Effect of Tolerancing on the Relative Positions of Parts in an Assembly

Frédéric Cazals [1] and Jean-Claude Latombe [2]

[1] iMAGIS-IMAG, BP 53, 38041 Grenoble cedex 09, France

[2] Robotics Laboratory, Department of Computer Science, Stanford University, Stanford, CA 94305, USA

Emails: Frederic.Cazals@imag.fr, latombe@cs.stanford.edu

Abstract

This paper analyzes the variations of the relative positions of the parts composing an assembly when part dimensions span their specified tolerance zones. We focus on the simplified case where each part is modeled as a toleranced polygon defined as follows: each edge of this polygon is supported by a line of fixed orientation lying anywhere in a strip bounded by two extreme lines; these lines are themselves defined in a coordinate system attached to the part. The relative placements of parts in an assembly A are defined by spatial relations, such as edge e of part P is parallel to edge f of part Q, at distance d. We define the relative position of any two parts, P and Q, in an instance of A as the transform (a translation in our case) between the coordinate systems attached to these parts. Because of the possible variations in the geometry of each individual part, the relative position of P and Q is not constant over multiple instances of A. Hence, the question: What is the range of possible relative positions of any two parts in A? This paper describes an efficient algorithm to solve this problem. Experimental results obtained with an assembly sequence planner that incorporates this algorithm are also presented.

Keywords: tolerancing, assembly sequencing, assembly planning, computational geometry, solid modeling.

1 Introduction

Part tolerancing addresses the fact that manufacturing processes are inherently imprecise and produce parts of variable shapes. A large body of work has been devoted to the definition of tolerance languages providing designers with symbolic means to specify acceptable variations [4, 6, 7]. The problem of determining the range of possible boundaries of a part given tolerance specifications has attracted much attention [4, 5]. But the problem considered in this paper — the effect of tolerancing on the relative positions of parts in an assembly — has received little attention.

An assembly A is described by a geometric model of its parts and their relative placement. Typically, each part P is described as a collection of features localized in a coordinate system associated with the part. Tolerances specify acceptable variations of the features' locations in this coordinate system. The relative placement of parts is defined by spatial relations between features, e.g., face e of part P is parallel to face f of part Q, at distance d. The relative position of any two parts, P and Q, in an instance of A is the Euclidean transform between the coordinate systems attached to these parts. Because of the possible variations in the geometry of each individual part, the relative position of P and Q is not constant over multiple instances of A. This raises the following question: What is the range of possible relative positions of any two parts in A? The main contribution of this paper is the description of an algorithm that answers this question efficiently in the simplified case where A is a planar assembly made of polygonal parts.

The above question is interesting in itself. However, it appeared to us as we were designing an assembly sequence planner with toleranced parts [2]. A building block of this planner is an algorithm computing how each part Q may possibly block the motion of another part P. The motion directions of P that are blocked by Q depend on the geometry of the two parts, as well as on their relative position. The algorithm proposed in this paper is part of the implemented planner presented in [3].

Related work is the computation of the union and intersection of toleranced polygons. It is shown in [1] that even with a simple tolerance language, these standard operations become complicated.

2 Formal Problem

We now formally state the problem addressed in this paper.

A *toleranced part* P is a simple polygon with an attached coordinate system $(p, \vec{i}, \vec{j})$, as shown in Figure 1(a). Each edge e of P is defined by a triple $(\theta, [a, b])$ as follows:

- The outer normal to e makes an angle $\theta \in [0, 2\pi)$ with the i-axis of P's coordinate system.

- The distance between the origin p and the line supporting e is allowed to span the interval $[a, b] \subset \Re$.

Hence, e's orientation is uniquely defined by θ, while its location can vary. The distance d is called a *variational parameter* (or *parameter* for short) and the interval $[a, b]$ is the *tolerance zone* of this parameter.

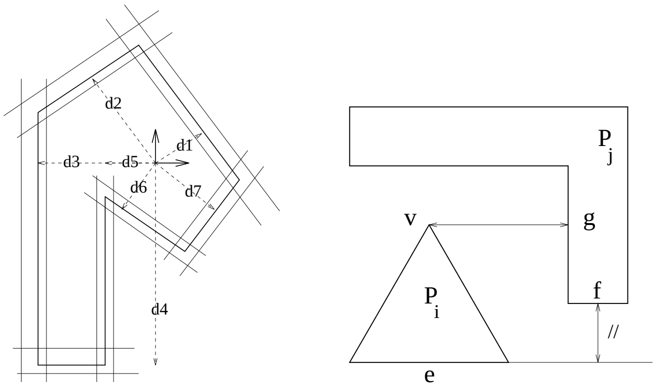

Figure 1: (a) Toleranced part (b) Spatial relation

A *toleranced assembly* is an assembly of toleranced parts whose relative placements are specified by spatial relations. Each *spatial relation* uniquely defines the relative placement of two parts independently of the particular geometry of these parts. As illustrated in Figure 1(b), a spatial relation consists of two *subrelations*:

- One subrelation states that two edges, one from each part, are parallel at a given signed distance, where the sign of the distance indicates whether the outer normals of the edges point in the same or opposite directions.

- The other subrelation states that a vertex of one part is at a given signed distance from the line supporting an edge of the other part.

The three edges involved in the same spatial relation must not be parallel.

The set of spatial relations in the description of an assembly A must be complete and non-redundant. By complete, we mean that if one randomly picks a valid geometry for every component of A, the relations determine a unique geometry for A. By non-redundant, we mean that removing any one of the relations makes the set incomplete. In order for the set of relations to be complete and non-redundant, it is necessary and sufficient that the undirected graph whose nodes are the components of A and whose links are the spatial relations be connected and without cycles. We call this graph the *relation graph* of A.

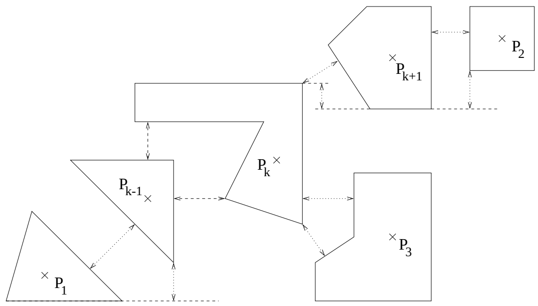

Figure 2: Relation graph

For any two parts, P_i and P_j, in their relative placement in A, we refer to the position of p_j in the coordinate system of P_i as the *position* of P_j *relative* to P_i. Although the relative orientation of their coordinate systems is fixed, the relative position of the two parts varies over several instances of A due to variations in the geometry of the parts composing A. The parts of A influencing the relative position of P_i and P_j are exactly the nodes along the path connecting P_i to P_j in the relation graph. For example, in Figure 2 the position of P_2 relative to P_1 depends on the geometry of P_1, P_{k-1}, P_k, P_{k+1}, and P_2. The position of P_3 relative to P_1 depends on the geometry of P_1, P_{k-1}, P_k, and P_3, etc.

For all valid geometries of the parts of A, we denote the locus of p_j with reference to p_i by $W_{i,j}$. (Note that this region is slightly different from its homonym in [2, 3].) We now present an algorithm to compute $W_{i,j}$.

3 Overview of the Computation

Given two parts P_i and P_j, let $L_{i,j}$ stand for the list of parts whose geometry influence their relative position. For example, in Figure 2: $L_{1,2} = (P_1, P_{k-1}, P_k, P_{k+1}, P_2)$, $L_{1,3} = (P_1, P_{k-1}, P_k, P_3)$, $L_{1,k-1} = (P_1, P_{k-1})$.

To simplify notations, we assume from now on that we wish to compute $W_{1,r}$ and that $L_{1,r} = (P_1, P_2, \ldots, P_{r-1}, P_r)$.

We denote the Minkowski sum of two point sets X and Y by $X \oplus Y$. By definition, $X \oplus Y = \{x+y | x \in X, y \in Y\}$.

We define the *region of influence* $I(P_k)$ of $P_k \in L_{1,r}$ to be the locus described by p_r when the only variational parameters allowed to vary are those of P_k and p_1 is fixed. Note that the only variational parameters of P_k that influence the position of P_r are those which define the locations (with respect to p_k) of the features of P_k which are used in spatial relations to define the placements of P_k relative to P_{k-1} and of P_{k+1} relative to P_k. There are at most six such parameters.

These parameters vary independently. Let us fix the values of all but one, d. When d spans its tolerance zone, p_r spans a line segment, which will be precisely defined in Sections 5 and 6. (Note that if $k \neq r$, all points p_i, $i = k+1, \ldots, r$, span parallel line segments of equal length.) So, when the variational parameters of P_k vary simultaneously, p_r spans a convex polygon of constant complexity that is the Minkowski sum of all the segments individually contributed by the parameters. This sum can be computed in constant time. Thus:

Proposition 1 *$I(P_k)$ is a convex polygon of constant complexity whose computation also takes constant time.*

When only the parameters of P_k are allowed to vary and $k \neq r$, the points p_i, $i = k+1, \ldots, r$, span polygons having the same shape and orientation, but at different locations. Thus, when all variational parameters in assembly A vary independently, the region $W_{1,r}$ is the Minkowski sum of all the regions of influence $I(P_k)$, $k = 1, \ldots, r$. The Minkowski sum of r convex polygons of constant complexity is known to be a convex polygon of complexity $O(r)$. It is computed by a divide-and-conquer algorithm in $O(r \log r)$ time. Hence:

Proposition 2 *The region $W_{1,r}$ is equal to the Minkowski sum $I(P_1) \oplus I(P_2) \oplus \cdots \oplus I(P_r)$. It is a convex polygon of complexity $O(r)$ that can be computed in $O(r \log r)$ time.*

The next three sections are devoted to the computation of the locus of p_r when a single parameter of P_k varies. As mentioned above, this locus is a line segment. To compute this line segment, we first distinguish among several types of parameters, depending on how they intervene in spatial relations.

4 Types of Parameters

There are at most six parameters of P_k which influence the position of p_r. We classify them into three types: those which are only involved in the relation between P_{k-1} and P_k, those which are only involved in the relation between P_k and P_{k+1}, and those which are involved in both relations. We call them the *input*, *output*, and *input-output* parameters, respectively.

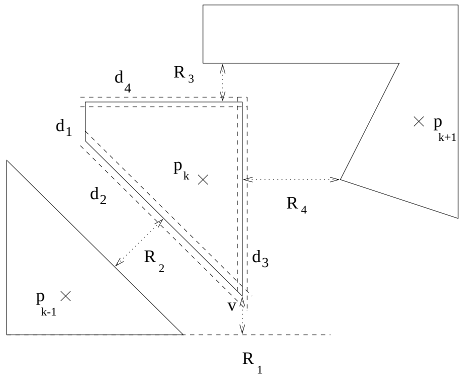

Figure 3: Types of parameters

To illustrate this classification, consider Figure 3:

- d_2 is an input parameter. When it varies, both p_k and p_{k+1} undergo the same vertical upward translation. (Note that this translation has not a 45dg orientation toward the right as one could first think. This is due to the fact that d_2 affects the positions of both the edge and the vertex of P_k that are respectively involved in the subrelations R_2 and R_1 with P_{k-1}.)

- d_4 is an output parameter. When it varies, p_{k+1} undergoes a vertical translation, while p_k remains fixed.

- d_3 is an input-output parameters. Its value affects the position of p_k (hence, the position of p_{k+1}) through the subrelation R_1 between P_{k-1} and P_k. It also affects the positions of p_{k+1} through the subrelation R_4 between P_k and P_{k+1}. Its combined effect on the position of p_{k+1} is still a translation, but it is more difficult to capture that for the other two parameters.

More generally, the variation of an input parameter of P_k directly affects the location of p_k and subsequently those of $p_{k+1}, \ldots, p_r$, all in the same way. Similarly, the variation of an output parameter of P_k directly affects the location of p_{k+1} and subsequently those of $p_{k+2}, \ldots, p_r$, all in the same way. The variation of an input-output parameter affects both the locations of p_k and p_{k+1}, but in different ways.

The somewhat non-intuitive effect of the parameter d_2 on the location of p_k in the above example leads us to refine our classification further. We say that an input or output parameter is an *edge* parameter if it is involved in a spatial relation through an edge only, a *vertex* parameter if it is involved through a vertex only, and an *edge-vertex* parameter if it is involved through both an edge and vertex. To clarify this terminology, consider again Figure 3: d_2 is an edge-vertex parameter, since it is involved in both R_1 (vertex) and R_2 (edge); d_4 is an edge parameter.

Let $\mathcal{R}(a, b)$ stand for "feature a of P_k is in subrelation with feature b of P_{k-1} or P_{k+1}." We say that an edge parameter d is in situation $\mathcal{R}(e, v)$ if the edge of P_k, whose position is determined by d, is in relation with a vertex of P_{k-1} (if d is an input parameter) or P_{k+1} (if d is an output parameter). Situations $\mathcal{R}(e, e)$ and $\mathcal{R}(v, e)$ are defined in the same obvious way. We say that an edge-vertex parameter d is in situation $\mathcal{R}(e - v, e - e)$ if the edge and vertex of P_k, whose positions are affected by d, are in relation each with an edge of P_{k-1} (if d is an input parameter) or P_{k+1} (if d is an output parameter). The situation $\mathcal{R}(e - v, v - e)$ is defined in the same way.

For input-output parameters, we concatenate the above refinements. For example, we say that an input-output parameter d is in situation $\mathcal{R}(e, .)\ \mathcal{R}(v, e)$ when it simultaneously determines the location of an edge of P_k that is a relation with an edge or a vertex of P_{k-1} and influences the position of a vertex of P_k that is in relation with an edge of P_{k+1}. In our notation, the dot "." stands for "any type of feature." Note that $\mathcal{R}(e, .)\ \mathcal{R}(v, e)$ and $\mathcal{R}(e, .)\ \mathcal{R}(v, .)$ are equivalent since the vertex of P_k referred to by $\mathcal{R}(v, .)$ can only be in a relation with an edge of P_{k+1}. The notations $\mathcal{R}(v, .)\ \mathcal{R}(e - v, .)$, $\mathcal{R}(e - v, .)\ \mathcal{R}(e - v, .)$, etc., are defined in the same way.

The set of all possible cases, which are analyzed in the next two sections, are listed in Figures 4 and 6.

We introduce a few more notations that are used below:

- We characterize the translation incurred by p_r when a parameter of P_k spans its tolerance zone by (t, β), meaning that a variation of this parameter of amplitude α causes p_r to undergo a translation of magnitude $\beta\alpha$ along vector t.

- Consider a parameter determining the location of an edge e of P_k. If it is an edge parameter we let d_e denote the direction of the line supporting e and leaving P_k's interior to its left; $\perp d_e$ refers to the outer normal to this line. If it is a vertex, we denote the same line by d_v and we let d_{av} designate the direction of the line supporting the edge adjacent to e at the corresponding vertex; d_{av} is chosen such that $\perp d_v \cdot d_{av} > 0$.

- Any spatial subrelation R involves at least one edge. We let $\perp R$ denote the direction perpendicular to this edge. The orientation chosen for $\perp R$ does not matter.

Note that P_1 has no input or input-output parameter, while P_r has no output or input-output parameter. The computation of $I(P_1)$ and $I(P_r)$ are therefore simple particular cases of the computation of $I(P_k)$ for $1 < k < r$.

5 Input and output parameters

In this section we precisely define the line segment spanned by p_r when an input or output parameter of P_k is the only variational parameter to vary. We distinguish among the 3 cases defined in the table of Figure 4. Case i-o:1 (the most delicate one) is illustrated by Figure 5.

index	*i* or *o*
i-o:1	$\mathcal{R}(v, e)$
i-o:2	$\mathcal{R}(e, e)$
	$\mathcal{R}(e, v)$
i-o:3	$\mathcal{R}(e - v, e - e)$
	$\mathcal{R}(e - v, v - e)$

Figure 4: Cases for Input or Output parameters

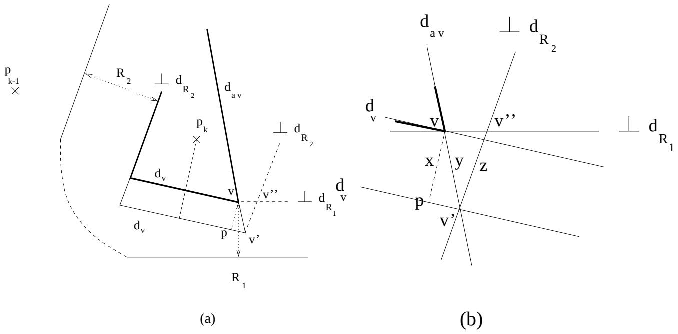

Figure 5: Case i-o:1: $\mathcal{R}(v, e)$ situation

Elementary trigonometric calculations yield the following propositions:

Proposition 3 (Case i-o:1) *An input parameter in situation $\mathcal{R}(v, e)$ induces the following translations:*

- *For p_k (hence, p_r):*
  $\beta_p = ((1/\sin(d_v, d_{av}))\sin(d_{av}, \perp R_1))/\sin(\perp R_2, \perp R_1)$
  t_p is supported by $\perp R_2$, such that $t_p \cdot d_{av} < 0$

- *For the vertex referred to by v in $\mathcal{R}(v, e)$:*
  $\beta_v = ((1/\sin(d_v, d_{av}))\sin(d_{av}, \perp R_2))/\sin(\perp R_2, \perp R_1)$
  t_v is supported by $\perp R_1$ such that t_v points into the same half-plane bounded by d_{av} as t_p .

- *For the edge whose position is determined by the parameter:*
  $\beta_e = \sin(d_v, \perp R_1)\beta_v$
  $t_e = - \perp d_v$

An output parameter of type $\mathcal{R}(v,e)$ induces a translation of p_{k+1} (hence, p_r) characterized by the coefficient β_p defined as above and the direction t_p supported by $\perp R_2$ such that $t_p \cdot d_{av} > 0$.

Proposition 4 (Case i-o:2) *A parameter in situation $\mathcal{R}(e,e)$ or $\mathcal{R}(e,v)$ induces a translation p_r characterized by the coefficient $\beta_p = 1/\sin(d_e, \perp R_2)$ and the direction t_p supported by $\perp R_2$ such that $t_p \cdot d_e < 0$ if the parameter is input and $t_p \cdot d_e > 0$ if the parameter is output.*

Proposition 5 (Case i-o:3) *A parameter in situation $\mathcal{R}(e-v, e-e)$ or $\mathcal{R}(e-v, v-e)$ induces a translation of p_r characterized by $\beta_p = 1/\sin(d_{av}, d_e)$ and $t_p = -d_{av}$ or $t_p = d_{av}$ depending on whether this parameter is input or output.*

6 Input-output parameters

As discussed before, the treatment of an input-output parameter requires considering three parts, P_{k-1}, P_k, and P_{k+1}, simultaneously. We distinguish among the 9 cases listed in the two tables of Figure 6.

index	i	o
io:1	$\mathcal{R}(e,.)$	$\mathcal{R}(e,.)$
io:2	$\mathcal{R}(e,.)$	$\mathcal{R}(v,.)$
io:3	$\mathcal{R}(v,.)$	$\mathcal{R}(e,.)$
io:4	$\mathcal{R}(v,.)$	$\mathcal{R}(v,.)$

index	i	o
io:5	$\mathcal{R}(e-v,.)$	$\mathcal{R}(e,.)$
io:6	$\mathcal{R}(e-v,.)$	$\mathcal{R}(v,.)$
io:7	$\mathcal{R}(e,.)$	$\mathcal{R}(e-v,.)$
io:8	$\mathcal{R}(v,.)$	$\mathcal{R}(e-v,.)$
io:9	$\mathcal{R}(e-v,.)$	$\mathcal{R}(e-v,.)$

Figure 6: Cases for Input-Output parameters

Again, elementary (but not so straightforward) trigonometric analysis yields the following propositions:

Proposition 6 (Cases io:{7,9}) *Consider an input-output parameter in any of the two situations $\mathcal{R}(e,.)\mathcal{R}(e-v,.)$ and $\mathcal{R}(e-v,.)\mathcal{R}(e-v,.)$. The variation of this parameter has no effect on the position of p_{k+1} (hence, p_r).*

Proposition 7 (Cases io:{1,2,5,6}) *Consider an input-output parameter in any of the situations $\mathcal{R}(e,.)\mathcal{R}(e,.)$, $\mathcal{R}(e,.)\mathcal{R}(v,.)$, $\mathcal{R}(e-v,.)\mathcal{R}(e,.)$, and $\mathcal{R}(e-v,.)\mathcal{R}(v,.)$.*

The variation of this parameter causes a translation of p_k characterized by (t_k, β_k), where both t_k and β_k can be derived from Proposition 3, 4, or 5.

The parameter variation causes a translation of p_{k+1} (hence, p_r) characterized by the coefficient $\beta = \beta_k \cos(t_k, R_4)/\sin(\perp R_4, \perp R_3)$ and the direction u supported by $\perp R_3$ such that $u \cdot \perp d_{e_4} > 0$, where d_{e_4} is the direction of the line supporting the edge of P_k involved in the subrelation between P_k and P_{k+1} in which the considered parameter is not involved.

Proposition 8 (Cases io:{3,4,8}) *Consider an input-output parameter in any of the situations $\mathcal{R}(v,.)\mathcal{R}(e,.)$, $\mathcal{R}(v,.)\mathcal{R}(v,.)$, and $\mathcal{R}(v,.)\mathcal{R}(e-v,.)$. The variation of this vertex parameter causes a translation of p_{k+1} composed of:*
- a translation characterized by the direction u supported by $\perp R_4$ such that $u \cdot t_4 > 0$, and the coefficient $\beta_u = \beta_3 \cos(t_3, R_3)/\sin(\perp R_3, \perp R_4)$,
- a translation characterized by the direction w supported by $\perp R_3$ such that $w \cdot t_3 > 0$ and the coefficient $\beta_w = \beta_4 \cos(t_4, R_4)/\sin(\perp R_3, \perp R_4)$,
where (t_3, β_3) and (t_4, β_4) characterize the translations undergone by the features of P_k involved in the subrelations R_4 and R_3 with part P_{k+1}. Both characterizations can be derived from Proposition 3.

7 Experiments

We have implemented the computations presented above. The corresponding software module is part of an assembly sequence planner that computes and queries a data structure called the strong non directional blocking graph (NDBG) [2, 3]. This data structure compactly represents the set of all extended translations of one part which are blocked by another part, for all pairs of parts in the assembly. Our implementation was done within the CAS.CADE environment of Matra-Datavision and consists of 11,000 lines of C++ code.

We run the assembly planner on several examples, including the simplified six-part "engine" shown in Figure 7. In this figure, for any two parts whose relative placement is defined by an input spatial relation, we have marked the edges involved in the edge-edge subrelation with short bold line segments, and the vertex and the edge involved in the vertex-edge subrelation with a bullet and a bold segment, respectively.

For simplicity, let us focus our attention on edge e of part 3 and assume that this is the only variational parameter with a non-zero tolerance zone. Figure 7 depicts the assembly at nominal size. We consider two tolerance zones for this parameter: the "positive" zone (the parameter is greater than its nominal value) and the "negative" zone (the parameter is smaller than its nominal value). Note on the figure that the nominal value is not at the center of the full tolerance zone which is shown between two dashed lines.

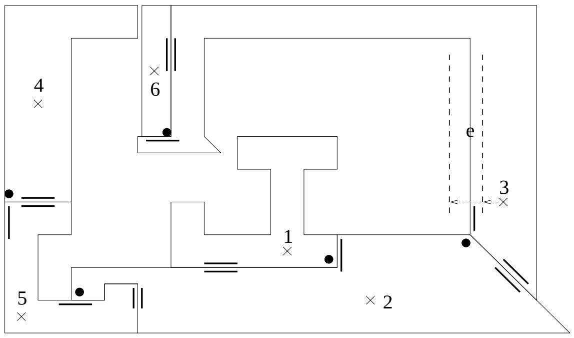

Figure 7: Engine example

We have run our assembly sequencer in three cases derived from this example: (1) edge e of part 3 is at its nominal location (hence, there is no tolerance); (2) it lies anywhere in the positive zone; and (3) it lies anywhere in the negative zone. In each case, we output two numbers computed by the planner: the number of guaranteed assembly sequences (i.e., sequences that are feasible for all possible values of the variational parameter) and the number of sub-assemblies that must be removed before part 1 can be withdrawn.

In case (1), the NDBG encodes 55 different assembly sequences. The shortest sequence to access part 1 is by removing parts 3 and 6 (as a rigid subassembly) with a vertical translation.

In case (2), the number of distinct guaranteed assembly sequences encoded by the strong NDBG shrinks to 23. Accessing part 1 is also more involved. It is no longer possible to vertically translate parts 3 and 6. Instead, part 4 can be removed first; then parts 3 and 6 can be translated toward the left-top. An alternative to this sequence is to remove 5 and 2 by a vertical downward translation. If fixturing issues are ignored, this is a valid sequence. But if they are not, parts 1, 3, and 4 will then have to be appropriately fixtured.

Finally, in case (3), the NDBG contains no guaranteed sequence. As one can easily observe, parts 4 and 6 can then overlap each other, which means that even the assembly state may not exist.

This simple example illustrates the dramatic impact that tolerances may have on valid assembly sequences. Thus, taking tolerances into account in assembly sequencing is of major practical importance.

8 Conclusion

This paper gives an algorithm to compute the relative position of two parts in a toleranced assembly defined by a simple, yet non-trivial tolerance language. As previously noticed in [1, 2, 3], finding exact geometric solutions for real problems involving toleranced objects turns out to be noticeably more difficult than the equivalent problems for nominal geometries. This difficulty comes from the number of cases to analyze as well as the complex interaction between the parameters.

It is therefore an important and interesting question whether such operations can be handled in three-dimensional assemblies with tolerances on feature orientations. This question remains open.

Acknowledgments: The work reported in this paper was done in the Computer Science Robotics Laboratory at Stanford University. F. Cazals was a Visiting Scholar supported by the Matra-Datavision company. The research was supported by NSF grant IRI-9306544-001 and a SIMA (Stanford Industrial Manufacturing Association) grant. The authors also wish to thank a reviewer for his careful reading and insightful comments.

References

[1] F. Cazals and G.D. Ramkumar. Algorithms for computing intersection and union of toleranced polygons with applications. To appear in *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 1997.

[2] JC. Latombe and R.H. Wilson. Assembly sequencing with toleranced parts. In *Proc. 3^{rd} ACM Symposium on Solid Modeling and Applications*, Salt Lake City, pp. 83-94, 1995.

[3] JC. Latombe, R.H. Wilson, and F. Cazals. Assembly sequencing with toleranced parts. *Computer Aided Design*, 29(2):159-174, 1997.

[4] U. Roy, C.R. Liu, and T.C. Woo. Review of dimensioning and tolerancing: representation and processing. *Computer Aided Design*, 23(7):466–83, 1991.

[5] H. Voelcker. A current perspective on tolerancing and metrology. *Manufacturing Review*, 6(4):258–268, 1993.

[6] R.K. Walker and V. Srinivasan. Creation and evolution of the ASME Y14.5.1 standard. *Manufacturing Review*, 7(1):16-23, 1994.

[7] N. Wang and T.M. Ozsoy. A scheme to represent features, dimensions, and tolerances in geometric modeling. *Journal of Manufacturing Systems*, 10(3):233–40, 1991.