Proceedings of the 1999 IEEE
International Symposium on Assembly and Task Planning
Porto, Portugal - July 1999

# Determining Near Optimal Interference-free Polyhedral Configurations for Stacking

**V.R. Ayyadevara**
Graduate Research Assistant
Department of Mechanical Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
va2a@andrew.cmu.edu

**D.A. Bourne**
Senior Systems Scientist
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
db@ri.cmu.edu

**K. Shimada**
Assistant Professor
Department of Mechanical Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
shimada@andrew.cmu.edu

**R.H. Sturges**
Associate Professor
Department of Mechanical Engineering and
Department of Industrial and Systems Engineering
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061
sturges@vt.edu

## Abstract

This paper uses a configuration space (c-space) approach to finding a satisfactory stack of polyhedral parts; we tested this method with industrial sheet metal parts. The optimal configuration for a new part, added to an existing stack, minimizes distance from a user-specified desired position, lies inside a given c-space region and, avoids interference with parts already in the stack. We present an iterative c-space based method that works with discrete orientations and yet produces interference-free configurations close to the desired part configuration. Two techniques are used to speed up the most computationally intensive step of c-space obstacle computation. An algorithm to compute orientation ranges within which connectivity graph topology of the obstacle stays constant is presented. Within such a range, extrapolation of c-space obstacle geometry from one orientation to another takes at most 15% of the time it takes to compute the obstacle from scratch. For every discrete orientation, we construct only a portion of the c-space obstacle in order to compute an interference-free configuration. For some complex parts, we determine the final configuration by considering less than 1% of all pairs of interfering convex components. The iterative method is guaranteed to converge.

## 1. Introduction

There are many applications that require the precise relative placement of pairs of complex polyhedral parts: e.g. packing, nesting, and stacking. Part stacking is especially difficult, because it combines the problem of final configuration (from packing and nesting domains) with stability concerns (from assembly planning). A desired part configuration can be determined to maximize stability and/or minimize floor space utilization. Most often interference between parts may preclude the desired part configuration. Therefore, it is required to compute an optimal configuration as close as possible to the desired configuration. Closeness is determined using a user-specified metric.

Determination of the optimal configurations can be performed in the configuration space (c-space) of the part. The desired position of the new part maximizes stack stability. Stack stability is maximized when the stack center of gravity (c.g.) lies at the centroid of the stack base and is as low as possible. If the new part in the desired position interferes with the stack, an interference-free configuration as close as possible to the desired position is computed. Two parts interfere with each other if the volume of their intersection is not zero. Hence, if they are just touching each other, they are not considered to interfere. The c-space of a part is six-dimensional: three position parameters $\{p_x,p_y,p_z\}$ and three orientation parameters $\{\phi,\psi,\theta\}$ (see Figure 1). These parameters describe the location of the part with respect to a world coordinate frame. A part configuration corresponds to a point in c-space. The parts already in the stack are stationary with respect to the world coordinate frame. A c-space obstacle corresponding to the stack is the set of configurations resulting in interference between the new part and the stack. Free space is the set of all interference-free configurations for the new part. For stacking application, the parameters $\{\phi,\psi\}$ can be chosen using stability maximizing heuristics such as selection of the

0-7803-5704-3/99/$10.00©1999 IEEE

largest face of the new part [1]. Therefore, the c-space of the new part is reduced to four-dimensional $\{p_x,p_y,p_z,\theta\}$-space. Consider adding a part to an existing part stack (Figure 2). A new part is inserted at the bottom of a three-part stack. The parameter $\theta$ is chosen initially as 0. In this initial configuration (Figure 2(c)), the new part interferes with the stack. In the final configuration Figure 2(d), the new part has been translated in +z direction and rotated about z-axis (changing $\theta$) to avoid interference.

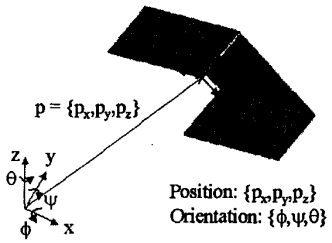
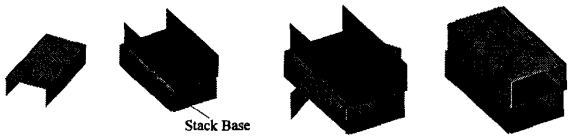



**Figure 1 Part Configuration Parameters**




(a) Part (b) Three Part Stack (c) Initial Configuration (d) Final Configuration

**Figure 2 Addition of a Part to a Three Part Stack**

This work applies c-space based techniques, used for robot path planning, to compute a near optimal interference-free configuration for a polyhedral part while adding it to an existing part stack. The final configuration minimizes distance from a user-specified position, lies inside a given c-space region and, avoids interference with parts already in the stack. The transfer of c-space based techniques from robot path planning domain to stacking domain is not straightforward. The desired location lies most often inside a c-space obstacle. Further, since the distance between the desired and optimal location, as measured by a user-specified metric, is to be minimized, connectivity information alone is insufficient. Only an exact description of the bounding surfaces of the c-space obstacle would enable the computation of the free configuration closest to the desired configuration. The requirement in path planning of guaranteeing that the path lies completely in free space does not apply to optimal part placement.

## 1.1. Previous Work

C-space based approaches have been popular with researchers working in path planning [2-6] and mechanism design [7-9]. In the case of path planning, given a start and a goal configuration in free space, a path, if one exists, is determined. If no path exists, the planner declares failure. It is important to show that the path lies completely in free space. Determining a feasible path, let alone an optimal one, is a difficult task. In the case of mechanism design, the desired behavior is plotted as a curve in c-space and lookup tables are used to suggest different mechanisms capable of achieving that behavior. The main difficulty with c-space approaches in four or more dimensions is the construction of the c-space obstacles. Researchers have been successful by limiting the dimensionality of the problem to three ($\{p_x,p_y,p_z\}$ or $\{p_x,p_y,\theta\}$) [4,8,9], by limiting the geometry of the parts to polyhedra [4], by considering only qualitative descriptions of c-space [7], or by performing incremental computation of c-space obstacles using probabilistic methods [3]. A c-space obstacle in $\{p_x,p_y,p_z\}$ space corresponding to a polyhedral obstacle is polyhedral. Schwartz and Sharir [5,6] study the effect of varying $\{\phi,\psi,\theta\}$ on the c-space obstacle in $\{p_x,p_y,p_z\}$ space and divide $\{\phi,\psi,\theta\}$ into non-critical and critical regions. In a non-critical region, the topology of the c-space obstacle stays the same. Joskowicz and Sacks [8] compute c-space obstacles for planar parts bounded by straight and circular segments. Sacks and Bajaj [9] investigate the effect of change in $\theta$ on the two dimensional obstacle in $\{p_x,p_y,p_z\}$ space. Avnaim and Boissonnat [10] present a polynomial time algorithm for construction of configuration space obstacles for one set of planar polygons with respect to another set of planar polygons. They determine the space of permissible configurations of a polygon set by obtaining the complement of the regions involving edge-edge intersections. The algorithm developed by Brost [11], on the other hand, uses facet intersections to construct the obstacles in configuration space.

## 1.2. Interference-free Configurations and Part Stacking

Optimal stacking polyhedral parts is very difficult. Even the two-dimensional form, i.e. optimal nesting of blanks on a sheet, with no stability concerns, has been shown to be NP-hard by Li and Milenkovic [12]. Hence, determination of a globally optimal part stack is virtually impossible. We are currently working on a stacking planner that uses a "generate and test" approach to generate good stacking plans. Such a planner needs tools for interference analysis and stack stability analysis. We have presented tools to evaluate stack stability in [1]. This paper focuses on another aspect of stacking: generating interference-free part configurations. This problem is also of interest to other areas such as part nesting, assembly planning and, packing.

We present an iterative method that works with discrete orientations and yet, produces very good configurations. Two techniques are used to speed up the most computationally expensive step of c-space obstacle construction. The algorithm has been implemented and tested successfully for planning stacking of sheet metal parts. The theoretical formulation and the iterative method can easily be extended to packing and nesting.

## 2. Problem Statement

Given a moving polyhedral set $\Phi$, a stationary polyhedral set $\Gamma$, a desired position $\mathbf{p_d}$ for $\Phi$, and a space $\Omega$ of candidate positions for $\Phi$, compute the configuration $Q^* \equiv \{\mathbf{p}^*, \theta^*\}$ as a solution to the problem:

$$\begin{aligned} &\min \left\| p_d - p^* \right\| \\ &\textit{subject to} \\ &\Phi(p^*, \theta^*) \cap \Gamma = \varnothing \\ &p^* \in \Omega, \theta^* \in [0, 2\pi). \end{aligned} \tag{1}$$

In the case of stacking, $\Phi$ is the new part and $\Gamma$ is the existing stack. The position $\mathbf{p_d}$ is chosen to center the stack c.g. with respect to the stack base and $\Omega$ is the set of positions such that the stack c.g. falls inside the convex hull of the stack base. This is an essential condition for part and stack stability. In Figure 2, the set $\Phi$ is the new part (Figure 2(a)), the set $\Gamma$ is the part already in the stack (Figure 2(b)), and $\Omega$ is the set of positions for the new part such that its c.g. falls inside the stack base in Figure 2(b). The space of candidate configurations $\Delta$ in $\{p_x,p_y,p_z,\theta\}$-space is obtained by sweeping $\Omega$ along $\theta$-axis from 0 to $2\pi$. The desired configuration in $\{p_x,p_y,p_z,\theta\}$-space is the line segment $\{\mathbf{p} = \mathbf{p_d},\ \theta \in [0, 2\pi)\}$. We use Euclidean distance $\|\mathbf{p_d}-\mathbf{p}\|$ from the desired position $\mathbf{p_d}$ as a metric for a candidate configuration $\{\mathbf{p},\theta\}$.

## 3. Approach

The optimal interference-free configuration can be obtained by constructing the c-space obstacle $C(\Phi,\Gamma)$ in $\{p_x,p_y,p_z,\theta\}$-space, subtracting it from $\Delta$, selecting the configuration $Q^*$ closest to the line $\{\mathbf{p} = \mathbf{p_d},\ \theta \in [0, 2\pi)\}$. However, computing a closed-form description of $C(\Phi,\Gamma)$ in $\{p_x,p_y,p_z,\theta\}$-space is much more difficult than computing $C(\Phi,\Gamma,\theta_k)$ in $\{p_x,p_y,p_z\}$-space for a discrete orientation $\theta_k$. First, we examine the effect of change in orientation on a c-space obstacle in $\{p_x,p_y,p_z\}$-space. Next, we show that the number of non-linear optimization problems to be solved to compute $Q^*$ using $C(\Phi,\Gamma)$ and $\Delta$ is exponential in the number of convex components of $\Phi$ and $\Gamma$. In the case of stacking, the number of problems is therefore, exponential in the number of stack parts and number of convex components of a part. Finally we describe a method that uses partial c-space obstacles and discrete orientations to compute near optimal interference-free part configurations.

### 3.1. Effect of Rotation on C-space Obstacle in $\{p_x,p_y,p_z\}$-space

Consider the pair of convex polygons A and B in Figure 3. B is stationary and A is allowed to translate, but not rotate. The polygons are shown on the left-hand side in world coordinates and the c-space obstacle corresponding to B in $\{p_x,p_y\}$-space is shown on the right hand side. The following discussion also applies to obstacles in $\{p_x,p_y,p_z\}$-space. The c-space obstacle $C(A,B,\theta_0)$ describes the set of positions of A, with orientation $\theta_0$, resulting in interference with B. It can be expressed as the Minkowski sum of B and A reflected about the origin of world coordinate frame [13]. Lozano-Perez [4] shows that $C(A,B,\theta_0)$ is convex and can be constructed using vertices $V_i^A$ of A and vertices $V_j^B$ of B using the following equation:

$$\begin{aligned} C(A,B,\theta_0) = Conv\left(\left\{P_{ij} \equiv V_j^B - V_i^A(p,\theta_0)\right\}\right), \\ i = 1,2,\cdots,n_A,\ j = 1,2,\cdots,n_B \end{aligned} \tag{2}$$

This computation can be performed in $O(n_A \cdot n_B \cdot log(n_A \cdot n_B))$ time where $n_A$ is the number of vertices of A, and $n_B$ is the number of vertices of B.

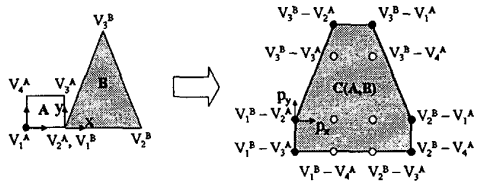



**Figure 3 Construction of C-space Obstacle C(A,B,0)**

Let us examine the effect on C(A,B) of allowing A to rotate about z-axis of world coordinate frame. The points $P_{ij}$ used to compute C(A,B) can be divided into two mutually exclusive sets: black extreme points of C(A,B) and white interior points. As A is rotated, all points $P_{ij}$ are transformed as follows:

$$P_{ij}(\theta) = V_j^B - R_{z,\theta} V_{0i}^A, i = 1,2,\cdots,n_A,\ j = 1,2,\cdots,n_B \tag{3}$$

where $R_{r,\theta}$ is the rotation matrix and $V_{0i}^A$, i = 1, 2, ..., $n_A$ are the vertices of A at $\theta$ = 0. As A is rotated, the connectivity graph topology of C(A,B) remains the same as long as the black points remain extreme points and the white points remain interior. As long as the topology remains the same, C(A,B) for a different orientation can be computed in $O(n_A \cdot n_B)$ time by just determining the new location of the black points using Equation (3). If one of the interior points becomes an extreme point or vice-versa (see Figure 4), the topology changes and C(A,B) has to be computed from scratch using Equation (2). The critical orientation $\theta_c$ when an interior point $P_{ij}$ becomes extreme is given by the following equation:

$$\begin{aligned} &\left[P_{ij}(\theta_c) - V_0^F(\theta_c)\right] \bullet n(\theta_c) = 0 \\ &P_{ij}(\theta_c) \times n(\theta_c) > 0. \end{aligned} \tag{4}$$

where $V_0^F$ is a vertex on one of the faces of C(A,B) and $\mathbf{n}$ is the normal of the same face.

The set $\Theta_c(A, B, r) \equiv \{\theta_{c1} < \theta_{c2} < \ldots < \theta_{cn\theta}\}$ of critical values for $\theta$ in the interval $[0, 2\pi)$ can be enumerated by solving a set of quadratic inequalities obtained using Equation (4) for every point $P_{ij}$. Within the interval $(\theta_{ci}, \theta_{ci+1})$, the topology of C(A,B) stays the same. Hence, this interval is called a constant topology orientation interval. The range $[0,2\pi)$ can be expressed as $[0,\theta_{c1}] \cup [\theta_{c1},\theta_{c2}] \cup \ldots \cup [0,2\pi)$. By computing geometry of C(A,B, $\theta$) for one orientation in every constant topology interval, the obstacle is completely characterized over the range $[0,2\pi)$.

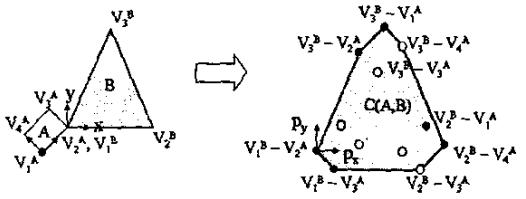



**Figure 4 Effect of Rotation of A on C(A,B,θ)**

## 3.2. Effect of Concavity of Φ and Γ on Computation of C(Φ,Γ)

First, consider the case when $\Phi$ and $\Gamma$ are convex. Let $C(\Phi,\Gamma,\theta)$ be known for $\theta \in [\theta_{cr-1},\theta_{cr}]$ and be described by faces $F_1, F_2, \ldots F_{nF}$. The position $\mathbf{p_d}$ lies inside $C(\Phi,\Gamma)$ if $\Phi$, when positioned at $\mathbf{p_d}$, interferes with $\Gamma$. For fixed $\theta$, the interference-free position closest to $\mathbf{p_d}$ lies on one of the faces. The position, closest to $\mathbf{p_d}$, lying on face $F_q$ with vertices $\{V_1^F, V_2^F, \ldots, V_{nq}^F\}$ over the interval $[\theta_{cr-1},\theta_{cr}]$, $Q_{rq}^* \equiv \{\mathbf{p}_{rq}^*, \theta_{rq}^*\}$, is the solution of the following non-linear problem:

$$\begin{aligned} &\min \left\| p_d - p_n^* \right\|^2 \\ &\textit{subject to} \\ &\sum_{k=1}^{n_q} w_k V_k^F(\theta_{rq}) - p_n^* = 0 \\ &\sum_{k=1}^{n_q} w_k = 1 \\ &w_k \geq 0,\ i = 1,2,\ldots,n_q \\ &\theta_{rq} \in (\theta_{cr-1}, \theta_{cr}) \end{aligned} \qquad (5)$$

This optimization problem can be solved using a quasi-Newton method like BFGS algorithm [13]. The objective function is smooth and the constraints are linear. The optimal configuration $Q^*$ from Section 2 is computed as the best configuration $Q_{pq}^*$ over all obstacle faces within a constant topology interval, and over all constant topology orientation intervals.

Computation of the optimal configuration is very expensive in the general case where $\Phi$ and $\Gamma$ are concave. First, $\Phi$ and $\Gamma$ are decomposed into sets of convex polyhedra $\{\Phi_i, i = 1, 2, \ldots, n\}$ and $\{\Gamma_j, j = 1, 2, \ldots, m\}$ respectively. The c-space obstacle $C(\Phi,\Gamma)$ is the union of the obstacles $\{C(\Phi_i,\Gamma_j), i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m\}$. The c-space obstacle $C(\Phi_i,\Gamma_j)$ is computed using Equation (2). The optimal position $\mathbf{p}^*$ lies on one of the faces of $C(\Phi,\Gamma)$ for some orientation $\theta^*$. A face of $C(\Phi,\Gamma)$ is either a face of one of its constituents, $C(\Phi_i,\Gamma_j)$, or a portion of such a face. The configuration $Q_{ij}^* = \{\mathbf{p}_{ij}^*, \theta_{ij}^*\}$, with $\mathbf{p}_{ij}^*$ lying on $C(\Phi_i,\Gamma_j,\theta_{ij}^*)$ is computed using Equation (5) and enforcing an additional constraint that $\mathbf{p}_{ij}^*$ lie outside the other constituent obstacles $C(\Phi_k,\Gamma_l,\theta_{ij}^*)$, $k \neq i$ or $l \neq j$. Consider one such obstacle $C(\Phi_k,\Gamma_l,\theta_{ij}^*)$. Let $G_1, G_2, \ldots, G_{nG}$ be the faces of $C(\Phi_k,\Gamma_l,\theta_{ij}^*)$. The exclusion constraint is equivalent to the following condition:

$$\exists r \in \{1,2,\cdots,n_G\} \textit{ such that } \left[p_{ij}^* - V_0^G(\theta_{ij}^*)\right] \bullet n_r(\theta_{ij}^*) \geq 0 \qquad (6)$$

where

$V_0^G$ is a vertex of face $G_r$ and

$n_r$ is the face normal of $G_r$.

From Equation (6), the number of non-linear problems to be solved to compute $Q_{ij}^*$ with $\mathbf{p}_{ij}^*$ lying outside $C(\Phi_k,\Gamma_l,\theta_{ij}^*)$ is at most $n_G$. The number of faces of a regular solid is of the order of the number of vertices [13]. Using Equation (2), we have $n_G = O(n_i n_j)$, where $n_i$ is the number of vertices of $\Phi_i$, and $n_j$ is the number of vertices of $\Gamma_j$. Since $Q_{ij}^*$ should satisfy Equation (6) for all $C_{kl}(\theta_{ij}^*)$, $k \neq j$ or $l \neq j$, we have to solve $n_i n_j^{mn}$ non-linear optimization problems. This computation has to be repeated for each constant topology orientation range. For the stacking problem, $n = M$, the number of convex components of the part, $m = NM$ where $N$ is the number of stack parts. The number of vertices of an obstacle $C(\Phi_i,\Gamma_j)$ is bounded by $V^2$, where V is the number of part vertices. Hence, if a new part is being added to an existing part stack, the number of problems to solve even for a single constant topology orientation range is $O(V^{2N(M^{\wedge}2)})$. This combinatorial explosion in the number of non-linear problems to be solved justifies the use of approximation methods with discretization in one or more axes to search for the optimal configuration.

## 3.3. Computation of $Q^*$ using Partial C-space Obstacles and Discrete Orientations

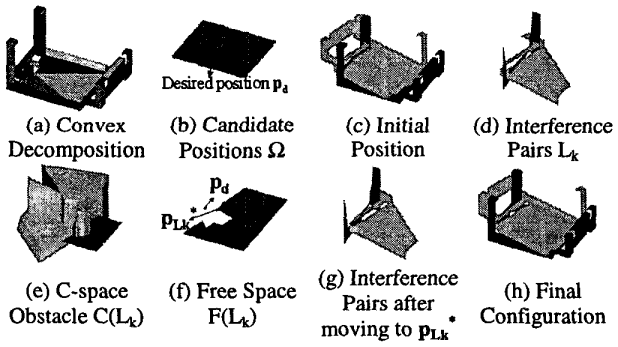
The high complexity of computing $Q^*$ arises primarily from the effect of change in orientation on the topology of $C(\Phi,\Gamma)$. The problem is simplified by considering only discrete orientations of $\theta$: $0, \Delta\theta, 2\Delta\theta, \ldots, 2\pi-\Delta\theta$. The quality of the final configuration computed is inversely proportional to the discretization step $\Delta\theta$. Hence, the discretization step should be chosen as small as possible. Once again $\Phi$ and $\Gamma$ are decomposed into sets of convex polyhedra $\{\Phi_i, i = 1, 2, \ldots, n\}$ and $\{\Gamma_j, j = 1, 2, \ldots, m\}$ respectively. For each discrete orientation $\theta_k$, the interference-free position $\mathbf{p}_k^*$ minimizing $\|\mathbf{p}_d - \mathbf{p}\|$ for

orientation $\theta_k$ is computed. The final configuration $Q^*$ is chosen as the best of all configurations $Q_k^* \equiv \{\mathbf{p}_k^*, \theta_k\}$.

Let $L_k$ be a subset of set of all possible convex polyhedral pairs $(\Phi_i,\Gamma_j)$, i = 1, 2, ..., n, and j = 1, 2, ..., m. Let $C(L_k)$ be the set of positions resulting in pair-wise interference between members of interference pairs of $L_k$ for orientation $\theta_k$. Clearly, $C(L_k)$ is a subset of $C(\Phi,\Gamma,\theta_k)$ and is obtained as follows:

$$C(L_k) \equiv \bigcup_{(\Phi_i,\Gamma_j)\in L_k} C(\Phi_i,\Gamma_j,\theta_k) \qquad (7)$$

Consider the region $F(L_k) \equiv \Omega - C(L_k)$. For any position (of $\Phi$) in $F_k$, there is no interference between the interference pairs of $L_k$. If $F(L_k)$ is empty, no interference-free position exists for orientation $\theta_k$. Otherwise, let $\mathbf{p}_{Lk}^* \in F(L_k)$ be the position closest to $\mathbf{p}_d$. Algorithm 1 shows the computation of $\mathbf{p}_{Lk}^*$. Whenever possible, $C(\Phi_i,\Gamma_j,\theta_k)$ is extrapolated from another orientation in the same constant topology orientation range. Otherwise, the obstacle is computed using Equation (2). In Figure 5, the part has 79 convex components. The initial configuration of $\Phi$, $\{\mathbf{p}_d,\theta_k\}$, is not interference-free. The space of permissible positions $\Omega$ is a region in $p_z = 0$ plane. The interference pairs are identified and the corresponding obstacle $C(L_k)$, free region $F_k$, and position $\mathbf{p}_{Lk}^*$ are computed. As no further interference is observed, $\mathbf{p}_{Lk}^* = \mathbf{p}_k^*$. From Figure 6, we can see that $C(L_k)$ is a small subset of $C(\Phi,\Gamma,\theta_k)$.




(a) Convex Decomposition (b) Candidate Positions $\Omega$ (c) Initial Position (d) Interference Pairs $L_k$

(e) C-space Obstacle $C(L_k)$ (f) Free Space $F(L_k)$ (g) Interference Pairs after moving to $\mathbf{p}_{Lk}^*$ (h) Final Configuration

**Figure 5 Computation of $\mathbf{p}_{Lk}^*$ using $C(L_k)$**

Algorithm 2 shows the procedure for computing $Q^*$ using partial c-space obstacle information. For each orientation $\theta_k = 0, \Delta\theta, 2\Delta\theta, \ldots, 2\pi-\Delta\theta$, the list $L_k$ of interfering convex polyhedral pairs (interference pairs) and corresponding position $\mathbf{p}_{Lk}^*$ are computed.

$$L_k \equiv \left\{(\Phi_i,\Gamma_j) \mid \Phi_i \cap \Gamma_j \neq \emptyset, i = 1,2,\cdots,n, j = 1,2,\cdots m\right\} \qquad (8)$$

This list can be constructed using algorithms for fast interference detection, e.g. RAPID [15]. The configuration $Q_{Lk}^* \equiv \{\mathbf{p}_{Lk}^*,\theta_k\}$ is stored in a priority queue with the associated cost $\|\mathbf{p}_d - \mathbf{p}_{Lk}^*\|$. The cheapest configuration is removed from the queue. The polyhedral set $\Phi$ is moved to configuration $Q_{Lk}^*$ and tested for interference with $\Gamma$. If no interference exists, $Q_{Lk}^*$ is the optimal configuration $Q^*$. Otherwise, the additional interference pairs encountered are added to $L_k$ and re-compute $\mathbf{p}_{Lk}^*$. The new configuration $\{\mathbf{p}_{Lk}^*,\theta_k\}$ is added to the priority queue with the new cost. If no valid $\mathbf{p}_{Lk}^*$ is found because $F_k$ is empty, the orientation $\theta_k$ is discarded. The process is repeated till an interference-free configuration is found or all orientations are discarded.

```
F_k ← Ω
For i = 1, 2, ..., n(L_k)
    If θ_k ∈ Θ
        Compute C(Φ_i,Γ_j,θ_k) using obsList and Eqn.(3).
    Else
        Compute C(Φ_i,Γ_j,θ_k) using Equation (2).
        Compute constant topology orientation range Θ_k
        Θ ← Θ ∪ Θ_k
        F_k ← F_k − C(Φ_i,Γ_j,θ_k)
        Add (C(Φ_i,Γ_j,θ_k), Θ_k) to obsList
    End if
End for
If F_k ≠ ∅
    Compute p_Lk* ∈ F_k minimizing ||p_d − p_Lk*||
End if
```

**Algorithm 1 Computation of $\mathbf{p}_{Lk}^*$ for an Convex Polyhedral Pair List $L_k$**

**Figure 6 C-Space Obstacle $C(\Phi,\Gamma,\theta_k)$**

```
p* ← {∞,∞,∞}, θ* ← 0°, Θ ← ∅
List obsList ← ∅, Priority Queue P ← ∅
For θ_k = 0, Δθ, 2Δθ, ..., 2π-Δθ
    Construct list L_k
    Compute p_Lk* using Algorithm 1
    If ||p_d − p_Lk*|| ≠ ∞
        Add (p_Lk*, θ_k, L_k, ||p_d − p_Lk*||) to P
    End if
End for
While P ≠ ∅
    Remove (p_Lk*, θ_k, L_k) from P
    Set configuration of Φ to {p_Lk*,θ_k}
    If Φ and Γ do not interfere
        Q* ← {p_Lk*, θ_k}
        Stop
    End if
    Add interference pairs (Φ_i,Γ_j) to L_k.
    Compute p_Lk* using Algorithm 1.
    If ||p_d − p_Lk*|| ≠ ∞
        Add (p_Lk*, θ_k, L_k, ||p_d − p_Lk*||) to P
    End if
End while
Q* ← {[∞ ∞ ∞]^T,0}
```

**Algorithm 2 Computation of $Q^*$ using Interference Pair Lists**

Algorithm 2 is essentially an A*-search. The heuristic cost at a node is the minimum distance to be moved by $\Phi$, from

the desired position $\mathbf{p_d}$, to avoid interference between member-pairs of $L_k$. This is a lower bound of the actual distance to be moved by $\Phi$ to avoid interference between $\Phi$ and $\Gamma$. Therefore, the method is guaranteed to perform as well as a discretization approach that considers all interference pairs for every discrete orientation. Further, this method is guaranteed to converge as every orientation $\theta_k$ eventually yields either a non-empty $F_k$ that can be used to determine $\mathbf{p_k}^*$ or a null-set implying that no solution exists for this particular orientation. The complexity of Algorithm 2 is $O(N_\theta \cdot n_L \cdot (n_\Phi \cdot n_\Gamma) log(n_\Phi \cdot n_\Gamma))$ where $N_\theta$ is the number of discrete orientations considered and $n_\Phi$ and $n_\Gamma$ are the number of vertices of $\Phi$ and $\Gamma$ respectively. The parameter $n_L$ is the total number of interference pairs considered and is the sum of the number of interference pairs in all $L_k$. The maximum value of $n_L$ is still the total number of interference pairs *mn*. However, we have seen through tests that for complex parts, $n_L << 0.01mn$.

## 4. Implementation and Results

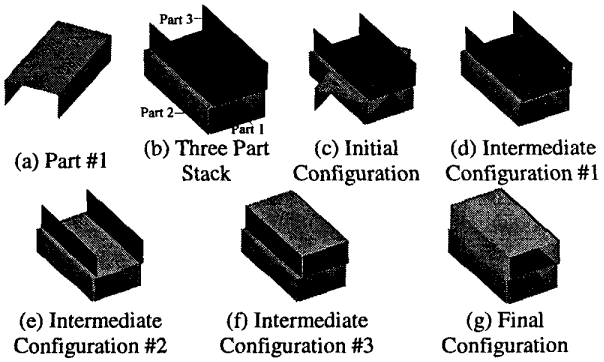
### 4.1. Implementation

Algorithm 2 has been used for planning stacking of polyhedral sheet metal parts. This implementation is part of a bigger project on automating process planning for sheet metal bending [16]. Sheet metal parts are difficult to stack because of flanges, holes and tabs. The algorithm described in the paper forms the interference analysis module of the stacking planner. The planner builds the stack incrementally and uses the interference analysis to evaluate candidate stacks. Stability based heuristics [1] are used to determine the parameters $\{\phi,\psi\}$ for the $i^{th}$ part. Thus, the c-space dimensionality is reduced from six to four: $\{p_x,p_y,p_z,\theta\}$. As far as the interference analysis module is concerned, when the $i^{th}$ part is added to a stack with *i-1* parts, the *i-1* parts are considered stationary. The algorithm has been implemented successfully using C++ on a PC with a 266 MHz Pentium Pro processor. The discretization step $\Delta\theta$ (see Section 3.3) is chosen as 1°. RAPID library [15] is used for interference detection, *qhull* library [17] is used for convex hull computation, and ACIS geometric kernel is used to model the parts and the c-space obstacles. Convex components of the sheet metal part are generated by triangulating all the concave faces of a zero sheet thickness model of the part and extruding the resulting polygons by the sheet metal thickness.

### 4.2. Results and Discussion

Two types of cases are presented. The first case is to add a new part to an existing three-part stack. The space of allowable positions $\Omega$ is three-dimensional. The desired position for the part $\mathbf{p_d}$ places the part c.g. at centroid of the stack base. The second case discussed restricts $\Omega$ to a plane parallel to the $p_z = 0$ plane and $\mathbf{p_d}$ is chosen as before. This case arises when the stacking planner has determined z-coordinate by choosing a set of edges and/or faces from the *i-1* parts to support the $i^{th}$ part [1]. That leaves only $\{p_x,p_y,\theta\}$ to be determined. For all the four parts, the stack orientation is chosen randomly and the initial new part orientation is chosen as 0°.

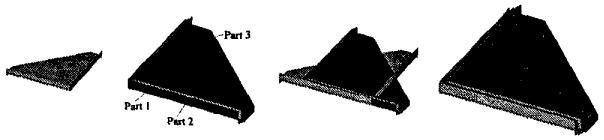
Parts #1 and #2 are considered for the first case. The corresponding stacks are shown in Figure 7 and Figure 8. In Figure 7(c), the new part interferes with some flanges of parts 1 and 2. The algorithm evaluates different orientations for the new part and chooses the orientation shown in Figure 7(d) (the new part is almost overlapping with part 1). In this configuration, the new part still interferes with parts 1 and 2, but the interference pairs are different. The algorithm moves the part up to avoid interference first with parts 1 and 2, and then finally with part 3. The final part configuration is such that the parts are nested. The algorithm works similarly for Part #2 in Figure 8 to produce a nested stack. This is one of the main advantages of using c-space representation. It helps us obtain nested stacks without explicitly looking for part features conducive to nesting.




(a) Part #1 (b) Three Part Stack (c) Initial Configuration (d) Intermediate Configuration #1

(e) Intermediate Configuration #2 (f) Intermediate Configuration #3 (g) Final Configuration

**Figure 7 A Four-Part Stack for Part #1**



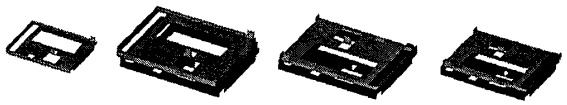

(a) Part #2 (b) Three Part Stack (c) Initial Configuration (d) Intermediate Configuration #1

(e) Intermediate Configuration #2 (f) Intermediate Configuration #3 (g) Final Configuration

**Figure 8 A Four-Part Stack for Part #2**

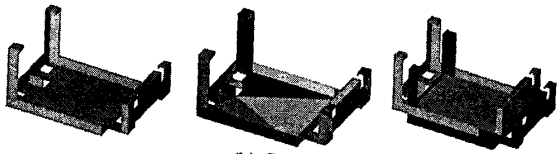
Parts #3 and #4 are tested for the application restricting $\Omega$ to a plane parallel to the $p_z = 0$ plane. The corresponding stacks are shown in Figure 9 and Figure 10. These parts have a large number of convex components and positioning them requires the accommodation of protruding flanges in holes. Once again, searching in c-space automatically enables us to accommodate flanges in holes without explicitly looking for such features.

However, the final configuration in Figure 10 might not be amenable to automated stacking as the flange is very close to the edge of the hole. Robot positioning error can cause parts to get tangled. One way of avoiding tangle is to prescribe an upper bound on robot positioning error and the final part configuration chosen should be at least that far from the c-space obstacle. Figure 10(b) shows the convex decomposition of the part into 79 convex components. One problem with the present triangulation scheme is evident from part configurations in Figures Figure 10(e) and Figure 10(f). In the configuration in Figure 10(e), there is a small triangular component of the moving part that is interfering with a component of the stationary part. An additional step is required to compute the small translation to the interference-free configuration in Figure 10(f). Such steps can be avoided by combining narrow and small triangles with neighboring triangles to create larger convex components.



(a) Part #3 (b) Initial Configuration (c) Intermediate Configuration (d) Final Configuration

**Figure 9 Planar Configuration for Part #3**



(a) Part #4 (b) Convex Components (c) Initial Configuration

(d) Intermediate Configuration #1 (e) Intermediate Configuration #2 (f) Final Configuration

**Figure 10 Planar Configuration for Part #4**

The performance of the algorithms for the four parts is shown in Tables 1-3. The times shown Table 1 are total computation time, time spent in interference checking, constructing c-space obstacles from scratch, and constructing them by extrapolation. Please note that the total computation time $t_{total}$ includes time taken for functions other than the above mentioned ones. Table 2 shows the computation time per function call. Table 3 shows the number of potential interference pairs $\{(\Phi_i, \Gamma_j) \mid i = 1, 2, \ldots, n, j = 1, 2, \ldots, m\}$ and the actual number of interference pairs for which c-space obstacle is computed. The interference detection routine takes only 0.1% of the time it takes to compute a c-space obstacle (see Table 2). It ends up analyzing up to 66% of all interference pairs for the simple parts #1 and #2 (see Table 3). This is not a problem as c-space obstacle computation for such parts is not time-consuming. For the complex parts #3 and #4, it analyzes less than 0.5% of the potential interference pairs (see Table 3) for c-space obstacle computation. This prevents the computation time from increasing rapidly as the number of convex components increases for a part. The extrapolation method for constructing C-space obstacles (presented in Section 3.1) takes only 10% of the time it takes to compute the obstacle from scratch (see Table 2). This enables us to choose a fine discretization step, 1°, for $\theta$. As expected, planar configuration problem for the more complex parts takes the same as the three-dimensional problem for the simpler parts.

| Ω | Part | $t_{interference}$ (s) | $t_{construction}$ (s) | $t_{extrapolation}$ (s) | $t_{total}$ (s) |
|---|---|---|---|---|---|
| 3 D | Part #1 M = 3 | 30.386 | 5.12 | 8.101 | 188.420 |
| | Part #2 M = 6 | 58.937 | 22.567 | 14.414 | 298.239 |
| 2 D | Part #3 M = 46 | 94.305 | 1.254 | 1.915 | 124.569 |
| | Part #4 M = 79 | 151.198 | 2.803 | 8.654 | 271.641 |

**Table 1 Computation Time for Interference Detection and C-space Obstacle**

| Ω | Part | $t_{interference}$ (s/Call) | $t_{construction}$ (s/Call) | $t_{extrapolation}$ (s/Call) |
|---|---|---|---|---|
| 3 D | Part #1 M = 3 | 0.0014 | 0.0105 | 0.0013 |
| | Part #2 M = 6 | 0.0015 | 0.0155 | 0.0016 |
| 2 D | Part #3 M = 46 | 0.0001 | 0.0104 | 0.0013 |
| | Part #4 M = 79 | 6.3e-5 | 0.0111 | 0.0016 |

**Table 2 Computation Time Per Function Call**

| Ω | Part | Total No. of Interference Pairs | No. of Interference Pairs Evaluated |
|---|---|---|---|
| 3 D | Part #1 M = 3 | 9720 | 6960 |
| | Part #2 M = 6 | 38880 | 9951 |
| 2 D | Part #3 M = 46 | 761760 | 1587 |
| | Part #4 M = 79 | 2246760 | 6751 |

**Table 3 No. of Interference Pairs Evaluated to Compute Q***

## 4.3. Limitations

All the configurations generated by the c-space approach satisfy the necessary condition for part stability, i.e., part c.g. should fall inside the stack base. However, no information is available currently about satisfaction of a

sufficient condition for stability: presence of part-part contacts that constrain the part and prevent it from tipping over. It will be useful to identify c-space regions that provide at least three contacts for the parts. This information combined with the necessary condition for stability will make searching for stable part configurations easier. Currently, the algorithm does not exploit part symmetry to reduce the number of discrete orientations considered. This might result in significant savings for certain part shapes.

## 5. Conclusions and Future Work

This paper shows that the complexity of using c-space based techniques for optimal polyhedral placement is $O(V^{2NMM})$. The paper also presents an extrapolation method to speed up c-space obstacle computation. The interval of orientations within which the topology of the c-space obstacle stays constant is obtained by solving a set of quadratic inequalities. A heuristic based method working with discrete orientations is presented. The method uses a small discretization step by taking advantage of the constant topology orientation interval. The final configuration is determined by looking at a small percentage of the potential interference pairs.

For applications like stacking, it will be advantageous to build in information about specific contacts (edge-face, face-face etc.) into the search for interference-free configuration. This will eliminate an additional stage of verifying that each part is constrained to be stable by contacts with the other parts. We also plan to work on using c-space information to determine if a stack is robust to robot positioning errors and tangling of parts.

## Acknowledgements

We thank US Amada Inc. for supporting this work. We would like to thank Mr. Duane Williams for his suggestions regarding the abstract and the paper. Thanks are also due to Dr. Cheng-Hua Wang for help with software development.

## References

[1] V.R. Ayyadevara and R.H. Sturges, "Automated Planning for Stacking of Bent Sheet Metal Parts," *Proc. 1997 ASME DETC*, Sacramento, CA, Sept. 1997.

[2] B.R. Donald, "A Search Algorithm for Motion Planning with Six Degrees of Freedom," *Artificial Intelligence*, Vol. 31, No. 3, pp. 295-353, 1987.

[3] L.E. Kavraki, P.Svestka, J.C. Latombe, and M. Overmars, "Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces," *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 4, pp. 566-580, 1996.

[4] T. Lozano-Perez, "Spatial Planning: A Configuration Space Approach," *IEEE Transactions on Computers*, Vol. 32, No. 2, pp. 108-120, Feb. 1983.

[5] J.T. Schwartz and M.A. Sharir, "On the Piano Movers' Problem I: The Case of a Two-Dimensional Rigid Polygonal Body Moving amidst Polygonal Barriers," *Communications on Pure and Applied Mathematics*, Vol. 36, pp. 345-398, 1983.

[6] J.T. Schwartz, "On the Piano Movers' Problem V: The Case of a Rod Moving in Three-Dimensional Space amidst Polyhedral Obstacles," *Communications on Pure and Applied Mathematics*, Vol. 37, pp. 815-48, 1984.

[7] T.F. Stahovich, *SketchIT: A Sketch Interpretation Tool for Conceptual Mechanical Design*, MIT Artificial Intelligence Laboratory Technical Report 1573, Mar. 1996.

[8] L. Joskowicz and E. Sacks, *Computer-Aided Mechanical Assembly Design Using Configuration Spaces*, Purdue University CS Tech Report 97-001, 1997.

[9] E. Sacks and C. Bajaj, "Sliced Configuration Spaces for Curved Planar Bodies," *International Journal of Robotics Research*, Vol. 17, No. 6, pp. 639-651, 1998.

[10] F. Avnaim and J.D. Boissonnat, *Polygon Placement under Translation and Rotation*, INRIA Report #899, Institut National de Recherche en Informatique et en Automatique, France, August 1988.

[11] R.C. Brost, *Analysis and Planning of Planar Manipulation Tasks*, Ph.D. Thesis, Carnegie Mellon University, January 1991.

[12] Z. Li and V. Milenkovic, "Compaction and Separation Algorithms for Non-Convex Polygons and Their Applications," *European Journal of Operational Research*, Vol. 84, pp. 539-561, 1995.

[13] J. O'Rourke, *Computational Geometry in C*, Cambridge University Press, 1994.

[14] S.G. Nash and A. Sofer, *Linear and Nonlinear Programming*, McGraw Hill, 1996.

[15] S.C. Gottschalk, M.C. Lin, and D. Manocha, *OBB-Tree: A Hierarchical Structure for Rapid Interference Detection*, Univ. of North Carolina CS Tech. Report TR96-013, 1996.

[16] S.K. Gupta, D.A. Bourne, K.H. Kim, and S.S. Krishnan, "Automated Process Planning for Sheet Metal Bending Operations," *Journal of Manufacturing Systems*, Vol. 17, No. 5, pp. 338-360, 1998.

[17] C.B. Barber, D.P. Dobkin, and H.P. Huhdanpaa, "The Quickhull Algorithm for Convex Hulls," *ACM Transactions on Mathematical Software*, Vol. 22, No. 4, pp. 469-483, Dec. 1996.