Geometric computation for assembly planning with planar tolerated parts

Yaron Ostrovsky-Berman and Leo Joskowicz
School of Engineering and Computer Science
The Hebrew University of Jerusalem, Israel
Email: yaronber@cs.huji.ac.il

Abstract

The assembly planning problem has received significant attention due to its importance in autonomous manufacturing. Typical assembly planners assume that parts have nominal shapes, while in reality their geometry varies according to the tolerance specifications. To account for tolerated parts, an assembly plan must be feasible for all possible variations of its components. Despite its practical importance, very few works address this problem.

This paper presents a general framework for mechanical assembly planning with tolerated planar parts and shows how to incorporate it into existing planners. Our framework uses a general tolerancing model for parts: vertices are standard elementary functions of the part dimensions, which are allowed to vary within tolerance intervals. The relative position of parts is uniquely determined by an assembly graph, which defines constraints between features of neighboring parts. The assembly graph supports placements of parts with rotational degrees of freedom, cyclic relations, and conditional constraints, which occur in non-nominal contacts between edges. Using this framework, we show how to augment existing algorithms for useful motion types, including single and multiple step translations, and infinitesimal rigid motions. We demonstrate the dramatic reduction in the number of valid assembly plans when tolerances are introduced.

keywords: motion planning, assembly planning, tolerance analysis, tolerance envelopes.

1 Introduction

Assembly planning of mechanical systems has received significant attention due to its practical importance in manufacturing. Nearly all assembly planners produce plans for nominal parts, assuming that the part geometry, position, and orientation in the assembled state are fixed and known. However, the inherent imprecision of the manufacturing process introduces uncertainties in the shape and size of the assembly parts, which results in uncertainty in their relative placement in the assembled state. Thus, the nominal assembly plan may not be feasible for certain instances of the parts, and a valid plan for one instance may not be suitable for another.

Manufacturing variability is modeled with tolerances, which allow engineers to specify part and assembly variability. Assembly planners for tolerated parts must output a plan which is valid for all possible instances of the assembly parts, or identify that none exists. For example, the nominal assembly sequence for the tightly fit parts of engine in Figure 1 is not feasible for a slight variation in the geometry of its parts.

In this paper we present a general framework for planar tolerated assembly planning. We develop new geometric methods to model polygonal part contact variations and show how to incorporate them into existing planners to support assembly planning for tolerated parts for common assembly motion types, including single and multiple step translations, and infinitesimal rigid motions. The framework is based on our previously developed parametric tolerance...
specification model [11], which unlike simple tolerancing models, supports both parameter dependencies between features of the same part and between different parts. It subsumes the model of Latombe et al. [9, 2].

We model the relative placement of parts in the assembly with an assembly graph, which defines contact or distance constraints between parts. For each instance of the part geometries, the assembly graph uniquely determines the rigid transformation that places the parts in the position and orientation that satisfies the constraints. Our model supports cyclic relations between assembly parts (closed contact chains) and conditional constraints, which occur in non-nominal contacts between edges.

We model the relative position variation of each pair of parts in the assembly graph with the tolerance envelopes of the motion space obstacles of pairs of parts [6]. The tolerance envelopes bound the blocked motions resulting from the worst case combination of the variational parameters. The envelopes are computed from the partial derivatives of their vertices with respect to the variational parameters of the assembly. By replacing the nominal obstacles with their tolerated counterparts, we obtain assembly plans that are valid for all instances of the parts. We demonstrate on the engine example the dramatic reduction in the number of valid assembly plans when tolerances are introduced.

2 Related work

The general assembly planning problem is known to be PSPACE-hard, even for nominal parts [20]. With additional assumptions about the parts, motion types, or sequence types, efficient and practical algorithms have been developed (see [6, 18, 12] for surveys).

The relative position of imperfect planar parts was studied by Turner [17], who reduces the problem to solving a non-linear system of constraints for a given cost function. Sodhi and Turner [14] later extended this work for 3D parts. Li and Roy [10] show how to find the relative position of polyhedral parts with mating planes constraints. These methods compute the placement of a single instance of the assembly, and thus cannot be extended to analyze the entire variational class of the assembly. Inui et al. [8] propose a method for bounding the volume of the configuration space representing position uncertainties between two parts. However, their method is only applicable for polygonal parts and is computationally prohibitive.

Very few works deal with assembly planning for tolerated parts. Thomas et al. [16]
compute translational assembly sequences for polyhedral parts with uniform tolerances. They model geometric variation by scaling the parts by a constant factor and ignore the effect of tolerancing on the relative positioning of parts. Latombe et al. [9] present a simple tolerancing model in which polygonal parts vary in the distance of their edges from the part origin, but not in their orientation. The relative positioning of parts in this model serves as a basic step in computing the assembly plan [2]. The assembly planner is a special case of the motion space approach for two handed single translations to infinity [6]. Latombe et al. acknowledge the limitations of their model and point to the need for developing a more general tolerancing model and for supporting other motion types.

3 Toleranced assembly specification

Assemblies of toleranced parts require a representation that accounts for part variations. For simplicity, we first describe assemblies of toleranced polygons, and then discuss the extensions for more general planar parts and polyhedral parts. In Section 3.1, we briefly present our general model for toleranced parts. In Section 3.2 we describe a new method for computing the variation in the relative position of two polygonal parts related by three constraints. In Section 3.3 we extend this method for well constrained assemblies of parts with possible cyclic relations. In Section 3.4 we show how to support conditional constraints, which occur when two non-nominal edges intersect at a point. In Section 3.5 we discuss the importance of datums in tolerated assemblies.

3.1 Toleranced parts

We model part variation with the parametric tolerancing model described in [11]. This model is general, reflects current tolerancing practice, incorporates common tolerancing assumptions, and has good computational properties. In this model, part variation is determined by $m$ parameter values $p = (p_1, p_2, \ldots, p_m)$, specifying lengths, angles, and radii. The parameters have nominal values and can vary along small tolerance intervals. The coordinates of the part vertices are standard elementary functions of a subset of the $m$ parameters. An instance of the parameter values determines the geometry of the part. Figure 2 shows an example of the tolerance specifications for engine part $P_1$. Typically, the vertex functions are derived from the dimensional tolerance specification, either manually or as output from a symbolic geometric constraint solver. Here we chose the functions and tolerance intervals to emphasize the difference between the types of tolerance envelopes.
In [11], we describe algorithms for computing the outer and inner tolerance envelopes, which are boundaries of the union and the intersection of all possible parts, respectively. The algorithms input the partial derivatives of the vertices according to the \( m \) variational parameters, and compute the envelopes under the linear approximation of the model. For a part with \( n \) vertices, the BSA algorithm computes the most accurate tolerance envelope in \( O(nm^2 \log m) \) space and \( O(nm^2 \log m) \) time, and the VEA algorithm computes a conservative approximation of the envelope in \( O(nm) \) space and \( O(nm \log m) \) time. Figure 2 shows the BSA and VEA envelopes of engine part \( P_1 \).

3.2 Relative position of neighboring parts

We now describe how to compute the possible variation in the position of part \( B \) relative to part \( A \) when \( B \) is positioned according to the specification and the variational parameters of both parts span their allowed values. The goal is to compute the coordinates of each vertex of \( B \) as a linear function of the variational parameters of \( A \) and \( B \).

Planar part \( B \) has three degrees of freedom, two for translation and one for rotation. Thus, to uniquely determine its position relative to \( A \), three independent constraints are needed. For each instance of the parts, there is a rigid transformation \( T = (t_x, t_y, \theta) \) that positions \( B \) relative to \( A \) and satisfies the constraints. Since the part variations are at least two orders of magnitude smaller than the nominal dimensions, we approximate the transformation angle with \( \cos(\theta) \approx 1 \) and \( \sin(\theta) \approx \theta \). Consider a vertex \( v_i = (v_{ix}, v_{iy}) \in B \) and a line \( n_{ix}x + n_{iy}y + c_i = 0 \) supporting edge \( e_i \in A \), where \( n_{ix}^2 + n_{iy}^2 = 1 \). Since \( B \) undergoes a rigid transformation, the coordinates of \( v_i \) are now functions of the original coordinates and the transformation variables:

\[
\hat{v}_{ix} = v_{ix} - v_{iy}\theta + t_x \quad \text{and} \quad \hat{v}_{iy} = v_{ix}\theta + v_{iy} + t_y.
\]

To constrain the distance between \( v_i \) and \( e_i \) to be \( d_i \), we write the equation \( \hat{v}_{ix}n_{ix} + \hat{v}_{iy}n_{iy} + c_i = d_i \), or:

\[
n_{ix}t_x + n_{iy}t_y + (v_{ix}n_{iy} - v_{iy}n_{ix})\theta + v_{ix}n_{ix} + v_{iy}n_{iy} + c_i - d_i = 0 \tag{1}
\]

which is linear in \((t_x, t_y, \theta)\). When \( e_i \in B \) and \( v_i \in A \) we get a similar equation after neglecting second order terms in the transformation variables. Three such vertex-line constraints result in a linear system of equations in the transformation variables. The system has a unique solution when the constraints are independent. Since the vertices and edges of the parts are functions of the variation parameters, so are the transformation variables. Therefore, the transformed vertices of \( B \), \( \hat{v}_i = (\hat{v}_{ix}, \hat{v}_{iy}) \), are now functions of the variation parameters of both \( A \) and \( B \). When these parameters span their allowed values, the boundary of \( B \) lies inside its outer and inner tolerance envelopes, computed with the transformed vertices.

We illustrate the relative position computation on parts \( P_2 \) and \( P_3 \) of the engine when only \( P_2 \) is tolerated (Figure 3). The equation of lines \( L_1 \) and \( L_2 \) are, respectively: \( 50x + (50 + p_2 - p_1)y - 8350 - 33p_1 - 17p_2 = 0 \) and \( (67 + p_1)y - 1139 - 17p_1 = 0 \). Substituting the coefficients into Equation 1, we get the following system of linear equations:

\[
50t_x + (50 + p_2 - p_1)t_y + (6650 + 150p_2 - 150p_1)\theta - 50p_1 = 0
\]

\[
50t_x + (50 + p_2 - p_1)t_y + (9950 + 183p_2 - 183p_1)\theta - 33p_2 - 17p_1 = 0
\]

\[
(67 + p_1)t_y + (10050 + 150p_1)\theta = 0
\]

The solution is:

\[
t_x = \frac{(p_1^2 - p_1p_2 - 83p_1 - 17p_2)}{(p_1 - p_2 - 100)}
\]

\[
t_y = \frac{150(p_2 - p_1)}{17(p_1 - p_2 - 100)}
\]

\[
\theta = \frac{(p_1 - p_2)}{(p_1 - p_2 - 100)}
\]

Notice that when \( p_1 = p_2 \), no rotation or vertical translation is required.
3.3 Relative positions of parts in an assembly

In the simple assembly model of Latombe et al. [9], the relation graph describes the relative position constraints between parts with two degrees of freedom. We extend this graph to include cycles and support parts with general tolerances and three degrees of freedom, and call it the extended relation graph, or simply the assembly graph.

Graph nodes correspond to parts and undirected arcs correspond to constraints between parts. Arc weights are either 1, 2, or 3, and indicate the number of point-line constraints between the two parts. The arc data structure holds additional information about each constraint, such as feature id of part A, feature id of part B, and the value (or parametric expression) of the distance between these features. Figure 4 shows the assembly graph of the engine in Figure 1.

The assembly specification is complete if for any assembly instance the relative position of all pairs can be determined from the constraints, and it is non-redundant if the removal of any constraint results in incompleteness. An assembly with both these properties is called well constrained. A necessary and sufficient condition for a well constrained assembly with \( N \) parts is that the sum of arc weights is \( 3(N - 1) \), and that for each cycle in the graph with \( N_c \) nodes, the sum of weights is \( 3(N_c - 1) \) and there is exactly one arc of weight 2 and one arc of weight 1 (a cycle with three arcs of weight 2 results in a non-linear system of six equations with no general solution). The above conditions are similar to the conditions for well constrained sketches in geometric constraint solving [4].

We now observe two properties of well constrained assembly graphs:

- When two parts are connected by a chain of arcs of weight 3, their relative position is determined link by link, where each link is solved as in Section 3.2. Note that such a chain
Figure 5: A cycle in the assembly graph decomposed into two rigid parts. The parts $C_1$, $C_3$ and $C_2$, $C_4$ of rigid bodies $X$ and $Y$, respectively, contain features constraining the relative position of the two bodies.

- A cycle of $N_c$ parts has $N_c - 2$ arcs of weight 3, one arc of weight 2, and one arc of weight 1. The last two arcs divide the cycle into two disjoint sets of parts $X$ and $Y$ connected by the arcs. The relative position between parts connected by arcs of weight 1 or 2 cannot be determined because it is under-constrained, but the relative positions between the rigid bodies corresponding to $X$ and $Y$ is well constrained. Figure 5 shows the general case. In the engine example, $C_1, C_3 = P_7$, $C_2 = P_3$, and $C_4 = P_1$.

The relative position between $P_i$ and $P_j$ is defined by the effect of the variational parameters of all the parts in the path from $P_i$ to $P_j$ on the vertices of $P_j$. Table 1 summarizes the algorithm for its computation. The algorithm finds a path between $P_i$ and $P_j$, and then computes the transformations that relate the positions of consecutive parts. Finally it computes the partial derivatives of the target part $P_j$ vertices. Since the parameters of a part participate in two transformations at most, there are at most two transformations which have non-zero derivatives according to parameter $p_k$, $1 \leq k \leq m$. From the partial derivatives we obtain the linear approximation of the vertex coordinates functions. Note that there is no need to evaluate function derivatives, because the new derivatives are functions of the original assembly vertices’ derivatives, which were given as input to the algorithm.

The complexity analysis is as follows. Let $r_{ij}$ denote the number of parts in the path from $P_i$ to $P_j$ (including cycle parts). Let $m_k$ be the maximal number of variation parameters affecting a vertex of the $k^{th}$ part, and let $m = \max_k \{m_k\}$. At each iteration of step 2 we compute the transformation $T_{kl} = (t_x, t_y, \theta)$ between two parts (or rigid bodies) and its partial derivatives according to the local variation parameters (there are up to 9 vertices participating in each set of constraints, so the number of local parameters is at most 9$m$). According to Section 3.2, the solution for $T_{kl}$ has constant size, and so do its derivatives. The $n_j$ vertices of $P_j$ depend on at most $mr_{ij}$ variation parameters, so the computation of their partial derivatives takes $O(n_jmr_{ij})$ time.

We note that this result is a generalization of the result of Cazals and Latombe [2] for parts with two degrees of freedom. With their tolerancing model, translations are sufficient to satisfy the constraints. This means that all the vertices of $P_j$ are translated uniformly, and the translation depends on $O(mr_{ij})$ parameters. The tolerance zone of the translation is a convex polygon inside which the origin of part $P_j$ varies. It can be computed in $O(mr_{ij} \log(mr_{ij}))$ time (see [11]), and since $m$ is constant (at most six), this compares with their result. Note also that in their model, the vertices are linear functions of the variation parameters, so the approximation is in fact exact.
In nominal assemblies, two parallel edges make contact in a line segment, but in toleranced holds. This method extends for the case of multiple nominal contacts between parts, where intent is to make contact between both pairs of edges: first with the horizontal edges (which assemblies the contact is generally a point. Consider parts

3.4 Conditional constraints

In nominal assemblies, two parallel edges make contact in a line segment, but in toleranced assemblies the contact is generally a point. Consider parts \( P_1 \) and \( P_2 \) in Figure 1. The design intent is to make contact between both pairs of edges: first with the horizontal edges (which are wider and therefore provide more stable contacts), then with the vertical edges. The former edges are termed primary mating edges, and the latter secondary mating edges. The secondary contact is generally between a vertex and an edge, but which vertex makes contact depends on the instance of the parts. For example, if the vertical edge of \( P_1 \) leans to the right and \( P_2 \) is nominal, then the upper vertex of \( P_2 \) will make contact, and if the edge leans to the left, the lower vertex will make contact. This is a conditional constraint.

We extend the assembly graph with an additional constraint type: secondary mating edges. This constraint type specifies that contact should occur between two edges, but does not specify which vertex is in contact. To do this, we assume that in the nominal case both contacts occur simultaneously, so that any infinitesimal variation in the variation parameters either forces the assembly into a state where there is only one contact or leaves the assembly in the same state. This assumption holds in nearly all practical uses. To find the collision-free transformation \( T \), we solve the system of linear equations once for the case where the first vertex is in contact, and once where the second vertex is in contact, and denote the resulting transformations \( T_1 = (t_{1x}, t_{1y}, \theta_1) \) and \( T_2 = (t_{2x}, t_{2y}, \theta_2) \), respectively. Now, an infinitesimal increase in a single parameter \( p_i \) either results in \( T_1 \) or \( T_2 \) being the correct solutions, or leaves both correct. In the latter case, \( \frac{\partial T}{\partial p_i} = \frac{\partial T_1}{\partial p_i} = \frac{\partial T_2}{\partial p_i} \). In the former case, we compute the left-hand and right-hand derivatives as follows. If \( T_1 \) is the correct solution, then \( \frac{\partial T}{\partial p_i} = \frac{\partial T_1}{\partial p_i} = \frac{\partial T^+}{\partial p_i} \) and \( \frac{\partial T}{\partial p_i} = \frac{\partial T_2}{\partial p_i} = \frac{\partial T^-}{\partial p_i} \), where \( \frac{\partial T^+}{\partial p_i} \) is the right-hand derivative and \( \frac{\partial T^-}{\partial p_i} \) is the left-hand derivative of \( f \). Otherwise the symmetrical relation holds. This method extends for the case of multiple nominal contacts between parts, where \( \frac{\partial T}{\partial p_i} \) is computed according to which of the multiple conditions holds.

Note that the combined transformation \( T \) is accurate when a single parameter varies, but may not be accurate when several parameters vary simultaneously. This is a limitation of the linear approximation, which sums the effects of all the parameters and ignores dependencies.

Table 1: Computing relative position between parts \( P_i \) and \( P_j \)

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Find a path in the assembly graph from ( P_i ) to ( P_j ).</td>
</tr>
<tr>
<td>2.</td>
<td><strong>Iterate</strong> on the path arcs ( a = (P_k, P_l) ) in order:</td>
</tr>
<tr>
<td></td>
<td><strong>If</strong> ( \text{weight}(a) = 3 ) <strong>then</strong> compute transformation ( T_{kl} ) positioning ( P_l ) relative to ( P_k ).</td>
</tr>
<tr>
<td></td>
<td><strong>Else if</strong> ( \text{weight}(a) &lt; 3 ) (cycle arc) <strong>then</strong></td>
</tr>
<tr>
<td></td>
<td>- Find rigid bodies ( X, Y ) from graph cycle (( X ) containing ( P_k ))</td>
</tr>
<tr>
<td></td>
<td>- Identify parts with constrained features ( C_1, \ldots, C_4 ) (Fig. 5)</td>
</tr>
<tr>
<td></td>
<td>- Compute ( T_{xi} ) positioning parts ( P_x \in X ) relative to ( P_i ).</td>
</tr>
<tr>
<td></td>
<td>- Compute ( T_{yj} ) positioning parts ( P_y \in Y ) relative to ( C_2 ).</td>
</tr>
<tr>
<td></td>
<td>- Compute the trans. ( T_{XY} ) positioning ( Y ) relative to ( X ) according to constraints in ( C_1, C_2, C_3, C_4 ).</td>
</tr>
<tr>
<td></td>
<td>- Continue path from the exit arc (if it exists).</td>
</tr>
<tr>
<td>3.</td>
<td><strong>For each</strong> ( u \in P_j ):</td>
</tr>
<tr>
<td></td>
<td><strong>For each</strong> variational parameter ( p_k ) in parts from ( P_i ) to ( P_j ):</td>
</tr>
<tr>
<td></td>
<td>- Find the two trans. that depend on ( p_k )</td>
</tr>
<tr>
<td></td>
<td>- Apply each trans. on ( u ).</td>
</tr>
<tr>
<td></td>
<td>- Compute derivatives of ( u ) according to ( p_k ) using derivatives of trans. and of initial vertex derivatives.</td>
</tr>
</tbody>
</table>
However, the corresponding tolerance envelope is conservative, that is it does not underestimate the worst case variation.

### 3.5 Datum features

Datums are features (points, lines, planes, ...) from which the location, or geometric relationship of other part features may be established. Datum specification is an integral part of the tolerancing standard [1]. In our framework, the datum may be an edge of a part with a fixed origin and orientation, or it may be a reference frame which does not belong to the assembly. In either case, the datum is represented in the assembly graph. The datum serves as a global coordinate frame which defines translation directions and a rotation axis for the robotic manipulators in the assembly design. It is therefore important to compute the position of all the parts relative to the datum as a function of the variational parameters.

Consider the path from the part $P_1$ containing the datum through $P_i$ to $P_j$. We separate the variational parameters into two sets: $S_1$ contains the variational parameters of the parts from $P_i$ up to and including $P_j$, and $S_2$ contains the rest of the parameters in the path. As noted in Section 3.3, variations in the parameters of $S_2$ cause the parts from $P_i$ to $P_j$ to undergo an identical rigid transformation. The following observation is useful in assembly planning:

**Observation 1.** When two parts $P_i$ and $P_j$ undergo an identical rigid transformation $(t_x, t_y, \theta)$, their configuration space obstacle, denoted by $P_j \setminus P_i$, rotates in angle $\theta$ with respect to the origin, but does not translate.

Let $\theta(p)$ be the rotation in the rigid transformation relating variations in $p_i \in S_2$ to the positions of $P_i$ and $P_j$, and let $\delta_i$ be the maximal variation of the parameter $p_i$, then the maximal rotation angle as a result of combined variations of parameters in $S_2$ is $\theta_{\text{max}} = \sum_{p_i \in S_2} \frac{\partial \theta(p)}{\partial p_i} \delta_i$. We term it the cumulative rotation angle.

### 4 Geometric computation for assembly planning

We now show how to incorporate our framework into existing assembly planners to support tolerated parts. Assembly planning algorithms are typically categorized in terms of the number of robotic hands (two-handed or $m$-handed), the number of allowed steps in removal of a subassembly (single or multiple steps), the motion type (translational, rotational, infinitesimal), and the sequence type (linear and/or monotone).

In his PhD thesis, Wilson [18] described a general approach for assembly planning. To efficiently search the space of disassembly sequences, his method searches in restricted spaces of increasing complexity, and if all else fails applies the general motion planning algorithm, whose complexity is exponential. His method was later generalized into the motion space approach [6].

In the motion space approach, assembly motions are parameterized such that each point in motion space represents a mating motion that is independent of the moving part set. The geometry of the assembly partitions the motion space into an arrangement of cells such that within each cell, the blocking relations between parts remains fixed. The arrangement is called the Non Directional Blocking Graph (NDBG), and the blocking relations of each cell is called the Directional Blocking Graph (DBG). The algorithm searches the NDBG for a DBG with a strongly connected component, which corresponds to a subassembly that can be separated from the other parts. The motion space approach is applied to one-step translations, infinitesimal rigid motions, and multi-step translations.

Next, we show how to extend these algorithms to compute the strong NDBG, which accounts for the variations produced by tolerated parts.
1. Find the path from the datum \( P_1 \) that goes through \( P_i \) and \( P_j \), identify the parameter sets \( S_1 \) and \( S_2 \), and compute the cumulative rotation angle \( \theta_{\text{max}} \) (Section 3.5).

2. Compute the position of \( P_j \) relative to \( P_i \) as a function of parameters in \( S_1 \) (Section 3.3).

3. For each pair of edges \( e_k \in P_i \) and \( e_l \in P_j \) do:
   a. Compute the parametric C-Space obstacle \( e_l \setminus e_k \) (nominally a parallelepiped).
   b. Compute the outer tolerance envelope of the C-Space obstacle.
   c. Compute the blocking directions of the edges: \( C_{kl} \) (a segment in polar angles).

4. Compute the union of all the pairwise segments:
   \( C_{ij} = \bigcup_{e_k \in P_i, e_l \in P_j} C_{kl} \).

5. Augment \( C_{ij} \) by \( \theta_{\text{max}} \) clockwise and counterclockwise.

### Table 2: Algorithm for computing blocking directions

<table>
<thead>
<tr>
<th>( e_k )</th>
<th>( e_l )</th>
<th>BSA</th>
<th>VEA</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{blocking} )</td>
<td>( \text{directions} )</td>
<td>( C_{kl} )</td>
<td>( v_l - v_k )</td>
</tr>
</tbody>
</table>

Figure 6: Illustration of the blocking directions of two edges \( e_i \) and \( e_k \). For clarity, the tolerance envelope of the C-Space obstacle is exaggerated.

#### 4.1 One step infinite translations

Two handed monotone sequences of one step translations to infinity were studied in [9] for tolerated polygons with fixed edge orientations. The planner computes the strong NDBG to find separating translation directions that are feasible for all part instances. The geometric primitive in this method is the computation of the cone of translation directions for which part \( P_i \) is blocked by part \( P_j \).

Table 2 presents the extension for general tolerated parts. Vertices of the C-Space obstacle are of the form \( v_l(p) - v_k(p) \), where \( v_l \in e_l \) and \( v_k \in e_k \), and \( p \) is the variational parameter vector. Thus, the obstacle is parametric and the partial derivatives of its vertices are known. Figure 6 illustrates step 3 of the computation. The cone \( C_{ij} \) represents the directions \( d \) for which there exist instances of \( P_i \) and \( P_j \) such that translation of \( P_i \) in \( d \) is blocked by \( P_j \).

Let \( m \) be the maximal number of variational parameters of a part in the assembly, and let \( r_{ij} \) be the length of the path from \( P_i \) to \( P_j \). Since only infinite translations are considered, it suffices to compute the conservative VEA envelope instead of the BSA envelope in step 3(b) (the cones corresponding to edges for which configuration space origin lies in the difference between the tolerance envelopes are subsumed by the cones of the rest of the edges). The time complexity of the VEA algorithm is \( O(mr_{ij} \log(mr_{ij})) \). Let \( n_i \) and \( n_j \) be the number of vertices of \( P_i \) and \( P_j \), respectively. Then the total complexity of computing \( C_{ij} \) is \( O(mr_{ij} + r_{ij}n_in_jm \log(mr_{ij})) \).

To construct the NDBG, we compute the cones of all pairs of \( N \) parts. The combined complexity of the relative position and NDBG computation is \( O(N^2rn^2m \log(mr)) \), where
Figure 7: The contact zonotope and its contribution to the strong NDBG. Vertices of the zonotope are shown with their corresponding contacts, in which the parts are slightly translated and rotated from the nominal.

\[ r = \max_{ij} \{ r_{ij} \} \text{ and } n = \max_i \{ n_i \}. \] For parts with two degrees of freedom, as in [9], it suffices to compute the tolerance envelope of the translation once for every pair of parts, so the running time becomes \( O(N^2 r(n^2 + m \log(mr))) \), which compares with Latombe et al’s result for constant \( m \).

4.2 Infinitesimal rigid motions

As observed in [18, 5], when all the parts of the assembly are in contact, an effective way of pruning the space of disassembly motions is to consider infinitesimal rigid motions first.

Infinitesimal motions are represented by the unit vector \( dX = (dx, dy, d\theta)^T \), where rotations are with respect to the datum’s frame. This motion causes a vertex \( v = (v_x, v_y) \in A \) to undergo a translation \( t_v = J_v dX \), where \( J_v = \begin{pmatrix} 1 & 0 & -v_y \\ 0 & 1 & v_x \end{pmatrix} \) is the Jacobian relating the differential motion of part \( A \) to the motion of \( v \). For each contact between a vertex \( v \) and an edge \( e \) with normal \( n_e \), the non-collision constraint is \( n_e J_v dX \geq 0 \). The vector \( n = n_e J_v \) is the normal to the plane that intersects the sphere of unit motions in a great circle. Since both \( J_v \) and \( n_e \) are functions of the variation parameter vector \( p \), \( n = n(p) \). Let \( \bar{p} \) denote the nominal values of the variation parameters and \( p_i \) denote the \( i^{th} \) parameter out of \( k \), then the linear approximation of the normal is:

\[ n(p) \approx n(\bar{p}) + \sum_{i=1}^{k} \frac{\partial n(\bar{p})}{\partial p_i} (p_i - \bar{p}_i) \]

As noted in [11], the body defined by \( n(p) \), when \( p \) spans the allowed tolerance hyperbox, is a 3-zonotope [3, 21], a convex centrally symmetric three dimensional polytope whose center is the nominal normal \( n(\bar{p}) \). The complexity of the zonotope is \( O(k^2) \) and it can be computed in \( O(k^2) \) time using the one to one correspondence between the topology of the zonotope and an arrangement of lines in the plane [3]. Each vertex of the zonotope represents an extremal instance of the contacting parts and the normal to their corresponding inequality plane (Figure 7). The plane intersects the sphere in a great circle. For the instance considered, one side of this plane represents motions that will cause collision, and the other side is collision-free. For the considered contact, only motions that lie on the positive side of all the planes defined by the vertices of the zonotope are collision free for all instances of the assembly. The arrangement of great circles defined by all contacts is the strong NDBG of the assembly. In the contact zonotope illustrated in Figure 7, the vertices \( v_1, v_2, v_3, v_4 \) project to the unit normals of the planes intersecting the sphere in the great circles \( c_1, c_2, c_3, c_4 \), respectively. The circles are the contribution of the contact to the strong NDBG.

In the worst case, the number of parameters affecting the contact, \( k \), is \( O(mr) \). Thus for
an assembly with $c$ contacts, the number of great circles in the strong NDBG for infinitesimal motions is $O(c(mr)^2)$.

### 4.3 Multi-step translational motions

For assembly plans that support multiple translations for removing a subassembly, Wilson et al. introduced the interference diagram [19]. The interference diagram is the overlay of the outer boundaries of the Configuration space obstacles of all the pairs of parts, and its origin represents the assembled state. A path from the origin to the external diagram cell defines a multi-step motion which possibly separates the assembly into two sets. Edges of the diagram represent transitions between motions that cause some two parts to overlap and motions for which they do not. The DBG of the motion is constructed by tracing the edges crossed by the path.

To compute the strong interference diagram, we replace the C-Space obstacles with the outer envelopes of toleranced C-Space obstacles. We first compute the relative position between all pairs of parts. Then for each pair $P_i$ and $P_j$, we compute the outer tolerance envelope of $e_l \cap e_k$, $E_{kl}$, using the BSA algorithm from [11], but considering only parameters in $S_1$. The outer cell of the arrangement of curves composed of all the pairwise edge envelopes $E_{kl}$ bounds the tolerance envelope of the C-space obstacle. Its complexity is $O(n^2\alpha(n))$, where $n$ is the size of the parts and $\alpha(n)$ is the extremely slowly growing inverse of Ackerman’s function.

According to Observation 1, variation in parameters of $S_2$ causes the C-space obstacle to rotate. Thus to account for parameters in $S_2$, we compute the outer boundary of the area swept by the C-space obstacle envelope computed in the previous step when it is rotated by $\theta_{max}$ clockwise and counterclockwise (this is done with a rotating sweep line centered at the origin). The overall time complexity of computing the obstacles in the strong interference diagram is $O(N^2n^2mr(\log n + mr\log(mr)\alpha(n)))$. A similar extension can be used for the case of two translations [7].

### 5 Example

We implemented the assembly graph data structure and relative position computation in MATLAB and ran it on the engine example. The input consists of $N = 7$ parts, 30 variational parameters with $\pm0.2mm$ tolerance intervals (0.5% of the average feature length in the assembly), maximum part complexity $n = 14$, maximal path length $r = 7$ (from $P_5$ to $P_4$, including cycle parts), and maximal number of local parameters $m = 3$. The time to compute all the pairwise relative positions was 1.98 seconds on a Pentium IV 2.4GHz with 512 MB RAM. Figure 8(a) shows the configuration space obstacle $P_4 \setminus P_5$ and its BSA envelope. The origin lies inside the envelope, indicating that there are instances for which the parts interfere, and thus no valid assembly plan exist for them.

Even when fewer variational parameters are taken into account, tolerancing drastically affects the number of valid assembly sequences. Figure 8(b) shows an instance of the engine when only two parameters vary. Vertex $v_1 \in P_2$ can translate horizontally $2.5mm$ and vertex $v_2 \in P_5$ can translate vertically $0.75mm$ (both variations are less than 2% of the nominal edge length). Nominally, $P_1$ is accessible after translating $\{P_4,P_5\}$ to the left or translating $\{P_3,P_6\}$ upwards. For the instance shown, $P_1$ is accessible only after both motions are carried out (with proper fixtures, an alternative is moving $\{P_2,P_3\}$ downwards).
Figure 8: Engine example. (a) The C-Space obstacle $P_4 \setminus P_6$ and its BSA envelope. (b) Non nominal engine instance with two variational parameters.

6 Conclusion

We now briefly discuss the extension of our framework to more general part shapes and motion types.

The computation of blocking cones in Section 4.1 is also useful in the $m$-handed simultaneous one shot translation problem studied by Schwarzer et al. [13]. The lines bounding the blocking cones of each pair define the linear inequalities used by the algorithm when testing for unboundedness of the composite configuration space.

The method of Section 4.2 is also useful in other assembly planners based on contact analysis of tolerated assemblies. Srinivasan and Gadh [15] propose an efficient method for disassembly aimed at removing a selected polyhedral component from the assembly. The geometric part of their algorithm computes blocked translation direction based on contacts between parts. The blocked directions are nominally half spaces, but with tolerances they become the union of half spaces derived from extremal contact instances which are vertices of the contact zonotope.

The algorithm of Thomas et al. [16] uses the stereographic projection of the configuration space obstacles of polyhedral parts to find separating translations. They propose scaling the parts by a small factor to support tolerances, but this approach ignores parameter dependencies in the shape variation. Instead, it is possible to compute the tolerance envelopes of the projections, similar to the method of Section 4.3, and find separating translations that are feasible for all instances.

To support tolerated parts whose boundary consists of line segments and circular arcs, the assembly specification is augmented by distance constraints between arcs, which are in the form of quadratic equations. The resulting system of equations now possibly has multiple solutions, but only one of them coincides with the assembled state in the nominal case. The computation of blocking directions for circular segments in Section 4.1 must use the BSA algorithm to compute the tolerance envelope, because the concavity property of line segment envelopes does not hold (see [11] for tolerance envelopes of circular segments). For multi-step translations (Section 4.3), the strong interference diagram is the overlay of the BSA envelopes of the outer boundaries of C-Space obstacles, which consist of line segments and circular arcs. The infinitesimal motion planning algorithm in [18] does not readily extend to parts with circular arcs because the non-collision inequalities are no longer linear.

To support polyhedral parts, the assembly specification determines the position and orientation of the parts relative to each other by six point-plane distance constraints. These are
augmented by secondary and ternary mating planes constraints with conditional contacts (see Section 3.4). This requires modifying the zonotope computation in the algorithm of Section 4.2. The zonotope is now a six-dimensional polytope with $O(k^5)$ vertices. Each of these vertices contributes one great circle to the strong NDBG. For the algorithms of Sections 4.1 and 4.3, we need to compute the envelopes of spatial tolerated triangles.

In this paper, we have presented a framework for tolerated planar assembly planning which is more general than existing approaches in terms of the tolerance model of the parts and the type of motions used for the assembly. However, more work is necessary before it can be applied in the industry. First, it is not apparent how to extend the method beyond polyhedral parts. Second, our framework assumes pre-determined contacts, while practical assembly analysis considers part placements that are optimal with respect to objective functions [14, 10]. Third, the linear approximation is good for small variations, but long chains of mated parts may cause non-negligible cumulative errors. We will address some of these issues in future work.

Acknowledgement: This work was partially supported by a grant from the Israeli Ministry of Science and Technology, Grant Number 01-01509.

References


