Published in *International Journal of Advanced Manufacturing Technology* 28:7–8 (April 2006), pp. 659–664;
doi: 10.1007/s00170-004-2415-x

Submitted May 4, 2004; accepted September 13, 2004; published online June 22, 2005.

Flexible Tools for Specifying Design Variation

Trichy Pasupathy, Xiaoping Zhao, and Robert Wilhelm

Department of Mechanical Engineering, University of North Carolina at Charlotte, Charlotte, North Carolina, USA

Corresponding author – Trichy Pasupathy, Department of Mechanical Engineering, UNC Charlotte, 9201 University City Blvd., Charlotte, NC 28223, USA, email trichy.pasupathy@jax.ufl.edu

Abstract
This paper describes flexible tools for specifying design variations that are based on nonuniform profile tolerance definitions. These tools specify bounds of design performance that can be used for negotiation among engineers in a collaborative design process. These specification methods allow for the capture of many different design functions that are not easily described with current tool designs. In addition, these specification methods lend themselves to efficient verification methods. Profile tolerance definitions provide the most general variation controls for complex mechanical surfaces. Common design practices and engineering standards for profile tolerances exhibit many weaknesses and limitations. We present a rationale for a complete specification approach using B-splines [1, 2] for profile tolerances, and illustrate the approach with examples. B-splines can be used to specify both uniform and nonuniform profile tolerance boundaries. Subsequently, algorithms for the evaluation of actual feature deviations and reporting methodologies for such tolerance zones are presented.

Keywords: design, profile, tolerancing

1 Introduction

The constant drive to create products that provide higher levels of performance and aesthetics has led to the design of complex shapes and part features. A common example is the airfoil design. Subsequently, manufacturers face the challenge of producing these parts in bulk, within the geometric variability specification and with little human intervention

to lower costs. They need to process tolerance information to perform analysis, synthesis, process planning, measurement, and verification. Computer-based tools facilitate the efficient execution of these tasks [3–7]. The foundation for the development of these tools lies in strong mathematical methods to specify and construct tolerance zones.

Profile tolerance specification is a powerful method that is extensively used to specify complex freeform and simple features. (Spheres and rectangles are examples of simple features as they can be defined by explicit equations). Profile tolerances are powerful since they offer the ability to simultaneously control size, form, and position tolerances.

Currently, uniform offsets are used to specify profile tolerances. In uniform offsets, all points on the nominal feature are offset in the normal direction by the tolerance value, and the extreme points are joined to give the boundaries of the tolerance zone. There are two variants: (1) the ASME [8]; and (2) the ISO [9]. In both these cases, however, problems arise in composing the tolerance zones of two adjacent features (as shown in Fig. 1(a) for ASME and Fig. 1(b) for ISO). Figure 1 (a) and (b) show one of the many potential design intents (excluding the undesirable regions) and how it cannot be met with the current standard definitions. The cause for this ambiguity can be attributed to the singular points [10]. The impact of this composition problem can be observed in the tolerance zone specification of real world parts. One example is the bent pipe that requires a varying tolerance boundary along its feature, as shown in [11]; another example is the airfoil in Figure 2. The airfoil also highlights the case of blending multi-tolerance zones of a single feature. Other challenges of the current specifications are discussed in detail in [10–12].

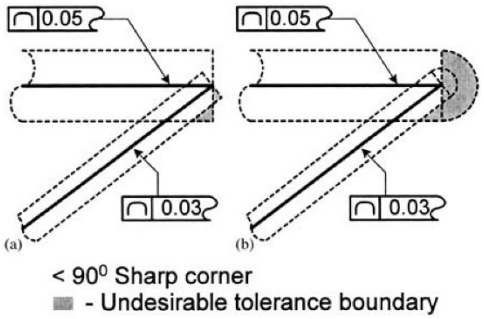

Figure 1. Undesirable zones in composition of features

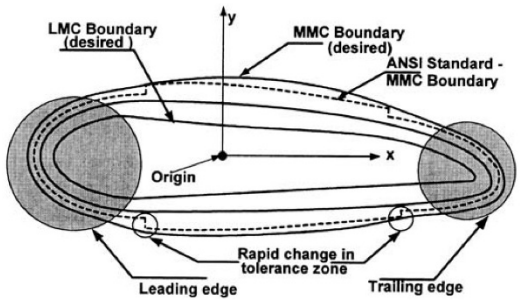

Figure 2. Desirable smooth tolerance boundaries

Additionally, designers should be given flexible tools to communicate their design intent without ambiguity. Such a tool should be portable in the manufacturing establishment to allow collaboration and discussion in a digital manner and enable the measurement of actual parts with coordinate measuring machines (CMM).

In this paper, we discuss the development of a flexible tool using B-splines for the specification of tolerance zones that provides a means for the unambiguous communication of design intent and measurement. A short literature survey is presented in the next section followed by a review of B-splines. Then we show the application of B-spline for the specification and describe two different algorithms for the measurement of actual deviations and a method for reporting the results.

2 Literature survey

Many methods are currently available to computationally construct uniform tolerance zones for freeform surface features [10]. Offsets obtained by manipulating the implicit equations constitute one group [13]; while offsets via Minkowski sums [14] are another. The third group consists of uniform offsets of base curves, which are called "B-splines" [15]. Although Minkowski offset and base surfaces defined by B-splines can describe a wide range of shapes, they do not fulfill the entire required range. Nonuniform offsets are not feasible with the above methods. In industries, for the special requirements described earlier, nominal offsets for critical sections of the part are specified, and the rest of the sections are either qualitatively judged or multiple parameters are used to verify conformance. In our airfoil example, the leading and trailing sections will only be specified. For the midsection, the maximum included circle is determined.

Cardew-Hall et al. [16] reported the use of a low-degree B-spline to join manually selected points on the tolerance boundary with straight lines for proof machining tolerances. This results in nonuniform offsets. Measurements of actual feature deviations and conformance verification of tolerances typical of current practices were not extracted or carried out. However, higher-degree B-splines that offer good controls, smooth boundaries, and accuracy in specification are required. New tools and methods are presented in this paper that apply recent methods of stationary point computation for measurement. The method presented in this paper can be utilized to describe advanced profile tolerance specification requirements.

3 B-splines for profile tolerancing

3.1 Review

There are many different mathematical curves and surfaces with applications in computer-aided geometric design (CAGD curves) for the construction of part features or surfaces. Table 1 illustrates the difference in properties of a selected list [2] of commonly used CAGD curves. A "✓" indicates that the curve possesses the desirable properties for the specification of tolerance boundaries, and an "×" denotes a lack of it.

Table 1. CAGD curves versus desirable properties for tolerance boundary specification

	Lagrange	Hermite	Bézier	B-spline any type $k = 2$	B-spline uniform $k > 2$	B-spline periodic/clamped $k > 2$
Convex hull	×	×	✓	✓	✓	✓
Variation diminishing	×	×	✓	✓	✓	✓
Ease of computing/ programming	✓	×	✓	✓	✓	✓
Inclusion of end points in the final curve	✓	✓	✓	✓	×	✓
Ab initio designs	×	×	✓	✓ (Excellent)	✓ (Excellent)	✓ (Excellent)
Smooth transition of curve	✓	✓	✓	×	✓	✓
Curve passing through all points on the curve	✓	✓	×	Techniques available	Techniques available	Techniques available

From the comparison of parameters such as ease, control in the graphical specification, and interpolating a given set of points (see Table 1), we conclude that the clamped/open type of B-spline is optimal to meet most requirements for the specification of both uniform and nonuniform profile tolerance boundaries. It possesses excellent properties for ab initio design, as the shape of the curve or feature can be modified in a predictable way.

The general form of B-spline for a 2D curve [1] is given by:

$$B(t) = \sum_j B_{j,k}(t) P_j; \quad 0 \le t \le 1 \qquad (1)$$

where $P = [P_j]$ is the array of control points; $B(t) = [B_j(t)]$ are the blending functions; and K is the knot vector obtained from the particular B-spline formulation. In Figure 3, the control points of the tolerance boundaries

$$P_1^{t+} \sim P_3^{t+} \text{ and } P_1^{t-} \sim P_3^{t-}$$

are shown. These control points are positioned with respect to the datum origin. The nonuniform blending function that creates boundaries with straight lines joining control points is used. The figure also shows that other blending functions can also be incorporated; a blending function to create a curved boundary is shown as an alternative curve. By interactively selecting the control points and the parameters for the blending function, user-defined nonuniform profile tolerance boundaries can be created. The same can be achieved for 3D surfaces. The general form for a B-spline [1] surface is given by:

$$B(u, v) = \sum i \sum j \, B_{i,j}(u) B_{j,l}(v) \, P_{ij} \; ; \quad 0 \le u \le 1; \quad 0 \le v \le 1 \qquad (2)$$

where the parameters k and l are knot vectors for u and v directions, respectively.

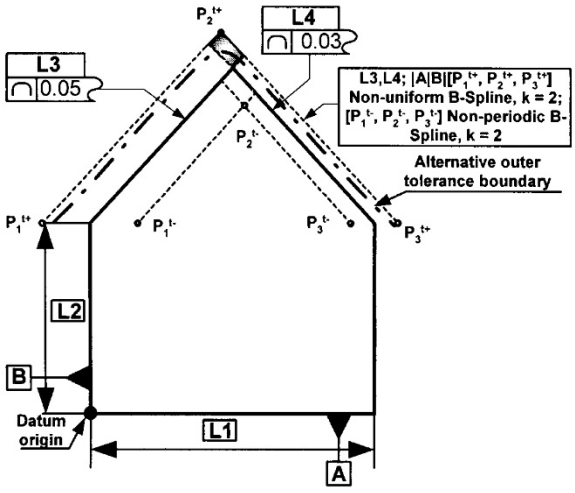

Figure 3. Nonuniform profile tolerance specification

3.2 Nonuniform profile tolerancing

Nonuniform profiles require variations in the set of rules to validate the feature's actual conformance in tolerance zones and in reporting deviations. In Figure 4, the difference in the observation of deviations is illustrated. In Figure 4(a) (uniform tolerance zones), the highest observed peak or valley is all that is required, since it indicates its position in the tolerance zone.

However, for nonuniform tolerance boundaries (Fig. 4(b)), meaningful information cannot be extracted from a single value. One potential format to report and visualize deviations is indicated in Figure 4(c). Here, magnitudes of the individual markers (vectors) are determined from the *L*/*D* ratio. The deviations can also be normalized to extract a single value, if necessary; this is another potential method. Apart from the above information, the regular rules for feature acceptability remain the same.

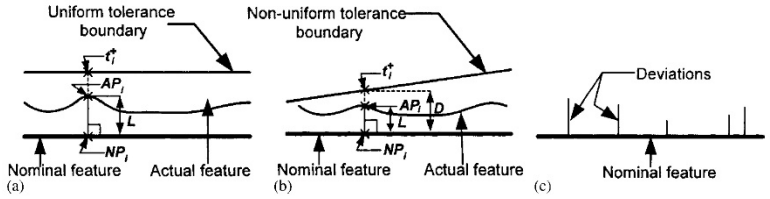

Figure 4. Deviations in nonuniform boundaries

To determine if a measured actual point is inside the tolerance zone requires stationary point computation. Stationary point is the point on the B-spline curve that is the closest to an arbitrary point. This helps to determine the location of an arbitrary point with respect to the B-spline curve. This information is also essential to edit the boundaries during the design and collaboration stage. Stationary point computation is not trivial, but it can be achieved with reasonable computing resources. This is discussed in section 3.3.

3.3 Actual feature validation algorithms and measurement

Stationary point computation is important in the implementation of this specification methodology. A formal representation is presented below.

The squared distance function between an arbitrary point $AP_0[x_0, y_0, z_0]^T$ and a curve $\mathbf{B}(t)$ (in 3D) is given by the equation:

$$D(t) = |AP_0 - \mathbf{B}(t)|^2 \qquad (3)$$

Likewise, the distance function between a point and a surface $\mathbf{B}(u, v)$ is given by:

$$D(u, v) = |AP_0 - \mathbf{B}(u, v)|^2 \qquad (4)$$

The t or the u, v values corresponding to the stationary point satisfy the condition $\nabla D = 0$. Processing the $\nabla D = 0$ operations for equations 3 and 4 results in the following:

$$D_t(t) = (AP_0 - \mathbf{B}(t)) \bullet \mathbf{B}_t(t) \qquad (5)$$

(Here $D_t(t)$ represents the differentiation of equation 4 with respect to t, and "•" represents the dot product.)

$$D_u(u, v) = (AP_0 - \mathbf{B}(u, v)) \bullet \mathbf{B}_u(u, v) = 0 \qquad (6)$$

and

$$D_v(u, v) = (AP_0 - \mathbf{B}(u, v)) \bullet \mathbf{B}_v(u, v) = 0 \qquad (7)$$

The roots from equations 5, 6, and 7 will give the stationary points on the curve. However, equations 5 through 7 are nonlinear. Alternative methods are required to determine the stationary point. In this paper, we present two approaches to the stationary point computation. The first detects multiple stationary points if they exist; the second detects only one point, but it is easier to implement. For tolerancing purposes, detection of multiple stationary points is not required, as the boundaries of tolerance zones are not supposed to self-intersect. The first method is the projected polyhedron algorithm, and the second is based on the binary subdivision method.

3.3.1 Projected polyhedron (PP) method

Partrikalakis et al. [15] have developed this procedure, and the complete methodology is available in their book. Only the basic procedure is explained here. An example curve representing equation 5 is presented in Figure 5. The curve intersects the t axis at the root, and this point has to be determined. The algorithm proceeds by first determining the convex hull of the curve. Next, the t values at the intersections of the t-axis and the convex hull are noted. The convex hull intersects the t-axis at t_1 and t_2. The section of the curve between the two t values is extracted and converted to B-spline/Bezier format, and the above steps are repeated until the convex hull collapses to a very small size, as per a preset tolerance value.

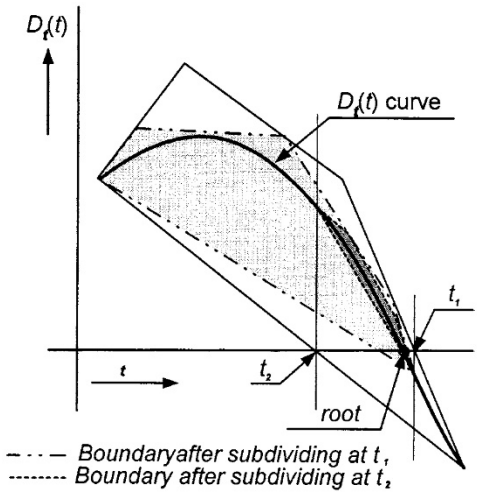

Figure 5. The projected polyhedron algorithm

A similar procedure is followed in determining the roots of the stationary point of surfaces. Two 2D convex hulls obtained from projections of the control points on the opposite planes of the coordinate frame have to be computed and subdivided. Additional distance computations between the given point and the corners of the surface have to be determined. By comparing the distances from the above computations, the root is established.

This method is useful in determining multiple roots. Additional work with regard to numerical stability has been carried out; this has resulted in the interval PP algorithm [15].

3.3.2 Binary subdivision method

Figure 6 explains this procedure. It shows a section of the curve between two subsequent nonrepetitive knot values, t_p and t_q. It also shows the given point AP_0 for which the stationary point on the curve has to be computed. The knot values are parameters on the t line that are obtained from the B-spline formulation. We start by subdividing the knot interval to obtain r at $(t_q - t_p)/2$. Next, we compare distances between $r + \Delta t$ and from $r - \Delta t$ to AP_0 to determine the section of the curve that is closer to the given point: either $t_p \le t \le r$ or $r \le t \le t_q$. (The distance computation is not straightforward, and this is discussed later in this section.) The value of Δt is arbitrary, and a smaller value would result in better accuracy. We choose the section of the curve that is closer and repeat the subdivision and selection process.

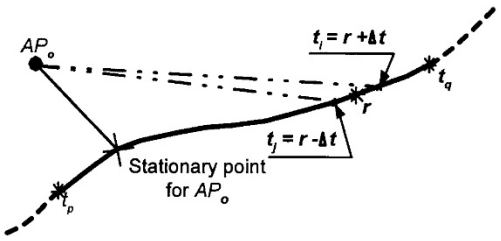

Figure 6. Binary subdivision method

The subdivision process is repeated until a preset minimum value is met. As the subdivision process proceeds, the distance between AP_0 and the closer section, either $r - \Delta t$ or $r + \Delta t$, is monitored. If the change in the distance between subsequent subdivisions is less than the minimum value, then we stop and record the value of t. We repeat this process with each nonrepetitive knot interval and determine corresponding t values that are closer to the given point. The distances from the recorded set of t values (reduced set) are compared to establish the stationary point. The complete pseudo-code for 2D and 3D stationary point computation is resented in [12].

Distance normalization. Distance computation between the arbitrary point and the curve is not straightforward because of the poor parameterization of the B-spline curve. This can be observed from a hypothetical case, as presented in Figure 7. The points on the curve corresponding to $r + \Delta t$ and $r - \Delta t$ that is computed from the B-spline function are not equally disposed about the computed point r. This may lead to wrong decisions in terms of which section of the curve should be discarded. The section with the point $r + \Delta t$ (Fig. 7) will be incorrectly chosen if the normalization is not carried out.

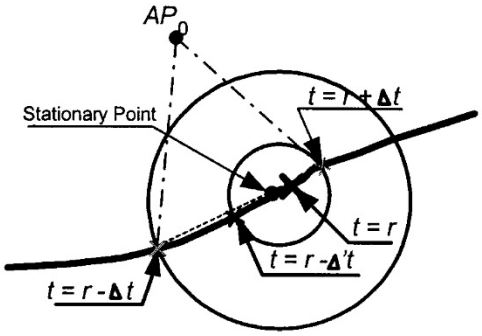

Figure 7. Normalization before distance computation

We normalize the distance and determine the point $r + \Delta' t$ or $r - \Delta' t$ (here $r - \Delta' t$) by interpolation before the decision to discard a section. The computational times for this algorithm are presented in section 4.

4 Examples and results

The binary subdivision algorithm was implemented and tested on several examples both for 2D and 3D curves and surfaces. Figure 8 illustrates an example for a 2D curve and a 3D surface.

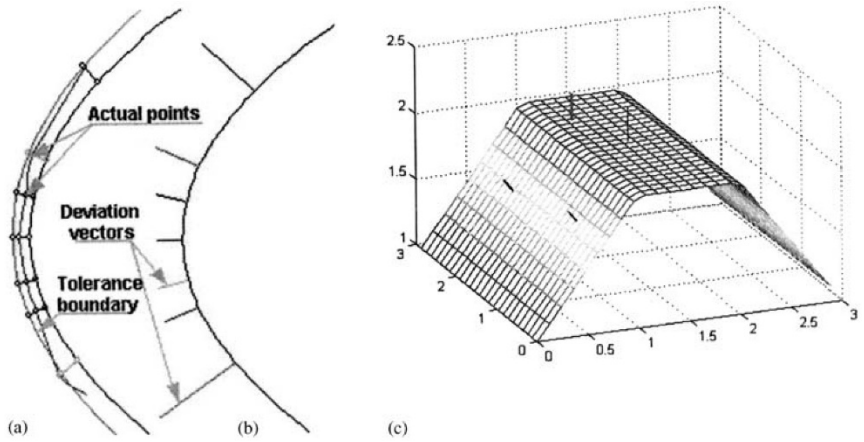

Figure 8. (a) Example feature, (b) deviations, and (c) deviations on a 3D surface

The algorithm was tested with B-splines that had a comparable explicit form, such as a straight line and a circle. Errors smaller than 10^{-10} were observed. This compares well with the tolerance preset that stops the search for the roots. In the example, the base curve and the outer boundaries were all described by clamped B-splines. The times for computing a stationary point depends on the number of control points, degree, and the preset minimum value. For instance, a seven-point degree-two 2D curve, the program took about 75 ms CPU time on a 450 MHz Windows PC. For a simple 3D surface with 16 points and degree-one curve, the CPU time was 100 ms. These times indicate that the tolerance boundary editing and measurement of actual features can be carried out in reasonable time.

Figure 9 shows a Web-based implementation. This application was developed using Java Technologies to build a Java Applet [17–19]. This applet was used to test different algorithms. Control points were input manually to draw the nominal feature. Subsequently, the interpolating algorithm from [1] was applied to draw the inner and outer boundaries with the points on the boundaries and the curve parameters. Additional editing was also carried out. In the evaluate mode, the binary subdivision program was used to calculate the stationary points on LMC and MMC boundaries.

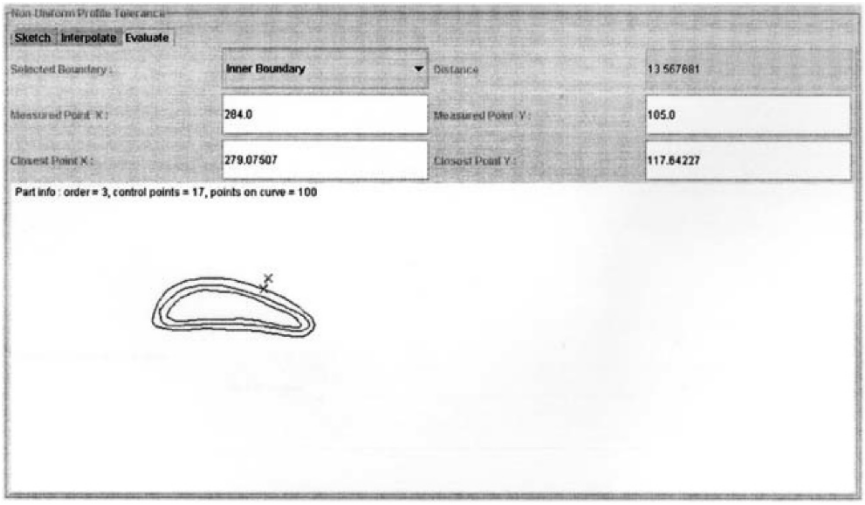

Figure 9. Web-based implementation

5 Conclusions

Nonuniform tolerance boundary specification is an important tool for designers and for collaborative manufacturing environments. It provides the flexibility to edit and modify the tolerance boundaries specific to a manufacturing environment and product functionality. Among various CADG curves, clamped B-spline curves were shown to be the most appropriate for this specification, as illustrated in the examples. The B-spline, however, is a nonlinear polynomial function. Thus, the evaluation of deviations of actual measured points is difficult and requires alternative methods. Two different approaches to the evaluation within a reasonable timeframe were presented. A Java applet was created to demonstrate the procedure involved in the specification and evaluation.

Acknowledgments – This work was possible thanks to the excellent research environment at the Center for Precision Metrology, UNC Charlotte. This material is based upon work supported by the National Science Foundation under Grant No. 9811556. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation (NSF).

References

[1] Rogers DF, Adams JA (1990) Mathematical elements for computer graphics, 2nd edn. McGraw-Hill, New York

[2] Mortenson ME (1997) Geometric modeling, 2nd edn. Wiley, New York

[3] Li JK, Zhang C, Zhang YZ (1989) Operational dimensions and tolerances calculation in CAPP systems for precision manufacturing. Ann CIRP 38(1):403

[4] Wilhelm RG, Lu SCY (1992) Tolerance synthesis to support concurrent engineering. Ann CIRP 41(1):197

[5] Elmaraghy HA et al. (1993) Evolution and future perspectives of CAPP. Ann CIRP 42(2):739

[6] Lapierre L, El Maraghy H (2000) Tolerance analysis and synthesis using Jacobian transforms. Ann CIRP 49(1):359

[7] Mathieu L, Marguet B (2001) Integrated design method to improve producibility based on product key characteristics and assembly sequences. Ann CIRP 50(1):85

[8] ASME (1994) Mathematical definition of dimensioning and tolerancing principles. ASME, New York

[9] ISO (1983) Technical drawings—geometrical tolerancing—tolerances of form, orientation, location and run-out—generalities, definitions, symbols, indications on drawings. ISO, Geneva

[10] Pasupathy TM, Morse EP, Wilhelm RG (2003) A survey of mathematical methods for the construction of tolerance zones. ASME Trans – J Comput Inf Sci Eng 3(1):64–75

[11] Henzold G (1997) Handbook of geometrical tolerancing. Wiley, New York

[12] Pasupathy TM (2003) Design automation geometric profile tolerance definition and placement of fluid power cartridge valves. Dissertation, University of North Carolina, Charlotte, NC

[13] Requicha AAG (1993) Mathematical meaning and computational representation of tolerance specifications. In: Proceedings of the International Forum on Dimensioning Tolerancing and Metrology, AMSE CRTD 27:61–68

[14] Rossignac JR, Requicha AAG (1986) Offsetting operations in solid modeling. Comput Aided Geom Des 3:129–148

[15] Patrikalakis NM, Maekawa T (2002) Shape interrogation for computer-aided design and manufacturing. Springer, Berlin Heidelberg New York

[16] Cardew-Hall MJ, Lebans T, West TG, Dench P (1993) A method of representing dimensions and tolerances on solid based freeform surfaces. Robot Comput Integr Manuf 10(3):223–234

[17] Capione M, Walrath K, Huml A (2001) The java tutorial, 3rd edn. Addison-Wesley, Boston

[18] Walrath K, Capione M (1999) The JFC swing tutorial, 2nd edn. Addison-Wesley, Boston

[19] Capione M et al. (1999) The java tutorial continued. Addison-Wesley, Boston