Acta Cybernetica **18** (2008) 669–680.

An Evolutionary Algorithm for Surface Modification

Evgeny Lomonosov* and Gábor Renner*

Abstract

Complex shapes bounded by free form surfaces are designed and modified by specifying the parameters of their mathematical descriptions. Although most graphics and design systems provide high level tools for free form shape modifications, they are not intuitive and suitable for a creative designer or a stylist. In the present paper a technique is proposed for the modification of the shape of a parametric B-spline surface through embedding a set of predefined characteristic curves. Shape modification is performed by an evolutionary algorithm. The evolutionary algorithm minimises the distance between the curve and the surface, while maintaining the global shape and smoothness of the original surface. Several experimental results are presented.

Keywords: shape modification, evolutionary algorithms, B-spline surfaces

1 Introduction

Complex shapes are described as free form surfaces and represented in parametric forms such as Bézier, B-spline or NURBS surfaces in computer graphics, computational geometry, and engineering design (CAD/CAM). They are usually created and modified by defining and modifying parameters of their mathematical representations, mainly the control points, eventually degrees, weights and knot vectors. Working with these requires mathematical knowledge and skills and is not suited to the engineering concepts and practice of shape design. Shape modification should be intuitive, easy to define, and resulting in the expected shape. Additionally, in engineering practice, acceptable compromise has to be found between different, sometimes contradictory requirements concerning shape and dimensions. Because of functionality, the resulting shape must meet tight dimensionality and shape constraints. It must also be aesthetically pleasing, free from unwanted bumps and wiggles. All these expectations can only be realised in a highly iterative process, which is costly and time consuming.

Methods for intuitively designing and modifying complex free form surfaces are described in several research papers. They try to control the shape by geometrical constraints and user parameters that are easy to manipulate. The so called

*MTA SZTAKI, Budapest, Hungary. E-mail: {elomonosov,renner}@sztaki.hu

Dynamic NURBS [9] introduces mass distribution, deformation energies and forces into the geometrical model in order to deform surfaces in a physically intuitive manner. In case of the so called variational design tool method, modification is governed by an underlying mechanical model; a bar network which is associated with the control polygon of the surface [3]. Due to the complexity of the deformation process, the allowable geometrical constraints are usually simple, for example, moving the surface to become tangent to a prescribed plane. Line constraints are more appropriate for surface modifications; they have been introduced into variational design by Pernot et al. [6]. Design tools of this kind can be integrated into a free form feature concept, providing high level tools for shape modifications [5]. Recently haptic tools were suggested for the direct manipulation of physically based B-spline surfaces [1].

A basic problem with the above approaches is that the effect of the modification is usually not precisely defined. They work well in the design phase, mostly in conceptual design. However, in many cases of shape modification, e.g. when redesigning an already existing object, well defined modifications are needed with precise shape and dimensionality constraints. Because of the free form characteristics of the surfaces, the modification must still be flexible, not constrained by the features of the mathematical description.

A typical example of the above type of shape modification is when a free form surface must be created which embeds a previously defined space curve. The designer usually starts with an initial shape, and the modified shape must be similar to the initial one as much as possible.

The algebraic complexity of free form curves lying on free form surfaces is high [8], and there is no real chance to embed analytically an arbitrary space curve into a smooth surface. However we can try to embed a good approximation of the curve, which also provides the possibility to satisfy additional design constraints, such as the definition of the surface region to be modified. Alternatively, the requirement of minimal distortion or of maximal smoothness of the final surface can be used.

These kinds of shape modification problems can be handled successfully by combining geometrical and numerical computations with evolutionary search methods. In this paper a method based on evolutionary algorithms is proposed to modify the shape of a complex surface with the constraint of embedding a space curve.

2 Evolutionary algorithms

Evolutionary algorithms are powerful and robust search procedures based on artificial adaptation. Evolutionary algorithms search for the solution of a problem simulating a process of evolution in nature. The problem is formulated as a search for the global maximum (or minimum, for some problems) of a given function which is called the fitness function. A fitness function depends on a set of parameters characterising the problem. It measures how fit is a potential solution, as specified by some combination of parameter values, for our purposes. The aim is to find the most suitable combination which corresponds to the maximal (or minimal) value

of the fitness function.

Conventional search techniques usually process one candidate solution a time. In contrast with these, evolutionary algorithms consider several potential solutions at once. A number of potential solutions form a population. An individual in a population represents one possible combination of parameter values. By analogy to natural genetics, the representation of a solution for an evolutionary algorithm is called a chromosome. A chromosome consists of genes which usually directly represent the parameters of the problem. Alternatively, an encoded representation may be used, for instance, a binary encoding proposed in [2]. It is often advantageous to incorporate problem specific knowledge into the problem representation for evolutionary algorithms.

Random numbers are normally used to generate an initial population. Another approach is to apply some heuristics to ensure that the individuals of the initial population cluster in those regions of the parameter space where the final solution is likely to be found. Subsequent generations are formed by selecting individuals with better fitness function values. The selected individuals are called parents. Individuals of the next generation (called offsprings) are obtained by applying genetic operators to parents. Similarly to the way it happens in nature, offsprings inherit properties of their parents and more fit individuals are more likely to pass their genes to the next generation.

A number of methods may be used to select parents, but it is essential that individuals with better fitness values have more chances to be reproduced in the next generation. Most commonly used genetic operators include crossover and mutation. The crossover operator exchanges genes between parents to form an offspring. The mutation operator changes some genes randomly. The evolutionary process is controlled by a number of parameters, the most important of which are the population size and the rates of mutation and crossover. As more fit individuals have more chances to pass their genes to the new generation, average fitness gradually improves. The pseudocode summarising the algorithm is shown below.

Algorithm 1 Evolutionary algorithm

```
1: Generate initial population
2: while Termination condition not satisfied do
3:    Evaluate fitness functions of individuals
4:    Select pairs of individuals to become parents
5:    Create offspring using crossover operator
6:    Apply mutation operator to offspring
7: end while
```

3 Surface shape modification

In this paper we consider the task of modifying an existing free-form surface to satisfy some constraints given as a number of free-form curves that should be em-

bedded into the resulting modified surface. The resulting surface should also retain similarity to the initial one and at the same time be smooth and visually pleasing. The surface is given by B-spline representation defined by the expression:

$$\mathbf{S}(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{p}_{ij} N_{ik}(u) N_{jl}(v), \tag{1}$$

where $\mathbf{p}_{ij}$ are control points forming a control net and $N_{ik}(u)$, $N_{jl}(v)$ are B-spline basis functions of order k and l [4].

The curves to be embedded into the surface can be given in an arbitrary representation commonly used in computer aided geometric design. In our experiments a cubic B-spline curve was used. Because exact representation of curves on surfaces has high algebraic complexity [8], we do not aim at producing a surface which embeds the given curve algebraically. Instead, the curve is sampled and the sampled points are used to calculate the fitness function. Curve sampling makes it possible to define the distance between a curve and a surface. The fitness function to be minimised is then defined as the sum of squared distances from the sampled points on the curve to the nearest points on the surface:

$$f(S) = \sum_{i=0}^{N} d^2(\mathbf{c}_i, \mathbf{s}_i), \tag{2}$$

where N is the number of sampled points, $\mathbf{c}_i$ is a sampled point on the curve, and $\mathbf{s}_i$ is the nearest point on the surface. A rough initial estimate and Newton-Raphson iteration are used to find $\mathbf{s}_i$.

All the parameters defining a B-spline surface can be used to modify its shape. The degree can be changed in each parametric direction, knots and control points can be moved, inserted, or removed. If the algorithm was allowed to change all the parameters of the surface it would lead to a very large search space and slow algorithm execution. To simplify the search space and to make the algorithm faster, we only consider moving control points in this paper, leaving the degrees unchanged and the knot vectors inherited from the initial surface.

An important property of the B-spline surface representation is its locality in the sense that a control point has influence only on a well defined region of the surface. As our goal is to obtain a surface which is similar to the initial one as much as possible, it is reasonable to alter only the patches of the surface nearest to the curve. Therefore the search space is further reduced by considering only control points which have influence on patches of the surface located near the curve. For each of the sampled curve points we find the nearest point on the surface and the corresponding knot intervals in the parameter space of the surface. Next, using the relations between control points and local basis functions, we determine the control points which have influence on this region of the surface. Alternatively, the region to be modified can be defined in an interactive way.

In Figure 1 an example of a surface is shown (a car body element) with parametric lines and modifying curves. Patches of the surface that have to be modified

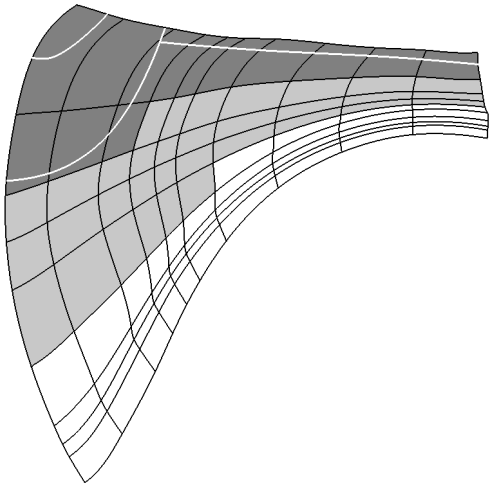

Figure 1: Modified patches

to embed the curve are shown in dark gray. Other patches that were affected by moving control points are shown in light gray and patches that were not modified are shown in white.

Some authors (e.g. Watabe [10]) suggest using multidimensional arrays as chromosomes when working with multidimensional data. However, our experiments showed that a one-dimensional array, where each gene corresponds to a displacement of one particular control point in the two-dimensional control net, performs better in our task. This can be attributed to the fact that in our case only part of the control net is varied by the algorithm, and the shape of this part is often not rectangular. Therefore using rectangular two-dimensional arrays as chromosomes would force the algorithm to process a lot of meaningless data, making it considerably slower.

As a consequence of the above experience and argumentation, a one-dimensional array of real-valued 3D vectors was used as the genetic representation (chromosome). Each vector corresponds to the displacement of one of the affected control points with respect to its initial position. Each individual surface is therefore defined by

$$\mathbf{S}(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n}(\mathbf{p}_{ij} + \Delta\mathbf{p}_k)N_{il}(u)N_{jm}(v), \tag{3}$$

where $\Delta\mathbf{p}_k$ are elements of the chromosomes (genes), $k = k(i,j)$. Mapping of genes onto the control net is illustrated in Figure 2.

New generations are formed using tournament selection of individuals. Two individuals are selected at random from the current population. Their fitness values are compared, and the individual with a better fitness value is used as a parent.

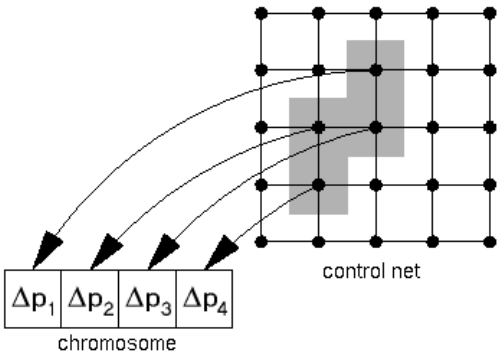

Figure 2: One-dimensional array of displacement vectors is mapped onto two-dimensional control net.

The second parent is selected in the same way. Crossover is performed as a random exchange of genes between the two parents. Any gene in the offspring has an equal probability of coming from any of the two parents.

Two mutation operators are used in the algorithm. The first mutation operator changes each component of the current displacement vector by a value not exceeding 10% of the allowed range. The other mutation operator replaces the current displacement vector in the chromosome with a randomly generated new one. In both cases each component of the new displacement vector is constrained to lie in the allowed range.

The allowed range for components of any displacement vectors is defined as a multiple of the maximum distance between a sampled point on the curve and the initial surface:

$$\Delta_{max} = C_\Delta \cdot max(d^2(\mathbf{c}_i, \mathbf{s}_i)), \tag{4}$$

where C_Δ is a coefficient. This effectively constrains control points of the resulting surface to lie inside a cube with the centre in the initial position of the point and with an edge size of $2\Delta_{max}$.

In order to make the best fitness non-increasing, elitism is used in the algorithm. A copy of the most fit individual in the current generation is preserved in the next generation with no crossover and mutation.

Besides embedding the curve, the resulting modified surface should retain similarity to the original one, being as close to it as possible. This is achieved without incorporating additional constraints in the algorithm or adding extra terms to the fitness function, by setting all displacement vectors in the initial population to zeros. Preliminary experiments showed that the algorithm that starts from randomly generated population, tends to move control points more than it is necessary, thus producing surface that is far from the initial.

The algorithm is terminated if the best fitness does not change for a number of generations or reaches a desired threshold value.

4 Surface smoothing

Many different methods can be applied to generate smooth B-spline curves or surfaces within an evolutionary process (see e.g. [7]). For instance, a smoothness criterion can be incorporated into the fitness function. The evolutionary algorithm, in this case, optimises the smoothness of the surface simultaneously with the above distance function. The resulting surface is a compromise between the desired shape and smoothness. In our case, incorporating a smoothness measure into the fitness function did not yield positive results. A possible explanation is that minimising the smoothness metric for the surface attempts to control smoothness globally, while we only need to preserve smoothness in the modified region and its surroundings. Control point modifications necessary for embedding the curve tend to spoil smoothness in these regions.

Another way to ensure surface smoothness is to perform additional smoothing as a separate post processing step, not taking smoothness into account when embedding curves. The drawback of this method is that the distance function usually deteriorates at the post processing step, and we can lose the desired shape of the surface. It may be more appropriate to control smoothness during the process of distance optimisation.

We developed a method that allows local control of surface smoothing. The proposed method tries to balance the change in the surface smoothness, which is introduced whenever the algorithm moves a control point, by means of moving other control points. If the original surface is smooth, embedding the curves which do not belong to the surface often makes the surface more bumpy. We are trying to compensate for this by moving neighbouring points whenever a control point is moved by the algorithm, to minimise changes in curvature as illustrated in Figure 3.

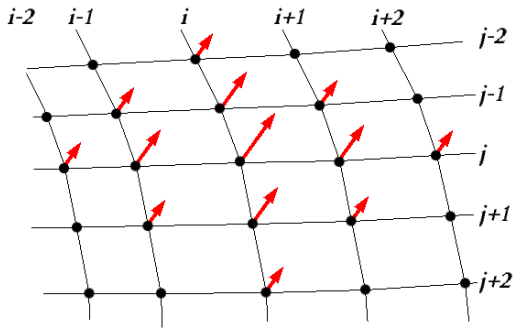

Figure 3: When a control point is moved, its neighbours also move.

If the algorithm moves point $\mathbf{p}_{ij}$ by $\Delta\mathbf{p}_k$, then neighbouring control points $\mathbf{p}_{i-1,j}$, $\mathbf{p}_{i+1,j}$, $\mathbf{p}_{i,j-1}$ and $\mathbf{p}_{i,j+1}$ are moved automatically by 2/3 $\Delta\mathbf{p}_k$. Furthermore, control points $\mathbf{p}_{i-2,j}$, $\mathbf{p}_{i+2,j}$, $\mathbf{p}_{i,j-2}$, $\mathbf{p}_{i,j+2}$, $\mathbf{p}_{i-1,j-1}$, $\mathbf{p}_{i-1,j+1}$, $\mathbf{p}_{i+1,j-1}$ and $\mathbf{p}_{i+1,j+1}$ are also moved, and their displacement is 1/3 $\Delta\mathbf{p}_k$. In this way we avoid

excessive relative control point movements. Therefore the algorithm produces a surface which is more visually pleasing and more similar to the initial one than those obtained by moving control points independently.

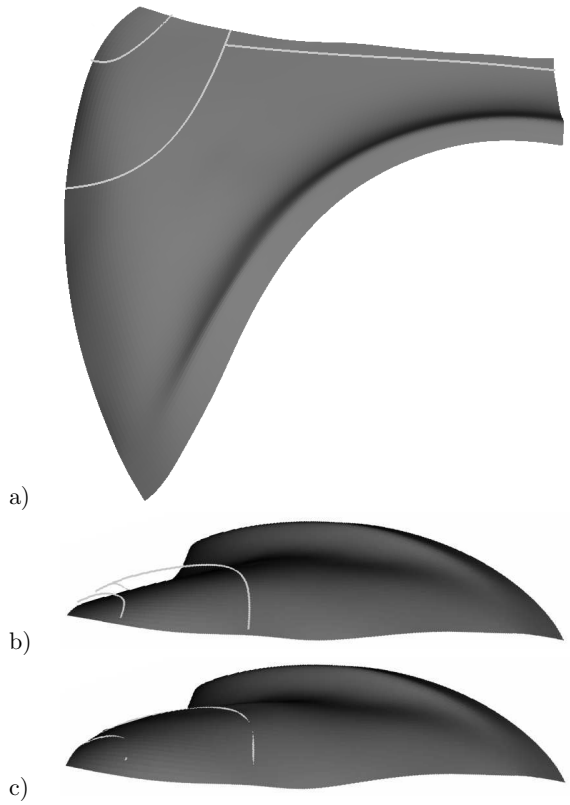

Figure 4: A car body part with modifying curves a) Front view b) Side view before the modification c) Side view after the modification

5 Results

The algorithm was applied to a number of different surfaces from various application areas, including technical and medical objects. Modifying curves of different shapes and spatial position were designed to guide the modification process. The aim was to obtain a surface that embeds the modifying curve and at the same time retains smoothness and shape similarity to the initial surface.

All the surfaces that we used were reverse-engineered from measured points using the GEOMAGIC Fashion 6 surface fitting package. The resulting surfaces

Figure 5: A medical replacement with a modifying curve (shown in white) and its projection onto the surface (shown in black)

are 3rd degree non-uniform polynomial B-spline surfaces. The modifying curves were designed using the Blender three-dimensional modelling system.

One of the most important application fields for our algorithm is automobile design. Therefore we used a real part of a car body, as the first surface for our experiments (Figure 4). We consider the situation when the designer alters the shape of an existing car body part to satisfy new functional and aesthetic constraints. For better visualisation of the relationship between the surface and the modifying curves we show both the front view and the side view. The result of surface shape modification is shown in Figure 4(c).

Another possible application field for surface modification algorithms is the reshaping of medical replacements (e.g. knee prostheses) in order to fit them to the shape and size of the patient. Figure 5 shows the tibia (lower limb) part of a knee prothesis with a modifying curve above the surface and its projection onto the surface. The third example is a sheet metal part surface which was chosen for its complexity and high curvatures (Figure 6).

For all the data sets, similar parameters were used in the evolutionary process, which were chosen after some preliminary experimentation. The crossover rate was set to 0.9. The mutation rate was 0.4 for the first mutation operator, and 0.05 for the second. The population size was set to 50 in all the cases.

For the car body example, both the original and the modified surfaces are shown, as the difference can be seen clearly. For the other two examples the resulting surface is very similar to the original one, and the difference cannot readily be seen. Therefore, the performance of the algorithm should be evaluated using numerical results.

The results of the experiments are shown in Table 1. Because of its stochastic nature, the evolutionary algorithm produces slightly different results each time we run it. Therefore, ten experiments were made with each surface. Table 1 shows the average values for ten runs of the algorithm. The following values are shown in the table:

1. the dataset name,
2. the number of control points of the surface,

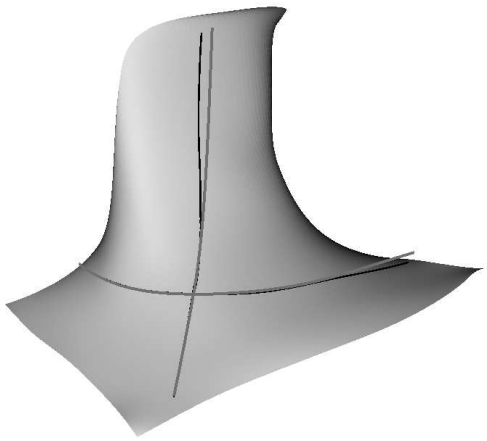

Figure 6: A metal sheet with a modifying curve (grey) and its projection (black)

3. the number of control points in the modified region,
4. the average number of generations before the fitness function reaches 0.1% of its starting value,
5. the average number of generations before the fitness function value stops changing,
6. the initial value of the fitness function (according to (2)),
7. the average final value of the fitness function,
8. the maximal final value of the fitness function.

Table 1: Numerical results of the experiments

1	Dataset name	Car body	Medical replacement	Metal sheet
2	Num. ctrl. points	156 (12×13)	238 (14×17)	72 (8×9)
3	Num. ctrl. points modified	58	56	59
4	Num. gen. before threshold	596	181	588
5	Num. gen. before stop	1874	1018	1944
6	Initial fitness value	6781.8	35.55	1128.8
7	Avg. final fitness value	2.6319	0.0003	0.3185
8	Max. final fitness value	4.2354	0.0006	0.4358

Two different termination conditions were employed in our experiments. The algorithm was terminated when the fitness function reached a pre-defined threshold, which was set to 0.1% of the initial fitness function value. The average number

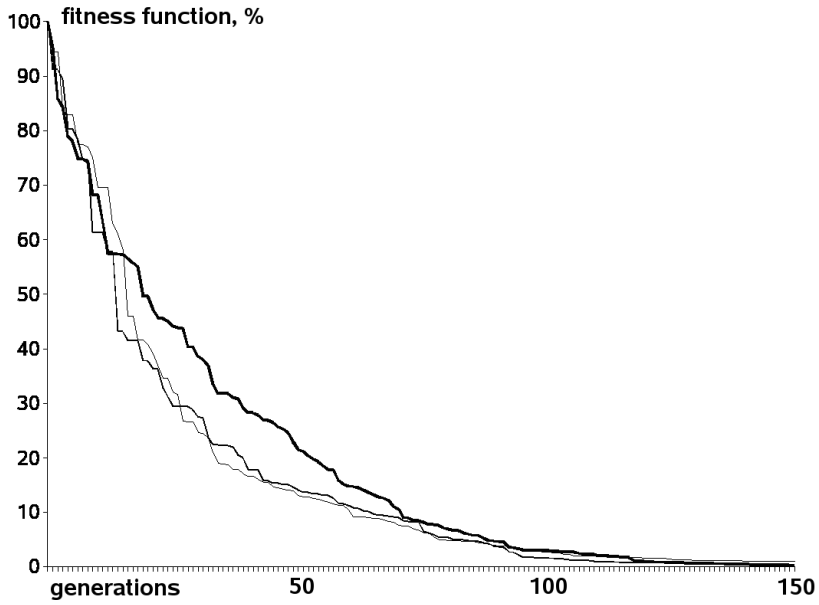

Figure 7: Fitness function change with generations

of generations needed to reach the threshold is given in the 4th row of Table 1. Then we continued the execution of the algorithm until the fitness function stopped changing its value. In practice, we terminated the algorithm as soon as the fitness function value did not change for 15 generations. The average number of generations before the fitness function value stopped changing is shown in the 5th row of the Table 1, the average final value of the fitness function in the 7th row, and the maximal final value of the fitness function in the 8th row.

Figure 7 shows how the the fitness function changes with the generations. Fitness is displayed here as a percentage of the initial value. Three typical cases were chosen, one for each example. It can be seen that an acceptable result was obtained in all the cases by the 150th generation. This result could later be improved only slightly.

It can be seen from Table 1 and Figure 7 that the algorithm reduced the fitness function value by three to five orders of magnitude. The resulting fitness function value is well below the required technical tolerance.

6 Conclusions

Free form shapes are modified in CAD/CAM systems by the parameters of their mathematical representations, mainly control points. This technique is tedious and not intuitive for a creative designer. In the paper a method is proposed to modify

the shape of a B-spline surface in order to embed a previously given characteristic curve into the surface. Control points of the surface are moved by an evolutionary algorithm. Genetic representation and control point modifications are constructed so as to minimise the distance between the curve and the surface, together with the changes in surface smoothness. Experimental results demonstrated the applicability of our method to several types of surfaces and curves. Examples from car body design and medical applications were given. Searching for other field of applications is in progress.

References

[1] Dachille, Frank, Qin, Hong, Kaufman, Arie, and El-Sana, Jihad. Haptic sculpting of dynamic surfaces, April 1999.

[2] Goldberg, David E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1989.

[3] Guillet, S. and Léon, J.C. Parametrically deformed free-form surfaces as part of a variational model. *Computer Aided Design*, 30:621–630, 1998.

[4] Hoschek, Joseph and Lasser, Dieter. *Fundamentals of Computer Aided Geometric Design*. A K Peters, 1993.

[5] Pernot, J-P., Falcidieno, B., Giannini, F., Guillet, S., and Léon, J-C. Modelling free-form surfaces using a feature-based approach. In *SM '03: Proceedings of the eighth ACM symposium on Solid modeling and applications*, pages 270–273, New York, NY, USA, 2003. ACM.

[6] Pernot, Jean-Philipe, Guillet, Stephane, Léon, Jean-Claude, Giannini, Franca, Catalano, Chiara Eva, and Falcidieno, Bianca. A shape deformation tool to model character lines in the early design phases. In *SMI '02: Proceedings of the Shape Modeling International 2002 (SMI'02)*, page 165, Washington, DC, USA, 2002. IEEE Computer Society.

[7] Renner, Gábor and Ekárt, Anikó. Genetic algorithms in computer aided design. *Computer-Aided Design*, 35(8):709–726, 2003.

[8] Renner, Gábor and Weiss, Volker. Exact and approximate computation of B-spline curves on surfaces. *Computer-Aided Design*, 36(4):351–362, 2004.

[9] Terzopoulos, Demetri and Qin, Hong. Dynamic NURBS with geometric constraints for interactive sculpting. *ACM Transactions on Graphics*, 13(2):103–136, 1994.

[10] Watabe, Hirokazu and Okino, Norio. A study on genetic shape design. In *Proceedings of the 5th International Conference on Genetic Algorithms, Urbana-Champaign, IL, USA, June 1993*, pages 445–451, 1993.