Subassembly Stability

Nico Boneschanscher
Hans van der Drift
Laboratory for Manufacturing Systems
Department of Mechanical Engineering
Delft University, The Netherlands

Stephen J. Buckley
Russell H. Taylor
Manufacturing Research
IBM T.J. Watson Research Center
Yorktown Heights, NY

Abstract

Planning a product assembly requires that we determine the order in which the product subparts are to be assembled. One constraint on this ordering is that the subassembly must be stable at each stage under the gravitational force and the insertion force of the next part to be assembled. In this paper, we discuss the stability problem for the case where the subassembly sits on a table. A program has been written to solve this problem for a class of subassemblies. The input to the program consists of a model of the subparts and their interconnections, and a set of external insertion forces. The program tests whether the total disturbance force is contained in the set of all stable forces between each subpart and the table. A linearized model of friction in six dimensions is used in the computation.

1.0 Introduction

In recent years, attention has focused on the need for an automatic system for planning product assemblies (Lozano-Pérez 1976, Taylor 1976, Liebermann and Wesley 1977, Lozano-Pérez et al 1987). Planning an assembly requires that we determine the order in which the product parts are to be assembled. In general, this is a difficult problem, because of the large number of possible solutions (Homem de Mello and Sanderson 1986). One constraint on this ordering is that the subassembly must be stable at each stage under the gravitational force and the insertion force of the next part to be assembled. In this paper, we discuss this stability problem for the case where a three-dimensional subassembly sits on a table, and the insertion force is given. We will assume that the parts of the subassembly can be accurately modeled by rigid polyhedra.

Fahlman (1973) investigated the stability of a subassembly of blocks (bricks and right triangular wedges), using an iterative numerical method which propagates forces through the subassembly. The correctness and computational complexity of his method have not been established.

Blum, Griffith, and Neumann (1970) implemented a subassembly stability test based on linear programming. For each part, force balance equations are written in terms of point contacts with adjacent parts and gravitational forces. Friction at each contact point is approximated by four linear inequalities. The program searches for a simultaneous solution to the force balance equations in which the forces acting on each body are either zero or internal to the body.

Palmer (1987) established the computational complexity of the subassembly stability problem for rigid polygons in the plane. He showed that **guaranteed stability** is NP-hard, under assumptions similar to what we call "limited superposition" in "2.2 Multi-Point Friction" on page 2. He also showed that **potential stability** is in P, and that for a special class of subassemblies, guaranteed stability without friction is in P.

This paper is organized as follows. "2.0 Friction" dicusses our representation for friction. Our stability algorithm is then described in "3.0 Algorithm" on page 3. "4.0 Experiments" on page 5 presents some of our experimental results. "5.0 Conclusions" on page 5 summarizes our contributions.

2.0 Friction

Consider Figure 1, which shows an insertion force applied to a two-dimensional block on a table. To compute the stability of the block with respect to the table, we must take friction into account.

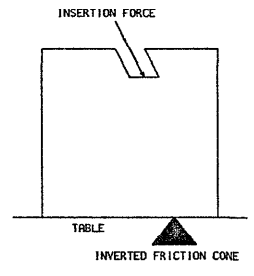

Figure 1. An insertion force applied to a two-dimensional block on a table.

The **friction cone** of the table with respect to the block is defined as the set of possible reaction forces that the table can exert in response to an applied force from the block (Coulomb, see Baumeister 1978). The angle of the friction cone is equal to $arctan(\mu)$, where μ is the coefficient of friction between the block and the table. Coulomb made the following empirical observation about friction cones: *if an applied force opposes a possible reaction force in the friction cone, then an equal and opposite reaction force will be generated, and no relative motion will occur.* The inverted friction cone of the table with respect to the block thus contains the set of stable forces that can be exerted on the block at the point of contact.

2.1 Point Friction

In this subsection, we present a mathematical formulation of the inverted friction cone for a point contact in three dimensions. Our formulation is taken directly from Erdmann (1984).

For a point contact in three dimensions, the number of degrees of freedom is six. However, only the three translational degrees of freedom are subject to friction. Thus, the friction cone is a three-dimensional subset of six-dimensional space. The inverted friction cone can be derived by writing down the equations for static equilibrium of an applied force/torque $\bar{f}$ at the contact point. The following constraints hold:

1. The applied torque must be zero.
2. The applied force must be interior to the contact surface.
3. (Coulomb's Law) The tangential component of the applied force must be less than or equal to μ times the normal component of the applied force.

Erdmann showed that these constraints can be written as the following system of linear equations:

$$\bar{f}_r = 0 \qquad [2.1]$$

$$\bar{f} \bullet \bar{n} \leq k_1(\bar{f} \bullet \bar{t}_{\wp}) \qquad [2.2]$$

$$\bar{f} \bullet \bar{n} \leq k_2(\bar{f} \bullet \bar{t}_{\wp}) \qquad [2.3]$$

$$\bar{f} \bullet \bar{t}_{\perp} = 0 \qquad [2.4]$$

where:

$$k_1 = \frac{\|\pi_T(\bar{t}_{\wp})\| - \mu\|\pi_N(\bar{t}_{\wp})\|}{\mu\|\pi_N(\bar{n})\|} \quad k_2 = \frac{\|\pi_T(\bar{t}_{\wp})\| + \mu\|\pi_N(\bar{t}_{\wp})\|}{-\mu\|\pi_N(\bar{n})\|}$$

$\bar{f}_r \equiv$ torque at the contact point
$\bar{n} \equiv$ six-dimensional outward normal vector
$\bar{t}_{\wp} \equiv$ a pure sliding vector
$\bar{t}_{\perp} \equiv$ a pure sliding vector perpendicular to $\bar{t}_{\wp}$
$\pi_T \equiv$ orthogonal projection onto the real space tangent plane
$\pi_N \equiv$ orthogonal projection onto the real space normal.

The constants $\bar{n}$, $\bar{t}_{\wp}$, and $\bar{t}_{\perp}$ can be computed from geometric models of the parts in contact. There are many possible choices for $\bar{t}_{\wp}$ and $\bar{t}_{\perp}$. For a particular choice, Equations 2.1-2.4 define a two-dimensional slice of the inverted friction cone. A linearized three-dimensional cone can be obtained by taking the Minkowski sum of a finite number of cone slices. In practice, we found that eight slices give a fairly accurate approximation. Let $\bar{f}_1 \ldots \bar{f}_8$ represent stable force/torques in the eight slices, where each $\bar{f}_i$ is defined similar to $\bar{f}$ in Equations 2.1-2.4. We can then write the linearized cone as

$$\bar{f}_c = \bar{f}_1 + \bar{f}_2 + \cdots + \bar{f}_8. \qquad [2.5]$$

Equation 2.5 can be combined with all of the slice equations in a system of the form

$$A\bar{x} = \bar{0}$$
$$B\bar{x} \geq \bar{0}$$

where $\bar{x}$ consists of $\bar{f}_c, \bar{f}_1, \bar{f}_2, \ldots, \bar{f}_8$, and $\bar{f}_c$ represents the forces in the inverted friction cone.

2.2 Multi-Point Friction

When two parts are in n-point contact, each possible reaction force can be viewed as a nonnegative linear combination of n forces, each of which comes from the friction cone of one of the contact points. We call this the **full superposition** assumption. Under full superposition, we can write the set of possible reaction forces of an n-point contact as

$$F_{composite} = \bigoplus_{i=1}^{n} F_i \qquad [2.6]$$

where $F_1 \ldots F_n$ are the friction cones at the contact points, and $\oplus$ denotes Minkowski sum. $F_{composite}$ can be viewed as the **composite friction cone** of the n-point contact. We will assume that Coulomb's criterion for stability extends to composite friction cones under multi-point contact. Note that the composite inverted friction cone of an n-point contact can be computed using Equation 2.6 by first inverting the point friction cones F_i.

There is reason to believe that some of the reaction forces under full superposition are impossible, or at least unreliable. If the applied force is exterior to some subset of the contact surfaces, then it is difficult to believe that these contact surfaces contribute to the reaction force. Let denote the set of outward normal vectors at the n contact points. Then we might write the composite friction cone as

$$F_{composite} = \bigcup_{M \subseteq N} \left[\left(\bigcap_{\bar{n} \in M} exterior(\bar{n}) \right) \cap \left(\bigoplus_{\bar{n} \in M} F(\bar{n}) \right) \right]$$

where

$$exterior(\bar{n}) = \{\bar{x} \mid \bar{x} \bullet \bar{n} > 0\}.$$

We call this the **limited superposition** assumption. For example, consider the planar two-point contact shown in Figure 2. For simplicity, ignore rotations. F_1 and F_2 represent the friction cones of the contact points. If we assume full superposition, then the composite friction cone of B with respect to A is given by $F_1 \oplus F_2$, which yields the entire plane. This implies that A may not slide at all. If we assume limited superposition, then the composite friction cone is equal to $F_1 \cup F_2$. (The applied force can never be strictly interior to both surfaces.) This implies that sliding will occur as long as the applied force is not in either of the inverted friction cones.

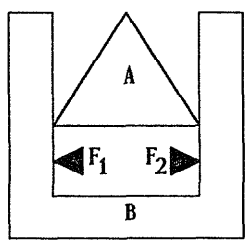

Figure 2. A planar two-point contact.

Ramifications of full and limited superposition on stability will be discussed later in this paper.

3.0 Algorithm

In this section, we will describe an algorithm which assumes full superposition. In general, the algorithm computes only potential stability. In "3.4 Guaranteed Stability" on page 5, we will discuss cases for which we conjecture that the algorithm computes guaranteed stability.

The input to the program is a geometric model of a subassembly and a set of disturbance forces. The modeler knows a set of basic bodies, namely cuboids, cylinders, cones and wedges, all represented as polyhedra. It also knows a basic set of contact situations:

AGAINST (vertex, face)	A vertex against a face.
INSIDE (face1, face2)	Face 1 is contained in face 2.
TIED (part1, part2)	Part 1 is rigidly attached to part 2.

The output of the algorithm is the set of orientations for which the subassembly will be stable on a table.

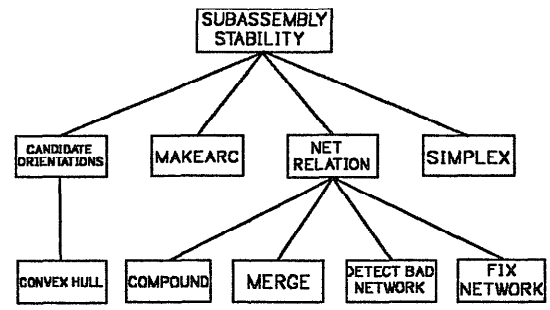

Figure 3. The structure of the program.

Figure 3 depicts the structure of the program. "Subassembly Stability" is the block controlling the entire flow of the algorithm. It uses "Candidate Orientations", "Makearc", "Net Relation" and "Simplex" to find all stable orientations.

"Candidate Orientations" determines all orientations in which the subassembly is stable on a table, assuming that the parts are rigidly attached to each other. With a convex hull algorithm, all "first guess" orientations are found. Then, orientations in which the center of gravity of the complete subassembly is not supported are pruned, leaving the set of orientation candidates.

Next, we relax the assumption that the parts are rigidly attached. In "Makearc", the geometric model is translated into a network of parts. A relation is created in the network for each pair of parts in contact. Then, for each part in the network, the network is reduced to a single relation between the part and the table in "Net Relation". The final relation is tested for stability in "Simplex".

If for a particular orientation all parts are found to be stable, then the orientation is stable. If one or more parts are found to be unstable, then the orientation is unstable.

3.1 Makearc

"Makearc" generates a relation for each contact point in the subassembly. A relation connects two parts, describing the stable forces that one of the parts can exert on the other through the contact point (ignoring the rest of the subassembly). The stable forces at a contact point are given by the inverted friction cone at that point, which is represented by the system of linear equations described in "2.1 Point Friction" on page 2.

3.2 Net Relation

"Net Relation" reduces a network of part relations to a single relation between a part and the table. Similar to strategies described by Smith and Cheeseman (1986), the actions to reduce the network are **merging** and **compounding**. Merging is performed when there are two par-

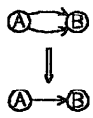

Figure 4. A simple merge action.

allel relations between two parts (see Figure 4). Assuming full superposition, merging is accomplished by computing the Minkowski sum of the two relations, and results in a single relation between the parts. In a compound action, a chain of two relations is reduced to a single relation by intersecting them, thereby eliminating the middle part in the chain (see Figure 5).

Consider Figure 6, which represents an abstract network of three parts. We are interested in determining the net relation of part A relative to part C. Network reduction is accomplished by iteratively repeating the following steps until a single relation remains:

1. Merge until no longer possible.
2. Compound until no longer possible.

In Figure 6, a compound action is performed first, eliminating part B. Then, the two remaining relations are merged to produce a single relation between A and C.

Sometimes, neither compounding nor merging is possible during the reduction of a network. Two possible causes for this are **bidirectional relations** and **loops**. Assume that we are interested in the stability of a part A with respect to the table. A bidirectional relation is a relation between two parts in which forces can propagate in either direction on the way from A to the table. A loop is a circular path of relations on the way from A to the table, but not including A or the table. When these structures occur, transformations must be performed on the network so that reduction can continue. We are investigating algorithms to perform these transformations.

3.3 Implementation of Merging and Compounding

Each initial relation in the network represents an inverted friction cone, represented by a system of linear equations. Let F_1 represent an initial relation, given by the following linear equations:

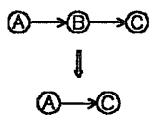

Figure 5. A simple compound action.

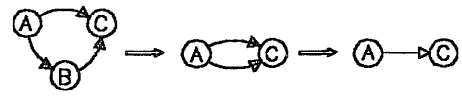

Figure 6. An assembly requiring merging and compounding.

$$A\bar{x} = \bar{0}$$
$$B\bar{x} \geq \bar{0}$$

Let $\bar{f}_1$ represent the stable disturbance forces in this system. Let F_2 represent another initial relation, given by the following linear equations:

$$C\bar{y} = \bar{0}$$
$$D\bar{y} \geq \bar{0}$$

Let $\bar{f}_2$ represent the stable disturbance forces in this system. During the reduction of a network, several merge and compound actions are performed, each adding new constraints to the initial system of equations. In the case of a merge action involving F_1 and F_2, the following six constraints are introduced:

$$\bar{g} = \bar{f}_1 + \bar{f}_2$$

$\bar{g}$ represents stable disturbance forces in the merged relation. It can be seen that merging is the Minkowski sum of the two relations. In the case of a compound action, the following six constraints are introduced:

$$\bar{f}_1 = \bar{f}_2$$

$\bar{f}_1$ represents stable disturbance forces in the compounded relation. It can be seen that compounding is the intersection of the two relations. This is a simplified version, without disturbance forces acting on the part to be eliminated. Disturbance forces can be added to the framework fairly easily.

Previous constraints remain undisturbed during network reduction. When the reduction is complete, one set of six variables represents the net relation of the part relative to the table. The test for stability is accomplished by assigning the disturbance force on the part in question to the final set of six variables, resulting in six additional equations. The complete system of equations is then passed to a Simplex program, which determines their feasibility.

3.4 Computational Complexity

This subsection gives the average running time of the stability algorithm. Our analysis assumes full superposition, and does not necessarily hold if bidirectional relations and loops exist in the subassembly.

Let m be the number of parts in the subassembly, and n the number of vertices. Then there are $O(n)$ initial relations in the subassembly network. Since each merge and compound action eliminates one relation from the network, network reduction can be performed in $O(n)$ steps. The output of network reduction is a system of $O(n)$ equations in $O(n)$ variables. These equations are passed to the Simplex program, which is known to run in time $O(n^3)$ on the average. Thus, it takes $O(n^3)$ steps on the average to determine the stability of a single part. Furthermore, it takes $O(mn^3)$ steps on the average to determine the stability of an entire subassembly at a given orientation.

In extreme cases, it is possible for the Simplex algorithm to take exponential time. Polynomial worst-case time can be guaranteed by substituting Karmarkar's algorithm (Karmarkar 1984).

3.4 Guaranteed Stability

Palmer (1987) proved that for subassemblies of polygons in the plane without friction, if no contact points have an interior angle of less than π on both sides of the contact, then the problem of guaranteed stability is in P, and can be solved by linear programming. We conjecture that this result extends to three dimensions by taking the minimum interior angle on both sides of each contact point. If so, then our algorithm computes guaranteed stability for a class of frictionless subassemblies, including all subassemblies that consist of stacked rectangular parts.

Another conjecture is that an algorithm based on limited superposition will compute guaranteed stability in the presence of friction. In many cases, limited superposition and full superposition yield the same result (the next section presents one such case). In these cases, we conjecture that our current algorithm computes guaranteed stability in the presence of friction.

We are attempting to establish these conjectures. In addition, we are investigating algorithms to:

- identify cases where limited and full superposition are equivalent.
- compute guaranteed stability for cases where limited and full superposition are not equivalent.

Although the latter problem is in general NP-hard, we anticipate that many practical cases can be solved by efficient search procedures.

4.0 Experiments

The described algorithm has been implemented in AML/X, linked to a Fortran Simplex routine. It has been tested on a number of geometric models. Figure 7 summarizes one of our tests. In this test, μ was equal to 0.1.

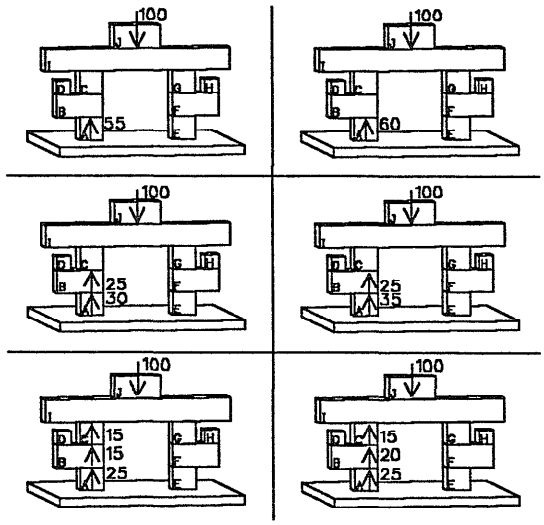

Figure 7. The right hand cases are unstable.

5.0 Conclusions

An algorithm to determine the static stability of a subassembly on a table has been developed. It converts a geometric model into a network of parts, and represents the relations between the parts as linear equations. A linearized model of friction in six dimensions is used to represent the contact situations in the subassembly. After reducing the network to a net relation between a part and the table, linear programming is used to determine the stability.

In general, the algorithm computes potential stability. We conjecture that it computes guaranteed stability in the absence of friction when no contact points exist which have minimum interior angles of less than π on both sides of the contact. We also conjecture that the algorithm computes guaranteed stability for cases where limited and full superposition are equivalent. We are investigating algorithms to recognize these cases, and to compute guaranteed stability for cases where limited and full superposition are not equivalent.

Our method is similar to that of Blum, Griffith, and Neumann (BGN), in the sense that we too use linear programming. However, we have gone beyond the BGN results in the following ways:

1. When a subassembly is unstable, the BGN algorithm does not indicate which parts are unstable. Since this information is useful to an assembly planner, we compute the stability of each part individually.
2. We address the issue of external insertion forces, while BGN limits its scope to gravitational forces.

3. Our method for linearizing friction is based on Erdmann's equations, rather than the BGN method. Erdmann's equations allow an arbitrary number of faces in a linearized friction cone, while the BGN method appears to be limited to four faces.
4. The BGN paper implicitly assumes full superposition. We have identified the alternative concept of limited superposition, and we are investigating an efficient algorithm which uses it to compute guaranteed stability.
5. We are investigating an algorithm to compute the **robustness** of a stable insertion force; that is, the "closeness" of a stable insertion force to an unstable force.

Acknowledgments

We would like to acknowledge contributions from the following people: Bela Musits, for valuable comments on the work; John Forrest, for advice on the Simplex program; Mike Erdmann, for advice on friction; Wally Dietrich and Lee Nackman, for advice on AML/X; V.T. Rajan, for advice on friction and linear programming; and Bob Wittrock, for advice on linear programming.

References

1. Baumeister, T., editor 1978. **Marks' Standard Handbook for Mechanical Engineers**, McGraw-Hill.
2. Blum, M., Griffith, A., and Neumann, B. 1970. "A Stability Test for Configurations of Blocks", AI Memo 188, MIT Artificial Intelligence Laboratory.
3. Erdmann, M. 1984. "On Motion Planning With Uncertainty", S.M. dissertation, MIT Department of Electrical Engineering and Computer Science, also AI-TR-810, MIT Artificial Intelligence Laboratory.
4. Fahlman, S. 1973. "A Planning System for Robot Construction Tasks", AI-TR-283, MIT Artificial Intelligence Laboratory.
5. Homem de Mello, L., and Sanderson, A. 1986. "AND/OR Graph Representation of Assembly Plans", CMU-RI-TR-86-8, The Robotics Institute, Carnegie-Mellon University.
6. Karmarkar, N. 1984. "A New Polynomial-time Algorithm for Linear Programming", **ACM Symposium on the Theory of Computing** 16, pp. 302-311.
7. Lieberman, L., and Wesley, M. 1977. "AUTOPASS: An Automatic Programming System for Computer Controlled Mechanical Assembly", **IBM Journal of Research and Development** 21(4), pp. 321-333.
8. Lozano-Pérez, T. 1976. "The Design of a Mechanical Assembly System", S.M. dissertation, MIT Department of Electrical Engineering and Computer Science, also AI-TR-397, MIT Artificial Intelligence Laboratory.
9. Lozano-Pérez, T., Jones, J., Mazer, E., O'Donnell, P., Grimson, W.E.L., Tournassoud, P., and Lanusse, A. 1987. "Handey: A Robot System That Recognizes, Plans, and Manipulates", **IEEE International Conference on Robotics and Automation**, Raleigh, North Carolina, pp. 843-849.
10. Palmer, R. 1987. "Computational Complexity of Motion and Stability of Polygons", Ph.D. dissertation, Cornell University.
11. Smith, R., and Cheeseman, P. 1986. "On the Representation and Estimation of Spatial Uncertainty", **International Journal of Robotics Research** 5(4), pp. 56-68.
12. Taylor, R. 1976. "A Synthesis of Manipulator Control Programs From Task-Level Specifications", Ph.D. Dissertation, Stanford University, also AIM-282, Stanford Artificial Intelligence Laboratory.