Atlas: An Automatic Assembly Sequencing and Fixturing System

Bruce Romney

Dept. of Computer Science
Stanford University
bruce@flamingo.stanford.edu

Abstract. We present a method to generate automatically both an assembly sequence and a fixture to hold all the intermediate subassemblies, given only a geometric description of the product. The main contribution here is that the sequence and fixture are generated concurrently rather than sequentially, unlike in previous assembly-planning work. Also, the representation of possible fixturing locations is more complete. This method has been implemented for the planar case, and we present experimental results below. We also discuss some of the key issues involved in the concurrent design of assembly sequences and fixtures.
The long-term vision for this work is a CAD tool which can provide immediate feedback to a product designer about the ease of assembling the proposed product.

1 Introduction

Early mechanical CAD systems were little more than electronic draftboards, providing some part-interference information but leaving most advanced analysis to the designer. Over the years, though, as geometric modelling and related CAD technology have improved, CAD systems have been able to provide increasingly sophisticated feedback to the designer, not only about the product's functionality but also about its manufacturability and serviceability. These advances now allow the designer to quickly and automatically identify potential problems in these areas, and devote more attention to higher-level considerations. Recently, for example, a number of prototype systems have been presented which can automatically generate assembly sequences for a given product [1, 2, 3, 4, 5].

The purpose of the present research is to increase this level of sophistication still further, by automatically generating not only the assembly sequence but also the mechanical fixture required to hold the product in place during assembly. Assembly fixturing and stability have a crucial impact on the feasibility of assembly sequences, so a system to generate both a fixture and a sequence would be substantially more useful than a sequencer alone. Now, one could conceivably generate first one and then the other; but given the strong interdependence between sequencing and fixturing, it seems reasonable to expect that an approach which combines the two analyses could potentially produce better results. We refer to this technique as sequence/fixture *co-design*, and we employ it in Atlas, the system described below. Although the current implementation of Atlas is restricted to planar assemblies, all of its techniques extend to the three-dimensional case as well.

A sequencing and fixturing system such as Atlas, when fully developed and incorporated into a CAD tool, would be of tremendous value to a team of product designers. It could alert them to hidden problems in the assemblability of a product, or, conversely, present non-obvious solutions to known problems. This would be all accomplished "on line", without the need to construct a prototype.

In this paper, we begin by describing some of the past work in sequencing and fixturing in section 2. In section 3, we then formally define the problem and state our assumptions. Next, in section 4, we describe Atlas' basic operating procedures. In section 5, we explore Atlas' decision-making process from the perspective of wrench space. In section 6, we present some experimental results. Finally, in section 7, we discuss some directions of future research and conclude.

2 Related Work

The problem of automatically generating assembly sequences is, in its full generality, PSPACE-hard [6, 7]. Even when every operation moves exactly two subassemblies into their final relative positions, the problem is still NP-hard [8]. As a result, much of the past and present work in this area focuses on even more restricted variants of the problem. For example, some systems insert parts only along the major axes [3, 9, 10, 11], while others insert only one part at a time [2, 4]. For the case of binary, monotone sequences using straight-line part insertions from any direction, Wilson and Latombe presented a complete polynomial-time algorithm in [12] using the *non-directional blocking graph (NDBG)* data structure.

One common thread that appears in much of the literature is the strategy of "assembly by disassembly", in which an assembly sequence is generated by starting with the completed product and working backwards through disassembly steps. We shall use this strategy as well.

The assembly-sequencing portion of Atlas is based on the STAAT system [1] and the related concept of the local translational freedom cone [13]. The approach has some similarities to techniques described by other authors [2, 5, 14, 15]. However, little consideration is given in these papers to the problem of fixturing the intermediate subassemblies.

There is a great deal of literature on the fixturing or grasping of a single part. Much of it focuses on establishing form closure [16, 17, 18, 19, 20, 21] or force closure [22, 23, 24] on that part. Some authors also propose figures of merit with which to compare different form- or force-closing configurations, often based on the necessary contact forces. However, in sequence/fixture co-design, even when treating an assembly as a single unit, we are not necessarily interested in a form-closing fixture; on the contrary, such a fixture would often be too restrictive to allow part removals or insertions. Instead, we merely need to stabilize the assembly and all subassemblies against gravity and the *insertion forces* (see below, section 3.1), while still admitting free insertion paths for the parts. Also, because of the difficulty of the combined sequencing/fixturing problem, this paper focuses on ways to find *one* feasible fixture and sequence, leaving the task of comparing multiple solutions to future efforts.

The field of *assembly* fixturing, in which all parts may move independently, is far less represented in the literature than that of part fixturing. Some early work was done by Fahlman [25] for structures of simple blocks. However, Fahlman himself acknowledged his stabilization techniques to be "a loose grouping of a very large number of essentially independent tricks". Later, Wolter and Trinkle [26] and Mattikalli [27] described a more systematic approach to analyzing and fixturing individual assemblies, by using linear programs to solve simultaneous force or velocity equations. Then, by scattering fixture elements all over the assembly and eliminating those that turned out to be unnecessary, these authors were able to stabilize assemblies with only a small number of fixture elements. Atlas' fixturing algorithm was originally inspired by these researchers' efforts.

However, to the author's knowledge, no one has previously addressed the problem of performing both assembly sequencing and assembly fixturing *concurrently*. This co-design approach raises a host of new issues, many of which were discussed in an earlier paper [28]. Atlas, then, represents the first attempt to both generate an assembly sequence and fixture that sequence, under a "unified" approach which recognizes and utilizes the interrelationships between fixturing and sequencing.

3 Problem Statement

The task we are trying to accomplish is as follows:

> Given a geometric description of a product, generate (1) a sequence of part insertions which will assemble the product from its parts, and (2) a fixture to hold the product in place during assembly, such that each intermediate subassembly is stable under gravity and all insertion forces.

3.1 Insertion Forces

Insertion forces are part-against-part forces which may arise from part insertions. For example, in Fig. 1, either friction or a slight misalignment in P_1's position could cause a force to be imparted on P_2 to the right. To allow for this possibility, we would need one or more additional fixture elements (or *fixels*) to the right of P_2.

To be conservative, we assume that these insertion forces could occur at any contact point(s), and in any combination of magnitudes. They could potentially be quite large if the insertions were performed by, say, heavy robotic machinery; therefore, we check for insertion-force stability in the presumed absence of gravity (recognizing that the insertion forces might overwhelm any gravitational resistance).

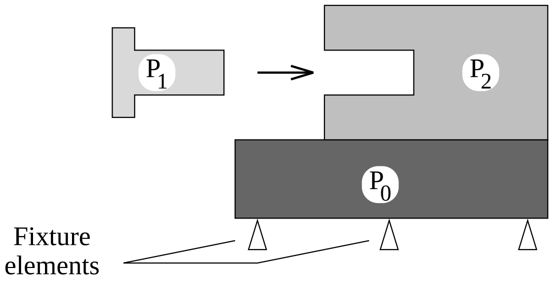


Fig. 1. An assembly operation which is unstable due to insertion forces.

3.2 Assumptions

We assume that the assembly is *polyhedral*, *rigid*, and defined by *nominal geometry* (*i.e.,* no tolerances).

We assume that the assembly sequences are *monotone* (parts inserted into a subassembly do not later move with respect to that subassembly) and composed of *single-step translations*. Extensions to rotational and/or multi-step part motions remain a direction for future research.

We assume that the fixture is *firmly secured* and *non-adjustable*. The advantage of a single, non-adjustable fixture is that it can be constructed once and then left alone throughout the assembly process; upon completion, it can then be recycled again and again. (The techniques presented here, however, extend easily to allow the dynamic addition of fixture elements during the assembly process.) Note that the single-fixture model does pose substantial restrictions on the range of possible sequence/fixture solutions. Most notably, it implies *linearity* in the assembly sequence (that is, each step involves the insertion of exactly one part). This is because the insertion of a multi-part subassembly would imply the previous use of another fixture to create that subassembly.

We assume the fixture elements are *frictionless* and can only apply *nonnegative force* (that is, they can push but cannot pull). We also treat them as having *negligible size*, so we describe them only at their points of contact, as in Fig. 1. To be sure, the actual shape of the fixture elements could pose a significant assembly-sequencing constraint; and if there were, say, a library of predefined fixturing elements available on a CAD system, these constraints could be carefully considered. However, this analysis is beyond the scope of the present work.

Finally, sometimes (but not always), we assume that parts are *glued together* (or similarly attached) upon insertion, so that the entire assembly is rigid and unified for stability-checking purposes. This assumption is based on observations from tours of actual automobile factories. When parts are described as not glued together, we shall neglect friction between them in our stability-checking (in order to be conservative).

4 Description of System

4.1 Overview

Atlas' basic approach to sequencing and fixturing is to enclose the final assembly in a completely immobilizing fixture (a process called *overfixturing*), and then to remove parts one at a time, reducing the fixture as necessary. The major steps are shown in Fig. 2 and described in greater detail below. During the disassembly process, the requirements imposed by the evolving assembly sequence constrain the evolving fixture, and vice-versa. In general, when we modify the proposed fixture, we are modifying the *one* fixture which will hold *all* the subassemblies; therefore, we must ensure the stability not only of the current assembly, but also of all previously-seen superassemblies.

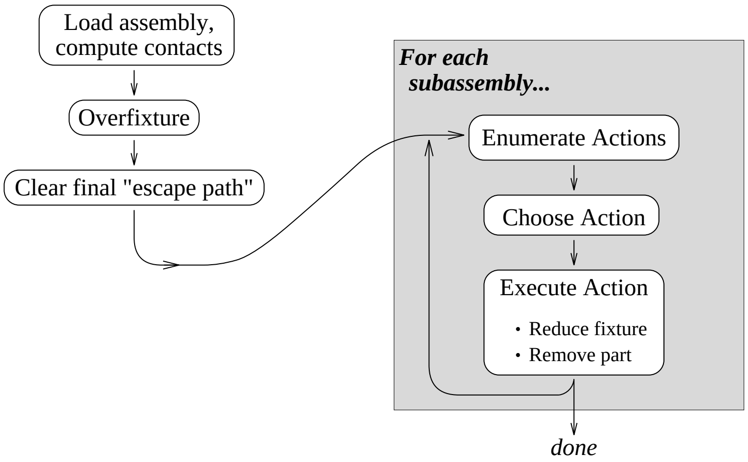


Fig. 2. Atlas' basic operating procedure.

Naturally, a fixture must not be so constraining as to prevent the product's removal after it has been assembled (usually in the upward direction). So we must perform one additional step after overfixturing, to eliminate all portions of the fixture which would block an upward motion of the entire assembly. (Some of the figures in this paper may, however, neglect the need for such an "escape path", for clarity in illustration.)

4.2 The Fixture Continuum

Atlas represents the candidate fixture as a *continuum* of possible fixel locations, rather than as a finite set of discrete fixels. (See Fig. 3.) Although a discrete representation, such as was used in [27, 28], is certainly valid, the continuous approach is a more complete representation of possible fixturing locations. This produces a potentially greater choice of stable disassembly steps. In essence, a continuous fixture can be thought of as an infinite set of differential fixture elements (the only caveat being that finite-sized forces can be applied at any point, unlike in the differential model).

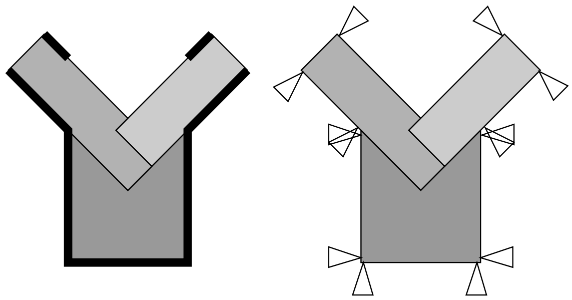


Fig. 3. A continuous fixture representation (left) describes possible fixturing locations more completely than a discrete fixture (right).

For purposes of analyzing stability, however, the fixture continuum is reduced to an equivalent, finite set of *virtual fixels* with which to balance the external forces. For planar assemblies, each "fixture segment" is reduced to two virtual fixels at the segment's endpoints, as shown in Fig. 4. Any net force and torque producible by the continuous segment can also be produced by these virtual fixels, and vice-versa; this makes virtual fixels useful for checking stability. In the three-dimensional case, the continuous fixture would be composed of two-dimensional "regions" rather than segments, and the virtual fixels would be at the corners of each region's convex hull.

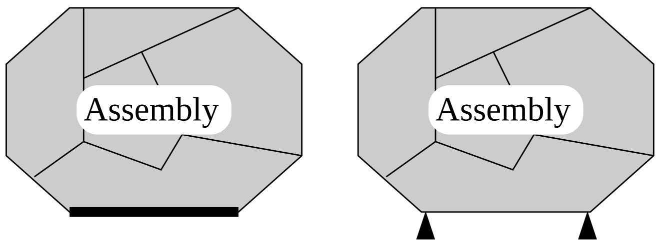


Fig. 4. To check the stability of a planar assembly, each continuous "fixture segment" (left) is reduced to two virtual fixels (right); together, these virtual fixels can produce any net force and torque that the original segment can produce.

This scheme resembles somewhat Wolter and Trinkle's use of two fixels at continuously-variable locations along each edge (in the planar case) [26]; however, the fixture continuum has the advantage of allowing segments to be broken up into arbitrarily many disconnected subsegments, as dictated by the evolving sequencing constraints.

4.3 The Disassembly Tree

In [29], Homem de Mello and Sanderson described a way to represent all possible disassembly sequences using an AND/OR graph. Each node in the graph represents a subassembly, and each AND'ed pair of children for that node represents two subassemblies into which the given subassembly can be decomposed. When there are multiple ways to break apart a subassembly, there are multiple pairs of children OR'd together.

Now, for linear disassembly sequences, such as those we are generating here, every assembly decomposition removes exactly one part. So one child node out of each AND'ed pair of children will represent an extracted part, and will therefore be a leaf node. For simplicity, we can suppress these nodes, reducing the AND/OR graph to a simple OR graph whose missing leaf nodes are implicit.

When fixture information is included in the nodes, however, the graph must become a tree in order to maintain consistency [28]. A partial example of such a tree is shown in Fig. 5. Note that the fixtures in most nodes represent partial solutions, admitting only the disassembly subsequence seen so far (at and above the given node). The final fixture solutions appear at the bottom, in the leaf nodes.

The arcs in the tree link a parent assembly and candidate fixture with each of its possible child subassemblies/fixtures. These arcs represent *actions*, of the form

Delete segments S from the fixture, and remove part P in direction D.

An action is said to be *feasible* if it involves no part collisions and the fixture modification violates no current or previous stability requirements.

Now, the continuous nature of the fixture implies that there may actually be an infinite number of distinct feasible children for a given node, each with at least a slightly different fixture. (See Fig. 6.) If we were able to somehow represent this continuum of children (and grandchildren, etc.) in our tree structure, and then exhaustively search the entire resulting tree, we would have a complete solution to the co-design problem; that is, given an assembly, we would be guaranteed to find a sequence and fixture for it when such a solution existed.

However, in practice, creating and managing such a tree would be difficult indeed. Equivalence classes of children are difficult to identify because the external forces to which they would respond equivalently are not all known until the sequence has been fully generated. Moreover, the tree would grow at least exponentially in the number of parts. To address the first problem, we shall discretize the continuum of children, by taking the continuum of possible actions and sampling from it a (hopefully) representative set of them. To address the exponential-growth problem, we shall explore only the most promising child which results; possible figures of merit for this are discussed at length below. To

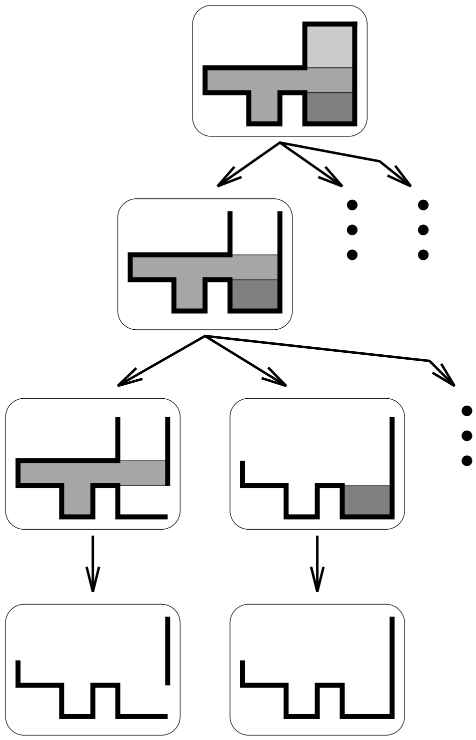


Fig. 5. A partial example of a disassembly tree.

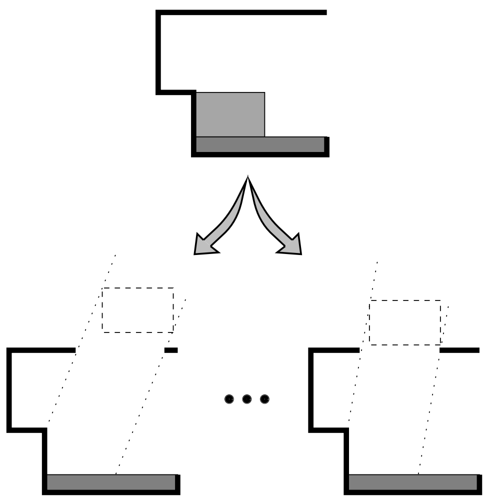


Fig. 6. Sometimes, there may be an infinite number of distinct feasible children, because the fixture is continuous.

be sure, completeness is sacrificed by these measures, but the resulting problem becomes manageable, and by choosing our child nodes very carefully, we can make failures few and far between.

4.4 Enumerating Possible Actions

A Complete Approach. In a complete algorithm, the set of all possible actions for an assembly can be found by considering each part in turn and generating its *extended translational freedom (ETF) cone*. An ETF cone is the subset of directions, from the entire sphere of directions, over which a given part can translate to infinity without colliding with another part. For example, in Fig. 7a, the possible motions of P_1 are described by the ETF cone of Fig. 7b. Note that in general, the ETF cone may not be cone-shaped at all; it may be non-convex, empty, degenerate, or composed of several disjoint regions.

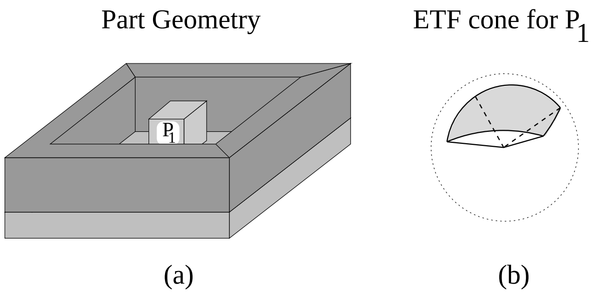


Fig. 7. An example of an extended translational freedom (ETF) cone.

Not all directions in the ETF cone produce actions which are feasible. For any given direction, a part motion in that direction may require the elimination of crucial portions of the fixture, in terms of the stability either of the current assembly or of some previously-seen superassembly. The set of directions corresponding to feasible actions is a subset of the ETF cone. Conceivably, we could identify this subset by projecting the assembly stability constraints onto the ETF cone. Doing so would yield the (continuous) set of all feasible actions, and thus ensure the completeness of the algorithm.

Atlas' Algorithm. Again, however, due to the difficulty of such an operation, Atlas uses a simplified approach, illustrated in Fig. 8. The cone of directions is first discretized, and feasibility is checked on each of the resulting discrete directions. Furthermore, instead of the extended translational freedom cone, Atlas' method is based on the simpler *local translational freedom (LTF) cone* [13]. An LTF cone is the subset of directions over which a given part can move *infinitesimally* without colliding.[1]

[1] For a more rigorous definition of LTF cones, the reader is referred to [30].

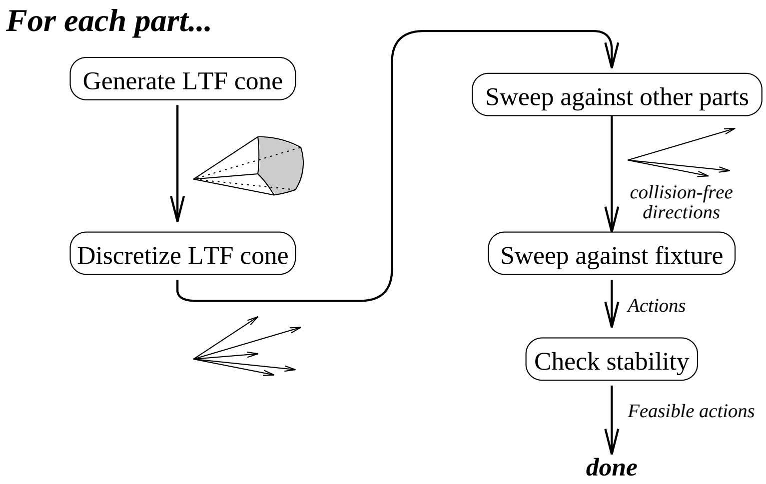


Fig. 8. Atlas' procedure for enumerating feasible actions.

For each part, Atlas generates its LTF cone with respect to all other parts, and from that cone selects a finite set of directions (such as the "corners" of the LTF cone). Atlas then "sweeps", or projects, the part out to infinity in each direction, to verify that it will not collide with more distant parts (see Fig. 9). For those directions that are indeed collision-free, a similar sweeping operation identifies the portions of the fixture which must be deleted. The result is a set of collision-free actions. Finally, for every such action, stability is checked for the current assembly and all previously-seen superassemblies, using linear programs. Each of these linear programs tries to find a combination of forces and torques from the virtual fixels and/or the interpart contacts to balance out the known external forces and torques. The formulas are presented briefly below, in section 5.1; for more information, the interested reader is referred to [26, 27].

One final point is worth mentioning about the enumeration of feasible actions. Sometimes it is possible to remove a part without reducing the fixture at all. In fact, such "fixture-preserving" actions may be preferred, since fixture reductions may reduce the ability of the fixture to stabilize future subassemblies. In order to try to find fixture-preserving actions when they exist, Atlas performs the preceding analysis a second time, this time with the fixture treated as unchangeable, like a big, immovable part. That is, in the second pass, the LTF cone includes constraints from the fixture as well as from other parts, and the sweeping module rejects any fixture-reducing actions entirely. Any feasible actions which emerge from this demanding process must be fixture-preserving, and are included in the list of feasible actions for consideration by the next phase of the co-design process.

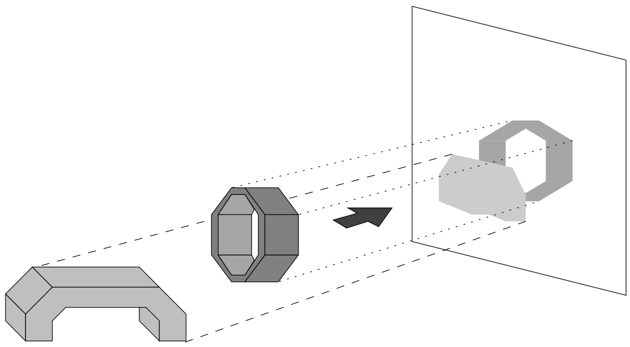


Fig. 9. By projecting, or "sweeping", parts in a proposed direction of motion, we can detect collisions between them. A similar operation detects part/fixture collisions.

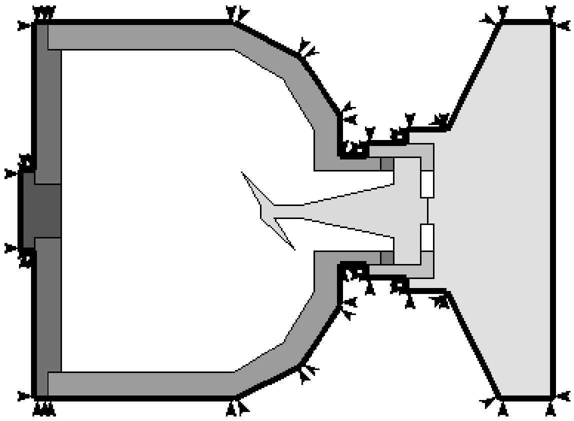
4.5 Choosing an Action

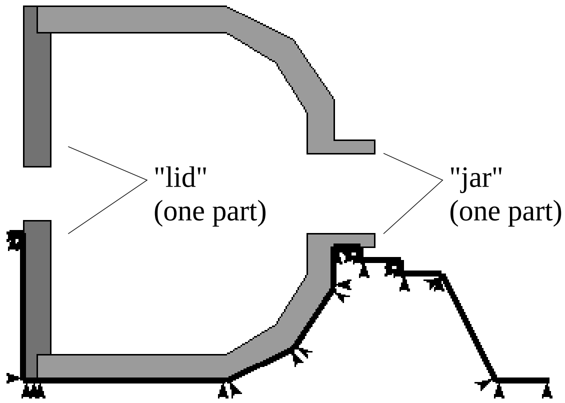
Having enumerated a set of feasible actions, Atlas' next step is choosing among them. This is a crucial decision, because exhaustively exploring all possible assembly sequences is not practical. In [1], which considered sequencing but not fixturing, the choice of action was guided by the optimization of an objective function, such as removing a key part as quickly as possible. However, in sequence/fixture co-design, we must first and foremost ensure that our fixture successfully stabilizes the entire assembly sequence. Since this sequence is not fully known until the end, deciding which action will best accomplish that goal is a difficult task indeed.

Nonetheless, several strategies have been found which work well in practice. For example, one approach, used in [28], is to minimize the number of deleted fixels (or deleted virtual fixels) in each step. In the planar case this is equivalent to minimizing the number of deleted fixture segments. The assumption here is that a fixture with more remaining fixels will be more likely to stabilize future, as yet undetermined, subassemblies.

While this method is generally a good one (as evidenced by the results reported in [28]), it can sometimes lead to dead ends in the disassembly process. For example, consider the "tilted blender" example of Fig. 10a. If we delete as few virtual fixels as possible in each step, we eventually arrive at the subassembly of Fig. 10b. No further actions are feasible, primarily because of the need to counteract insertion forces. If we attempt to remove the lid to the left, we must delete the leftmost part of the fixture, which would leave nothing to counterbalance the insertions of the rightmost parts. Conversely, if we attempt to remove the jar to the right, we will be unable to counterbalance the insertion of the small "lid cap" from the left.

We can gain additional insight into this problem, and infer a better heuristic for choosing actions, by considering the notion of *wrench space*.


(a)



(b)

Fig. 10. The "tilted blender" example: (a) After overfixturing, with virtual fixels shown explicitly. (b) The dead-end situation that results from always deleting as few virtual fixels as possible.

5 Wrench Space and Convex Hulls

5.1 Theory

A *wrench* is a force and a torque together:

$$\mathbf{w} = \begin{pmatrix} \mathbf{F} \\ \boldsymbol{\tau} \end{pmatrix}$$

Each fixel (or virtual fixel) imparts a wrench on the assembly, as do external forces such as gravity and insertion forces; the net sum of these wrenches must be zero to achieve stability. As noted earlier, none of the fixels can apply negative force. So if $\mathbf{w}_i$ is the unit-magnitude wrench imparted by fixel i, then there must be some $\alpha_i \in \Re$ such that [26, 27]:

$$\alpha_0 \mathbf{w}_0 + \alpha_1 \mathbf{w}_1 + \cdots + \alpha_{m-1} \mathbf{w}_{m-1} + \mathbf{W}_{ext} = \mathbf{0}, \qquad \alpha_i \geq 0$$

or,

$$\alpha_0 \mathbf{w}_0 + \alpha_1 \mathbf{w}_1 + \cdots + \alpha_{m-1} \mathbf{w}_{m-1} = -\mathbf{W}_{ext}, \qquad \alpha_i \geq 0 \tag{1}$$

For a planar assembly with glued parts, $\mathbf{w}_i \in \Re^3$. For a three-dimensional assembly, $\mathbf{w}_i \in \Re^6$. And if the n parts are not glued together, then $\mathbf{w}_i \in \Re^{6n}$, with six force/torque components acting on each part (most of which will be zero). In the unglued case, we must also include wrenches from the interpart contacts.

Atlas checks stability by seeking a solution $\{\alpha_i\}$ to eqn. (1) using a linear program. (For more information on these formulations, the reader is referred to [26, 27].)

Graphically, what eqn. (1) implies is illustrated in Fig. 11. For the system to be stable, each negated external wrench $-\mathbf{W}_{ext,j}$ must be expressible as a nonnegative linear combination of the fixel wrenches $\mathbf{w}_i$, which means that each $-\mathbf{W}_{ext,j}$ is on or inside the "cone" of $\mathbf{w}_i$'s. If we were to project the wrenches onto a unit sphere, as Brost and Mason did in [31], the projected $-\mathbf{W}_{ext,j}$'s would have to lie on or inside the projected convex hull of the $\mathbf{w}_i$'s. Alternatively, the *ordinary* convex hull of the $\mathbf{w}_i$'s and the origin must contain $-\epsilon \mathbf{W}_{ext}$ for some suitably small ϵ. Note that in practice, this convex hull may not be cone-shaped; instead, it may take a variety of convex forms, such as a flat disc, or a "polyhedral sphere" completely surrounding the origin.

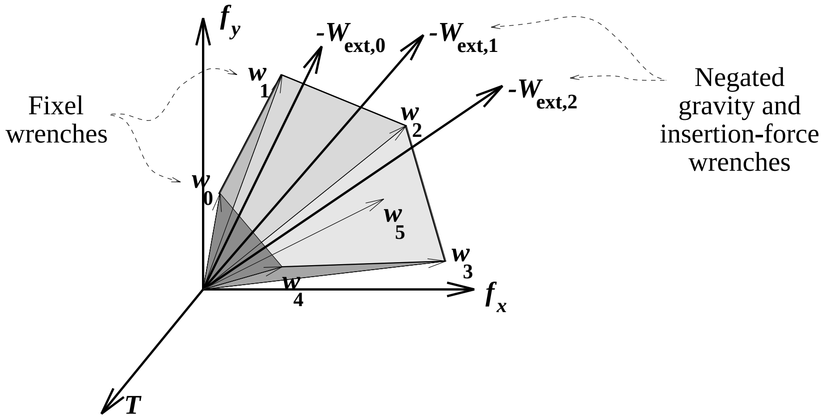


Fig. 11. The negated external wrenches $-\mathbf{W}_{ext,j}$ must be contained in the "cone" of fixel wrenches $\mathbf{w}_i$, in order to achieve stability.

Over the course of the sequence/fixture generation, we will be deleting fixels (and their wrenches) to make way for part removals, and we will be adding new external wrenches corresponding to the gravity and insertion forces on different subassemblies. Throughout this process, we must ensure that all the external wrenches are contained in the fixel-wrench convex hulls in this manner.

5.2 Application

An example of an actual wrench-space configuration is shown in Fig. 12. This situation corresponds to the "tilted blender" example of Fig. 10, under the glued-parts assumption, and in particular to the dead-end case of Fig. 10b. As can be seen, the disassembly subsequence leading up to this point has produced a number of gravity and insertion-force wrenches which must be contained.

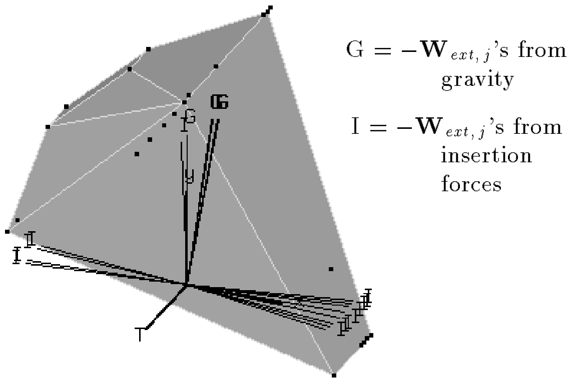


Fig. 12. The wrench-space representation of Fig. 10b.

And here we see the difficulty of the current predicament: the external wrenches are so spread apart that maintaining containment of them requires the use of most of the remaining fixels. As a result, Atlas does not have sufficient flexibility in deleting fixels. This observation suggests a disassembly strategy which keeps these wrenches more "clustered" together, or seeks redundancy in the spacing of the fixel wrenches, or both.

Clustering among the external wrenches can be measured in a variety of ways. If the parts are glued, the simplest might be to compute the average dot product among the external wrenches (after scaling the torques to bring their magnitudes in line with those of the forces). A more sophisticated approach might analyze the convex hull of the external wrenches, since containment of the wrenches on that convex hull implies containment of all externals.

For the case of unglued parts, the dot-product criterion would have to be modified, since any external wrenches ($\in \Re^{6n}$) affecting different parts would necessarily have a zero dot product. One modification would be to first project these wrenches onto $\Re^3$ or $\Re^6$ before comparing them; in essence this treats the assembly as a single unit for purposes of comparing external wrenches, despite the assembly's many internal degrees of freedom.

Redundancy in the fixel wrenches is much harder to measure, and identifying useful measures of fixture redundancy remains the subject of current research. One possible approach, however, might be to find the minimum number of fixels whose removal would destabilize the assembly in some way.

Atlas currently selects the action which maximizes clustering, as measured by the dot-product test (using an $\Re^3$ projection in the case of unglued parts). This simple strategy was found to work well on all the examples tested.

5.3 Convex Hull Complexity

In general, a convex hull of m points in d dimensions has complexity $O(m^{\lfloor d/2 \rfloor})$. Therefore, if we assume the parts are glued together and act as a rigid body (so d=constant), the complexity is polynomial in the number of fixels m (but constant in the number of parts). If the n parts are not glued, however, the wrench space has dimension $d = 3n$ or $6n$, and the complexity is exponential in the number of parts. Therefore, for larger assemblies, an explicit convex hull computation is only practical under the glued-parts assumption. This may constrain the development of more sophisticated criteria for selecting actions.

Furthermore, a number of rather subtle complications exist to the convex hull structure described thus far. We now turn our attention to some of them.

5.4 Convex Hull "Shells"

When parts are removed, the fixels which touched them become "inactive"; that is, they are no longer useful in stabilizing the remaining subassemblies, although they are still valid for stabilizing the earlier subassemblies. For example, in Fig. 10b, the fixels on the lower right are inactive. Note that this situation is different from that of *deleted* fixels, which cannot be considered at all.

In wrench-space terms, inactive fixels may be used to contain external wrenches which affect the earlier subassemblies, but not wrenches affecting the later ones. Therefore, different convex hulls are applicable at different stages of the disassembly process. Since these hulls get progressively smaller as the disassembly progresses, we have a series of concentric convex-hull "shells". And the external wrenches do not all correspond to the same shell; our requirement for containment of an external wrench by the convex hull actually means containment by the particular shell corresponding to the subassembly which that wrench affects. This observation further complicates the assessment of a fixture's future usefulness in stabilizing the assembly sequence.

5.5 Disconnected Assemblies

Another complication arises when a subassembly becomes disconnected; that is, when its parts can be split into multiple connected components which do not touch each other. In this case, our assumption that the parts are glued together upon contact no longer ensures that the assembly will behave as a single rigid body (for indeed, components which do not touch cannot be glued together).

In this situation, the effect of a fixel on an assembly is properly described by a wrench of dimension $3c$ (for a planar assembly) or $6c$ (for a three-dimensional assembly), where c is the number of connected components. In wrench space, this is similar to an assembly of c unglued parts, but with two key differences. First, the absence of contacts between the components decouples the stability check into c independent subproblems. Second, the previously-generated subassemblies are still described by simple 3- or 6-dimensional wrenches (if parts are glued), and later subassemblies may be, too. So the wrench space is of non-constant dimensionality.

Fortunately, the dot-product "clustering" test can still be made to work. The $\Re^3$ projections of the $\Re^{3c}$ wrenches correctly describe the effect of each wrench on a particular component. And if the pivot point for the torques remains the same, then these projections are entirely comparable to the $\Re^3$ wrenches from the earlier subassemblies. Therefore, maximizing the dot product between the earlier $\Re^3$ external wrenches and the projections of the new $\Re^{3c}$ external wrenches will realize our goal of containing these wrenches with a relatively small number of fixel wrenches.

6 Experimental Results

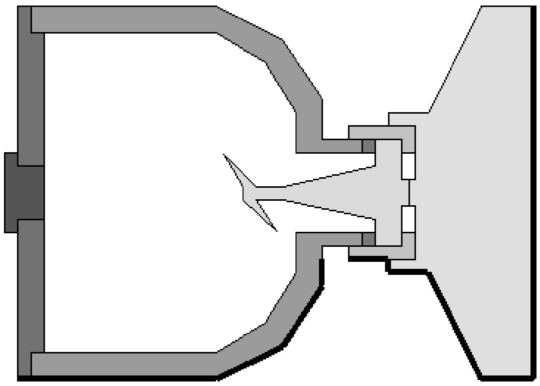


Fig. 13. The "tilted blender" assembly in its final (non-minimized) fixture, following a successful disassembly. The generated assembly sequence inserts the rightmost part downward, and then all other parts from left to right.

Using this dot-product "clustering" strategy, Atlas was able to successfully disassemble the blender example from Fig. 10. The final fixture is shown in Fig. 13. The corresponding assembly sequence inserts the rightmost part downward, and then all other parts from left to right. The final wrench-space configuration is shown in Fig. 14. For the case of glued parts, this solution was generated in 0.07 seconds of CPU time on a DEC alpha workstation. For the case of unglued parts, the same solution was generated in 0.84 CPU seconds. The higher time reflects the much greater complexity of the stability analysis in the unglued case.

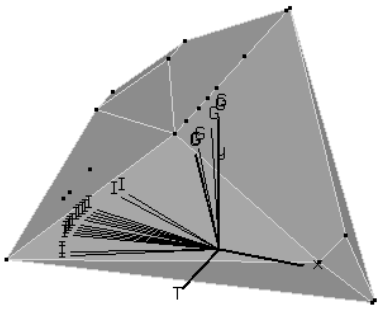


Fig. 14. The wrench-space representation of Fig. 13, showing all the external wrenches from the assembly process.

In another example, Atlas was run on an 11-part cross-section of an "A' Discriminator", a safety device being developed at Sandia National Laboratories. It generated the fixture of Fig. 15 in 0.32 seconds (glued parts) or 3.00 seconds (unglued parts), along with a sequence that inserts all parts straight down. Again, the same solution was found for both cases. Although this solution seems fairly straightforward, the example is interesting nonetheless because one subassembly is disconnected (consisting specifically of the two largest parts), and therefore the projection approach described in section 5.5 had to be applied.

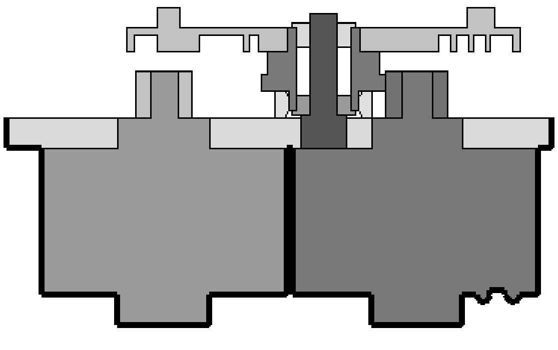


Fig. 15. The A' Discriminator example (courtesy Sandia National Laboratories), in its final fixture.

A third example is shown in Fig. 16. The corresponding sequence inserts the largest part straight down and then all other parts from left to right. This solution was generated in 0.02 CPU seconds (glued parts) or 0.14 CPU seconds (unglued parts).

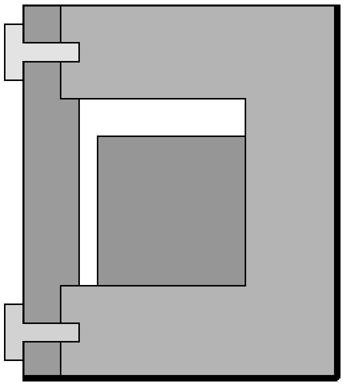


Fig. 16. The "box-lid-cargo" example, in its final fixture.

7 Conclusion

A software system has been presented which can quickly generate an assembly sequence and a fixture for a product, given only a geometric description of that product. Stability against both gravity and insertion forces is guaranteed. To the author's knowledge, this system, and its predecessor in [28], represent the first attempt to address the problems of assembly sequencing and assembly fixturing in a unified fashion. The system is viewed as an important first step toward the realization of a CAD tool which can provide immediate feedback to a team of product designers about a product's assemblability.

However, in order to achieve that goal, much work remains to be done. One important subgoal will be a system which can return not just any sequence/fixture solution, but the "best" one according to some criterion. To accomplish this, some means will have to be found to search sequence/fixture trees effectively but non-exhaustively.

Another important direction for future research will be to release many of the assumptions described earlier, such as the restriction to polyhedral assemblies, the use of only one fixture, the linearity of the assembly sequence, and the ideality of the fixels. But it should be noted that Atlas' present application only to planar assemblies is merely a matter of implementation; the underlying theoretical basis for all of Atlas' techniques extends equally well to three-dimensional assemblies.

Finally, experiments with more example assemblies would be useful, in order to verify and/or refine the algorithm presented above.

Acknowledgement

This research was supported by a grant from the Stanford Integrated Manufacturing Association (SIMA), and by NSF/ARPA grant IRI-9306544.

References

1. B. Romney, C. Godard, M. Goldwasser, and G. Ramkumar, "An efficient system for geometric assembly sequence generation and evaluation," in *Proc. ASME International Computers in Engineering Conference*, (Boston), pp. 699–712, 1995.
2. J. Wolter, "On the automatic generation of assembly plans," in *Proc. IEEE International Conference on Robotics and Automation*, vol. 1, (Scottsdale, AZ), pp. 62–68, 1989.
3. R. Hoffman, "Automated assembly planning for B-rep products," in *Proc. IEEE International Conference on Systems Engineering*, (Pittsburgh), pp. 391–394, 1990.
4. T. Woo and D. Dutta, "Automatic disassembly and total ordering in three dimensions," *Journal of Engineering for Industry*, vol. 113, no. 2, pp. 207–213, May 1991.
5. A. Lin and T.-C. Chang, "3D MAPS: Three-dimensional mechanical assembly planning system," *Journal of Manufacturing Systems*, vol. 12, no. 6, pp. 437–456, 1993.
6. B. K. Natarajan, "On planning assemblies," in *Proc. 4th ACM Symp. on Computational Geometry*, pp. 299–308, 1988.
7. J. D. Wolter, *On the Automatic Generation of Plans for Mechanical Assembly*. PhD thesis, Univ. of Michigan, 1988.
8. L. Kavraki, J. Latombe, and R. Wilson, "On the complexity of assembly partitioning," in *Information Processing Letters*, vol. 48, pp. 229–235, 1993.
9. J. Miller and R. Hoffman, "Automatic assembly planning with fasteners," in *Proc. IEEE International Conference on Robotics and Automation*, vol. 1, (Scottsdale, AZ), pp. 69–74, 1989.
10. S. Lee and Y. Shin, "Assembly planning based on geometric reasoning," *Computers and Graphics*, vol. 14, no. 2, pp. 237–250, 1990.
11. A. Subramani and P. Dewhurst, "Automatic generation of product disassembly sequences," *Manufacturing Technology CIRP Annals*, vol. 40, no. 1, pp. 115–118, 1991.
12. R. Wilson and J. Latombe, "Geometric reasoning about mechanical assembly," *Artificial Intelligence*, vol. 71, no. 2, pp. 371–396, 1994.
13. L. Homem de Mello, *Task Sequence Planning for Robotic Assembly*. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, 1989.
14. L. Homem de Mello and A. Sanderson, "A correct and complete algorithm for the generation of mechanical assembly sequences," in *Proc. IEEE International Conference on Robotics and Automation*, vol. 1, (Scottsdale, AZ), pp. 56–61, 1989.
15. A. Sanderson, "Automatic generation of mechanical assembly sequences," in *Geometric Modeling for Product Engineering*, (Amsterdam), pp. 461–482, North-Holland, 1990.
16. H. Asada and A. By, "Kinematic analysis of workpart fixturing for flexible assembly with automatically reconfigurable fixtures," *IEEE Journal of Robotics and Automation*, vol. RA-1, no. 2, pp. 86–94, 1985.

17. J. Bausch and K. Youcef-Toumi, "Kinematic methods for automated fixture reconfiguration planning," in *Proc. International Conference on Robotics and Automation*, vol. 2, (Cincinnati, OH), pp. 1396–1401, 1990.
18. R. Brost and K. Goldberg, "Complete algorithm for synthesizing modular fixtures for polygonal parts," in *Proc. IEEE International Conference on Robotics and Automation*, (San Diego), pp. 535–542, 1994.
19. X. Markenscoff and C. Papadimitriou, "Optimum grip of a polygon," *International Journal of Robotics Research*, vol. 8, no. 2, pp. 17–29, 1989.
20. B. Mishra, J. Schwartz, and M. Sharir, "On the existence and synthesis of multifinger positive grips," *Algorithmica*, vol. 2, no. 4, pp. 541–558, 1987.
21. Y. Xiong, D. Sanger, and D. Kerr, "Geometric modelling of boundless grasps," *Robotica*, vol. 11, pp. 19–26, 1993.
22. C. Ferrari and J. Canny, "Planning optimal grasps," in *Proc. International Conference on Robotics and Automation*, vol. 3, (Nice, France), pp. 2290–2295, 1992.
23. Z. Li and S. Sastry, "Task-oriented optimal grasping by multifingered robot hands," *IEEE Journal of Robotics and Automation*, vol. 4, no. 1, pp. 32–44, 1988.
24. V.-D. Nguyen, "Constructing force-closure grasps," *International Journal of Robotics Research*, vol. 7, no. 3, pp. 3–16, 1988.
25. S. Fahlman, "A planning system for robot construction tasks," *Artificial Intelligence*, vol. 5, pp. 1–49, 1974.
26. J. Wolter and J. Trinkle, "Automatic selection of fixture points for frictionless assemblies," in *Proc. IEEE International Conference on Robotics and Automation*, (San Diego), pp. 528–534, 1994.
27. R. Mattikalli, *Mechanics Based Assembly Planning.* Ph.D. thesis, Carnegie Mellon University, Pittsburgh, 1994.
28. B. Romney, "Issues in the co-design of assembly sequences and fixtures." Manuscript. Available at `http://robotics.stanford.edu/users/assembly`, 1995.
29. L. Homem de Mello and A. Sanderson, "AND/OR graph representation of assembly plans," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 2, pp. 188–199, 1986.
30. R. Wilson, *On Geometric Assembly Planning.* Ph.D. thesis, Stanford University, Stanford, CA, 1992.
31. R. Brost and M. Mason, "Graphical analysis of planar rigid-body dynamics with multiple frictional contacts," in *Robotics Research: The Fifth International Symposium* (H. Miura and S. Arimoto, eds.), pp. 293–300, MIT Press, 1990.