. The elevation is only applied to the best individual whereas all others are re-initialised.
An adaptation of a control volume cannot be achieved with this multilevel approach. The use of Bezier curves as basis function of the Free-Form Deformation limits the possible refinements. Due to the definition of Bezier curves where every control point effects the whole Bezier curve, only one refinement possibility exists. Furthermore the degree elevation always refines the whole curve. A refinement at a certain area of the design is not possible.
Additionally the refinement of a Bezier curve changes all inner control points. In an Evolution Strategy this disturbs the self adaptation. The information about the search space accumulated in the strategy parameters becomes useless because the influence of the control points on the design is altered. Thus the strategy parameters have to be adapted again.
For designs where local deformations in a certain region are crucial for the quality many refinements have to be applied until these local modifications are possible. This leads to an unnecessary amount of optimisation parameters.
At the Honda Research Institute [11] an Evolution strategy with an adaptive FFD is used to optimise a 2D airfoil. Their aim of the adaptive representation is to solve the trade-off between compactness and completeness and to keep the amount of required problem specific knowledge to build a proper representation as small as possible.
For the adaptation of the B-spline FFD, the idea of sub-populations is applied. The degree of freedom is increased by an explicit insertion of new control points whose positions in the lattice are randomly chosen. The coordinates of the new control points are calculated by a linear interpolation of the neighbouring control points. After evolving several different populations for ten generations the population with the best performance is selected to proceed. Its representation seems to provide the most useful variability.
They also want to realise a search in sub-spaces. A coarse representation at the beginning of the optimisation shall enable to find promising areas. Then the representation is refined to allow new variability. Because the strategy parameter of the old object variables are typically decreased, the search is limited to the actual region.
Their introduced approach has some drawbacks. Due to the explicit inserting of new control points, the previously modified shape has to be frozen again. For complex designs and especially for the CFD mesh this computation is a costly operation.

Refreezing changes the influence of the control points and hence the self-adaptation of the strategy parameter is disturbed. Thus the refinement is similar to a restart with a finer control volume.
The direct manipulation extension is used as representation in an optimisation for the first time at the Honda Research Institute [12]. They want to analyse the effect of this representation on the optimisation. They assume that the use of the direct manipulation reduces the influence of the initial control volume and additionally increases the impact of the design variables on the design. Therefore a two dimensional turbine blade is optimised with a CMA-ES. For the evaluation of the fitness function a CFD-simulation is used. In order to avoid a re-meshing procedure the used CFD mesh is deformed simultaneously with the design.
To validate their assumptions they compare a general FFD representation with the direct manipulation approach. For the direct manipulation tests they use two different numbers of object points and two control volumes with a different number of control points. The coarser control volume consists of a 4×4 grid whereas the finer uses a 6×6 control points. For the general FFD approach the coarse grid is used.
They could show that the direct manipulation speeds up the optimisation while a similar fitness value is achieved. Furthermore the use of the different control volumes in the direct manipulation does not influence the convergence behaviour much. The fitness and the convergence behaviour mainly depends on the used object points.
They mention one problem of their approach. Their direct manipulation approach can destroy the structural composition of the CFD mesh if a fine control lattice is used. But fine grids are needed when local modifications are required to improve the quality. Since the use of the pseudoinverse only moves control points which have influence on the object points, in fine control lattices very large local deformations are applied. These deformations change the order of the CFD mesh points and lead to invalid meshes.

Chapter 3

Free-Form Deformation as representation

The kind of representation is a very important part in the optimisation. The representation determines the search space. Only designs which can be coded by the representation can be found by the optimisation. The representation has to be flexible to provide a large variation of designs. That requires a large number of parameters, especially for the representation of complex designs containing many edges and ridges.
On the other hand the search space should be as small as possible to reduce the number of possible shapes which enables a fast convergence of the optimisation.
Using Free-Form Deformation as representation allows a good trade off between completeness and compactness. Especially for complex shapes, which require a huge amount of parameters to be represented, the number of parameters can be reduced while still a high variability is provided. In the FFD the moveable control points of the lattice are used as object parameters. Since they are decoupled from the embedded design, their number can be chosen independent of the design. Also very complex designs can be represented with a few parameters. The number of object points can even be chosen with respect to the optimisation state. If the initial design is a roughly approximation of the optimum, a coarse grid can be used to improve the performance of the design very fast by applying global changes. If the embedded shape is already a good approximation, a finer grid can be used to enable more local changes in order to improve the design further. The representation of the embedded design has not to be changed for the different control volumes.
In context of evolutionary algorithm the representation specifies the map from genotype to phenotype. One important property for this map is that changes in the genotype result in similar changes in the phenotype. Small modifications of an object parameter should lead to small changes of the shape whereas large modifications should result in large shape changes. This strong causality constraint is essential to ensure that the self-adaptation of the mutation strategy parameters works.


(a) undeformed control volume 1 (b) deformed control volume 1

(c) undeformed control volume 2 (d) deformed control volume 2

Figure 3.1: The figures show the influence of the control point positions on the possible deformations. The two control volumes differ only in the positions of the two inner columns of control points. The design deformed with control volume 1 cannot be produced by control volume 2. For control volume 2 the rectangle will always be on the top of the hill and never on the right slope.

Free-Form Deformation fulfils this strong causality property. Small modifications of a control point lead to small deformations of the shape whereas large modifications result in large shape changes.
Another very important advantage of the FFD occurs when a CFD or FEM simulation is used to evaluate the performance of a design. The generation of the required CFD grid for every evaluation can be avoided. Instead of this costly computation, the deformation of the design is applied to the grid too. To maintained the structural composition of the grid after the deformation, the deformation has to be restricted. More about this restriction is explained in section 4.
A drawback of the Free-Form Deformation is the influence of the control point positions on the deformation of the embedded object. This influence is illustrated in figure 3.1. Dependent on the initial lattice, some deformations cannot be achieved. Therefore it is necessary to place the control points in the sensitive areas of the design. This placing requires problem specific knowledge and comprehension about the behaviour of splines. If the sensitive areas are unknown, a proper control lattice cannot be build.
Furthermore the influence of the control points decreases with the distance. The largest deformations are applied close to the control points. To maximise their influence on the shape they have to be placed close to the embedded design. But due to the local definition of the B-splines the influence of the neighbouring control points is already smaller even if they are placed near the design (figure 3.2). Some of the control points even have no influence.

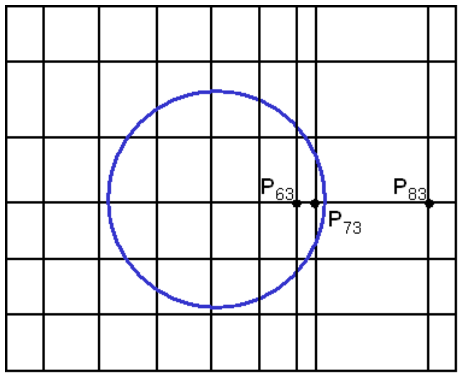


Figure 3.2: Point P_{73} has the largest influence on the embedded circle. The impact of point P_{83} is smaller due to the larger distance. The influence of point P_{63} is smaller because of the local definition of the B-spline function used in the FFD.

One option to overcome the problem of finding the sensitive regions is to adapt the control volume automatically by refining the control lattice. Therefore one has to determine promising positions for the new control points and to decide when the control volume should be refined. Such an adaptive FFD representation is developed in chapter 5. The problem of maximising the influence of the control points can be reduced by the direct manipulation of Free-Form Deformation. Here the deformation is not specified by the control points but directly by a desired deformation of the object. The direct manipulation approach is represented in chapter 6.

3.1 Implementation

The Evolution Strategy with Free-Form Deformation as representation is implemented in C/C++ for 2D and 3D problems. The 2D program is still based on a 3D FFD volume but projects it onto 2D.
Because the Evolution Strategy can easily be parallelised a master/slave architecture, runnable on a cluster is used. For the communication between master and slave the cppvm library [6], an C++ interface to the parallel virtual machine software package, is used.
For the operations and data types required for the Evolution Strategy the shark library [21] is used. The implementation of the Free-Form Deformation operations like freezing, deforming and insertion are provided by HRI.

3.1.1 Encoding

In the FFD the designs are specified by the control points. The coordinates of the control points, which are allowed to move, are used as object parameter and have

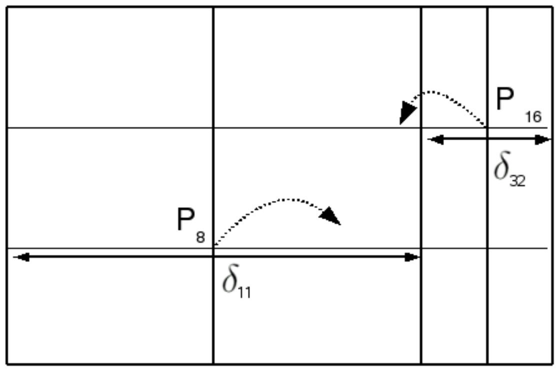
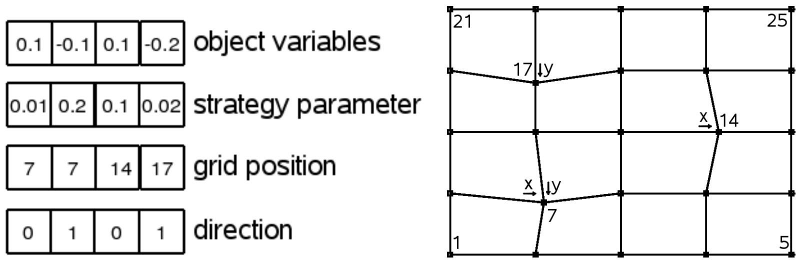


Figure 3.3: The movement of the control point is scaled dependent on the distance δ_{ij} of the adjacent points in the corresponding direction. A large movement of P_{16} would change the order of the control points, whereas small movements would require many steps to achieve a large deformation.

to be encoded into the chromosome of the individuals. This encoding is done in the following way:
Each individual consists of four chromosomes. The first holds the object variables. They contain the displacement of the control points relative to the initial positions. Additionally they are scaled with the distance δ_{ijk} of the adjacent points of the initial control lattice. This scaling aids to keep the order of the control points. Keeping the order of the control points is similar to the constraints which is introduced in chapter 4 to keep the structural composition of the computational grid. Figure 3.3 illustrates the benefit of this scaling. Without the scaling a large displacement would change the order of the control point P_{16} with a high probability, whereas for point P_8 the same displacement will still maintain the order. A smaller shift would also keep the order at point P_{16} but then the deformations are small. To achieve large deformations in the region of point P_8 several step are required. The scaling allows that the movement of points, which are far apart, is larger than the movement for points, which are close together. Even if the same strategy parameter is used. Thus in areas with a few control points large deformations can be achieved in less steps while the order is also kept in the dense areas of the control lattice.
The displacement is divided into the three different directions (x, y, z). To determine which control point and which direction is specified at a certain position in the object chromosome a unique number specifying the position of the control point in the lattice and the direction (x, y, z) of the movement is saved in two separated chromosomes. Figure 3.4 depicts the encoding of an exemplary control lattice.
The strategy parameters are encoded in a further chromosome.



(a) The genotype

(b) The corresponding control lattice.

Figure 3.4: The encoding of a control lattice.

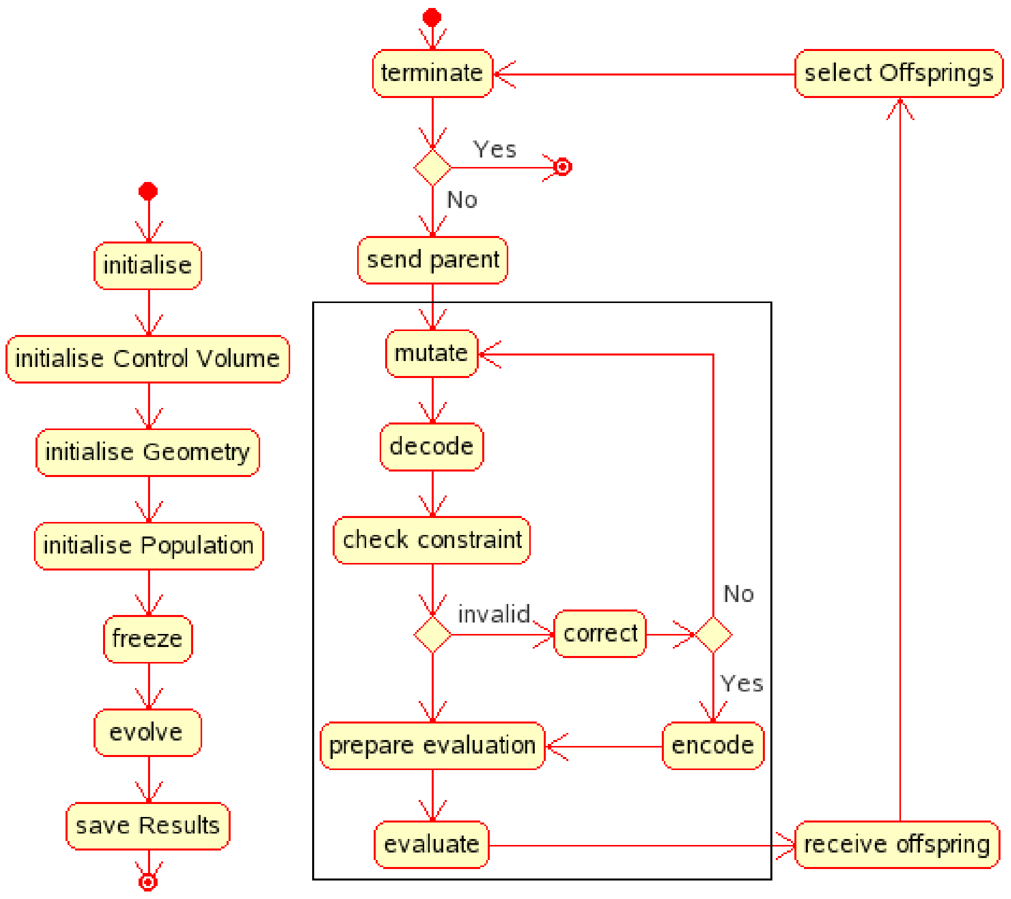
parameter name	description
λ	number of offsprings
μ	number of parents
σ_0	initial mutation step size
Strategy	strategy for self adaptation 0 - CMA 1 - global step size adaptation 2 - individual step size adaptation
Generations	max. number of generations

Table 3.1: Parameter for Evolution Strategy

3.1.2 Evolution cycle

The program starts with the initialisation of different parameters which determine the Evolution Strategy. They are listed in table 3.1. They have to be determined by the user in a parameter file.
This parameter file also specifies the input data containing the initial shape and the initial control lattice. The shape has to be given as a set of point. The control volume is defined by a 3D control lattice, the degree p and the three knot vectors of the B-splines.
After loading this data, the initial shape is frozen. Then the first population is build by encoding the moveable control points of the initial lattice into the chromosomes. Afterwards the evolution cycle depicted in figure 3.5 begins. The generation and evaluation of the offsprings is parallelised to speed up the optimisation. The master selects a free host, starts the slave process there and sends a random parent for the offspring. When all hosts are busy or for all offsprings a slave was already started, the master switches into the receiving mode and waits for the slaves.
When the slave has received the parent, the offspring is generated by copying the parent. Then the offspring is mutated according to the selected strategy. To evalu-

ate the fitness of the design, first the control lattice is decoded from the genotype. The next steps of the program depends on the approach which is used to solve the constraint optimisation. Two different approaches, a penalty function (4.1.1) or a correction method (4.1.2) can be used.
In the case of the correction method in the next step the control lattice is checked if it fulfils the constraint which keeps the structural composition of the computational grid. If this test fails, the lattice is corrected. Afterwards the corrected control points are coded back into the chromosome. In the case that the grid could not be corrected the slave creates a new offspring.
The penalty approach skips these steps.
The decoded control lattice is then used to deform the frozen shape according equation (1.1). Afterwards the fitness is evaluated and assigned to the offspring. The used fitness function also depends on the approach used to solve the constraint optimisation. The exact formulations are given in the next section. The penalty terms used in the penalty methods are explained in section 4.1.1.
After the fitness is evaluated the slave sends the new offspring back to the master. When it has received all λ offsprings, the selection operator is applied. The μ best offsprings are selected to build the parent population for the next generation.
When the termination condition is fulfilled, the evolution cycle is interrupted and the results are saved. This termination condition can be specified by a fixed number of generations or a fitness threshold.



(a) Initialisation

(b) Evolve activity in detail.

Figure 3.5: Activity diagram of the program using the correction method. The activities inside the box are performed by the slave.

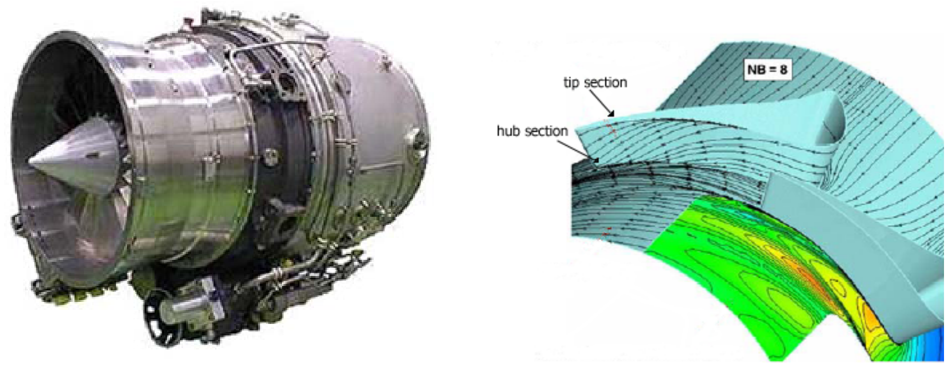


Figure 3.6: Gas turbine and a turbine stator blades with hub and tip section [8]

3.2 Test problems

Two problems should be solved to test and compare the Free-Form Deformation representation with the introduced adaptive Free-Form Deformation and direct manipulation. To keep the computational effort low, two dimensional target matching problems are chosen. An initial shape defined by a set of points, has to be deformed to match a given target point set. As target a dolphin and a turbine stator blade are used.

The dolphin test is an artificial problem. As initial shape for the dolphin a circle is used. Here large deformations are necessary to match the target. The fins contain strong local curvatures which requires that the control points move in different directions. It is difficult to keep the additional constraint which in a practical problem would maintain the structural composition of the computational grid. In the dolphin test problem this constraint is used to prevent that the deformed shape contains loops.

The turbine blade problem was selected to have also a practical problem. The blade is part of a gas turbine used in jets (3.6) and was already optimised in [8] with a CMA-ES. There the D turbine blade consists of a hub and a tip section which are both represented with a closed non-uniform rational B-splines (NURBS). To obtain the 3D stator blade they interpolate both curves linear. They evaluate the fitness with an in-house Navier-Stokes flow solver which is adjusted to the given problem and uses a structured grid. Thus it was possible to mesh the design automatically.

In this thesis the initial and optimised designs are used to define a target matching problem. Both initial designs have to be deformed to match the optimised shape. The control volume which deforms the initial blade to the target shall be found. To get a 3D control volume both lattices are linear interpolated. This control lattice then is used by another group to deform also an unstructured grid which was generated for the initial design. The aim is to show that the FFD is able to deform the computational grid and a re-meshing can be omitted. The results of the CFD-simulation of the deformed design and mesh should be analysed. The initial

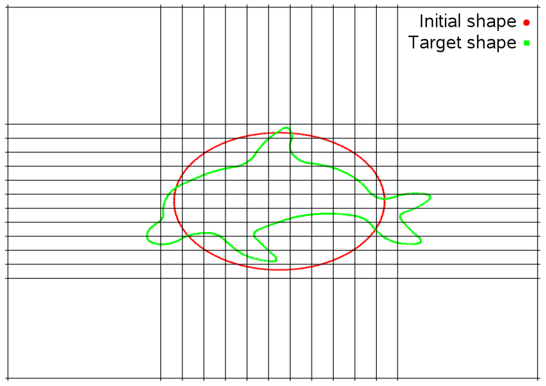
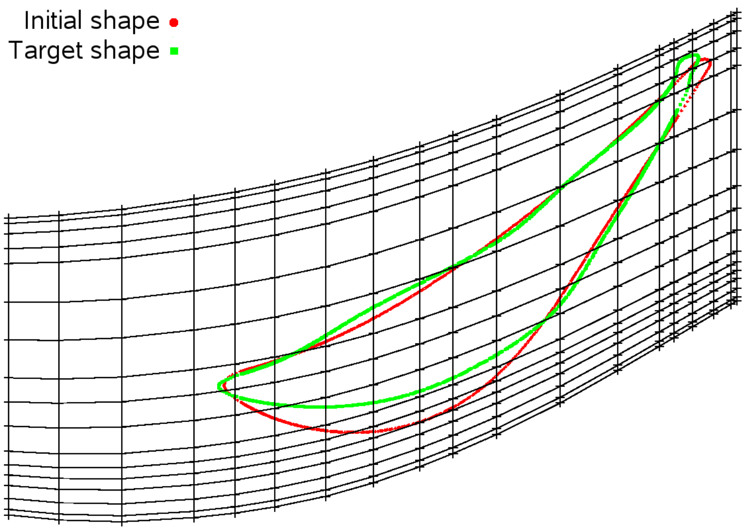


Figure 3.7: The initial shape, target shape and the control lattice for the dolphin test problem.

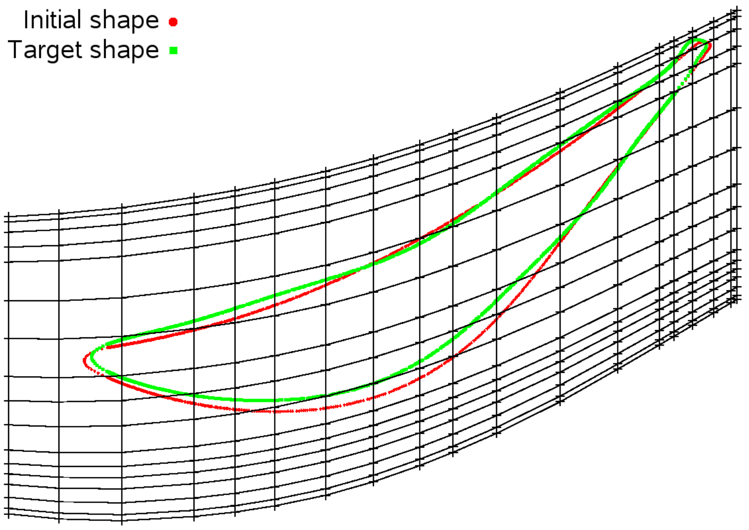
and target designs of the hub and tip section and their control volumes are shown in figure 3.8. In this diploma thesis only the results of the target matching problem using the hub section will be shown because the tip section is very similar.
In all target matching problems the fitness function to achieve the matching calculates the average distance between the two point sets. For every point of the deformed shape the minimal distance to the target is computed and vice versa. Mathematically the fitness for the two point sets X and Y is defined by:

$$f(X,Y) = \frac{1}{|X|+|Y|}\sum_{x\in X} min_{y\in Y}(\|x-y\|)^2 + \sum_{y\in Y} min_{x\in X}(\|y-x\|)^2 \qquad (3.1)$$

This function has to be minimised.
The control lattice used for the representation is shown in figure 3.7 and 3.8. For the dolphin case the lattice consists of 14×14 control points. For the dolphin problem also other volumes have been tested. But using a finer grid does not achieve better results. Whereas using a coarser lattice is not able to match the target as good. The rows and columns of control points are placed near the circle to increase their influence. The control lattice for the turbine blade problem is adapted by hand. The density of the lattice is highest in the region of the trailing edge where the most local deformations are required to match the target.
In all optimisation test runs the inner control points are allowed to move in x and y direction. The points of the boundary are always kept fix. In the case of CFD simulations this is required to keep the continuity at the boundaries. This result in 288 object variables for the dolphin test and 510 for each turbine blade. This number is much higher than in the NURBS representation used in [8], but the FFD representation is developed for complex designs which are not representable with splines easily. Furthermore the spline representation is not practicable if the CFD mesh generation have to be done manually like in the case of unstructured grids for designs with many edges.



(a) hub section



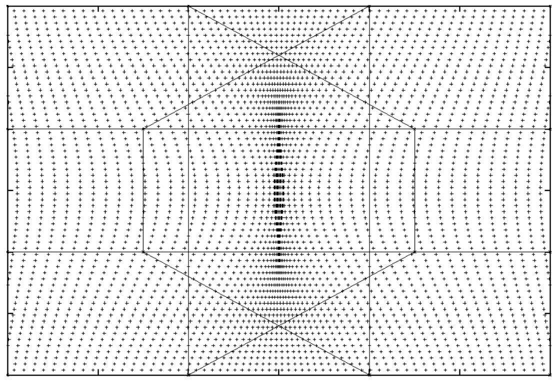
(b) tip section

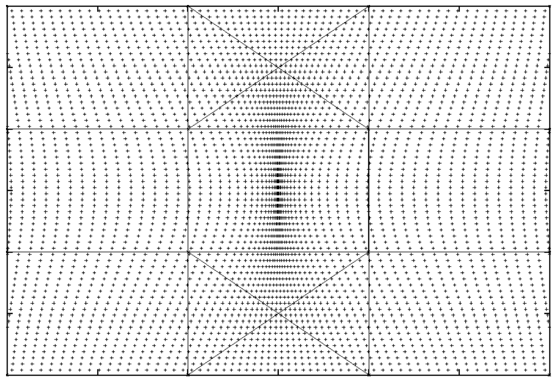
Figure 3.8: The initial shape, target shape and the control lattice for the turbine blade problem.

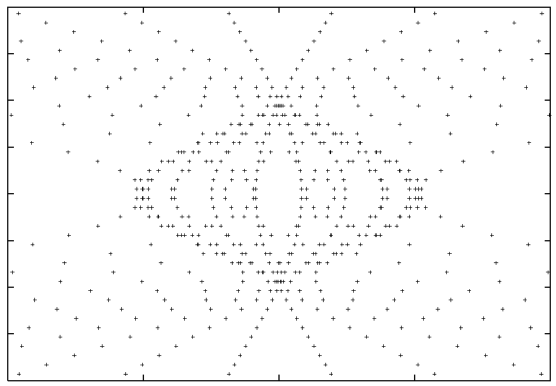
Chapter 4

Mesh deformation

In the context of shape optimisation the quality is usually defined on flow properties. To evaluate these properties, a computational fluid dynamic simulation is necessary. This simulation calculates the flow by a discretisation of the problem with an computational grid. To achieve realistic simulations, the mesh has to be adjusted to the shape. Since the shape is changed in every optimisation step, the computational grid has to be adjusted again.
Adapting the mesh can be realised in two ways: creating a new mesh or repairing the existing one. For complex designs these methods often requires human interactions which prevent an automatic optimisation process. Automatic meshing methods for unstructured grids are only available if the method is adapted to the given problem or the design is very simple and does not contain many edges and ridges. Furthermore the meshing requires costly computation time. In such cases a re-meshing procedure is not practicable.
Representing the shape with Free-Form Deformation allows to deform mesh and shape simultaneously. The quality of the mesh is kept and a costly re-meshing can be avoided. This can be realised as follows:
Before the optimisation starts a computational grid for the initial shape needs to be created. Together with the shape this grid is then embedded into the FFD control volume. This costly computation of the local coordinates is needed only once. If only a part of the CFD mesh is within the volume, continuity to the outlying mesh is guaranteed by fixing the control points of the boundary.
Before the fitness of a shape is evaluated the encoded design and the initial computational grid are deformed with the same control lattice. The mesh will follow all deformations applied to the shape. This technique has already been applied successfully in several optimisations [4][14][11].
Using Free-Form Deformation for repairing the mesh requires limitations on the deformations. The deformations can destroy the structural composition of the mesh. Large deformations can change the order of the grid points. That creates meshes with overlapping cells and cells with negative volume (figure 4.1). Such destroyed


(a) Invalid mesh


(b) Valid mesh


(c) Invalid mesh

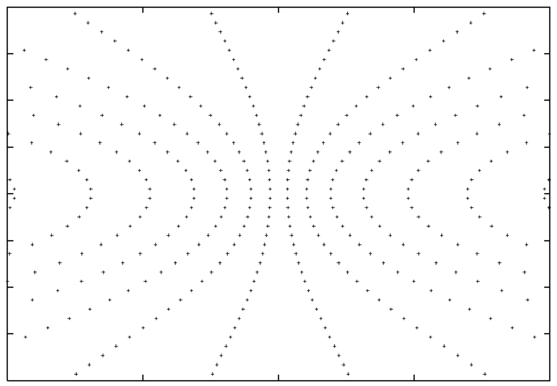

(d) Valid mesh

Figure 4.1: The pictures show the deformation of equidistant grid points. On the left side the order of the grid points has changed. On the right the control points are corrected with the algorithm from 4.1.2. In the lower figures the critical area is zoomed.

meshes cannot be used in a CFD-simulation. In such cases a new mesh for the deformed shape has to be generated.
Restrictions are also necessary to avoid self-intersections of the shape. Large control point displacements can lead to loops of the shape (figure 4.2). Such designs are typically incorrect because they violate manufacturing conditions. For the target matching problems loops also have to be avoided. Both mesh and shape can be kept feasible by the same restrictions which prevent self-intersections.
Restrictions on the deformations have to be made by additional constraints to the displacements of the control point. These restrictions have to be chosen carefully. On the one hand invalid shapes and meshes have to be avoided while on the other hand the deformations should still be as large as possible to achieve a high flexibility.
Gain and Dodgson introduced in [5] an injectivity test to prevent self-intersections for FFD. They developed an exact but computationally costly and an efficient but only approximate injectivity test.

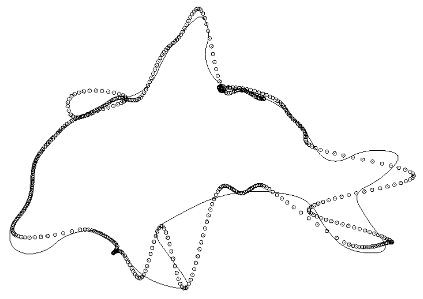


Figure 4.2: Intermediate result of a target matching problem with an unrestricted Free-From Deformation. The shape contains several self intersection.

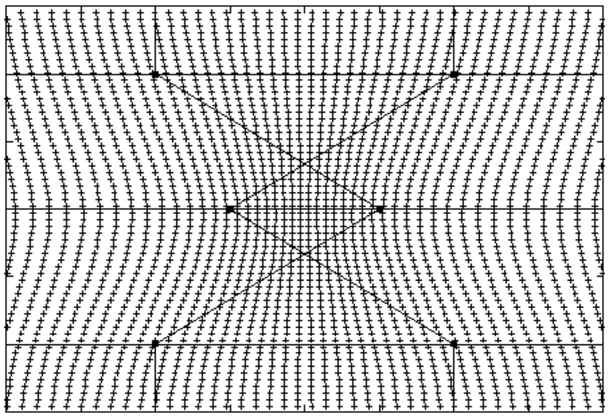


Figure 4.3: A Free-Form Deformation failing the approximative self-intersection test of [5]. However, the deformation is free from self-intersections. The convex hull constraint is satisfied by the displayed control lattice.

Both tests are based on the existence of continuous first partial derivatives and the Jacobian J of the deformation function (1.1). The Free-Form Deformation is a homomorphism if the determinant of the Jacobian, $det(J)$, is positive. The exact approach is necessary and sufficient but even for small control volumes too time consuming to perform. The test has to be performed before every fitness evaluation. That increases the time required for one optimisation step.
Their efficient test which approximates the determinant is very restrictive. Only very limited deformations are possible. Figure 4.3 presents a small Free-Form Deformation which does not fulfil the efficient test, but the deformation is free from self-intersections. The accuracy of the test can be increased by recursively subdividing the control volume. However, the subdividing process increases the computational effort.

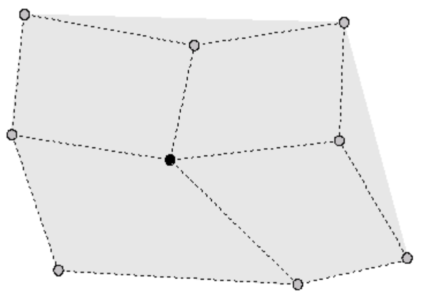


Figure 4.4: The cell of the black control point for two dimension. The corresponding convex hull is highlighted in grey. To fulfil the convex hull constraint, the black control point has to lie within that area.

4.1 Convex hull constraint

A constraint which provides good results and is already used at the Honda Research Institute is the convex hull constraint. There exists no proof that this constraint can not lead to invalid computational grids but at HRI the use of this constraint has successfully maintained the CFD mesh.
The convex hull constraint demands that every control point is inside the convex hull of its adjacent control points. Mathematically the following equation describes this constraint for control point P_{ijk}:

$$P_{ijk} = \sum_{l=1}^{26} \lambda_l P_l : \forall \lambda_l \geq 0, P_l \in \{P_{i-1,m-1,n-1} \ldots P_{i+1,j+1,k+1}\} \setminus P_{ijk}$$

This constraint can be checked successively for every control point. For point P_{ijk} of the lattice a cell of $3 \times 3 \times 3$ control points containing the neighbouring points is build. Then the convex hull of this cell is determined. Figure 4.4 depicts a control point and the convex hull of its cell in a 2D case. In the implementation the incremental algorithm from computational geometry [13] is used to calculate the hull. This algorithm starts with four points which build a convex polyhedron. Then the remaining points are added one by one. If the added point lies outside the current polyhedron, the current convex hull is updated. After all cell points are added it is tested if the control point P_{ijk} is within the convex hull.
Using these limitations for the control point movement leads to a constrained optimisation problem. Several methods are possible to treat these restrictions. A good overview of different approaches for EA is given in [2]. In the following sections the used approaches and their results are discussed.

4.1.1 Penalty function

A technique often applied in constrained optimisation are penalty functions. The idea is to transform a constrained optimisation problem into an unconstrained problem by adding a penalty term to the fitness function. This penalty has to be proportional to the constraint violation and is scaled with a weight which can be static or increased over the optimisation steps.
This weight is crucial for the optimisation and has to be chosen carefully. A large value discourage the exploration of the invalid regions of the search space. If several feasible disjoint regions exist, the algorithm will move to one of them and never leave it again. On the other hand a low penalty term will spend a long time for exploring the invalid regions. Besides, it is more likely that the final result of the optimisation is invalid.
One problem of the penalty approach in the case of CFD-simulations is that the regular quality function needs to be skipped due to the invalid computational grid. Thus the quality of an invalid solution depends only on the violation of the constraint. In the best case the results of the CFD-simulation can be approximated by similar but valid solutions. But such solutions are not easy to found.
Two different penalty approaches have been tested. In both cases the penalty function is proportional to the violation of the convex hull constraint. The violation is calculated by approximating the distance of the invalid control points to the convex hull of the corresponding cell.

$$p(x) = C \sum_{i,j,k} \|P_{ijk} - S\|^2$$

Here C is a static penalty factor and S the intersection of the convex hull and the connection line between control point P_{ijk} and the centre of its cell.
The first penalty function only depends on the violation of the constraint. The evaluation of the original fitness function is skipped. To ensure that infeasible solutions have a higher fitness, the penalty term contains additionally a constant value higher than the quality of feasible shapes. The fitness is defined as follows:

$$f_{p'}(x) = \begin{matrix} f(x) & \text{if x is feasible} \\ p(x) + c & \text{otherwise} \end{matrix} \tag{4.1}$$

where $p(x)$ is the penalty function as described before, and c is a constant offset.
In the second penalty function the constant factor is replaced with the fitness value of an approximated feasible solution. This solution is calculated with the correction algorithm described in the next section. If a correction was not possible, the same offset than in (4.1) is used.

$$f_{p''}(x) = \begin{matrix} p(x) + f(x') & \text{if x}' \text{ is available} \\ p(x) + c & \text{otherwise} \end{matrix} \tag{4.2}$$

where $x^{'}$ is the corrected solution of x.
However, the presence of an approximation enables also a more promising approach to solve constrained optimisation problems. Such an approach is explained in section 4.1.2.

4.1.1.1 Death penalty

Another way similar to the use of a penalty function is to reject invalid offspring and create new ones until a feasible solution is found. This approach, also called death penalty, can only produce valid solutions.
A drawback of this approach is that no information from the infeasible search space are exploited to guide the search. It can take a long time until a valid control point set is created. In fact it is not even ensured that a valid lattice is created ever. This drawback especially appears if the optimum requires large deformations as in the dolphin problem. For example in the dolphin case the number of mutation trials increases drastically during the optimisation. After 150 generations thousands of reproductions were necessary until a valid offspring was found.
To increase the probability of getting a valid offspring, the mutation step size can be decreased. But the reduction of the step size leads to a small variance in the offspring population. This results in a stagnation of the optimisation and prevents that the optimum is found.

4.1.2 Correction algorithm

A more sophisticated way to fulfil the convex hull constraint is to repair the control lattice. A correction algorithm has been developed which moves the control points so that the constraint is satisfied. To minimise the changes of the deformation, the displacement should be as small as possible. But this optimisation problem requires a considerable computational effort. Therefore a simpler algorithm which uses a small but not minimal displacement is applied.
In a random sequence all control points are tested if the convex hull constraint is hold. If an invalid point P_{ijk} is found, it is moved towards the centre of gravity M of its cell. Due to the random order, all control points have the same probability to keep their position. No control point is privileged. Otherwise it could happen that always the same control points are moved back. The length t of the displacement is determined by the intersection S of this connecting line with the convex hull and an additional small offset ϵ. The intersection is computed by solving a linear system of three equations for every face of the hull. The correct intersection point is found if the point S lies within the face.
The new position $P^{'}_{ijk}$ is calculated by the following equation:

$$P^{'}_{ijk} = P_{ijk} + t\,(M - P_{ijk})\,, \quad t = \|S - P_{ijk}\| + \epsilon$$

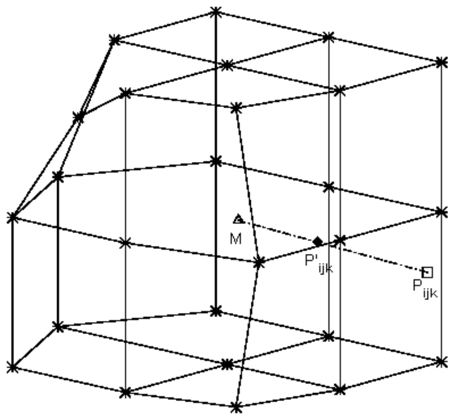


Figure 4.5: The invalid control point P_{ijk} is moved to the new position P'_{ijk}. M is the centre of the cell.

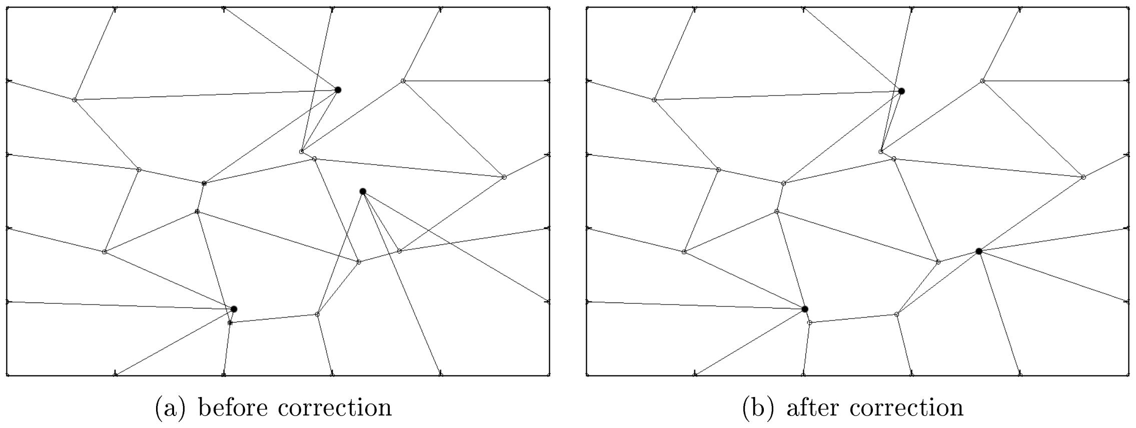
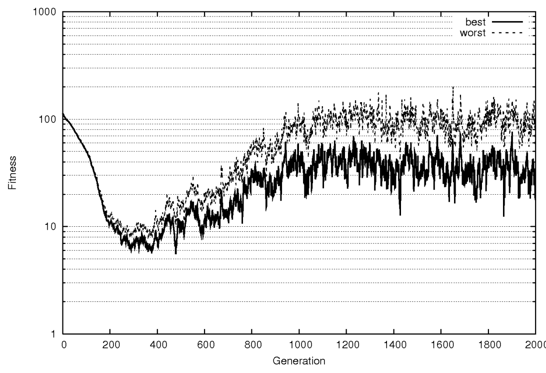


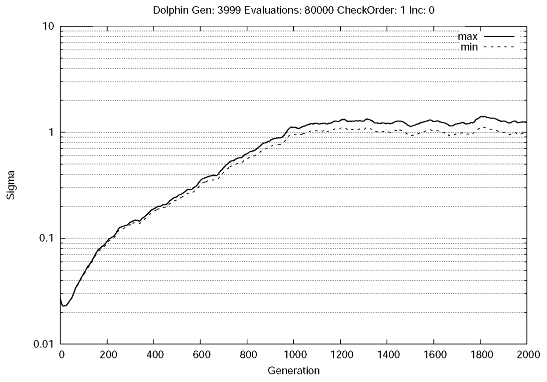
Figure 4.6: The control points of an exemplary control volume. In the left figure the marked points violate the convex hull constraint whereas on the right their coordinates are modified by the correction algorithm.

The correction of one control point is illustrated in Figure 4.5.
After shifting the invalid point P_{ijk}, the cells are checked again. This iteration is necessary because the displacement alters the convex hull of other cells. This can result in violations of the constraint at neighbouring control points. Chains of displacements can be generated. To interrupt potential infinity chains, a maximal number of iterations is introduced. These chains can especially occur if not adjacent control points change their position[1]. But since the modifications of the coordinates are limited by the mutation step size, such chains should be unlikely. Figure 4.6 represents the result for an exemplary 6 × 6 control volume in 2D before and after applying the correction algorithm.

[1]Two adjacent cells are mirrored over their conjointly edge.



(a) fitness



(b) maximal and minimal σ^2

Figure 4.7: The fitness progress of the dolphin test problem with the correction algorithm and CMA A. The value of the global strategy parameter is not reduced.

4.1.2.1 Interruption of self adaptation

With the introduction of the correction algorithm into the optimisation a problem in context of the CMA self adaptation of the mutation parameters occurs. The Shark library provides two different CMA implementations. One of them, called CMA A in this thesis, encodes the covariance matrix and the evolution path into the chromosomes of each individual similar to the global and individual self adaptation. This implementation updates the parameters immediately after the mutation is applied. The correction is applied after the mutation and the strategy parameter update. It changes the object variables. These changes can lead to a reduction of the attained variance. As explained in section 1.2.2.2 in this case the global step size should be reduced. But the update is based on the uncorrected object variables, and thus the step size is kept. So the global strategy parameter always increases and the ES does not converge. Due to the large mutation, the fitness even increases (figure 4.7).
To overcome this problem, the global step size is reduced proportional to the required correction. This reduction has to be done carefully. If the step size is reduced too much, the process converges to a local optimum. Two different approaches for the reduction of the global step size have been tested. In the first test (CD1) the global strategy parameter are reduced accordingly to the length of the required control point repairing:

$$\sigma^{2'} = \sigma^2 \frac{1}{exp(l_r/c)}$$

where l_r is the sum of the control point displacement of the correction algorithm and c a constant scaling factor. The exponential function should prevent that the search is going to far into the invalid area. But this function leads to a small step size which leads to a convergence at a local optimum in the area of the chest fin (figure A.3).

Therefore in the second test (CD2) the above described function is only applied if the algorithm was not able to repair the control lattice. As figure A.4 illustrates the higher step size allows to overcome the local optimum in the area of the chest fin, but the algorithm does not converge clearly. As one can see the global strategy parameter rise again.
In both cases the additional reduction of the global strategy parameter cannot repair the corrupted self adaptation. As soon larger modifications are applied by the correction algorithm the fitness rises due to the wrong adapted global strategy parameter.
Later another CMA implementation (CMA04) was used which provides a separate update function. This implementation does not encode the parameters into the chromosomes and uses the same parameters for all individuals. The update of the strategy parameters is carried out after the selection and is based on the attained variance. The changes in the genotype caused by the correction algorithm are considered by the parameter adaptation. The step size is reduced if the attained variance in the new parent population is smaller than the expected. Hence, the additional reduction as for the CMA A can be omitted.
The results of this approach (CD3) are shown in A.5. They depict that the self-adaptation of the strategy parameters works in the CMA04 also if the correction is applied. As a consequence the achieved fitness value is better than in the CMA A test runs, even though the optimisation converges at a local optima in the area of the chest fin.

4.1.3 Results

To compare the different approaches, the dolphin problem is used. Here large deformations are required which makes it difficult to keep the convex hull constraint. In all cases a $(3, 20)$-CMA Evolution Strategy is applied.
The fitness progress of the different approaches is depicted in figure 4.8.
The comparison of both penalty function (4.1) (run PD1) and (4.2) (run PD2) shows that the miss of the original fitness function leads to a much slower convergence. The lack of information about the fitness landscape if equation (4.1) is used does not enable to proceed towards the optimum in the invalid search space. After 4000 generations still no good approximation of the dolphin is found. The results of run PD1 are shown in figure A.1.
The number of fitness evaluations is reduced if the invalid designs are approximated with a valid shape. In this case the invalid solutions contain information about the fitness landscape. The search is able to move towards the optimum also in the invalid search space.
However, the result of the test run PD2 does not fulfil the convex hull constraint. As illustrated in figure A.2(a) the shape contains a loop at the chest fin. To get a

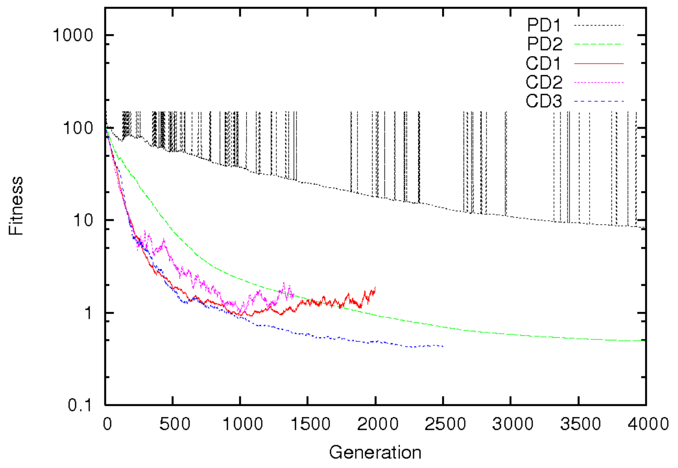


Figure 4.8: The fitness progress of the different non adaptive test runs.

	fitness	generations	evaluations
PD1	8.37427	4,000	80,000
PD2	0.49088	4,000	80,000
CD1	0.90988	1,089	21,800
CD2	0.96078	1,042	20,840
CD3	0.42121	2,267	45,340

Table 4.1: Results of the different test runs. The columns contain the values of the best solution found in the optimisation.

feasible shape, a higher penalty factor should be used. The results of run PD2 are shown in figure A.2.
These results expose that the quality of the shape has to be approximated. Otherwise the use of penalty functions results in an impracticable number of fitness evaluations.
The use of the correction approach can reduce the number of fitness evaluations further. Since the search space is limited to the feasible areas if the correction is applied, the optimisation converges faster as in the case of penalty functions.
The comparison of the two different CMA implementations shows that as soon as a fine adjustment to the boundaries of the feasible search space is required the CMA04 approach performs better. Since the CMA A is not able to adjust the global strategy parameter as good as the CMA04, the CMA04 achieves much better results even if the local optima could not be overcome. The correction approach using the CMA A even performs worser than the penalty which uses the correction algorithm to get an approximation for the fitness evaluation. The wrong adjusted strategy parameters prevent a convergence and better results. The results of the different test runs are summarised in table 4.1.

Chapter 5

Adaptation of representation

The creation of an adequate representation is difficult. The number of object parameters has to be chosen carefully. On the one hand the the number has to be small to speed up the optimisation. On the other hand the design has to be kept flexible enough to achieve the global optimum. A huge amount of experience and knowledge about designing and the methods used for the representation is necessary to satisfy this trade-off. Designers have to know how the optimal shape roughly looks like and which properties and parts of the shape are important. These assumptions are used by the designers to select the optimisation parameters for the design. These heuristics can prevent possibly new and unexpected solutions. If no previous knowledge about the sensitive areas is available, a suitable representation cannot be build.
On the other hand if the amount of object variables is not restricted the optimisation becomes infeasible. Due to the large search space, the optimisation needs many steps to converge.
An adaptive representation can overcome this trade-off between completeness and compactness even if no a priori knowledge is available. During the optimisation the representation is adapted automatically to the problem by increasing or decreasing the number of object parameters. The changed variability enables to find solutions which cannot be represented with the initial coding. The problem specific knowledge of the designers is less crucial for the optimisation.
The representation can also be adapted to the optimisation process. At the beginning of the optimisation, when the whole design space is searched for promising areas, a small set of object parameters, which allows only global modification, is sufficient. Later in the optimisation progress the number of object parameters is increased. New variation opportunities are enabled which allows to approximate the optimum more precisely. The search in the now higher dimensional space can be concentrated on the previously found promising area. Adapting the representation in such a way to the optimisation process realises a search in sub spaces and promises to reduce the number of optimisation steps.

The adaptation of the representation is, similar to the adaptation of the mutation strategy parameters, an optimisation of second order. The fitness values of the individuals are not improved directly, but the increased variability enables new capabilities for improvements of the fitness.
An important property for the adaptation of the mutation parameters is the strong causality. This property allows to draw conclusions from the changes of the genotype to the phenotype. Since the modifications of the coding are discrete, the changes have a minimal size which makes it difficult to realise the strong causality. To be able to predict the effect of the modified representation on the phenotype, the changes in the representation should not alter the phenotype. This neutral mutation decreases the probability of lethal modifications.
To adapt the representation, one has to increase the variability in promising regions. These sensitive areas and have to be found. This can be realised by comparing different representations and selecting the most promising one to proceed. The selection of the representation cannot be based directly on the fitness, because the neutral mutation does not change it. The quality of the changed variability can only be detected based on the optimisation progress. Individuals with a better adapted representation improve their fitness faster than the others. To determine this faster improvement of the fitness, a population is required. This population is evolved for some generations. The fitness gain which is achieved during these generations, can be used to compare several populations. The population with the highest improvement is the most promising.

5.1 Adaptation of FFD

5.1.1 Explicit insertion

The Free-Form Deformation representation can be adapted in several ways. The degree of freedom can be increased by inserting new control points while keeping the shape of the embedded object. Also a reduction of control points without modifying the design is possible. These modification can be implemented in the same way: Due to the decoupled arrangement of design and representation, any control volume can be chosen. The number of control points and their positions can be changed arbitrarily. After changing the control volume that way, the local coordinates of the current object have to be computed again.
This approach of explicitly modifying the control volume has some drawbacks. The computation of the local coordinate is costly especially for complex designs and CFD grids which contain many points. Furthermore the influence of the control points on the design is changed. In the case of an Evolution Strategy this fact destroys the adaptation of the mutation strategy parameters. The information about the search

space accumulated in the strategy parameters becomes useless since the fitness landscape is changed. Thus the explicit modification can be considered as a restart of the optimisation with a new representation and a better initial shape.

5.1.2 Implicit insertion

Another option is to insert control points implicitly. As already mentioned the degree of freedom of the used basis polynomials can be increased without changing the deformation. The advantage of this implicit insertion is that the embedded objects have not to be frozen again.
In the case of Bezier curves the degree elevation algorithm can be applied. Due to the global influence of the control points, all coordinates of the inner control points are changed. All object parameters have to be updated. As in the explicit modification case, the adaptation of the mutation strategy parameters is interrupted.
For B-splines the knot insertion algorithm can be used to increase the number of control points. In contrast to Bezier curves only the positions of the adjacent control points are changed. Their strategy parameter are wrong adapted after the refinement. The strategy parameters of the unchanged control points can be kept.
Additional the local definition allows to refine the representation in certain areas. The increased variability is not distributed over the whole control volume as in the case of Bezier curves. That enables to create areas of different density. That is important to get a compact representation. In sensitive areas, where local modifications are necessary to improve the fitness, a denser control lattice should be used than in areas which only require global modifications.
Due to the definition as a trivariate tensor product, always a complete plane of control points is integrated. That circumstance and the displacement of the adjacent control points limit the possibilities of pointedly refinements (figure 5.1).

5.2 Implementation

The adaptation of the Free-Form Deformation (AFFD) representation is implemented by refining the control lattice with implicit insertion at promising positions. These points are found by comparing different control volumes and choosing the best of them to proceed. Several populations are created. They all use a different control volume which emerge from a implicit insertion of control points into the current lattice. The exact creation of one population is explained later. Each of these sub-populations is evolved for a fixed number of generations. The best of them is selected and evolved for further generations until the refinement criteria is fulfilled

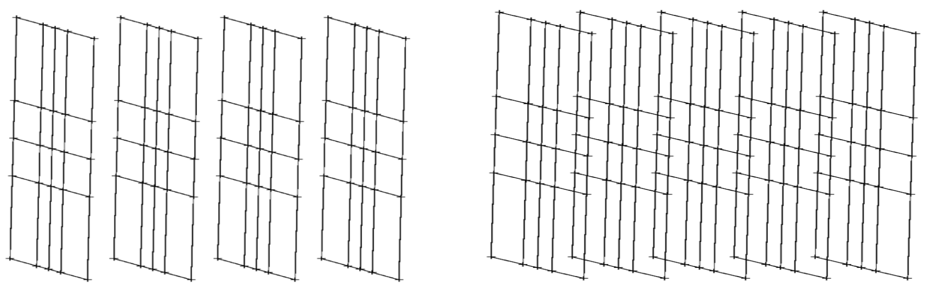

(a) before insertion (b) after insertion in x-direction

Figure 5.1: The refinement of a control volume always introduces a whole plane of control points. To keep the figure clearly, the connections in x-direction are not displayed.

again or a stop criteria is reached. The activity diagram of the main cycle is depicted in figure 5.2.

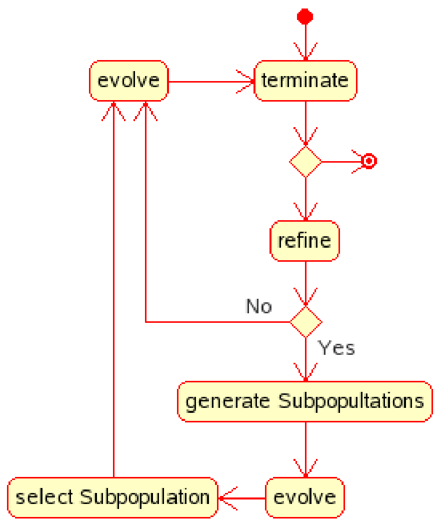
To compare the different representations, their quality have to be measured. As already mentioned a proper quality is to determine the fitness gain accomplished during some generations. Representations which achieve a larger fitness gain than others are better adapted to the optimisation process and are more promising to enable the representation of the optimum. Their control points have more influence in an area of the shape where deformations are required to achieve an improvement.

A crucial point of the adaptation is the point in time when the refinement takes place. Increasing the variability too late wastes time, because no further fitness improvement is possible. On the other hand inserting new control points too early foils the advantages of the rising variability and the approach becomes more and more similar to one with fix representation.

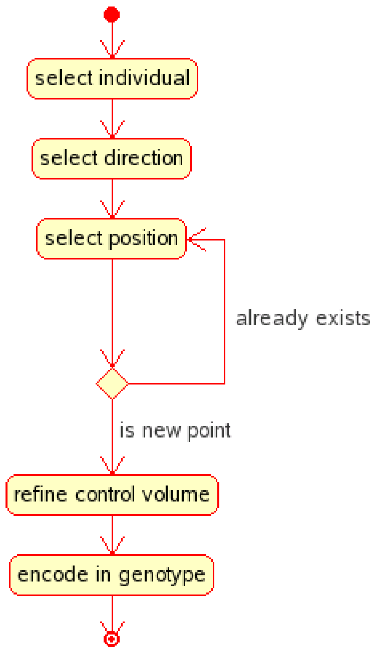
In the implementation first the refinement criteria was specified by the number of generations since the last refinement and the variance of the best fitness values of the last generations. When the minimal amount of generations is elapsed and the variance of the fitness values is below a constant threshold, the evolution of the population is stopped and a refinement is applied. A small variance of the fitness values indicates the convergence of the optimisation. Increasing the dimension of the search space can enable to find a better solution.

The use of a constant threshold for defining the refinement criteria is not sufficient. In a coarse representation at the beginning of the optimisation changes of the genotype lead to larger modifications of the phenotype than in a finer representation. As a consequence the variance of the fitness values is higher in the first generations and decreases during the optimisation process. A static threshold cannot be adjusted to the whole optimisation process. As figure A.9(c) shows between generation 25 and 50[1] a wrong adjusted threshold wastes time. Therefore in the later tests the refinement criteria is defined only by a minimal number of generations between two

[1]In generation 50 the representation was refined for the first time.



(a) Evolution cycle of the adaptive approach. The evolve activity is the same as in the non adaptive case 3.5.



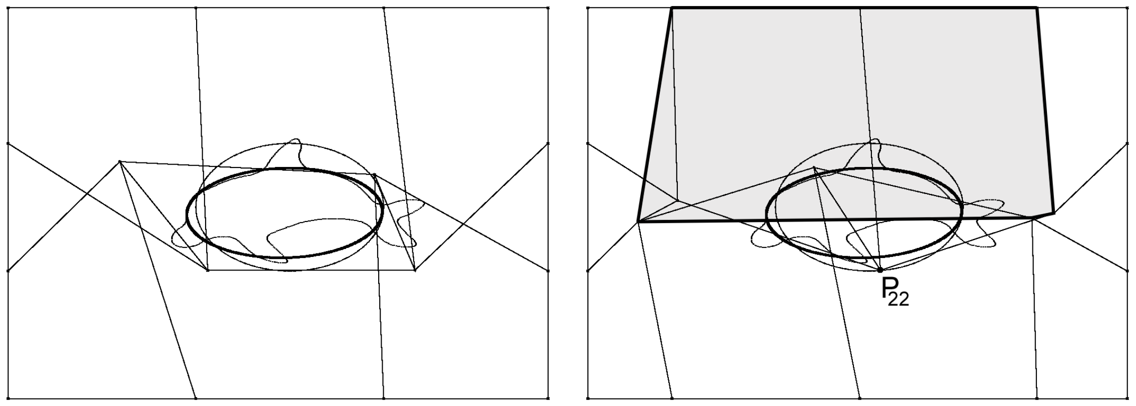
(b) Activity diagram of the generation of one sub-population.

Figure 5.2: Activity diagram of the adaptive FFD optimisation.

refinements. This number is increased to respect the slower convergence caused by the enlarged number of object variables.
The convenient time for insertion could also be detected on the basis of the global step size. A low value indicates the convergence and hence the variability has to be increased to enable further improvements. Waiting until the step size is small waste time. First some optimisation steps pass without improving the fitness significant. After the refinement further improvements are possible. But due to the small step size, more steps are required to carry out this improvement even if the strategy parameter adaptation enlarge the step size again.
To avoid that the control lattice is refined too early, one of the sub-population can use an unchanged representation. This can improve the adaptation of the representation to the optimisation progress. If the lower number of object parameters is sufficient at the current state, this population can be selected. On the other hand the unnecessary fitness evaluations of the refined populations could be omitted by a later insertion operation.
However, more investigations are necessary to improve the define a proper refinement criteria.

5.2.1 Generation of populations

The different populations are created by selecting a random individual and refining its control volume. One of the three directions is randomly chosen. Then a random point is inserted into the knot vector of this direction. If this point already exists or is close to an existent knot point, a new point is selected. The aim of the minimal distance is to prevent that control points are too close together. In such lattices it is more difficult to achieve large deformations, because keeping the convex hull constraint requires more correlated deformations. The minimal distance decreases after every refinement to allow dense lattices in a later optimisation stage. In figure 5.2(b) the different activities are depicted.
The refinement is done for the initial unchanged control volume and the modified control volume encoded in the individual. The refinement of the initial control volume is necessary because the encoding of the control points coordinates is relative to the initial control points.
After the control volume is refined, the genotype of the individual must be updated. The four chromosomes are extended according to the number of new inner control points. The old values of the unique control point numbers have to be changed with respect to their new grid position. The specification of the direction can be kept. The object variables of the neighbouring control points have to be updated. For B-splines of degree 3 these are the two adjacent planes of control points. The parameters required for the encoding of the new control points are then appended at the end of each chromosomes.



(a) before (b) after

Figure 5.3: The control volume before and after insertion in direction X. In the left figure the violation of the convex hull constraint of control point P_{22} is depicted.

The strategy parameters are also updated. In the case of a CMA strategy the evolution path and the covariance matrix are enlarged. The parameters of the unchanged control points are kept, whereas the parameters of the changed control points are approximated by their old values. The strategy parameters for the new object variables are initialised with default values. For the covariance matrix this are unit vectors.
When only a global strategy parameter is used, its old value is kept. For an individual step size adaptation the values of the old object variables are kept whereas the step size for the new ones are initialised with a default value determined by the user.

5.3 Constraint violation caused by insertion

In context with the convex hull constraint one problem occurs when the implicit insertion is applied. The convex hull constraint has to be kept during the whole optimisation process to ensure that the computational grid is valid. Refining the control lattice by an implicit insertion can violate this constraint. Control points which are added or changed due to the insertion can be located outside the convex hull of their cell. One example is shown in figure 5.3. Before the refinement the constraint is fulfilled. After the middle column of control points is introduced, the control point P_{22} violates the constraint.
Even though the implicit insertion does not change the deformation and the structural composition of the computational grid is still valid, it has to be prevented that the constraint becomes violated. Otherwise one cannot assure the grid validity in the next optimisation steps. In the next steps the convex hull constraint has to be fulfilled again. To overcome this problem, three different approaches are discussed: a late as possible, a penalty and a correction approach.

The problem of the constraint violation especially occurs in the death penalty approach, which was firstly used. Although the refined individual has a valid computational grid it cannot be used as parent for the next generation. In the death penalty approach new solutions are produced until a valid one is found. When the parent violates the constraint, the creation of a valid offspring is very unlike. Therefore the refinements have to fulfil the convex hull constraint. One has to find point and direction pairs, which do not lead to a violation. But finding such pairs is difficult. Especially in areas with large deformations it is likely that the insertion violates the constraint, whereas in areas where the deformations are small it is easier to find them. But in particular the areas with large deformations promise to have more benefit from a refinement.

5.3.1 Late as possible

The late as possible approach was used for the death penalty method to overcome the problem of a violated constraint after the refinement of the representation. Before the optimisation begins, different point and direction pairs for the next refinement are selected. When the refinement criteria is fulfilled, these pairs are used to create the sub-populations. Additionally in every generation it is checked if the refinement of the offspring population at the selected points would violate the constraint. When the constraint is fulfilled, the optimisation proceeds. Otherwise the parents of the offspring population are refined immediately even if the refinement criteria is not complied.
This as late as possible approach has some drawbacks. The large deformation required for the dolphin test problem violates the convex hull constraint very often. In the test runs only a few generations pass between two refinements. This circumstance complicates the comparison of the different control volumes. The number of generations to evolve the different populations is too small to get reliable results for the fitness gain.
Additionally the new insertion points and directions still cannot be chosen freely. The insertion of the new selected points into the control lattices of the current population has to fulfil the constraint.

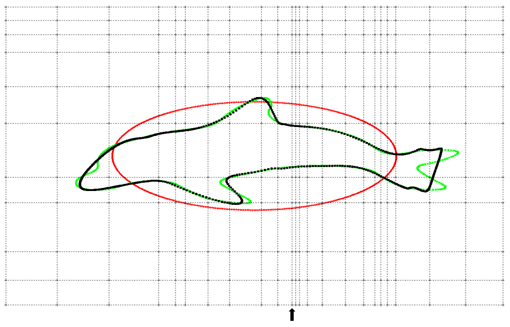
5.3.2 Penalty function

One option is the use of a penalty function already presented in section 4.1.1. In this case the hard constraint is replaced by a soft constraint. This allows to chose arbitrary points and directions for the refinement.
On the other hand the penalty approach does not fulfil the neutral mutation property. Due to the violation, the fitness value of the refined individual is higher than

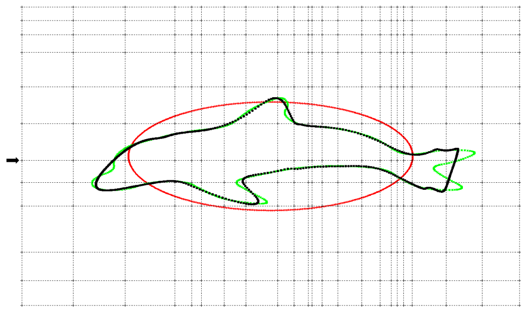
the unrefined although the individuals represent the identical shapes.
The penalty further interfere the comparison of different control lattices. It is difficult to conclude from the fitness gain on the quality of the new representation. Dependent on the penalty factor refinements with a large violation have a much higher fitness. The reduction of this violation enables to achieve a high fitness gain without improving the original fitness. Due to the high fitness gain, these populations are preferred in the selection. Thus the quality of the variability cannot be measured because the comparison between the different representations is mainly based on the size of the violation introduced by the refinement.
An other option is to use the absolute fitness to select one of the sub-populations. That is similar to a selection which is based on the fitness gain without the penalty caused by the insertion. In this case refinements which result in a large violation are not preferred. On the contrary they are disadvantaged, because they have to overcome the violation. The results of a test run (APD1) are given in figure A.8. In this test a single sub-population was created till generation 180. Afterwards their number was set to 15. The fitness progress depicts that until this generation refinements, which lead to a violation, occurs. After generation 180 only populations with a small violation were chosen whereas the others are dropped.
However, a refinement in an area with large modifications seems more promising. An example from the test run is given in figure 5.4. The refinement shown on the right would enable new variability but was neglected due to the high penalisation. Instead the refinement on the left was selected. There no penalty occurs. As a consequence in the areas of the head and the caudal fin no further improvements are possible. The optimisation converges at a fitness of 0.904.
One possibility to reduce the influence of the violation is to evolve the populations for more generations. This would increase the opportunity that refinements which lead to a violation but enable new and necessary variability are selected instead of unneeded ones.
On the other hand the amount of fitness evaluations increases simultaneously.

5.3.3 Correction algorithm

Another option is to correct the control point lattice to fulfil the constraint. After refining the lattice the invalid control points are displaced into the convex hull of their cell. This correction is done by the same method previously explained in section 4.1.2.
As in the penalty approach the positions and the time of the refinement can be selected freely. Besides the comparison of the different representations is not influenced as much as in the penalty approach, because the fitness gain depends mainly on the refined position especially if the fitness gain is used. Even if the correction modifies the deformation and accordingly the shape the changes of the fitness values



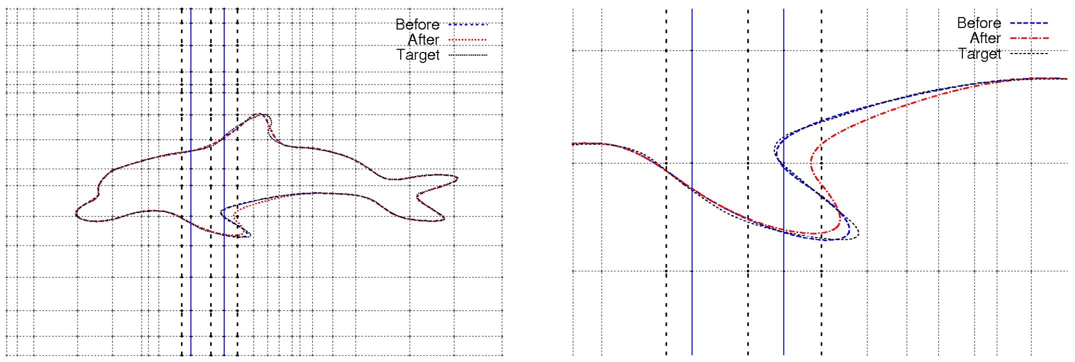
(a) The selected control lattice.



(b) A control lattice which seems to provide a more useful variability. To achieve a better matching at the head and at the tailing fin of the dolphin, control points in this area are required.

Figure 5.4: Two control lattices used by different populations during a test run of the adaptive penalty method. The new control point row or column are marked with an arrow. The population using the left lattice was selected even if it does not provide any new useful variability. The refinement one the right leads to a violation of the convex hull constraint. The solutions of that population have a higher fitness value although the refinement provides new variability.

are smaller than in the penalty approach.
Since the correction algorithm modifies the phenotype, the fitness has to be computed again. However, the different test runs have shown that in most cases the population with the highest gain also has achieved the best absolute fitness value. Using the absolute fitness instead the fitness gain can omit the additional evaluations.
The correction does not satisfy the neutral mutation property completely. The displacement of the invalid points changes the deformation and accordingly the shape. Especially in areas with large deformations and dense lattices the refinement can lead to violations of the convex hull constraint. The correction algorithm modifies the deformation in that area and thus also the shape. These annulled deformations have to be recovered again. Figure 5.5 depicts an example. After inserting a new point in x direction the chest fin is matched clearly less. To achieve the old deformation, several optimisation steps are necessary. Such refinements slow down the optimisation. On the other hand the insertion at such points are necessary to provide a sufficient variability as run APD1 exposes.



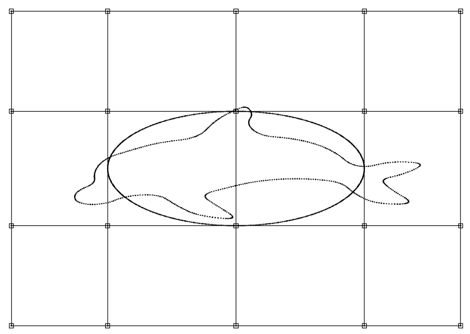
(a) Full shape (b) Zoomed at changed part

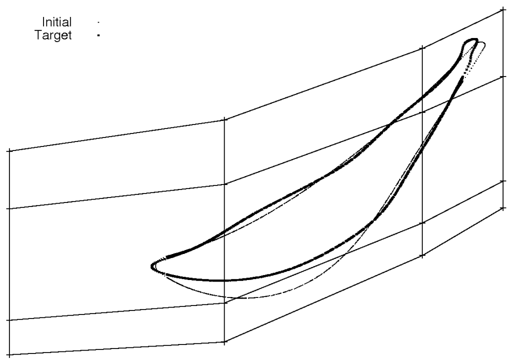
Figure 5.5: The control lattice and the shape before and after a refinement in x direction are shown. The solid blue lines displays the edges of the control lattice which are replaced by the refinement. The new edges are depicted as bold dotted lines. Due to the correction of the convex hull constraint, the dolphin matches the target worse after the refinement.

5.4 Results of adaptation

Several test runs with different number of sub-populations, depicted in table 5.1, have been used to solve the dolphin or blade problem. All tests using a $(3,20)$-CMA-ES. Except for run APD1 which uses the penalty approach, in all runs the correction approach is applied. The initial control lattices are shown in figure 5.6. The sub-populations are evolved for 15 generations before the best one is selected according to their fitness gain.
In test run AD1 the refinement criteria was determined by the fitness variance and a minimal number of generations between two refinements of 15. This number was raised in generation 400 to 20 and in generation 800 to 25. Till generation 300 the minimal amount of generation elapsed before the variance of the fitness values has reached the threshold. Afterwards the point in time for the refinement was only determined by the minimal number of generations because the fitness variance was always beyond the threshold. From generation $1,000$ no refinement was applied any longer.
In all other runs the refinement criteria was specified only by the minimal number of generations which starts at 15 and is incremented by two after each insertion cycle.

A comparison of the adaptive penalty and correction approach is given in figure 5.7. In the first generations both approaches seem very similar. But from generation 600 the correction approach achieves much better results. After that generation the penalty approach could not increase the required variability.


(a) dolphin



(b) turbine blade hub section

Figure 5.6: Initial control volume for adaptive FFD tests.

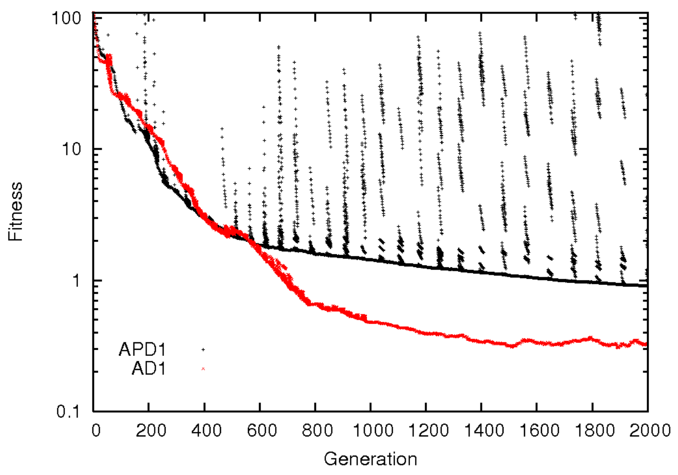
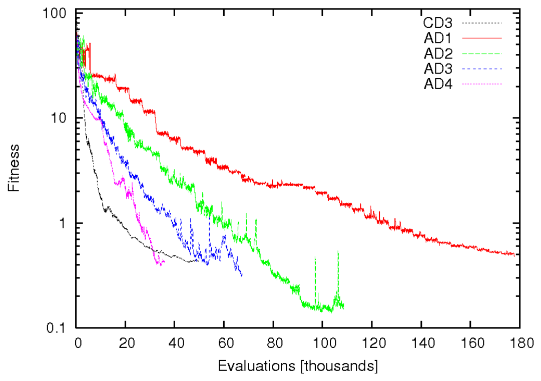


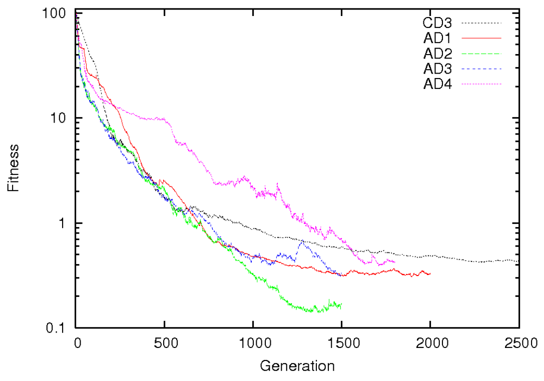
Figure 5.7: Fitness progress of the adaptive penalty and correction approach. The fitness values of the rejected sub-populations are included. In the first generations both approaches performs similar. But after generation 600 the penalty approach could not increase the required variability, because the refinement leads to a large violation of the convex hull constraint. As the peaks indicates the violation of some refinements increases right before that generation.

5.4.1 Adaptation to optimisation process

One goal of the adaptive approach is to adapt the representation to the optimisation process in order to speed up the convergence. As the strategy parameters indicate the ES (in both approaches adaptive and non adaptive) starts with a global search. In the first generations a large step size is used and then reduced by the strategy parameter adaptation.
Starting with a coarse lattice, which allows only global changes, and refining the representation more and more supports the search strategy of the optimisation process. The combination of both enables a search in subspaces. In the first steps a rough approximation of the optimum is found. This rough estimate is refined more and more during the optimisation process until a good approximation of the optimum is found.
Regarding some interim steps of the dolphin test problem one can see that this search in subspaces is realised. The initial circle is firstly deformed to an eclipse. Then it is bended to approximate the body of the dolphin. Afterwards the fins and the lip are developed more and more until the dolphin is almost matched. The progress of one deformation is depicted in figure A.14. In the non adaptive runs every part of the circle is moved for itself to match the dolphin A.15. This search in subspaces allows to find a solution near the global optimum for the dolphin test problem. As the shapes in section A.3 depict the adaptive test runs do not stick in the local optima with two chest fins as the most non adaptive test runs do. Therefore the fitness values of the adaptive approach are smaller.
To determine if this search in sub spaces speeds up the optimisation, the fitness progresses of the different test runs are compared in figure 5.8(a). One can see that the adaptive approach requires much more fitness evaluations even if the number of generations is reduced (5.8(b)).The picture also shows that the number of fitness evaluations increases with the number of used sub-populations. Only the test run AD4, where a single sub-population is created, achieves a faster convergence. However, this run is not faster than the non adaptive CD3 over the whole optimisation process. Due to the initial global representation, the fitness of the adaptive tests decrease faster especially in the first steps of the optimisation. But then the non adaptive run makes up leeway and performs better between generation 150 and 1500. Afterwards the adaptive test run AD4 performs better again. A better fitness is achieved while less fitness functions were evaluated.
During the generations 150 and 500 the fitness gain of run AD4 is very small in comparison to the other test runs. It seems probable that the refinements between these generations do not lead to the required variability. Using a heuristic to determine the insertion point instead the random selection can decrease the probability of such refinements.
In test run AD5 a simple heuristic was used. The direction of the refinement is determined by the number of lattice points. Always the direction with the lower number of control points is selected. If their number is equal, the x direction is used. The



(a) The fitness according the fitness evaluations.



(b) The fitness according the generations.

Figure 5.8: The fitness progress of the adaptive test runs.

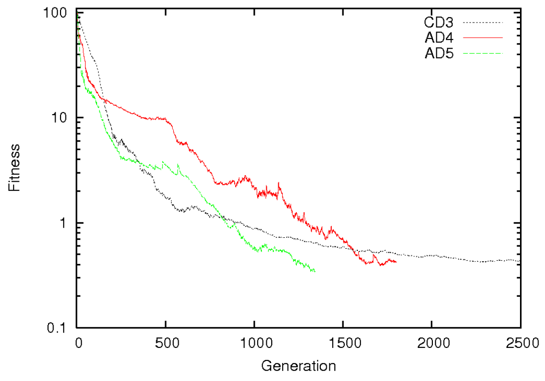
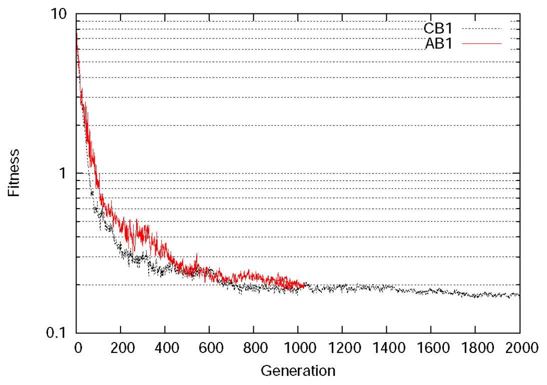


Figure 5.9: The fitness progress of the adaptive tests with a single sub-population.

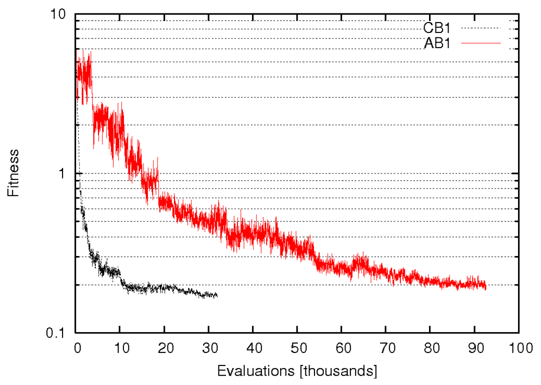
new knot points halve the largest interval of the corresponding knot vector[2]. The enlargement of the search space is stopped when a 19×18 control lattice is reached.

Figure 5.9 illustrates that the heuristic reduces the number of fitness evaluations while simultaneous a better solution is created than in the run CD3 and AD4. Regarding the fitness progress of the turbine blade test problems (5.10) one can see that the adaptive test does not reduce the number of generations. The adaptive approach is not even faster in the first generations. Even a single population will not speed up the optimisation. This can be explained with the better start point. The given design does not need as much global changes as the circle in the dolphin case. In such a case the initial grid should already be finer. Otherwise several refinements have to be applied before the variability is sufficient to enable the required deformations.

[2]If several intervals are equal, the first of them is halved.



(a) The fitness according the generations.



(b) The fitness according the fitness evaluations.

Figure 5.10: The fitness progress of the adaptive turbine blade tests.

5.4.2 Adaptation of control lattice

Another aim of an adaptive representation is to create a compact representation while still providing a sufficient variability for the current search stage. To evaluate the adaptation of the representation to the problem, one has to regard the number of object parameter and the achieved fitness value. The runs of the adaptive approach all achieve a slightly better fitness value. This improvements stem from the overcoming of the local optima in the area of the chest fin. The variability of the non adaptive control lattice is high enough. One the other hand they use a much higher number of object variables than the non adaptive solutions. The solution of run AD2 for example uses a control lattice of 21×16 points. That corresponds to 532 object variables, whereas the non adaptive solutions only uses 288. This numbers do not testify a compact representation. However, it has strongly to be mentioned that the control lattice used for the non adaptive tests is already adjusted by hand.
The problem in the automatic adaptation is that control points lose their influence on the embedded shape. Refining the control lattice always introduces a complete plane of control points. Even if the refinement is only needed in a smaller part. The control points of the adjacent planes are relocated. Additionally the area of influence becomes smaller because of the piecewise definition of the B-splines. Control points can lose their influence on the shape. Especially the control points positioned at the boundary of the shape are moved more and more to the outside and lose the effect on the design. In run AD1 for example the first five and the last three rows of the control lattice and the control points inside the circle have no more influence on the embedded shape.
Besides, the refinements of the representation depend only on the number of generations. Refinements are applied even if the variability is sufficient for the optimisation. To achieve a more compact representation, the refinements should not be applied if no further improvement is possible. More research is necessary to develop conditions for the definition of a sufficient refinement criteria. As already explained in section

	#pop[3]	fitness	generations	evaluations	control lattice	control points with influence	
						number	ratio
CD2	-	0.9608	1,042	20,840	14×14	128	88,88
CD3	-	0.4212	2,267	45,340	14×14	128	88,88
AD1	16	0.3058	1,894	177,900	23×18	205	61.01
AD2	10	0.1376	1,379	108,590	21×16	144	54.14
AD3	5	0.3097	1,492	67,215	22×18	134	41.88
AD4	1	0.3882	1,709	34,233	17×25	139	41.44
AD5	1	0.3133	1,356	27,148	19×18	120	44.12

Table 5.1: Results of the adaptive test runs. For comparison the two non adaptive test run CD2 and CD3 are shown again. The columns contain the values of the best solution found in the optimisation.

	#pop[3]	fitness	generations	evaluations	control lattice	control points with influence	
						number	ratio
CB1	-	0.1636	2,267	45,340	19×17	137	53.73
AB1	15	0.1840	960	86,139	14×15	110	70.51

Table 5.2: Results of the turbine blade test problem. The columns contain the values of the best solution found in the optimisation.

5.2 the fitness value and the strategy parameter can be used as a point of origin.
For the turbine blade problem the results look different. The resulting control lattice of the adaptive test run (AB1) consists of less control points than the one adjusted by hand. The number of object variables is reduced from 510 to 312 while the achieved fitness is slightly worse. The non adaptive test run achieves a fitness value of 0.164 the adaptive 0.184. This more compact representation is achieved because the minimal number of generations between two refinements was increased to 100 at generation 700. That avoids many unnecessary refinements.
This adaptation is expensive. AB1 uses 15 sub-populations and thus the number of fitness evaluations increases from $32,000$ to $86,139$. The results are summarised in table 5.2.
One option to reduce the number of object variables appreciably is to encode only the control points which have influence on the shape, into the genotype. In the adaptive test runs their number depends on the is used sub-populations and is about 50%. In run AD2 for example the number of object parameters would be reduced from 532 to 144. Table 5.1 contains the exact values of all test runs. In comparison to the control lattice adapted by hand the adaptive approach obtain a similar number.
This information could also be used to improve the correction algorithm. Control

[3]number of sub-populations

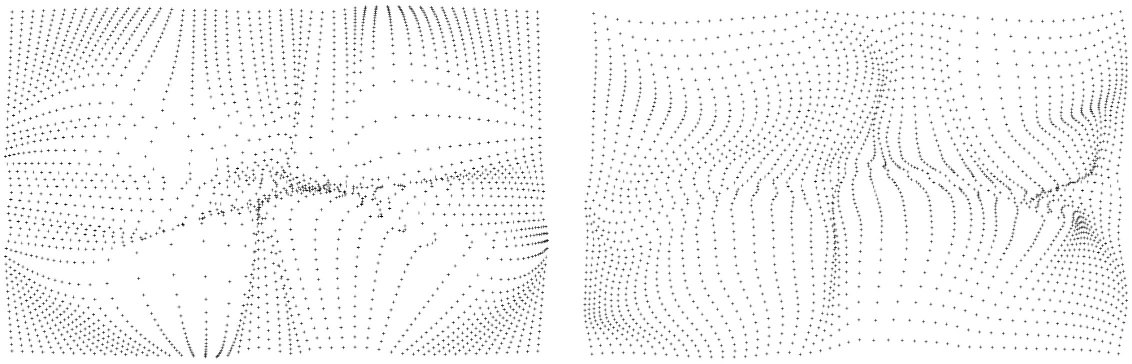

(a) non adaptive run CD3 (b) adaptive run AD1

Figure 5.11: Deformation of a equidistant point grid.

points which do not effect the design can be relocated preferred, whereas points with influence remain unchanged more likely.
A further option to obtain a more compact representation could be the use of another spline basis function. A basis function which enables more local refinements by adding a single control point and does not change the influence of the old control points, is preferable. Possible candidates for such spline basis could be hierarchical splines or T-splines. They enable very local refinements.
The global representation at the beginning and the relocation of the control points also have an advantage. Since the more global deformation before the refinement is kept by the insertion and also maintained in the following optimisation steps, the deformation is smoother than in a non adaptive approach. In figure 5.11 this is illustrated by deforming an equidistant grid of points. The lattices used for the deformations stem from run AD1 and CD3 of the dolphin problem. In the manual adjusted control volume the placement of the control points leads to an unbalanced deformation. Such a smoother deformation will lead to CFD meshes with a better quality. A CFD mesh for test run CD3 would have cells of very different volume whereas the cells in run AD1 have a more similar size.

5.4.3 Problem specific knowledge

The adaptive representation should reduce the required problem specific knowledge to build an adequate representation. That is important if no a priori knowledge about possible solutions is available and the designers cannot build a proper representation.
The results of run AD1 (A.9(b)) show that the control volume is adapted to the problem. As expected, the control volume is finest in the region of the fins. There the most local modifications are necessary to match the target. In the other regions,

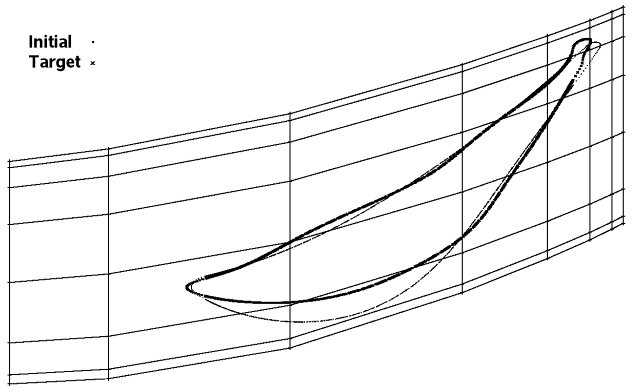


Figure 5.12: The control volume of the adaptive test run AB1 in generation 216. In the area of the trailing edge the grid is denser.

where the shape has less curves, the control lattice is coarser. The adaptation is more obviously in the blade test problem. The target turbine blade contains the largest curvatures at the trailing edge. In that area local deformations are required. The adaptive approach has adapted the representation. The control lattice is most fines in that area (A.7(c)). Regarding the control lattice of an earlier state (5.12) the different densities of the control lattice are obvious.
The control lattice produced by the AFFD method is very similar to the manual adapted (3.8(a)). This indicates that the adaptive approach can automatically detect the sensitive areas of the design. The influence of the previous knowledge of the designer is reduced. Even if no knowledge of possible solutions is available, an adequate representation is build. That enables to find unexpected solutions not representable with the initial coding created by designers.
The adaptation depends on the number of populations. Using only 10 or less does not lead to a control volume which looks adapted. Especially in run AD4 many control points seem unnecessary. However, the adapted control volume in test run AD1 does not lead to a better fitness value than in the other adaptive test runs. On the contrary the fitness is worse than in run AD2. The very dense control lattice in the area of the chest fin creates some problems. As already mentioned the refinement in that dense region leads to a violation of the convex hull constraint. The correction annuls the deformations achieved so far. To obtain the old deformation again, neighbouring control points have to move correlated. The more control points are involved the more correlated movements are required. This slows down the optimisation. In a coarser grid fewer correlated mutations are necessary. That enables to achieved large deformations faster as the chest fins in the figures A.10(b) and A.12(b) show.

5.4.4 Conclusion adaptive approach

The results demonstrate that the AFFD approach can reduce the influence of the initial representation. That enables to find also unexpected solutions. The AFFD approach depends less on the a priori knowledge of the designers. However, to obtain a proper representation, several sub-populations and a huge amount of fitness evaluations are needed.
For costly fitness evaluations the adaptive approach with several sub-populations will not be feasible due to the high number of evaluations. On the other hand if no previous knowledge about possible solutions is available the adaptive approach can be used to detect sensitive areas.
A reduction of the fitness evaluations can be obtained if global deformations can improve the fitness and if a single sub-population is used. For good start points global deformations are not necessary to improve the fitness. In such a case the initial lattice should be already finer.
Achieving both goal, a reduction of the fitness evaluations and the creation of an proper representation, seems difficult to realise.
One possibility to achieve an adapted control lattice with fewer fitness evaluations is to use a increasing number of sub-populations. In a coarse lattice the benefit of a refinement point depends mainly on the direction of refinement. The positions of the new points are less important, because the area of influence is still large. When the number of sub-population depends on the current number of control points, especially in the first generations the number of fitness evaluations can be reduced. The adaptation of the lattice should still be possible since for the later refinements a sufficient variety of representations is available.
An other option is to use more sophisticated heuristics. They can be based on the information of the control point displacement. Areas where the positions of the control points are changed more seem to be sensitive for the design, especially the areas with a large relative movement of adjacent control points. In such areas large deformations are applied. Regions with only a small deformation seem to be optimal or insensitive for the optimisation. However, more investigations are necessary to develop good heuristics. On the other hand the use of heuristics can prevent to find new and unexpected solutions which is one goal of the automatic adaptation.

Chapter 6

Direct Manipulation

As already mentioned it is important to have a small number of object parameters to represent a shape while on the other hand a high flexibility has to be provided. To achieve a good trade off between compactness and completeness for the Free-Form Deformation, the initial control lattice has to be created properly. The influence of the control points on the embedded shape has to be maximised. A high impact is especially important in the constraint optimisation. If the movements of the control points are restricted, the influence of the control points on the shape has to be high to achieve anyway large deformations in a few steps. Since the influence of the control points decrease with their distance to the design, the maximisation is usually done by placing the control points near the sensitive regions of the design. The creation of such a problem specific control volume requires some knowledge about the behaviour of FFD and splines. Moreover it is not possible to achieve a high influence of all control points. Due to the piecewise definition of B-splines, in fine control lattices the influence of a control point decrease fast with the distance. Some points even have not effect the shape.
The direct manipulation technique can reduce the problem of maximising the influence. The specified deformations are always achieved independent of the influence of the control points. The object points are moved to the target positions even if the control points are far away. The accurate placement of the control points is less important. To achieve the target deformation exact, one has to assure that the number of object points is not higher than the number of control points.
Of course, the possible deformations still depend on the control lattice and all deformation which can be achieved by the direct manipulation can also be achieved by the general FFD approach. In both cases the deformation is ultimately defined by the control lattice. But the desired deformation should be achieved faster, because the influence of the object variables on the phenotype is larger. As already mentioned in the general FFD approach the influence of the various control points is different.
In the direct manipulation the impact of the object variables, which are the target

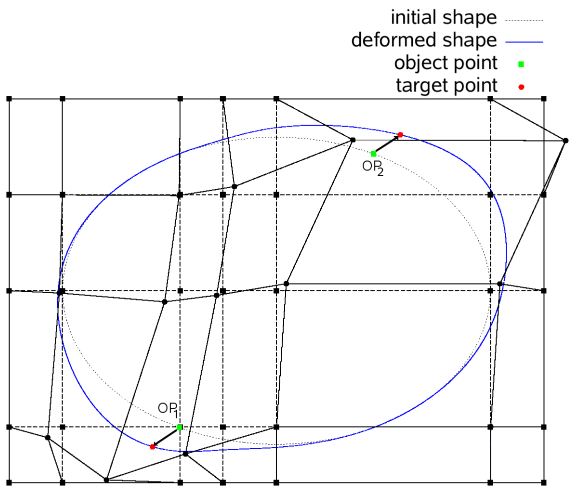
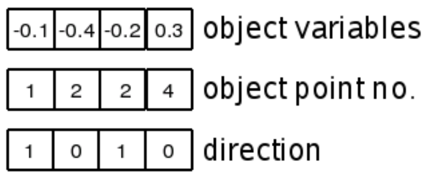


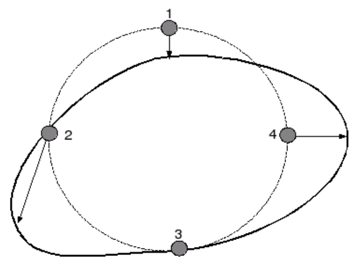
Figure 6.1: The displacements of the two object points are of equal size. Whereas the displacements of the control points are quite different. To achieve the displacement of object point $OP1$, two control points have to be shifted far because their influence on the embedded circle is smaller. This indicates the larger and straighter influence of the object points on the shape.

deformations, is straighter. A change in an object variable leads to an equivalent alteration of the design. Large modifications of the object variables lead to large changes of the design whereas small ones modify the shape only slightly. Thus the strong causality property is better implemented as in the general FFD approach. There small changes of control points with a large influence change the phenotype more than large modifications of control points with little impact. Figure 6.1 gives an example. The changes of both object points are of the same magnitude. The changes of the control points are different.
The enhanced influence allows to reduce the number of object parameters while still a similar variability is provided. Of course, if the number of object points is smaller than the number of control points the direct manipulation will not be able to provide the same variability as the general FFD approach. As already explained in such a case several solutions exist to achieve the target deformation but only one of them can be specified. However, the differences of these deformations on the design should be quite small. Since the target points are matched of all these different solutions, the differences in the parts close to the object points have to be small.
The reduction of object variables should especially prune the variability which has no or only a small effect on the embedded shape. If a control point has no effect on the embedded shape, but is necessary for a correct defined control volume, its unneeded variability is reduced.
On the other hand this small differences sometimes lead to a significant better quality. Similar to the general FFD approach the object parameters have to be selected carefully to create a proper representation. However, the selection of adequate object points is more intuitive than the selection of control points. A precise knowledge of



-0.1	-0.4	-0.2	0.3	object variables
1	2	2	4	object point no.
1	0	1	0	direction

(a) The genotype



(b) The object points and target deformation

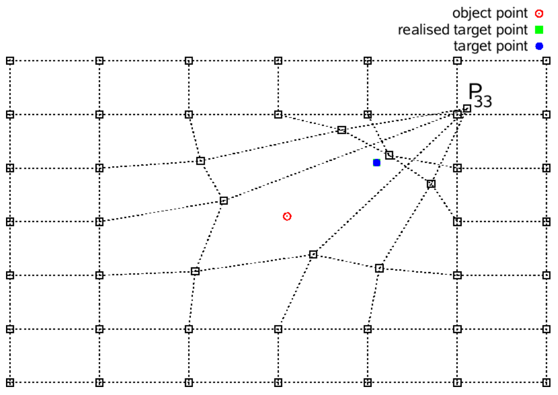
Figure 6.2: The encoding of object points and target deformations.

splines is no required.
These both advantages of the direct manipulation, the compacter representation and the increased influence on the design, should speed up the optimisation.
Despite these advantages the direct manipulation is still limited. Due to fixed representation, the initial control volume and object points determines which deformations are possible and which shapes cannot be represented. The control volume and the object points have to be adjusted to the problem and sensitive areas have to be found. The number of control points should not be as important as in the general FFD approach because they do not define the object variables. A higher number should not slow down the optimisation. To overcome this problem of building a proper representation, an adaptive DMFFD representation can be used. First investigations in that topic are made in section 6.4.

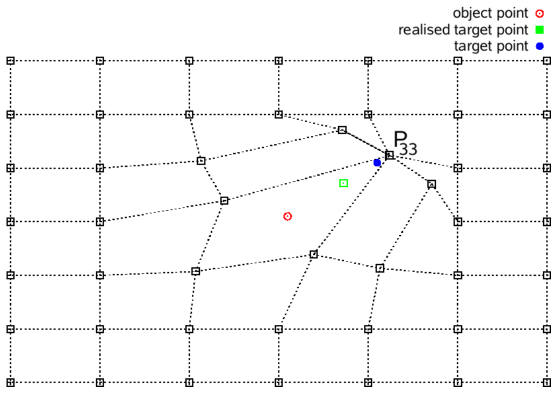
6.1 Implementation

6.1.1 Coding

In contrast to the FFD and AFFD approach where the control points are used as object variables, in the DMFFD method the displacements of the object points adopt this role. The displacement is divided into the three directions and measured in the (x, y, z)-coordinate system relative to the initial position of the corresponding object point. This object point is specified by its index in the initial point set. For each object parameter a point number and the direction of the target deformation are encoded. Figure 6.2 gives an example.
Furthermore each individual contains a control volume which fulfils the point deformations specified in the genotype. This control volume consists of an initial and a deformed control lattice as well as a knot vectors for each directions.
The strategy parameters are not encoded into the genotype because the CMA04 implementation is used.



(a) direct manipulation with pseudoinverse



(b) direct manipulation with pseudoinverse and correction algorithm

Figure 6.3: To match the displayed target points, the control volume is calculated with the pseudoinverse. The resulting control lattice does not fulfil the convex hull constraint at control point P_{33}. To fulfil the constraint, control points, which have a small or even no impact on these object point, have also to be moved. On the right side the correction algorithm is applied. As one can see the realised deformation is much smaller.

6.1.2 Computing control points

Unfortunately the efficient and exact approach of Hsu explained in section 1.4 cannot be used to compute the control point coordinates. To keep the structural composition of the computational grid, the convex hull constraint was introduced. This constraint has to be fulfilled to ensure that the fitness can be evaluated with the deformed grid.
The use of the pseudoinverse leads to invalid control lattices. Since the pseudoinverse calculates the solution with minimal displacement, only the control points which have influence on the object points are moved. The others keep their positions. This can result in a violation of the convex hull constraint if large deformations are needed to match the target points. Figure 6.3 shows an exemplary deformation calculated with the pseudoinverse. The resulting control lattice does not satisfy the constraint.
The implemented correction algorithm cannot be used to correct the computed lattice. This algorithm undo most of the deformations because the invalid control points are moved back. The size of the possible deformations depends on the distance of the control points. In a coarse lattice the control points can be displaced farther than in a tighter one. As a consequence local deformations will be small. Larger displacements can only be achieved with a coarse lattice and thus have to be more global. If such restricted deformations are sufficient for the optimisation, the pseudoinverse together with the correction algorithm could be applied. However, normally it is unknown with deformations are necessary.

Therefore the control point coordinates are not calculated with the pseudoinverse. They are determined by solving a target matching optimisation problem. The aim of this optimisation is to get a control lattice which matches the object points as good as possible while keeping the convex hull constraint. For solving this problem the FFD algorithm described in chapter 4 is used.
To initialise the optimisation, several dates have to be provided. The object points encoded in the genotype of the individual determine the initial points for the control point calculation. The target point set is defined by the initial object points shifted by the target deformation.
The control volume used in the control point calculation is initialised with the control volume encoded in the individual. That reduces the computational effort. The calculation of the control points starts with the deformed control lattice of the last generation. The deformation specified by this control lattice matches the target points of the last generation. Since the differences of the target points between two generations are not large, the necessary control point displacements should be small and easy to calculate.
The strategy parameters of the FFD method are initialised with the global parameter of the object point optimisation. This step size indicates how far the object points are displaced in comparison to their last positions. The displacement of the control points should be in a similar order of magnitude.
The fitness function of the FFD method is defined as the average distance between the deformed object points and the corresponding target positions:

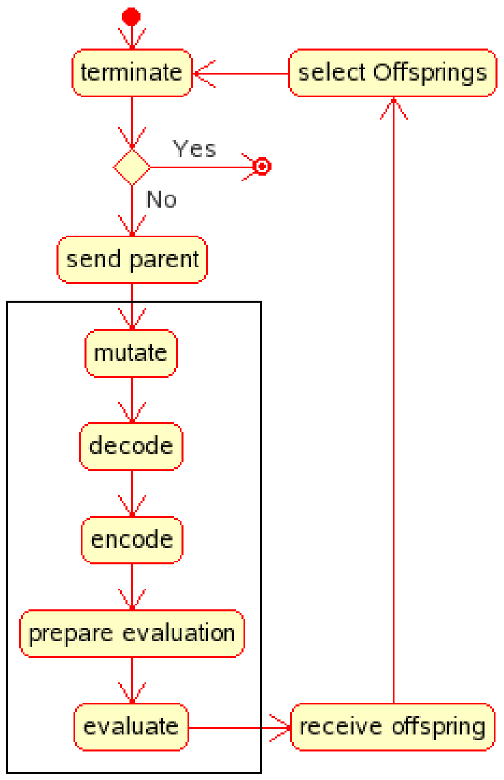
$$f(x) = \frac{1}{n}\sum_{i}^{n} \|OP_i - T_i\|^2$$

where n is the number of object points, OP_i are the coordinates of the object points and T_i of the target points.
The solvability of the constrained problem is not as clear as in the unconstrained case. Since it is possible that the problem has no feasible solution and to limit the computational effort, the optimisation is stopped if a certain fitness value or a maximal number of generations is reached. After the control volume is calculated, it is encoded into the genotype of the individual.
The FFD program used to calculate the control points does not use a master/slave architecture. Due to the limited computational capacity[1] the advantage of the parallelisation is negligible. Instead of the parallel offspring generation in the FFD approach the offsprings are all created by the master one by one.

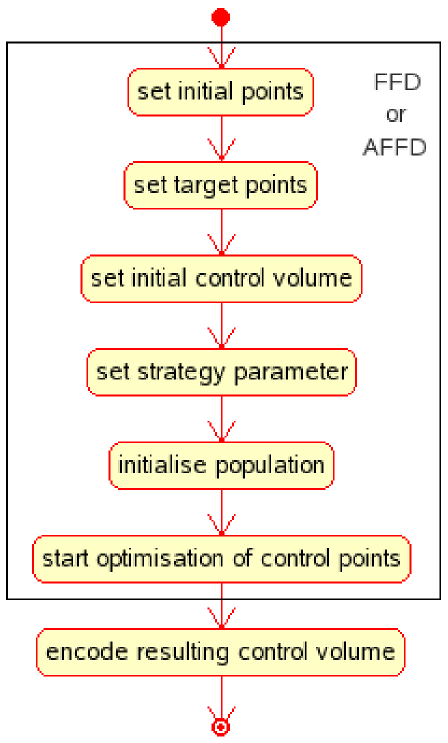
6.1.3 Evolution cycle

The Evolution Strategy using the direct manipulation technique (DMFFD-ES) is implemented similar to the one explained in chapter 3. The DMFFD program again

[1]The number was limited to two double core cluster nodes.



(a) The evolution cycle of the DMFFD approach. The part inside the box is realised by the slave.



set initial points
FFD
or
AFFD
set target points
set initial control volume
set strategy parameter
initialise population
start optimisation of control points
encode resulting control volume

(b) Activities of the control point optimisation to match the target positions of the object points. The box marks parts computed by the modified FFD program.

Figure 6.4: Activity diagram of the direct manipulation approach.

uses a master/slave architecture. The evolution cycle is depicted in figure 6.4(a). First the initial design, the target design and the initial control lattice defined in the parameter file are loaded. Then the initial design is frozen. After the population for the first generation is created, the evolution cycle starts.
The master sends a random parent and the mutation strategy parameter to every free host and then switches into the receiving mode.
After the slave has received the parent, a new individual is generated by mutating an exact copy of the parent and evaluating its fitness. Therefore the phenotype has to be encoded from the genotype which consists of the following steps:

1. decoding the object and target points
2. calculate the control point positions which fulfil the deformation specified by the object and target points
3. deforming the embedded shape according to the calculated control points

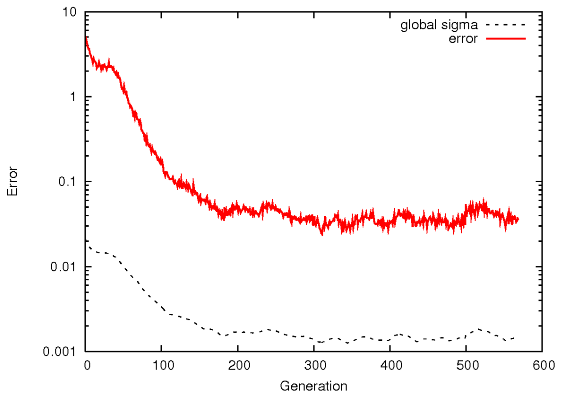
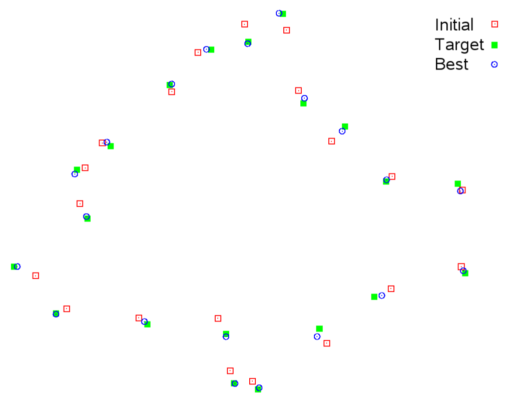


Figure 6.5: A high error in the control point calculation decreases the step size. The smaller target deformation in the next step enables to achieve better results in the control point calculation.

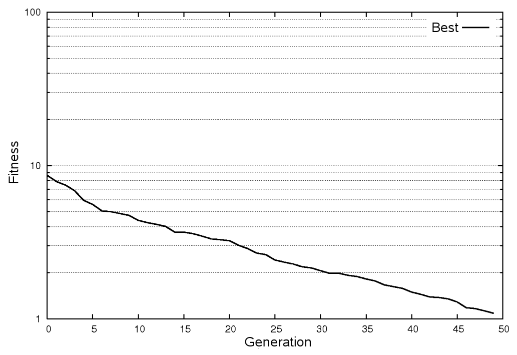
The object points and target deformations are used to initialise the calculation of the control points. Then the control points, which satisfy the object point deformation, are computed by the modified FFD approach. The resulting control volume is used to deform the whole design according to equation (1.1). When the deformed design is available the fitness is evaluated with the same function (3.1) as in the FFD approach.
Since it is not ensured that the calculated control points move the object points exactly to the target positions, the realised positions of the object points are coded back into the object variables. The distances between the initial object points and the object points deformed by the calculated control points, are computed in the Cartesian coordinate system and written into the chromosomes. Due to the encoding of the realised object point displacements, the self adaptation takes the realised variance into account. If the target points could not be matched, the realised variance is smaller as the expected. The global step size is reduced accordingly. As a consequence of the smaller step size the computation of the control lattice will be easier for the individuals of the next generation. Thus the error of the control lattice optimisation is limited by the self adaptation of the DMFFD-ES. Figure 6.5 illustrates this limitation.

6.2 Problems with the control point calculation

In the first test runs the optimisation of the control lattice was stopped as soon as a maximal number of generations was reached or the fitness value was below a threshold of 0.01. This results in a truncation of the DMFFD optimisation. If the changes in the object points become too small, it happens that already the initial



(a) The target deformations could not be achieved due to the limited number of generations.



(b) The fitness progress of the control point calculation. Using more generations would enable to find a better solution.

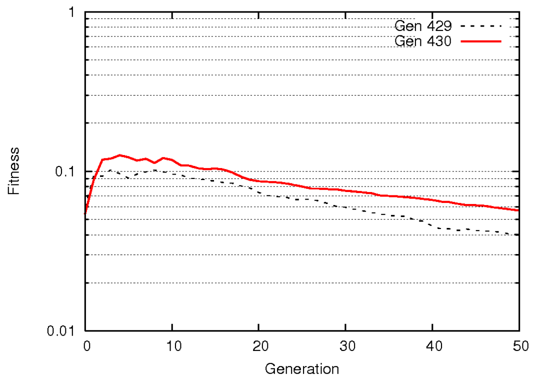
Figure 6.6: Example for the realised target deformation.

solution was better than the threshold. In such a case the offspring is equal to its parent because the control lattice is unchanged. The realised variance decreases and the strategy parameter adaptation reduces the step size. As a consequence in the next steps the probability that this problem appears again increases. In the end the step size becomes zero. Therefore in the later tests the threshold was set to zero, which signifies an exact solution.
For practical reasons the number of generations used for the computation of the control points is limited to 50. Such a maximal number is necessary to assure the termination. It cannot be guaranteed that a solution of the constrained optimisation exists or will be found by the ES.
The value of this number has to be chosen carefully. On the one hand a small number makes it impossible to realise large displacements and to match the target positions exactly. This reduces the realised variance of the population and therewith the step size of the DMFFD-ES. The reduced step size slows down the optimisation because more optimisation steps are required to obtain a large object point displacement. Figure 6.6 shows an example. The target points could not be matched because the number of generations is not sufficient to realise the deformation.
On the other hand a large number of generations increases the time for the control point calculation. As a consequence the time for one optimisation step of the DMFFD-ES is enlarged.
The choice of the maximal generation number depends on the cost of the fitness evaluation. The more time is required the more time should be spent for the control point calculation to reduce the number of fitness evaluations. This reduction can annul this time consumption required for the control point calculation.
Another problem occurs when the convex hull is just fulfilled and the difference between the previous and actual target points are very small. Then in the first steps of the control point calculation the fitness becomes worse (figure 6.7(a)).

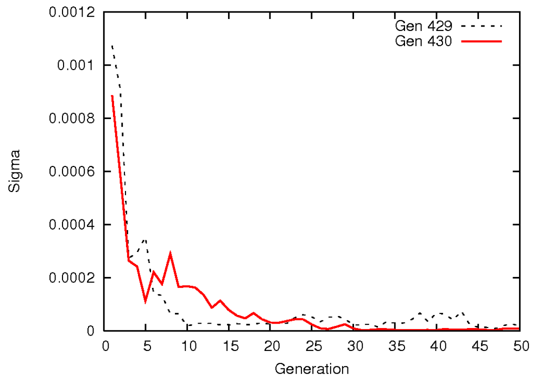
The reason could be that the initial strategy parameter is too high. In the previous control point calculation the control point positions were adjusted to the boundaries of the valid search space. This adjustment is realised with very small steps as the dotted lines show. The large changes in the first generations of the new calculation lead to a large violation of the convex hull constraint as figure 6.7(c) illustrates. After the correction of this violation the fitness becomes much worse.
In the optimisation depicted with the solid line it was not possible to make up this leeway. If the initial solution could not be improved within the limited number of generations, the initial control points are returned to the DMFFD-ES. As a consequence the offspring is equal to its parent. Due to the reduced variance, the step size of the DMFFD-ES decreases. In run DMD2 and ADMD1 this problem occurred several times. The optimisation was truncated due to a step size of almost zero (figure A.17(c) and A.18(c)).

6.3 Results of direct manipulation

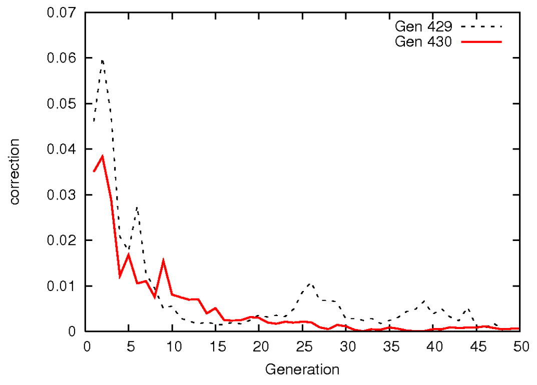
Two test runs for the dolphin problem have been accomplished. Both runs use a (3,16)-CMA ES, whereas the calculation of the control points uses a (1,16)-ES with only a global mutation strategy parameter (GSA). Run DMD1 uses 20 and DMD2 40 object points equidistant distributed over the circle. All object points can be moved in x and y direction. That means a reduction of the object parameters from 288 to 40 respectively 80 in comparison to the general FFD approach.
The comparison of the results of the DMFFD approach with the non adaptive FFD shows that the direct manipulation requires less fitness evaluations (figure 6.8). Run DMD2 requires 435 generations whereas run CD3 needs 843 to achieve the same fitness. This corresponds to a reduction of fitness evaluations from $16,860$ to $6,960$. The enhanced influence and the compacter representation lead to a reduction of about 58.7%.
On the other hand the fitness value achieved by the FFD approach was not realised by the direct manipulation. This can be explained with the lower degree of freedom used in the DMFFD runs. Comparing the results of DMD1 and DMD2 one can see that the increased number of object points leads to a clear better solution. Whereas run DMD1 achieves a fitness of 1.9593 test run DMD2 results in 1.0690. Furthermore the truncation of the optimisation explained in the previous section prevents better results.
Run DMD1 points out a further aspect of the direct manipulation approach. Figure A.16(a) depicts that not only the areas near an object point are optimised to match the target shape but also regions between object points which are far apart. In these areas the target is not matched as good as in the areas next to an object point but the shape is also adjusted to the target. Regarding the tail fin one can see that the centre is matched although there is no object point. This aspect is



(a) The fitness progress



(b) The progress of the global strategy parameter



(c) The length of the control point displacements caused by the correction algorithm

Figure 6.7: Two successive control point calculations are shown. The solid line depicts a run where the initial fitness could not be improved. In the first generations the fitness increases. In these generations the step size is clearly higher and larger corrections are required. The dotted lines show the data of the calculation right before.

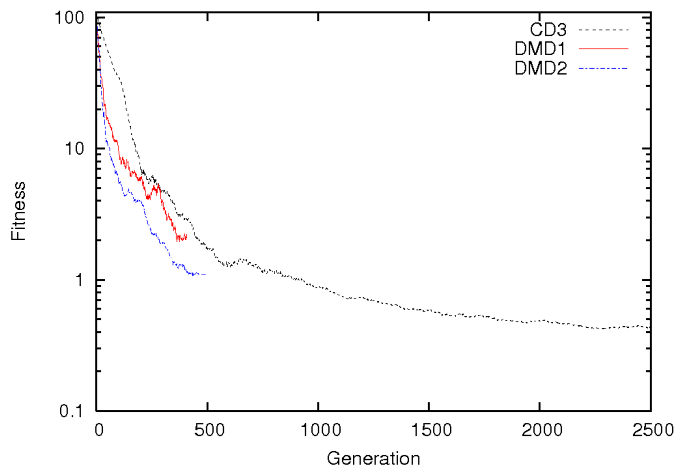


Figure 6.8: The fitness progress of the DMDFFD tests

	fitness	object variables	generations	evaluations
CD3	0.4212	288	2,267	45,340
DMD1	1.9593	40	363	5,808
DMD2	1.0690	80	435	6,960

Table 6.1: Results of direct manipulation test runs

more obvious in the results of the adaptive direct manipulation test run ADMD1 A.18. In this run the whole rear is nearly matched although there are only four object points. That indicates that also the control points, which do not influence the object points, are optimised to match the target, even if their calculation does not have any informations about the target shape. Since the phenotype of an individual is defined by its encoded control volume, the selection is mainly based on the control points and only secondary on the displacement of the object points which specify the movement of the control points. The control points are not only optimised to match the object points but also to match the target shape. This optimisation is much slower because it is not guided by object points.

6.4 Adaptation of direct manipulation

The direct manipulation approach has some limitations. As in the general FFD approach the possible deformations of a shape are determined by the control volume. In order to not exclude the optimal shape from the search space the control lattice still has to be created carefully. The sensitive parts of the design have to be found. As already mentioned in 5 this is difficult and requires a huge amount of problem

specific knowledge. In order to reduce the effort which is required to create an adequate control volume, an adaptive representation (ADMFFD) can be used which automatic increases the variability in promising areas.
The direct manipulation Free-Form Deformation can be adapted at two different parts:

- the control volume
- the set of object points

The control volume can be adapted during the calculation of the control points to match the specified object point deformation. This adaptation can be implemented as described in chapter 5. The FFD method used to calculate the control points is replaced by the developed AFFD approach. But several questions remain. One has to decide when the control lattice should be refined. One possibility could be to refine the lattice if the target points could not be matched. But refining the control lattice does not ensure that the error of the control point calculation decreases. A better object point matching can also be prevented by the convex hull constraint. Therefore this refinement criteria can lead to fine control lattices. As already mentioned such fine lattices make it difficult to achieve large deformations while keeping the convex hull constraint. Additional conditions are necessary to limit the number of control points.
An other question is where the control lattice should be refined. Since the calculation of the control points is a target matching problem, more information is available than in chapter 5 to select promising areas. There one has to proceed on the assumption that a CFD-simulation is used. One option is to use the already implemented sub-population approach with randomly selected insertion points. But the use of several sub-populations slows down the control point computation. To reduce the computational effort, a good choice could be to insert the new control points near the object point which has the largest distance to its target position. But again if the convex hull constraint impedes a better matching, the insertion will complicate the matching.
The adaptation of the control volume in the direct manipulation is not discussed further in this diploma thesis. But the results of the adaptive general FFD approach indicates that an adaptation will improve the optimisation in several aspects. In the general approach the adaptive control volume could speed up the optimisation and also achieve better results. These improvements should also be possible in the direct manipulation.
The direct manipulation of FFD can also be adapted by modifying the set of object points. New points can be added or existing points removed. In both cases the modifications of the representation are neutral mutations because the control points remain unchanged. The stored control volume exactly matches the target positions and thus a new control point calculation would not change the control lattice.

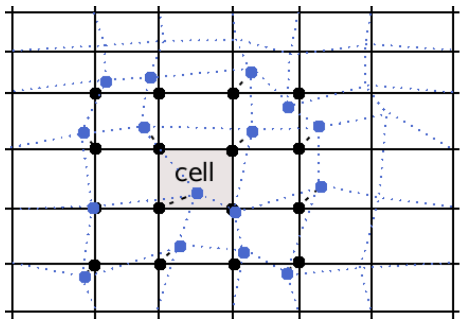


Figure 6.9: A cell and the corresponding control points are depicted. The distances between the control points at the last refinement (black points) and their current position (blue points) are totalises.

As for the adaptation of the control volume one has to decide when and where the representation should be changed. In the executed tests the set of object points is modified after a fixed number of generation by adding a new object point. The removal of points is not used. For the adaptation of the object point set the idea of sub-populations is implemented again. Due to runtime reasons in all tests a single sub-population is used.
In this thesis the influence of a changed object point set is analysed. Therefore tests which use different strategies for the selection of a new object point are used and compared with the non adaptive direct manipulation. The first strategy always selects the design point with the largest distance to the target point set. This strategy is only available for the given target matching problem. It should be tested if the adaptation of the object point set can improve the performance of the direct manipulation.
Furthermore a strategy is used which selects a random point of the design.
Based on the analysis of the direct manipulation a heuristic (LCD) for the object point selection is developed and compared with a random strategy. The heuristic is based on the control volume. As the results of the DMFFD approach show the control points are also adjusted to the dolphin in areas where no object point lies. That indicates that the control points are optimised indirect to match the target. The idea is that in areas where the deformation is changed much, further improvements are possible. In such areas the optimisation has not converged. The control points are still moved to find their optimal placement. To speed up the convergence in this area, an object point of this region is added. The new object point should enable a directed search in this region. In the implementation a cell of the control lattice is used to specify the area. All control points, which have an effect on this cell, are used to measure the deformation of the area. Their total displacement since the last refinement is calculated. The new object point is selected randomly from the cell with the highest value. Cells which already contain an object point are neglected. Figure 6.9 shows an example.
To estimate this heuristic, it is compared with a method which selects the FFD cell

randomly. Again only cells which do not already contain an object point can be selected.

6.5 Results of adaptive direct manipulation

To analyse the adaptive direct manipulation FFD, two test runs are executed. In both runs the set of object points is increased. In run ADMD1 the representation was modified every 15th generation by adding the design point with the largest distance to the target shape. In run ADMD2 a random design point is added every 10th generation. The control volume used in both tests is the same as in the general FFD approach (figure 3.7) and is kept fix. Both tests use a (1,16)-CMA-ES and start with 5 object points evenly distributed over the whole shape. As initial shape again the circle is used.
For the control point computation a (1,16)-ES with global strategy parameter adaptation (GSA) is applied. In run ADMD2 the computation of the control points was stopped if the error is below a threshold of 0.01.
The fitness progress in 6.10 shows that the rising number of object points leads to a faster convergence. The number of fitness evaluations could be reduced from $5,520$ in run DMD2 to $3,520$ in run ADMD2. That corresponds to a ratio of about 36%. This reduction can be explained with the smaller number of object parameters. The reduced search space speeds up the optimisation. But the sole adaptation of the object point set does not lead to a search in sub spaces as in the adaptive FFD approach. The deformed design is determined by the control lattice. Its resolution is already in the first generation fine. It is not possible to produce a roughly approximation of the optimum which is refined more and more during the optimisation. However, more investigations are necessary to analyse the reasons for this reduction more detailed.
The achieved fitness of ADMD2 is between the one of the fixed test runs, whereas the fitness of ADMD1 is slightly better. But it has to be mentioned again that the heuristic used in ADMD1 is based on information of the target. For real problems this heuristic is not available. The truncation of the optimisation, caused by the problems already described in section 6.2, prevents better results. Especially in run ADMD2, which uses the fitness threshold in the control point calculation, better results will be possible. The results of the adaptive direct manipulation test runs are shown in A.18 and A.19. The dates are summarised in table 6.2.
These first results indicate that an adaptive representation is able to improve the direct manipulation representation. Using a more sophisticated heuristic to select a new object point instead using a random one will improve the results further. Also the combination of an adaptive control volume and a variable set of object points should be subject of future research. The results of the adaptive general FFD approach and of the adaptive direct manipulation show that an adaptive approach can

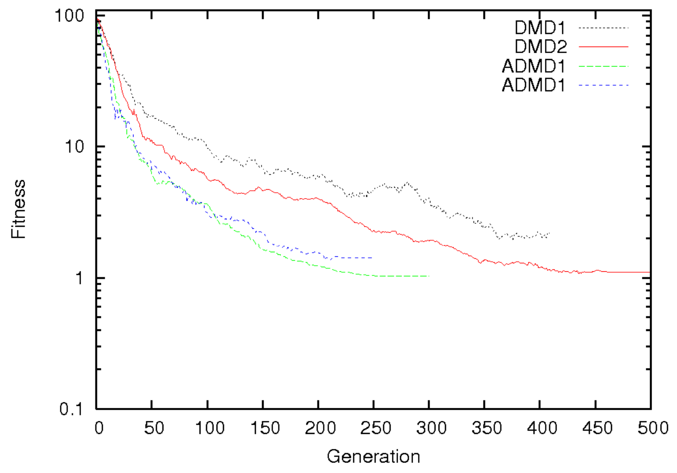


Figure 6.10: The fitness progress of the adaptive direct manipulation test runs.

	fitness	object variables	generations	evaluations
DMD1	1.9593	40	363	5,808
DMD2	1.0690	80	435	6,960
ADMD1	1.0317	10 . . . 54	273	4,384
ADMD2	1.3616	10 . . . 56	219	3,520

Table 6.2: Results of adaptive direct manipulation test runs

improve the performance of the optimisation. If both are combined, the optimisation should be improved additionally.
For the comparison of the developed strategy for object point selection with the random strategy the initial shape is changed. Starting with a circle would not allow any conclusions about the heuristic because in every area of the shape large deformations are required. Therefore as initial shape a dolphin without fins is used (figure 6.11). In this case the developed heuristic should be inserted the object points especially in the regions of the fins. Here the largest deformations are required.
However, the results of the test runs do not demonstrate this assumptions. The object points are distributed over the whole design. In the areas of the fins firstly in the 18th insertion operation an object point was placed.
The problem is that in the first generations the already adapted areas are modified, too. To achieve the initial good solution, these areas have to be adjusted again to the target. Thus similar to the circle problem in every area of the shape object points are required.
As a consequence the comparison of the heuristic and the random strategy does not allow many conclusion. Figure 6.12 illustrates the fitness progress. For the tested problem both strategies perform very similar.

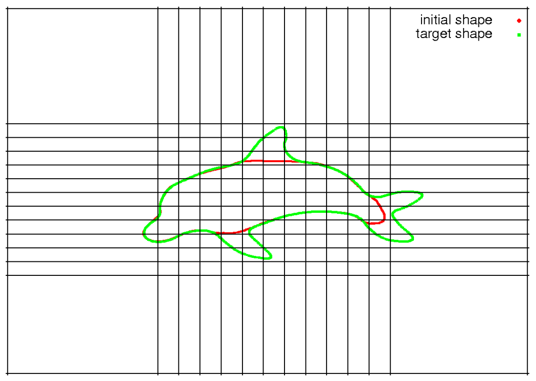


Figure 6.11: The initial shape used for the comparison of both object point selection strategies.

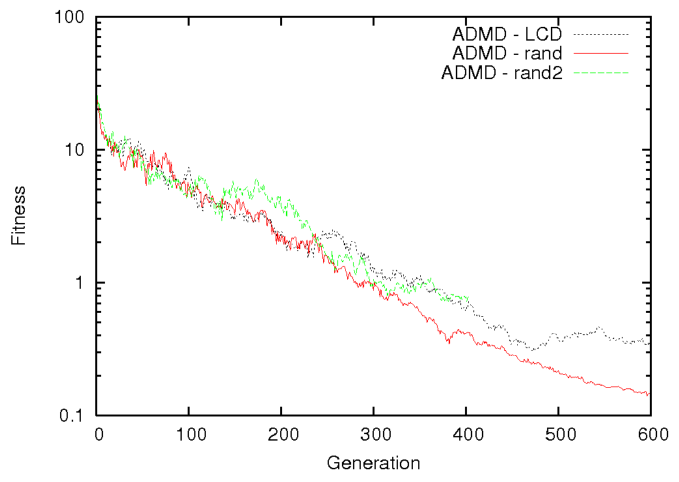


Figure 6.12: The fitness progress of both object point selection strategies.

Chapter 7

Conclusion and outlook

In this diploma thesis an adaptive Free-Form Deformation and a direct manipulation of Free-Form Deformation were used in an Evolution Strategy to represent a design. Both methods represent deformations of a control volume which contains the initial design. In the case of the FFD method these deformations are specified by the coordinates of control points. Instead of using the control points directly to represent the design in the direct manipulation FFD the control point coordinates are defined by a set of object points and their target positions.
Since the Free-Form Deformation specifies deformations, it can be used to deform the computational grid required for the fitness evaluation. In the case of complex designs a costly and eventually manual re-meshing procedure can be omitted. To keep the structural composition of the grid after the deformation, the convex hull constraint was introduced to restrict the deformation. The analysis of the problems occurring in this constraint optimisation resulted in an algorithm that repairs the control lattice to satisfy the constraint. The results of the tests have shown that the developed correction algorithm reduces the number of fitness evaluations in comparison to penalty methods.
The major goal of the adaptive representation was to reduce the dependence on the initial representation created by designers. The adaptation should allow to find new and unexpected solutions which are not codeable by the initial representation. The implemented approach which uses several sub-populations with randomly refined representations achieves this goal. The FFD control volume was automatically adapted to the problem but at the expense of fitness evaluations. For costly fitness evaluations instead the random strategy with several populations a single heuristic should be used. A test of a simple heuristic has shown that the convergence of the optimisation and the quality of the solution could be improved even if the representation was not adjusted to the specific problem. The use of more sophisticated heuristics could probably improve the performance further.
Due to the limited refinement possibilities of the B-spline basis functions and the

insufficient definition of the refinement criteria, the approach did not create compact representations. Conditions for the detection of a suitable point in time of a refinement have to be developed to avoid unneeded variability.
The FFD representation has also some limitations. The influence of a control point on the design depends on its distance to the embedded design. To reduce the problem of placing the control points such that their influence is maximised, a direct manipulation of FFD was introduced. It could be shown that the direct manipulation increases the influence on the design and thus reduces the number of fitness evaluations.
To maintain the convex hull constraint the computation of the control points had to be changed. Instead of the pseudoinverse approach the previous implemented general FFD Evolution Strategy was used to solve the constraint target matching problem. The influence of this computation on the direct manipulation approach was analysed.
Possibilities of an adaptive direct manipulation were discussed. The first tests have shown that the enlargement of the object point set can reduce the number of fitness evaluations. However, more investigations are necessary to analyse the behaviour of the adaptive direct manipulation FFD representation. Especially the combination of an adaptive set of object points and an adaptive control volume seems very promising and should be subject of further research.

Bibliography

[1] M. Andreoli, A. Janka, and J.A. Désidéri. Free-form deformation parameterization for multilevel 3d shape optimization in aerodynamics. Research Report 5019, 2003.

[2] C. Coello. Theoretical and numerical constraint handling techniques used with evolutionary algorithms: A survey of the state of the art, 2002.

[3] Sabine Coquillart. Extended free-form deformation: a sculpturing tool for 3d geometric modeling. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, pages 187–196, New York, NY, USA, 1990. ACM Press.

[4] Jean-Antoine Désidéri and Ales Janka. Multilevel shape parameterization for aerodynamic opimization - application to drag and noise reduction of transonic/supersonic business jet. In *European Congress on Computational Methods in Applied Sciences and Engineering ECCOMAS 2004*, 2004.

[5] James E. Gain and Neil A. Dodgson. Preventing self-intersection under free-form deformation. *IEEE Transactions on Visualization and Computer Graphics*, 7(4):289–298, 2001.

[6] Steffen Goerzig. Cppvm - a c++ interface to pvm. version 1.5.0.

[7] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[8] Martina Hasenjäger, Bernhard Sendhoff, Toyotaka Sonoda, and Toshiyuki Arima. Three dimensional aerodynamic optimization for an ultra-low aspect ratio transonic turbine stator blade. In *Proceedings of the ASME Turbo Expo*, 2005. ASME Paper No. GT2005-68680.

[9] William M. Hsu, John F. Hughes, and Henry Kaufman. Direct manipulation of free-form deformations. *Computer Graphics*, 26(2):177–184, 1992.

[10] Stefan Menzel, Markus Olhofer, and Bernhard Sendhoff. Application of free form deformation techniques in evolutionary design optimisation. In *6th World Congress on Structural and Multidisciplinary Optimization*, 2005.

[11] Stefan Menzel, Markus Olhofer, and Bernhard Sendhoff. Evolutionary design optimization of complex systems integrating fluent for parallel flow evaluation. In *In European Automotive (CFD) Conferemce (EACC), 2005.*, 2005.

[12] Stefan Menzel, Markus Olhofer, and Bernhard Sendhoff. Direct manipulation of free form deformations in evolutionary design optimisation. In *9th PPSN International Conference on Parallel Problem Solving from Nature, accepted, to be published*, 2006. accepted, to be published.

[13] Joseph O'Rourke. *Computational geometry in C*. Cambridge University Press, New York, NY, USA, 1994.

[14] Ernest C. Perry and Steven E. Benzley. Shape optimization of fluid flow systems. In *ASME FEDSM'00. 2000 ASME Fluids Engineering Summer Conference, Boston, Massachusetts, 2000*, 2000.

[15] Ingo Rechenberg. *Evolutionsstrategie '94*. Werkstatt Bionik und Evolutionstechnik. Frommann-Holzboog, 1994.

[16] Jamshid A. Samareh. A novel shape parameterization approach. Technical Memorandum NASA/TM-1999-209116, National Aeronautics and Space Administration, 1999.

[17] Jamshid A. Samareh. A survey of shape parameterization techniques. Technical report, 1999.

[18] Jamshid A. Samareh. Aerodynamic shape optimization based on free-form deformation. In *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2004.

[19] Hans-Paul Schwefel. *Evolution and Optimum Seeking*. Werkstatt Bionik und Evolutionstechnik. Wiley & Sons, 1995.

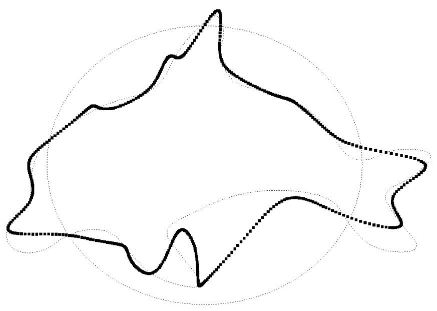
[20] Thomas W. Sederberg and Scott R. Parry. Free-form deformation of solid geometric models. In *SIGGRAPH '86: Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 151–160, New York, NY, USA, 1986. ACM Press.

[21] Shark - a modular c++ libraries for the design and optimization of adaptive systems. version 1.4.2.

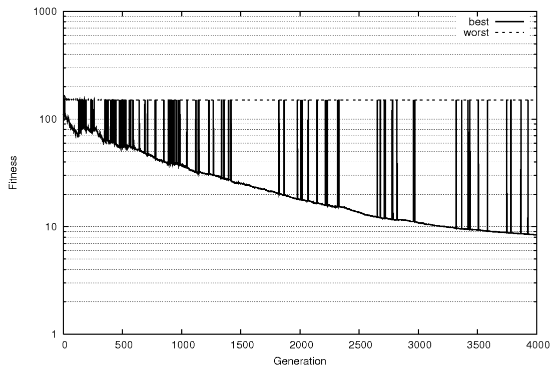
Appendix A

Results

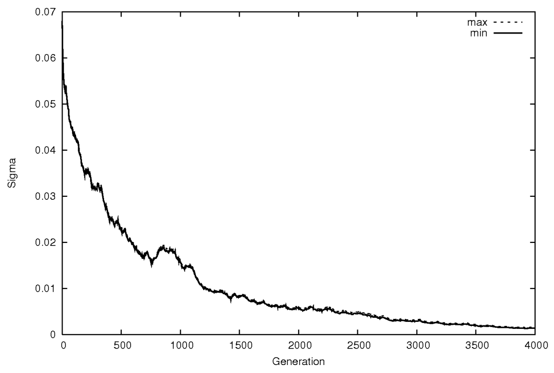
A.1 Non adaptive approach



(a) The resulting shape

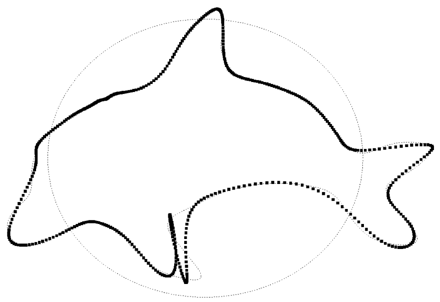


(b) Fitness value of the best and worst offspring

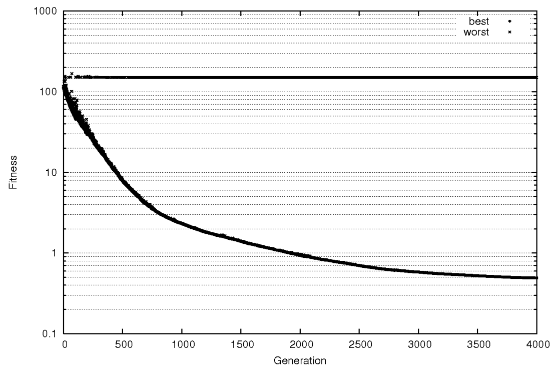


(c) Maximal and minimal strategy parameter of the CMA

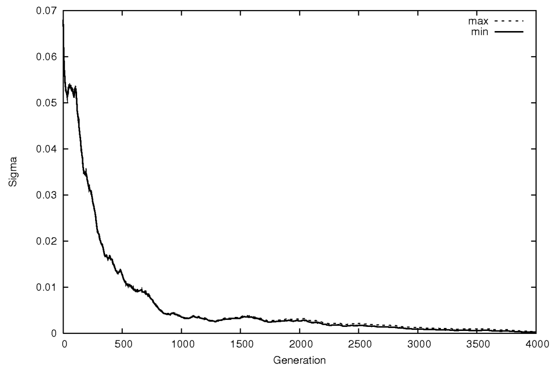
Figure A.1: Results of run PD1. The fitness function (4.1) with a fixed offset for invalid solutions is used. The convex hull constraint is fulfilled.



(a) The resulting shape

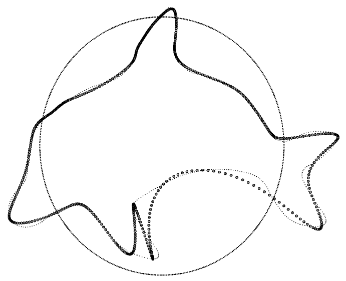


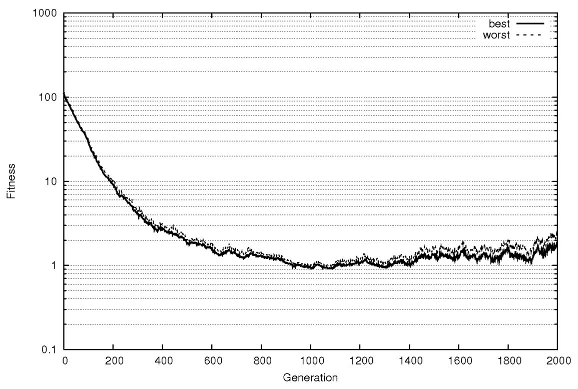
(b) Fitness value of the best and worst offspring



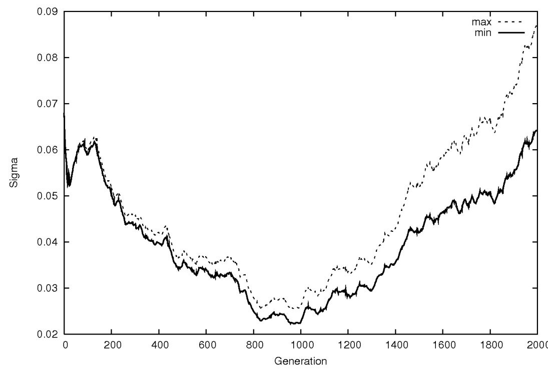
(c) Maximal and minimal strategy parameter of the CMA

Figure A.2: Results of run PD2. The fitness function (4.2) with an approximation of the fitness is used. The convex hull constraint is not fulfilled.

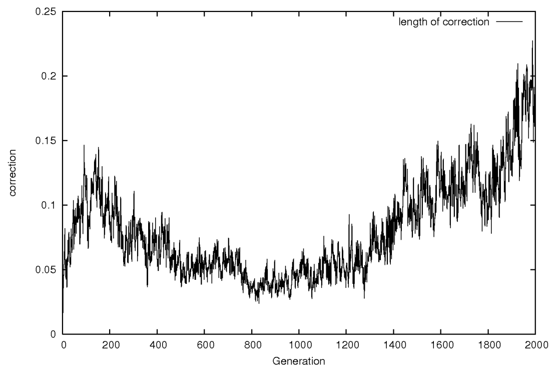

(a) The resulting shape.



(b) Fitness value of the best and worst offspring.

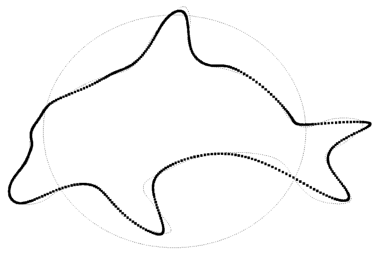


(c) Maximal and minimal strategy parameter of the CMA.

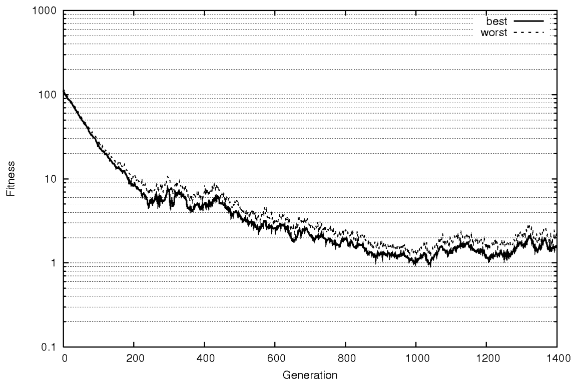


(d) Length of the displacements by the correction algorithm.

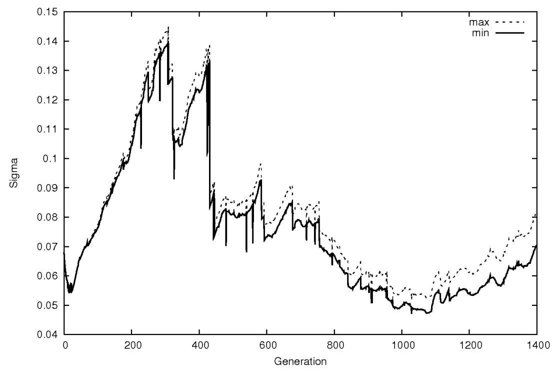
Figure A.3: Result of run CD1. All invalid solutions are corrected. The self adaptation is done by the CMA A methode. The global strategy parameter is reduced according to the length of the required correction. The optimisation does not converge.



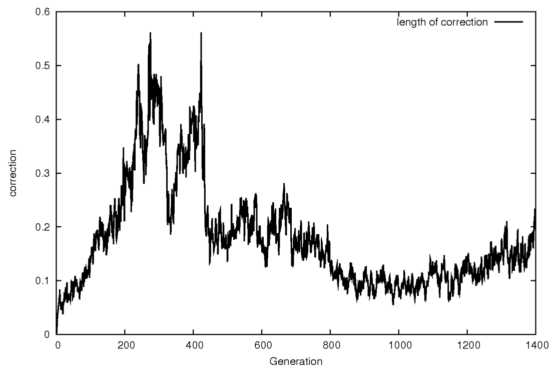
(a) The resulting shape. The local optimum at the chest fin is overcame.



(b) Fitness value of the best and worst offspring. The fitness value wavers between one and two.

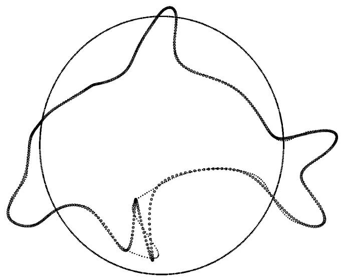


(c) Maximal and minimal strategy parameter of the CMA. The discret reduction is visible.

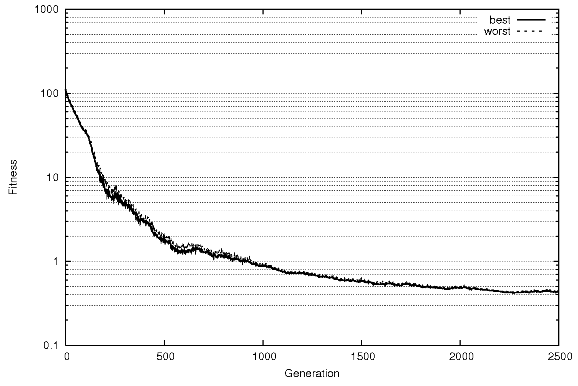


(d) Length of the displacements by the correction algorithm. If the correction becomes too large, σ^2 is reduced.

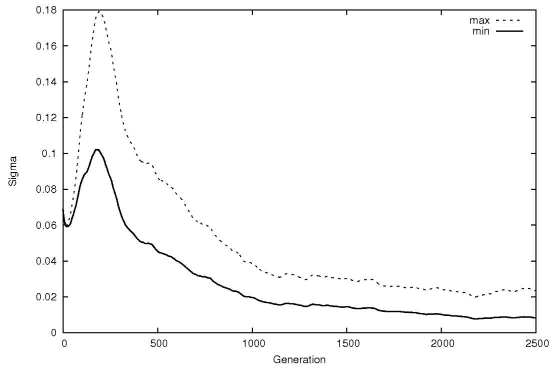
Figure A.4: Result of run CD2. Invalid solutions are corrected. The global strategy parameter is reduced only if the correction algorithm fails. As the step size indicates the optimisation does not converge.

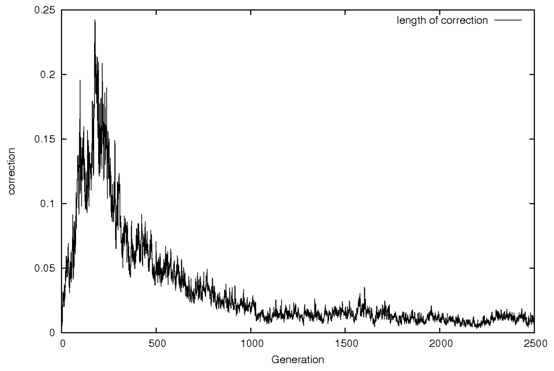


(a) The resulting shape.



(b) Fitness value of the best and worst offspring.



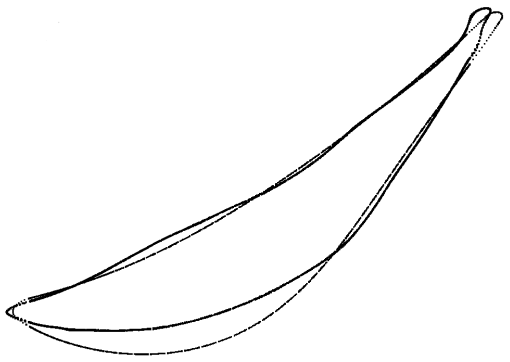


(c) Maximal and minimal strategy parameter of the CMA.

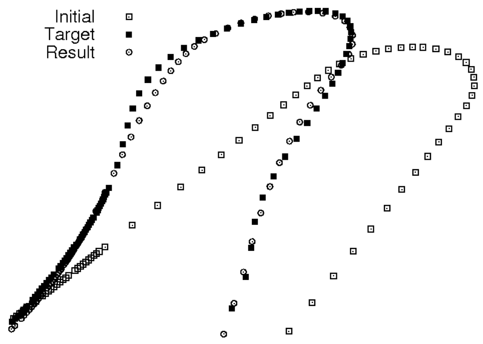
(d) Length of the displacements by the correction algorithm.

Figure A.5: Result of run CD3. The strategy parameters are adapted only by the self adaptation. As the step size shows the optimisation converges.

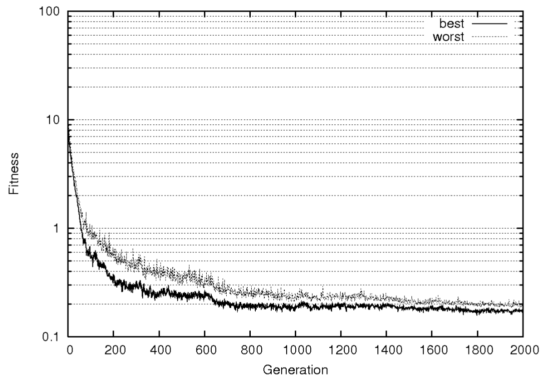
A.2 Turbine blade



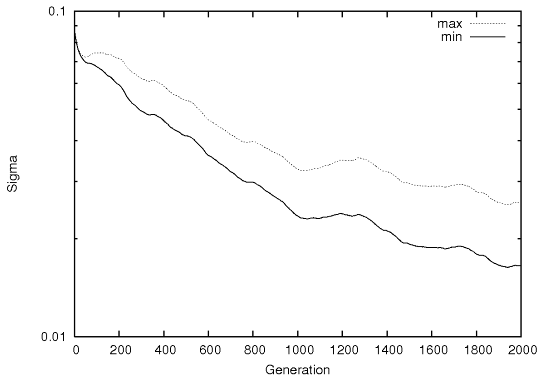
(a) Shape with smallest fitness.



(b) Trailing edge of the shape

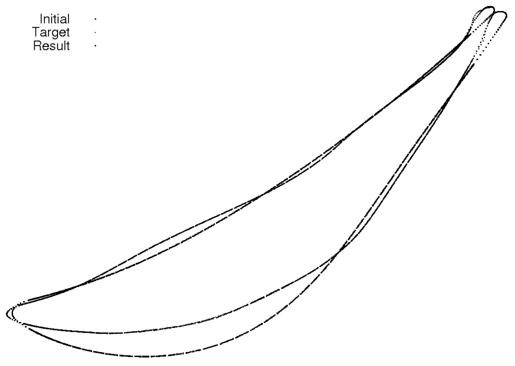


(c) Fitness progress

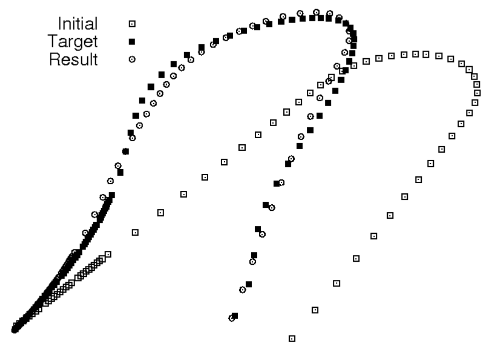


(d) Minimal and maximal strategy parameter

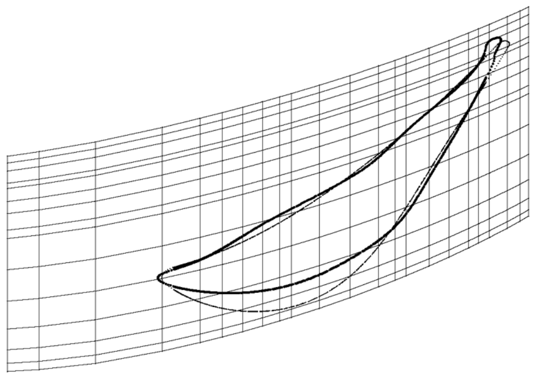
Figure A.6: The result and progress of test run CB1.



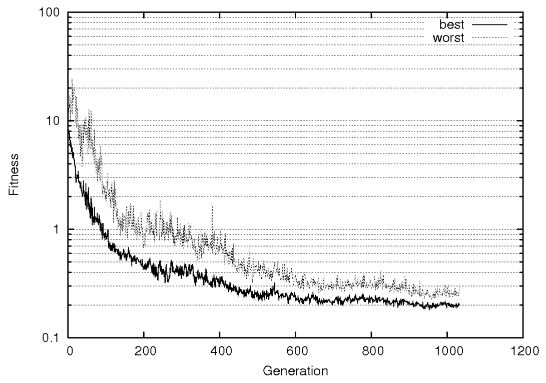
(a) Shape with smallest fitness.



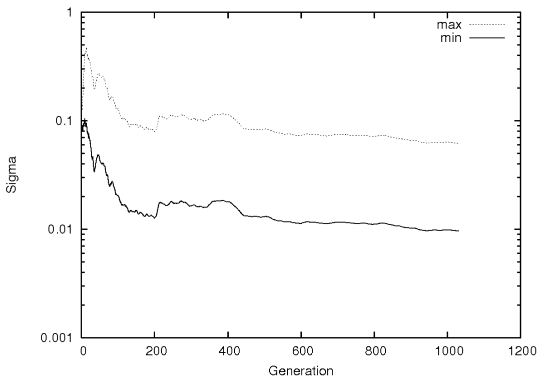
(b) Trailing edge of the shape



(c) Resulting undeformed control lattice



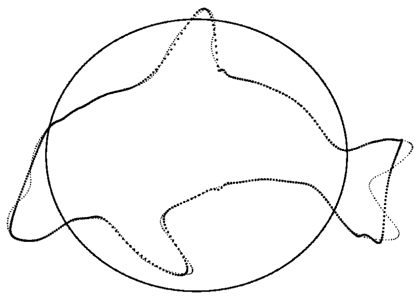
(d) Fitness progress



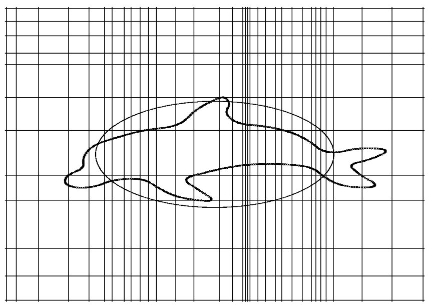
(e) Minimal and maximal strategy parameter

Figure A.7: The result and progress of test run AB1.

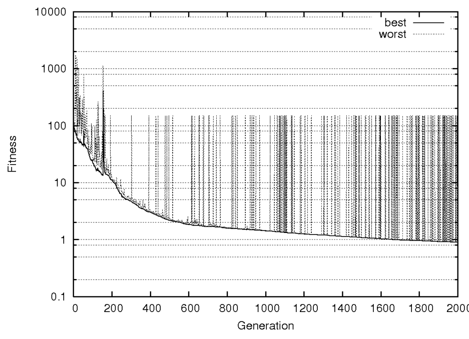
A.3 Adaptive approach



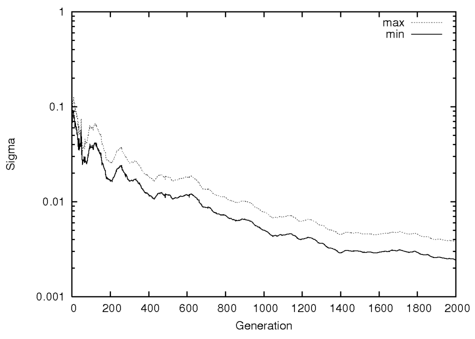
(a) Shape with smallest fitness.



(b) Resulting undeformed control lattice

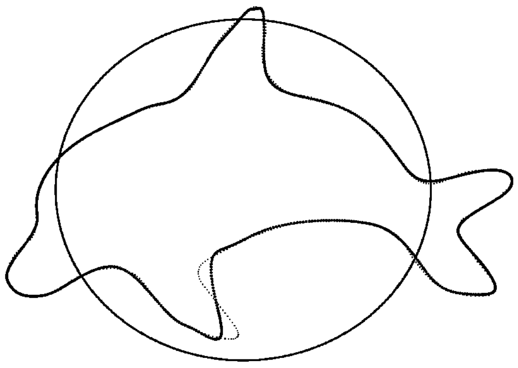


(c) Fitness progress

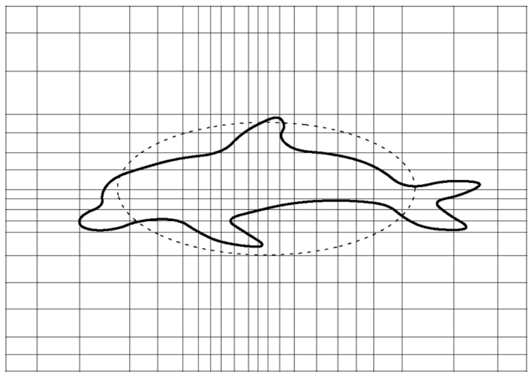


(d) Minimal and maximal strategy parameter

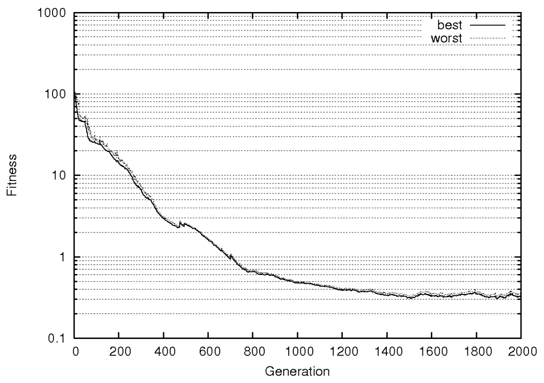
Figure A.8: The result and progress of test adaptive penalty approach. The selection of the sub-population is based on the absolute fitness value.



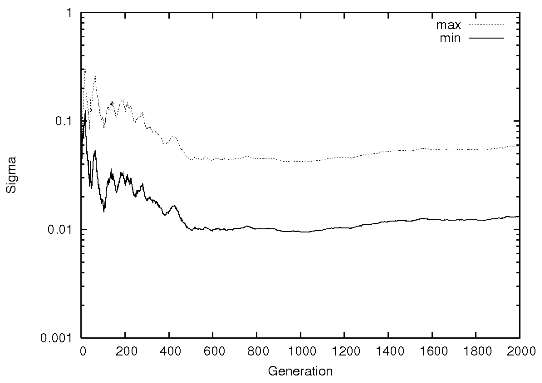
(a) Shape with smallest fitness.



(b) Resulting undeformed control lattice

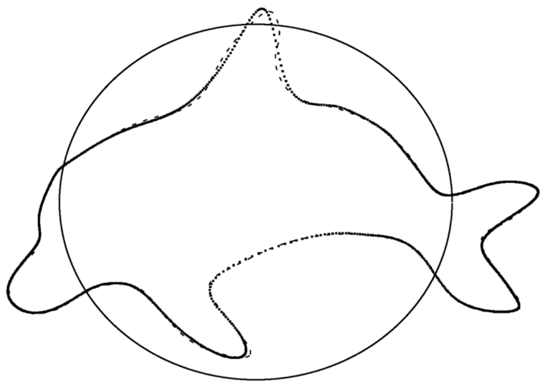


(c) Fitness progress

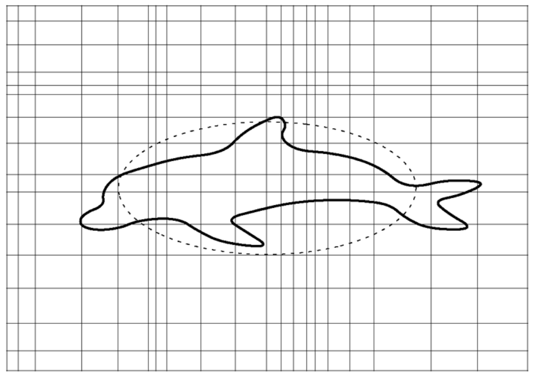


(d) Minimal and maximal strategy parameter

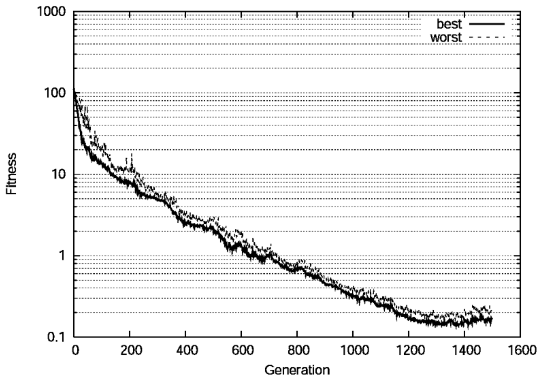
Figure A.9: The result and progress of test run AD1.



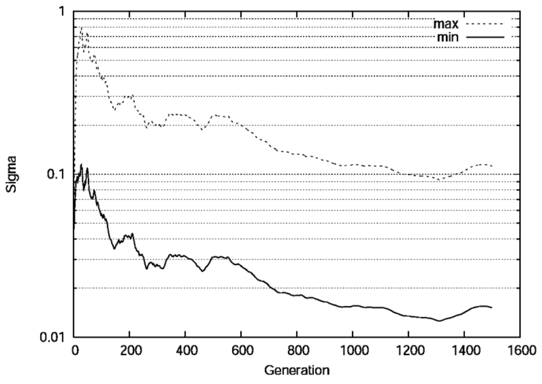
(a) Shape with smallest fitness.



(b) Resulting undeformed control lattice

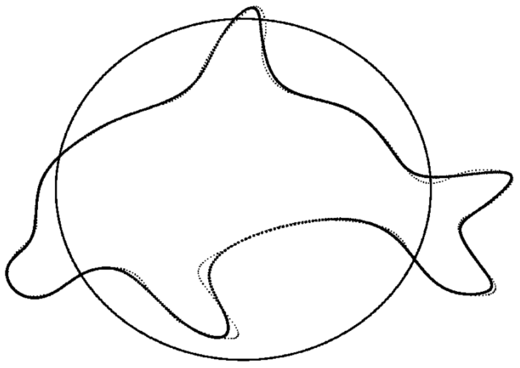


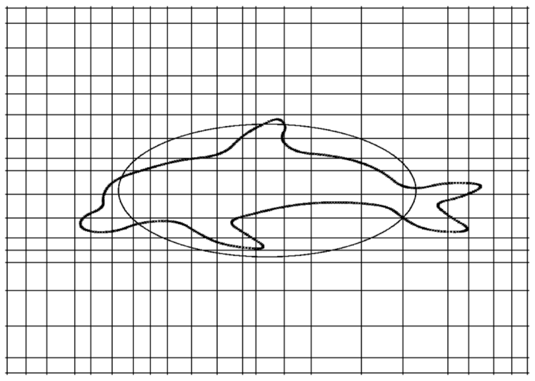
(c) Fitness progress

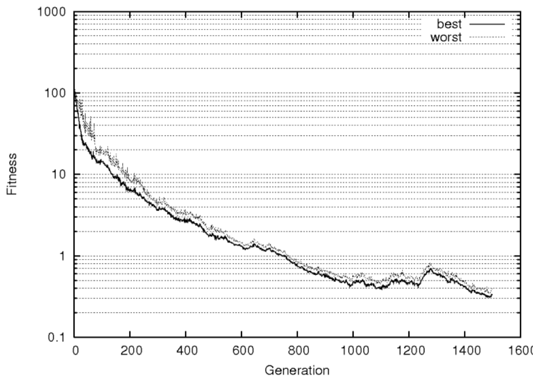


(d) Minimal and maximal strategy parameter

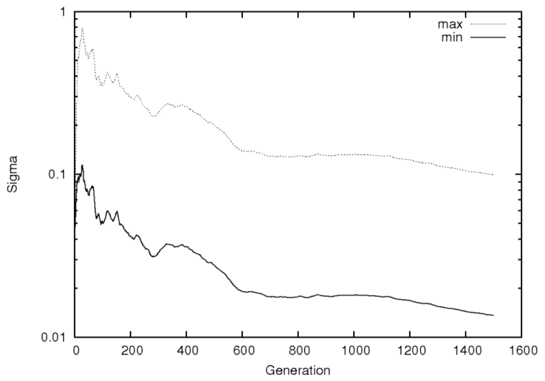
Figure A.10: The result and progress of test run AD2.


(a) Shape with smallest fitness.


(b) Resulting undeformed control lattice

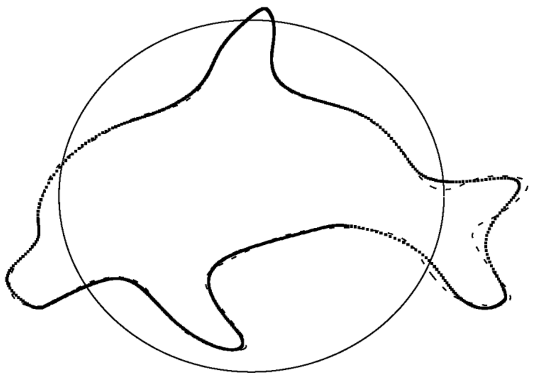


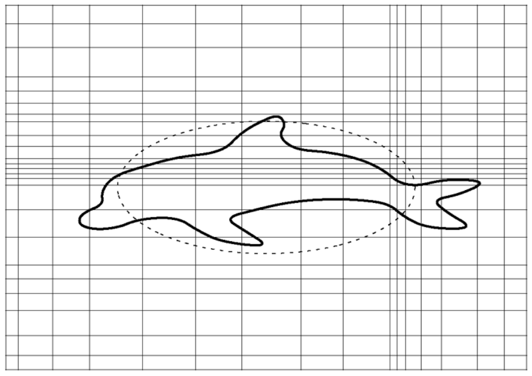
(c) Fitness progress

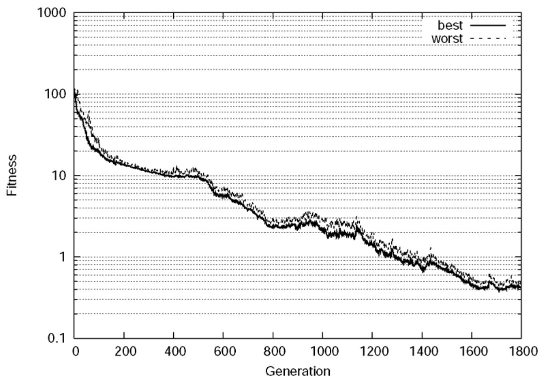


(d) Minimal and maximal strategy parameter

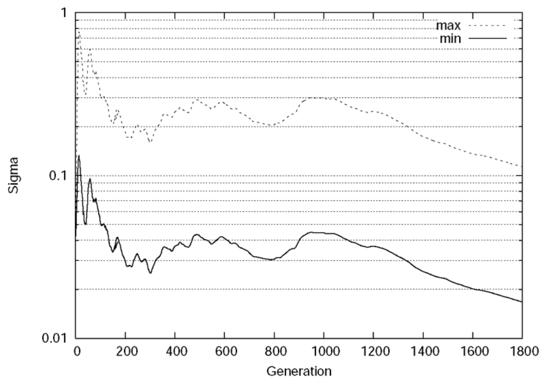
Figure A.11: The result and progress of test run AD3.


(a) Shape with smallest fitness.


(b) Resulting undeformed control lattice

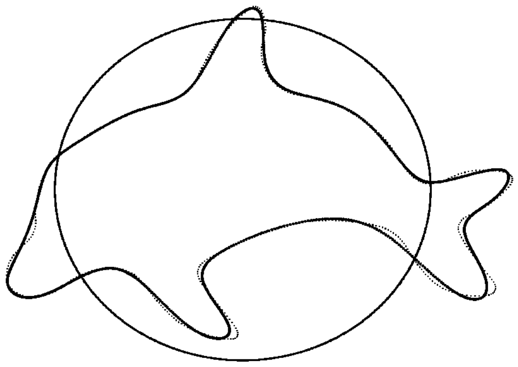


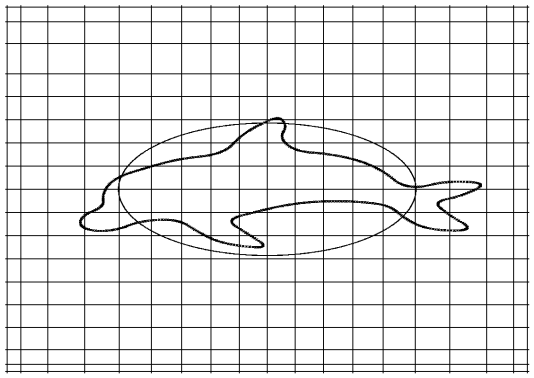
(c) Fitness progress

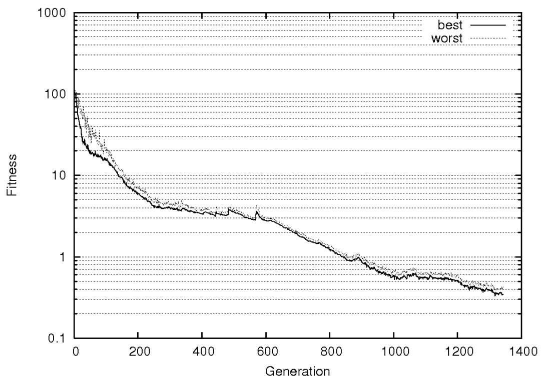


(d) Minimal and maximal strategy parameter

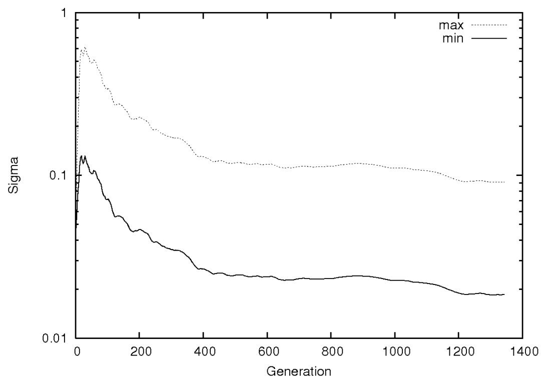
Figure A.12: The result and progress of test run AD4.


(a) Shape with smallest fitness.


(b) Resulting undeformed control lattice

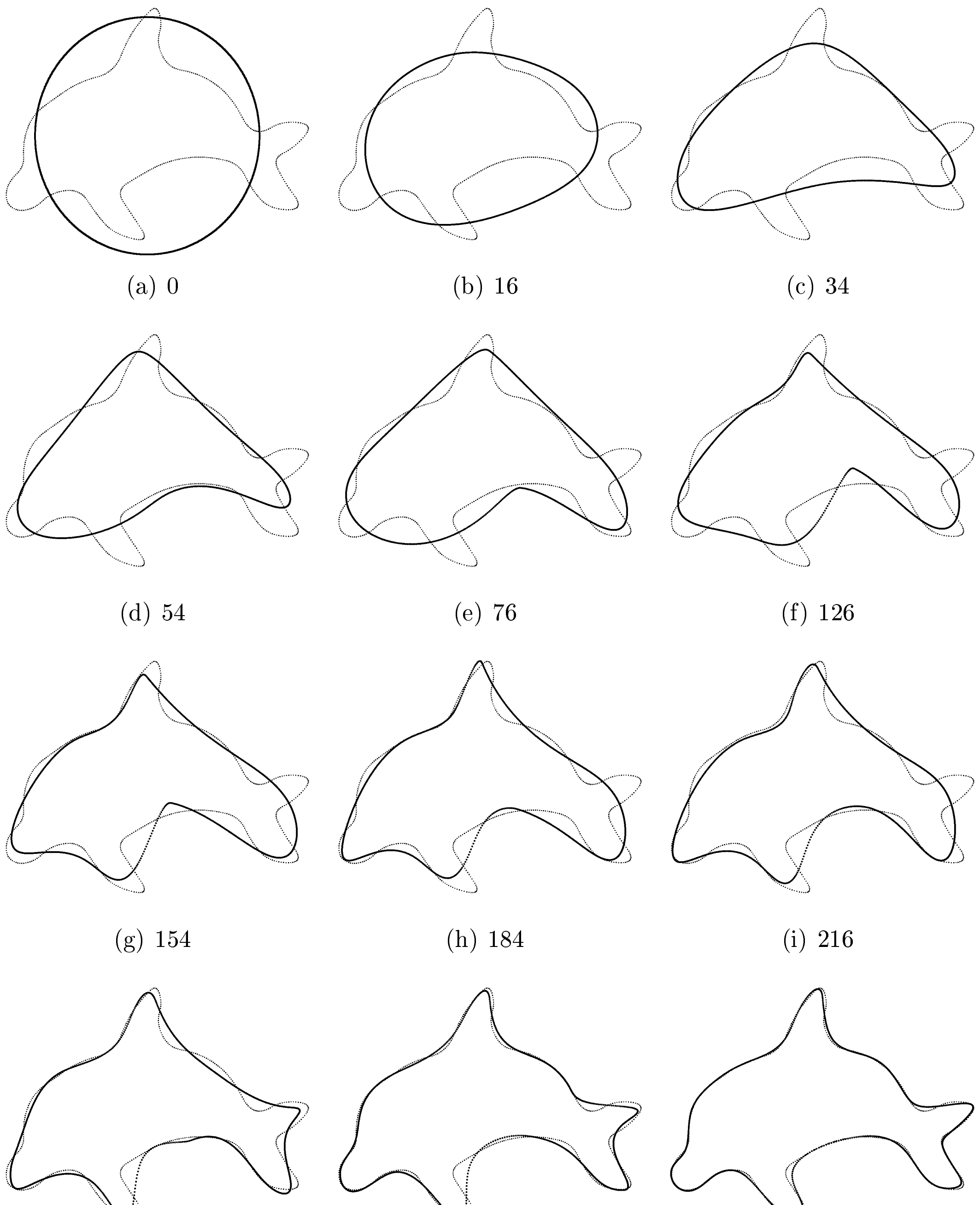


(c) Fitness progress



(d) Minimal and maximal strategy parameter

Figure A.13: The result and progress of test run AD5.



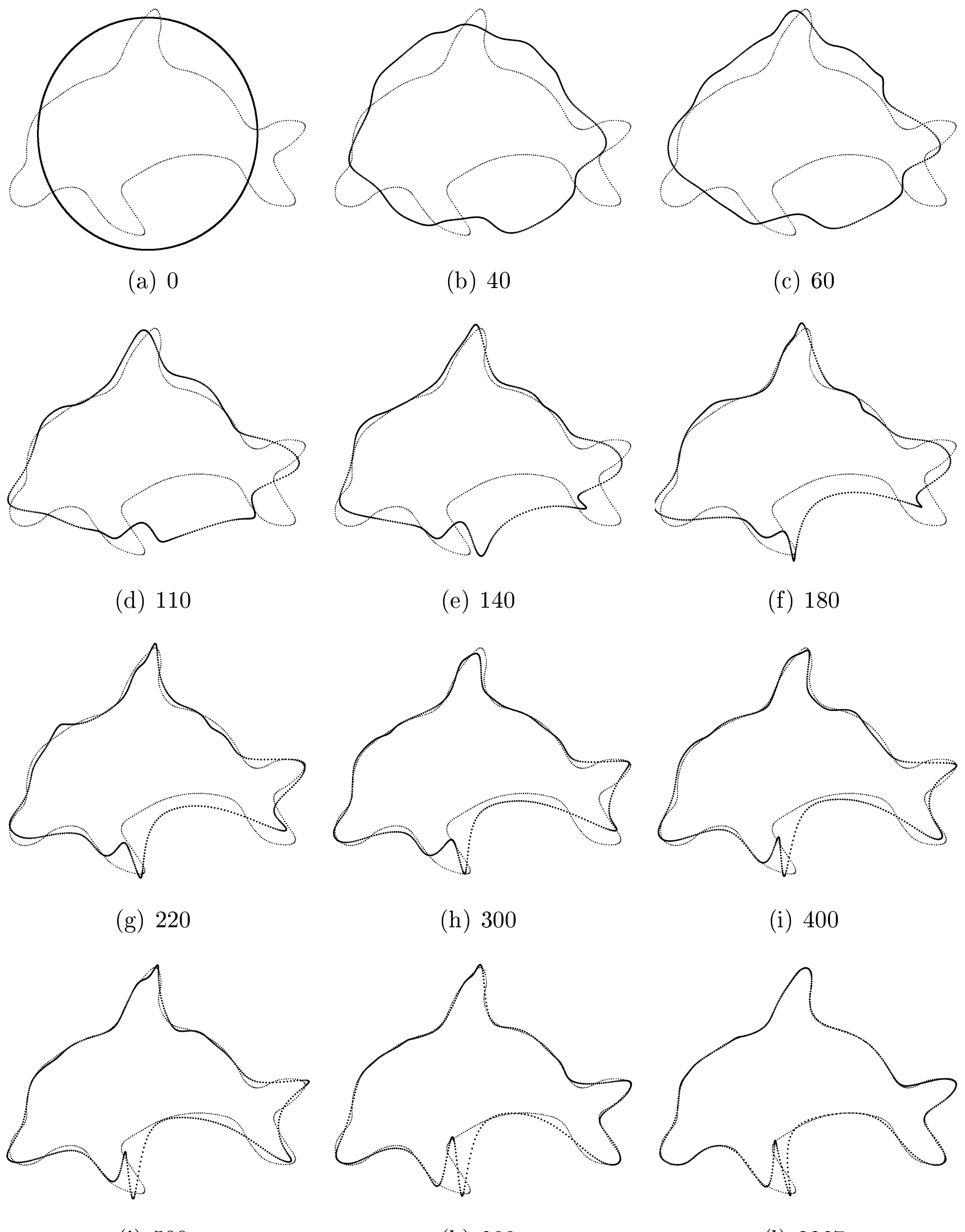
(a) 0 (b) 16 (c) 34

(d) 54 (e) 76 (f) 126

(g) 154 (h) 184 (i) 216

(j) 406 (k) 700 (l) 1492

Figure A.14: The progress of the shape deformation of run AD3. The subtitle shows the generation of the shape.



(a) 0 (b) 40 (c) 60

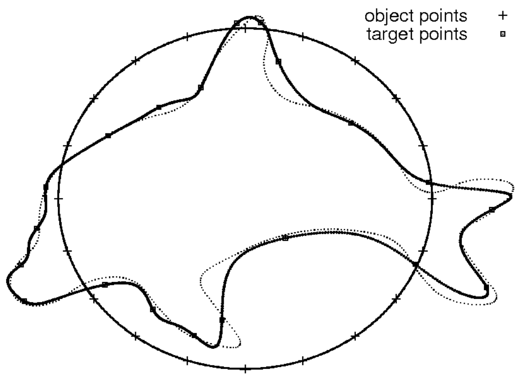
(d) 110 (e) 140 (f) 180

(g) 220 (h) 300 (i) 400

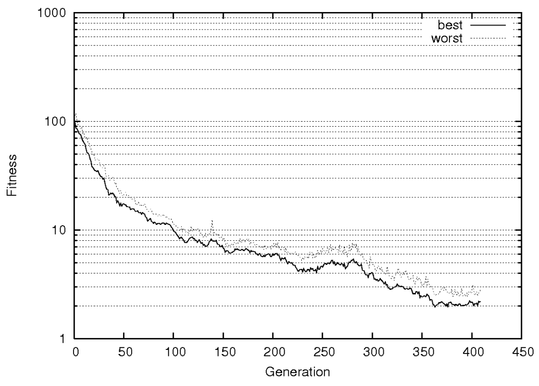
(j) 500 (k) 900 (l) 2267

Figure A.15: The progress of the shape deformation of run CD3. The subtitle shows the generation of the shape.

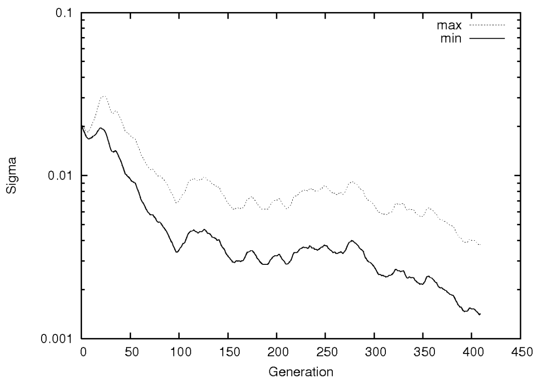
A.4 Direct manipulation approach



(a) Shape with smallest fitness. The object and target points are also displayed.

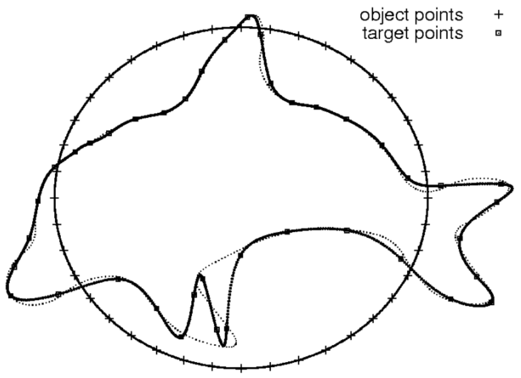


(b) Fitness progress

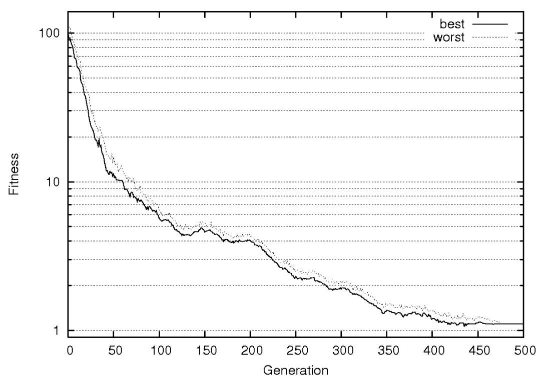


(c) Minimal and maximal strategy parameter

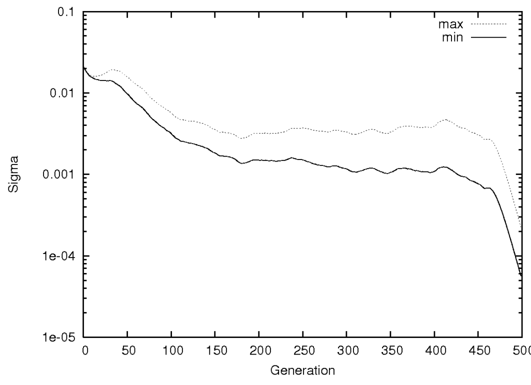
Figure A.16: The result and progress of test run DMD1.



(a) Shape with smallest fitness. The object and target points are also displayed.



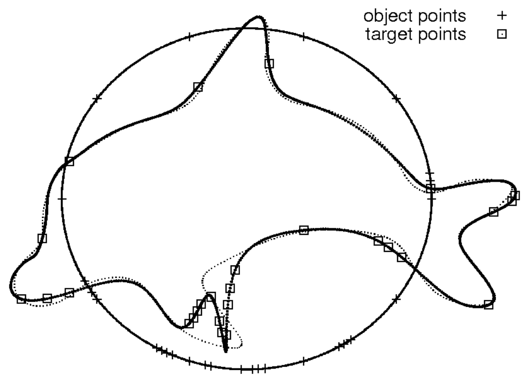
(b) Fitness progress



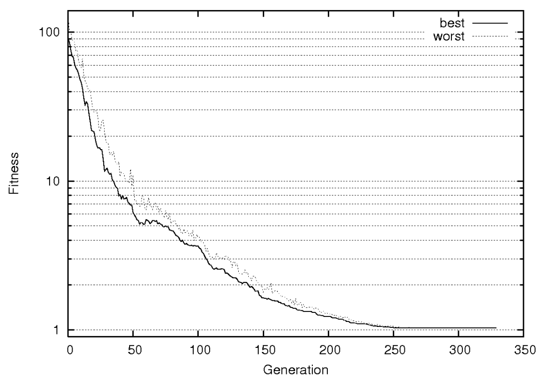
(c) Minimal and maximal strategy parameter

Figure A.17: The result and progress of test run DMD2.

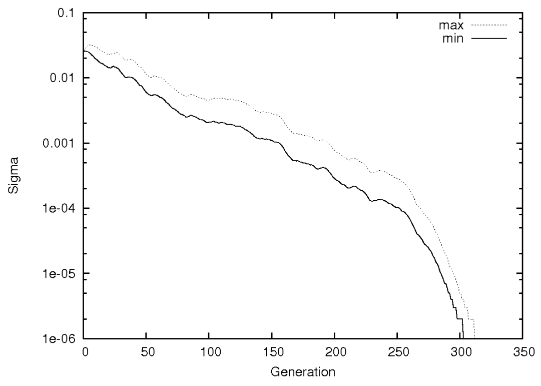
A.5 Adaptive direct manipulation approach



(a) Shape with smallest fitness. The object and target points are also displayed.

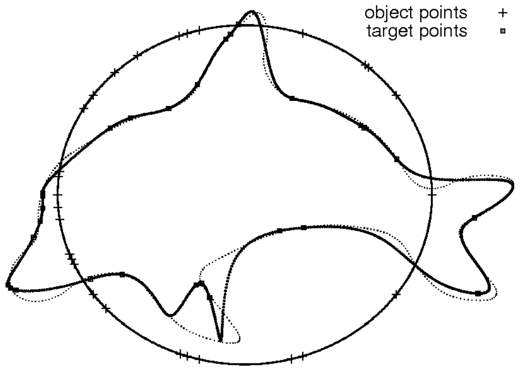


(b) Fitness progress

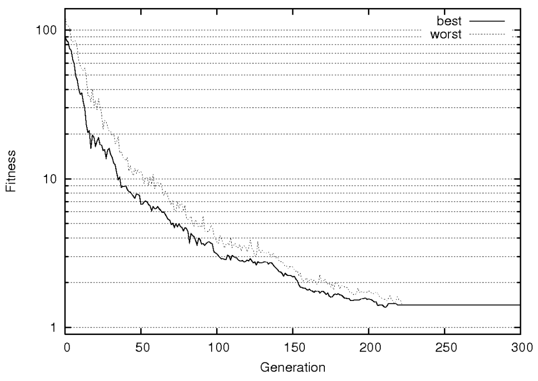


(c) Minimal and maximal strategy parameter

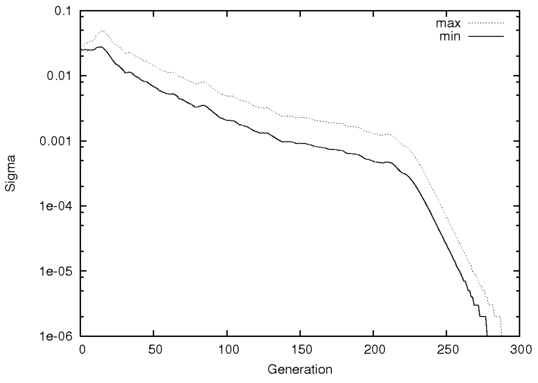
Figure A.18: The result and progress of the adaptive direct manipulation test run ADMD1. Every 15th generation the design point which is farthermost from the target is added into the genotype.



(a) Shape with smallest fitness. The object and target points are also displayed.

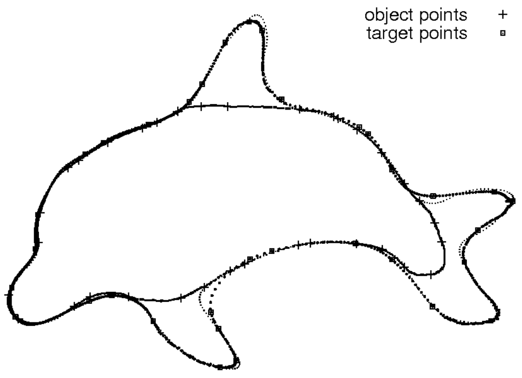


(b) Fitness progress

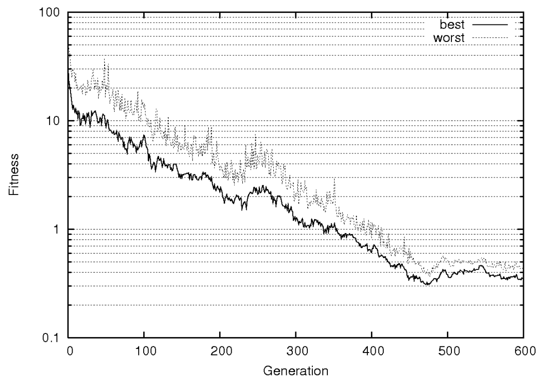


(c) Minimal and maximal strategy parameter

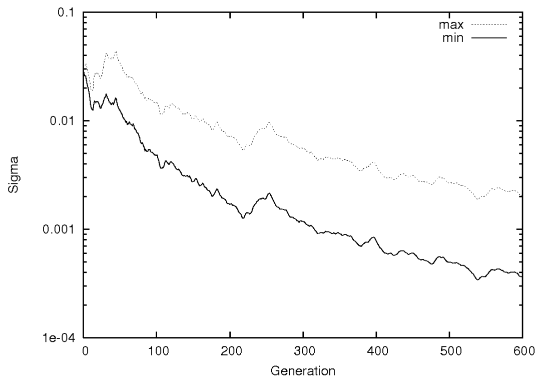
Figure A.19: The result and progress of the adaptive direct manipulation test run ADMD2. Every 10th generation a random point of the design is added into the genotype.



(a) Shape with smallest fitness. The object and target points are also displayed.

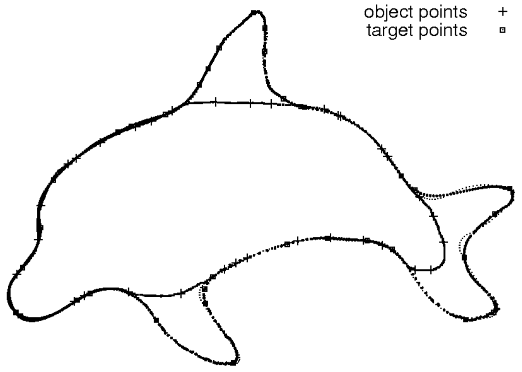


(b) Fitness progress

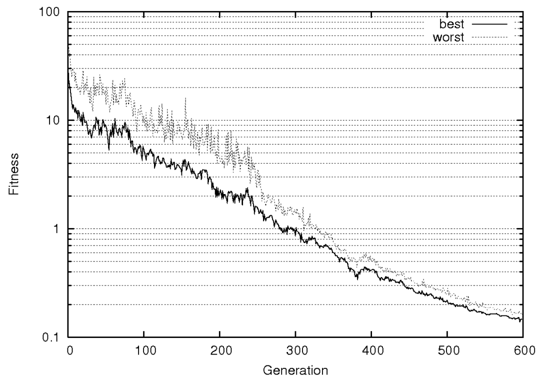


(c) Minimal and maximal strategy parameter

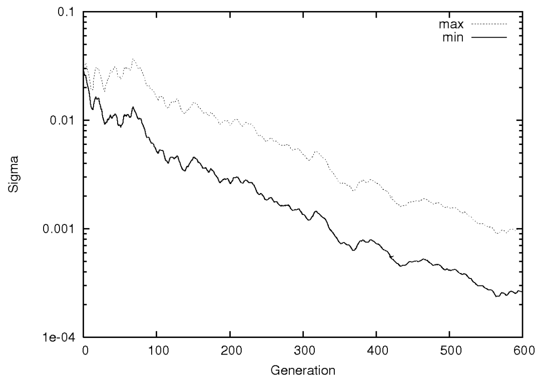
Figure A.20: The result and progress of the adaptive direct manipulation test run ADMD3. Every 10th generation a new point is added into the genotype. The point is selected randomly of the FFD cell which was changed most since the last refining and does not already contain an object point.



(a) Shape with smallest fitness. The object and target points are also displayed.



(b) Fitness progress



(c) Minimal and maximal strategy parameter

Figure A.21: The result and progress of the adaptive direct manipulation test run ADMD4. Every 10th generation a design point of FFD cell which does not already contain a object point, is selected randomly.

Ehrenwörtliche Erklärung

Hiermit versichere ich, die vorliegende Diplomarbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 20. Juli 2006

Thomas Bihrer