ON THREE DIMENSIONAL HEURISTIC PACKING FOR SOLID FREEFORM FABRICATION

Alejandro Beascoechea and Michael J. Wozny †

Center for Advanced Technology
Rensselaer Polytechnic Institute
Troy, NY 12180-3597

† Manufacturing Engineering Laboratory
National Institute of Standards & Technology
Gaithersburg, MD 20899-0001

Abstract

Optimal packing, in any dimension, is computationally intractable. Here we address the particular problem of automatically generating good layouts of arbitrary solid three-dimensional shape models into a container region that represents a given freeform fabrication manufacturing chamber. We put forth a multistage approach that combines modern heuristic optimization methods with computational geometry techniques. Our aim is to secure satisfactory and practical results for a problem for which reaching optimal packing configurations is computationally unattainable.

1. Introduction

The possibility of manufacturing several parts simultaneously can pay obvious dividends. The aggregated time to complete a given production plan can be substantially reduced, together with reductions in fabrication materials waste. At the same time, the throughput and utilization of the fabrication device can be significantly improved.

If reducing time/cost has certainly some appeal, there are occasions when packing can be the only alternative. For example, it is not uncommon that some parts to be molded result in a mold pattern which dimensions exceed those possible for any freeform fabrication device or simply those of the device available to us. If we "cut" the model of the pattern into subparts and pack them and build them together, we could still obtain a mold pattern after gluing with tighter tolerances than those possible using other forms of mold fabrication. Naturally, certain problems can require higher accuracy and better finish that what SFF technology can currently offer. But there is a clear tendency in the direction just described.

We can anticipate current and future need for computer software tools that help to cope with similar situations automatically. Some examples in this document will clearly suggest that manual operation, even for the simplest shapes, is out of the question for a non-trivial number of elements. Unfortunately from a computational point of view packing, as many other combinatorial optimization problems, is very easy to state but very difficult to solve. Given the NP-hard nature of the problem optimality , except for the most trivial cases, can hardly ever been achieved.

We can still strive for obtaining adequate part layouts, or at least better than a human operator can do manually, in "practical" interval of time. That is our final objective.

In order to meet our objectives and as opposed to research on 2D packing (mostly by the operations research community), we stress the geometric components. Based on sound computational geometry techniques, we define abstractions of part geometries to make the packing operation more efficient. In passing we combine the geometric elements with the use of modern heuristic tech-

niques and mix them novelly to create an automatic packing optimization method for solid three dimensional shapes into a container ready for fabrication.

Details about the techniques and their implementation, together with experimental results showing their adequacy to our problem are presented in the next dew sections.

2. The Problem

We consider the packing problem in an industrial solid freeform fabrication (SFF) context, where arbitrarily complex 3D parts can be built, and define the architecture and implementation of an practical environment to support this operation. In this case we can state the problem as follows:

Given a list $L_n = (S_1,...,S_n)$ *of* $n \geq 1$ optimally oriented geometric solids, pack as many of them as possible into a rectangular fabrication chamber U such that:

The bodies do not overlap one another or the boundaries of the container.
The total occupied volume density per layer is maximized, minimizing the total height.

As stated, the problem incorporates several application specific assumptions that differentiate it from other typical packing problems:

All the most popular fabrication processes require certain preferred orientations to achieve maximum effectiveness. Optimal part orientation can be determined by such factors as trapped volume, height, aliasing and number of slices required. Some processes also require to consider support structures to prevent overturns, support floating components and/or support overhanging material.

Because optimal part orientation is an active topic of research and is process dependent, we assume that the list of parts to be packed has previously been optimally oriented for whichever target layer manufacturing technology is applied.

Among the possible sequence permutations of the manufacturing plan that fit all their elements into the chamber, the ones that require less layers (assuming uniform thickness layers) will be the ones favored. If none of these exists, among the nests with discarded elements those of higher density are preferred.

Finally its necessary to note that the container is regular but the parts are irregular. This constraint is consistent with most of the cases considered in the packing literature and represents an adequate abstraction for most of the currently available SFF devices.

We shall use such specificities as constraints to convey additional problem-specific information that will help to make the solution searching process more focused.

3. Approach

Starting from a list of candidate exact computer models of the parts' geometries to be packed (here based on the availability of ACIS, a leading open geometric modeler C++ class library) the packing process has been divided in three major interdependent activities: part preprocessing, insertion scheduling and part placement, and layout optimization and compression. Next, each of these activities and the relevant terminology is introduced, followed by more detailed explanatory description:

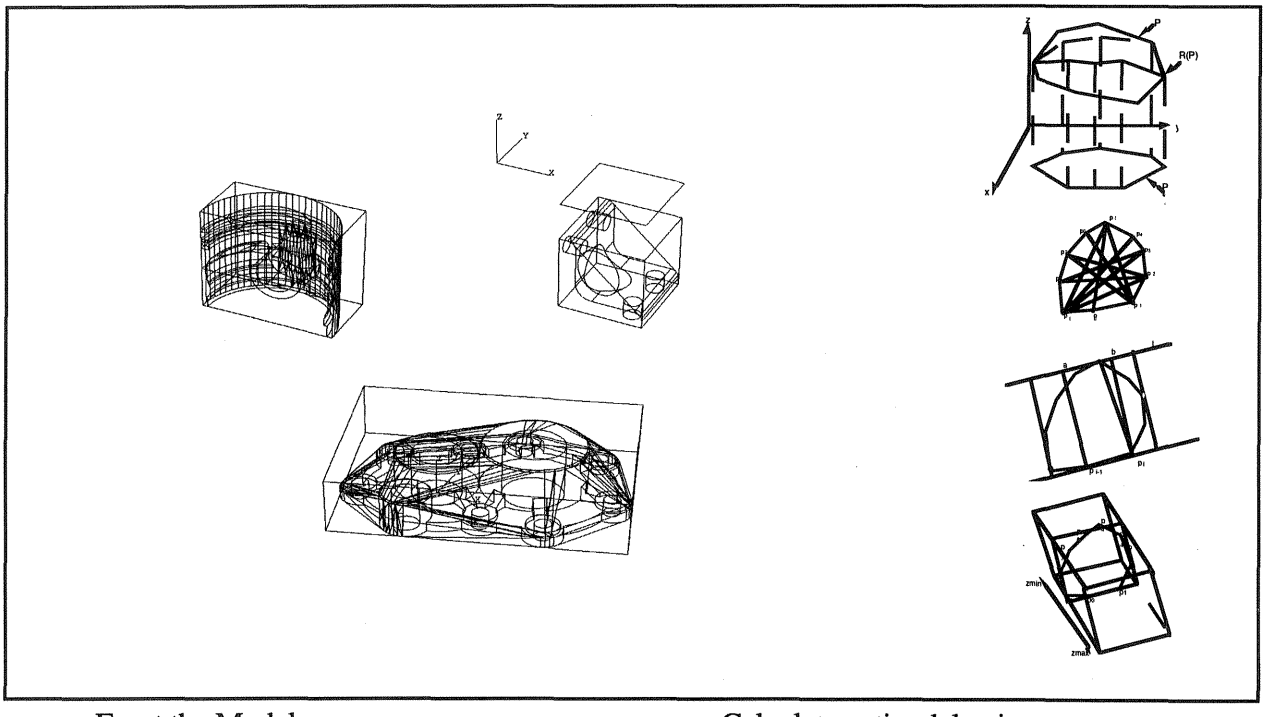
3.1. In order to facilitate efficient packing with minimum computational we obtain from every optimally oriented computer part model the following two part abstractions:

The minimal bounding box of the body in the direction of fabrication.
The orthogonal profiles of the part.

It is common that bounding regions are used to approximate more complex shapes. In a solid freeform fabrication environment the set of object shapes to deal with can be extremely heterogeneous. In practice is very frequent that parts are thin-walled components.

Our experiments suggest that bounding boxes usually represent the best alternative among the different possible elementary bounding regions as it is frequent that the most relevant features tend to align with the faces of the bounding box (Table 1).

Obtaining a tight bounding box for a complex 3D shape is not trivial. We have developed a fast algorithm to compute the optimal bounding box in the direction fabrication of the inscribed part. The algorithm to calculate the minimum bounding box follows the steps illustrated in the table below:



- Facet the Model
- Obtain the contour w.r.t. the direction of fabrication x3
- Project on x1-x2
- Calculate antipodal pairs
- Calculate the minimum bounding rectangle
- Extrude to obtain the minimum bounding box

Table 1 - Minimum Bounding Box of some preoriented ACIS parts

3.2. From the set of parts chosen to be packed, evolve schedules of insertion into the container. Additionally, as parts in a given schedule accede the container, determine an initial base location in

the manufacturing chamber (if available) for their minimum bounding boxes using a part placement scheme.

For every sequence packing procedure, we can identify two components that are the focus of heuristic evaluation.

- The sequence insertion schedule, i.e. the order in which a sequence of parts to be packed is going to be considered for its positioning within the container.

- The geometric placement of each individual part into the container. This is, its position within the container with respect to the parts already placed.

Borrowing from the 2D packing literature the part sequencing problem could be addressed simply by classifying such sequences by some form of total ordering relation function of the dimensions of the boxes (e.g. decreasing height, volume, etc.). For the placement of the parts component, a possible heuristic could be one of those known as level heuristics. A part can fit into a level if its height is less than that of the initial part arranged for a level (layer).

In the table 2 below we show some results of the application of one pass sequencing heuristics using two possible block positioning schemes we have devised (See section "Flexible part Placement"). Elements have been sorted in ascending/descending order by area or volume, and compared with random sequences. These simple heuristics produce feasible results very rapidly, however their lack of specificity with respect to the set of parts limits the potential packing efficiency.

BLOCK PACKING DENSITY		DESCENDING		ASCENDING	
Positioning Scheme	Num. Parts	25 BLOCKS	50 BLOCKS	25 BLOCKS	50 BLOCKS
FIRST FIT	Random	37	43	37	43
	Area	52	64	36	37
	Volume	58	73	29	32
LAYER BY LAYER	Random	37	45	37	45
	Area	44	51	19	17
	Volume	41	53	19	16

Table 2: Results of Rectangular Block Packing Using Various Heuristics

We propose the introduction of two additional elements to achieve greater heuristic performance in both fronts (sequencing and part placement):

3.2.1. Evolving part schedules

We use order-based genetic search to determine those sequences that produce superior results starting from a seeded population of sequences obtained using single-pass heuristic as the ones reported above. Members with higher than average performance will tend to be selected more frequently for crossover operations, which purpose is to preserve and agglutinate those subsequences of chosen individuals that most contribute to their higher performance through the creation of new

individuals that inherit their favorable features and the elimination of those that do not contribute to improvement.

Appropriate crossover operators play a fundamental role in this process. Based on order operators presented in the GA literature that have shown good performance for other sequencing problems such as the TSP and scheduling, we can generate our own problem specific layout operators that use as their base encoding the graphs obtained as the result of the part insertion and matching process. As in the edge-recombination operator the objective is to preserve favorable common edge structures of the packing graph.

3.2.2. Flexible part placement.

Placing rules used for rectangular packing have been usually limited to very simple layer/shelve/level schemes. When rectangular elements are inserted into a container layer-by-layer the non-overlapping constraint among parts is guaranteed, and only intersection with the container has to be checked. Other placement schemes are less frequent because of the additional computational resources involved when potential overlapping of boxes has to be considered. However, these more elaborate placement policies, such as first fit, can, in many cases, lead towards improved utilization (see table above).

The purpose of seeding the a population using well known heuristics is to help the search mechanism to start from a favorable initial states that guarantee a solution quality lower bound. If we initialize a population with poor initial guesses, and the insertion mechanism that generates individual is too simplistic the improvement we can get using genetic algorithms can be inferior than that than can be achieve using by using sophisticated placement scheme.

In our approach, each element inserted into the fabrication chamber is connected with those that have preceded forming a tree of available candidate available positions that can be efficiently traversed. With adequate free space mapping its then very simple to support several positioning schemes (including a sophisticated form of tight layer-by-layer, and first fit). Our starting search states are comparable or superior to those reported for related applications (some of them after additional combinatorial optimization). The results in Table 2 clearly suggest that even then, first fit tends to result in better packing efficiency.

3.3. *For each element in the scheduled sequence accepted for insertion, determine its* ***local context*** *(the set of neighboring parts the new part can interact with during the coupling process) and pack the part as tightly as possible with respect to the parts already in the container using a* ***matching function****. If a particular* ***coupling*** *is outstanding, consolidate the corresponding part blocks into a unique cluster and propagate to schedule creating process.*

Elements scheduled for insertion have to select, among the candidate placement positions, one that guarantees a non overlapping allocation. Additionally, once that an element is guaranteed a position into the container, the manipulation of a more detailed representation of its geometry incarnated by the part orthogonal profiles (see next section "Part Coupling"), and the possible interaction with is neighboring elements takes first place.

It is evident that given the nature of the non-overlap constraints as expressed in the statement of the problem (See section 2), testing for intersection between elements (including the container) or (their idealizations) is a fundamental operation in any packing scenario. On its efficiency, as well as on our ability to minimize the number of times it has to be performed depends, to a great extent, the possible practicality of the system.

To this end we act in two fronts. First configure the container region using a 3D uniform grid data structure that supports boundary box tests for intersections very efficiently and at the same time serves to localize the operation context during the subsequent part fine positioning. Second, we define a simplified representation of the body surface and match parts so described exclusively in the direction of the axes.

3.3.1. Orthogonal Array Surface Representation and Pair Coupling

Given an arbitrary three-dimensional solid body whose minimum inscribing bounding box has been obtained by the method described in section 3.1. Its set of orthogonal profiles is formed by six arrays of sampled distances from a rectangular grid of points on the bounding box to the corresponding intersections with surface of the part. The rays are shot towards the interior and normal to the surfaces of the encasing bounding box. In practice, a rectangular array is a snapshot image of the surface of the part, with the distances representing the gray level values.

Continuing with the analogy, matching two images for maximum overlap without explicitly checking for part boundaries intersection can be done using the following convolution/correlation type matching function (See Figure 1):

$$match(n_1,n_2) = \max(\prod_{k1=0}^{n1}\prod_{k2=0}^{n2} u(k1,k2) + w(k1+n1,k2+n2)),$$

where $n1,n2 \in \{-N+1,-N+2,...,-1,0,1,...,-N-2,-N-1\}$

and $\prod_{k\in K} a_k$ denote the minimum of the numbers a_k

Of the two profile images involved in a matching operation one, the source, corresponds to the part to be inserted into the container, and the second, the destination, is the image obtained from the aggregate union of the images of the parts already in the container that form the local context and in which the new part will integrate.

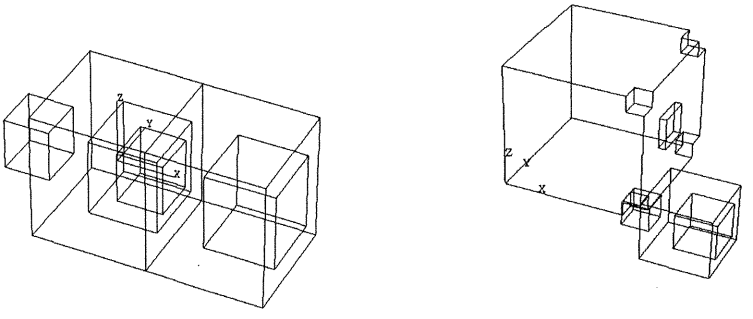


Figure 1. Pair wise coupling of simple ACIS solids.

Not all the positions for every pair of profile matrices have to be checked, though. It is possible to design favorable matrix addition sequences. For example, we could use a spiral sequence .

In the their most simple form, the images can be represented as array objects and the sampling can be uniform and defined as a multiple of the layer thickness. This strategy leads to a very straightforward computer implementation using very simple operations. However, other traditional image representation schemes can be used and, for very high resolution samples, specialized image processing hardware could be applied. In any case, the computation time is a function of the sam-

pling rate and (in the uniform sampling mode) is independent of the complexity of the surfaces involved.

3.3.2. Insertion in the Container and Part Interaction During Coupling

There are two moments during the packing process as described so far that require boundary box testing for overlaps : Insertion of a box into a container and detection of overlapping of a part with the parts in its local context during coupling.

In its most rudimentary form, testing for intersection during part insertion requires $O(n^2)$ test for intersection operations, testing each candidate against the list of parts already packed. This is not specially grave because typical SFF deals with a relatively small number of parts for a given packing operation. However, the part coupling process requires one test for overlap in a location for each pair wise combination of the facing images. The brute force approach is then unacceptable. To cope with this apparent bottleneck we generalize the use of the uniform grid data structure to three dimensions.

The uniform grid is a paradigmatic technique first proposed by Franklin in 1978 shown to be useful to efficiently solve a variety of important geometric problems. Very succinctly, for the three dimensional case, the space in which objects can exist is partitioned into equal sized axis-aligned rectangular buckets or cells. The dimensions of the rectangular "cells" are governed by the input data defining the objects occupying sections of the universe volume so defined. For each element to insert, the algorithm determines the range of cells of the grid it intersects and then it tests whether the candidate object intersects or not with any of the ones that have been inserted into the grid structure in proceeding steps of the iteration. If the candidate element does not intersect with any other on any of the cells tested, there is no intersection, and it can be safely inserted into the grid, adding a new element to the collection of laid out objects.

The uniform grid has been usually only associated to 2D polygon applications because its generalization to 3D has the potential drawback of requiring large amounts of memory. In a context like SFF, the number of parts that can intersect is relatively limited, their dimensions are in the same order of magnitude, and the distribution over the space practically uniform. For this reason the uniform grid represents an almost ideal construct for this application.

Additionally, the structure the grid imposes on the packing region is fundamental to define local contexts, i.e. regions of the space surrounding an insertion position. Once the local context and its component parts have been determined, checking for overlaps during part coupling is greatly simplified. Because there is coherence of parts in the local context of the new insertion , only those in the local context have to be tested for overlap .

3.3.3. Consolidation of Parts

We call consolidation to the operation by which two shape inscribing bounding boxes will be transformed into a single one of lesser volume grouping selected pairs of parts for which we are achieving above average couplings using the method previously described. Very succinctly, the procedure follows the following steps:

1)Determine the union of the projections. The projection of the convex hull of a 3D body has been obtained as a by-product of the computation of its minimum bounding box. The union of two convex polygons is the convex hull of the union of their polygon point sets.

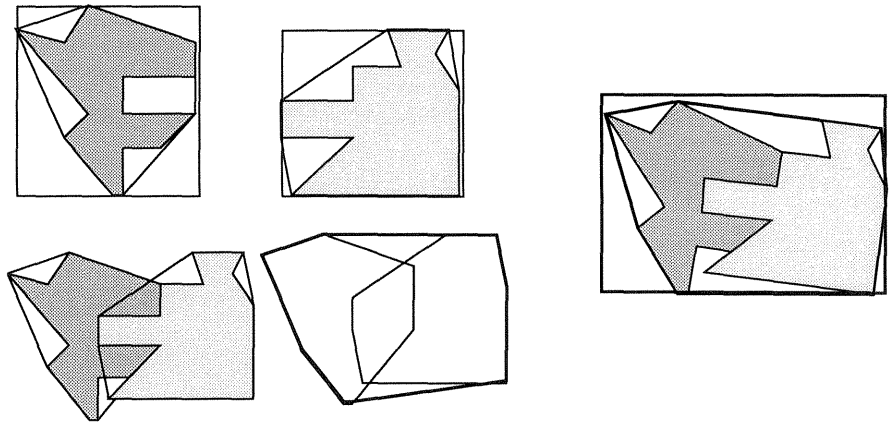


Figure 2: Schematic consolidation operation.

2) Determine the minimum inscribing rectangle of the union. We can use for this purpose the same procedure used during the computation of the minimum bounding box because the union of two convex sets is also convex.

3) Compute the minimum bounding box of the union.

4) Calculate the volume of the new box and compare with others.

3.4. Finally, ***select the best schedule*** *among the fittest individual of the population.*

4. Conclusion

We have discussed the main components of an approach to practically cope with the difficult problem of packing for SFF. To our knowledge it represents one of the first efforts to deal with irregular parts, using for this end a variety of techniques from different disciplines. The combined effect of the ideas render this problem tractable and their computer implementation can represents a valuable tool in the arsenal of the SFF practitioner.

5. References

Akman, V. Franklin, Wm. R., Kankanhalli, M., Narayanaswami, C. "Geometric Computing and Uniform Grid Technique." Computer-Aided Design (September 1989).

Allen, S., & Dutta, D. (1994). On the Computation of Part Orientation Using Support Structures in Layered Manufacturing (Technical Report No. UM-MEAM-94-15). University of Michigan.

Burns, M. (1993). Automated Fabrication: Improving Productivity in Manufacturing. PTR Prentice-Hall.

Davies, L. (1991). Handbook of Genetic Algorithms (1st ed.). Van Nostrand Reinhold.

Dowsland, K. A., & Dowsland, W. B. (1992). Packing Problems. European Journal of Operations Research, 56, 2-14.

Jacobs, P. F. (1992). Rapid Prototyping & Manufacturing - Fundamentals of StereoLithography. Society of Manufacturing Engineers.

Integration of a Solid Freeform Fabrication Process into a Feature-Based CAD System Environment

Yong S. Suh† and Michael J. Wozny‡

†Center for Advanced Technology in Robotics, Automation and Manufacturing
Rensselaer Polytechnic Institute
Troy, NY 12180

‡Manufacturing Engineering Laboratory
National Institute of Standards and Technology
Gaithersburg, MD, 20899

ABSTRACT

The design and implementation of a feature-based software system for the Solid Freeform Fabrication (SFF) process is discussed. For the SFF process to be an effective design tool in a manufacturing system, the tight integration of the process into a feature-based CAD system is desired. A feature set for determining the optimal part orientation is identified, and the procedure is described.

1 INTRODUCTION

1.1 SFF software: feature vs. geometric reasoning

To cope with the increasing market competition, the *concurrent engineering* (CE) concept is being adopted by many companies to reduce the cost and the cycle time for manufacturing quality parts. To build a successful CE system where designers and manufacturing experts work simultaneously, the appropriate management of the product information flow among the users is essential. The *product information* include high-level data such as design intent, part functionality and manufacturing processes, which traditional CAD systems cannot support. To support such high-level information beyond geometric data in the CE system, feature-based CAD systems have been introduced to associate engineering meaning to the shapes of the CAD model components. In these systems, users can manipulate the CAD models in terms of *features*, and software algorithms can simulate the human behavior by manipulating the high-level feature entities, as oppose to the low-level geometric reasoning processes with blind searching algorithms.

One of the primary application of the current SFF processes is to fabricate design prototypes for fast design verification. The process is identified to be a valuable tool in the CE environment because it can reduce the significant amount of design cycle time. Therefore, it is desirable that the SFF process software is fully integrated into the environment by taking a feature-based approach. As the process requires extensive geometric reasoning procedures that are time consuming and require complex algorithms, the feature-based approach is appropriate, and more intelligent processing is possible. Also, an algorithm can be easily customized in the feature-based approach by simply adding new features or modifying the existing features.

1.2 Design of a feature-based approach for the SFF

The feature-based systems have been classified into two big categories; feature extraction systems and design-with application feature systems.[hen94, hou93, kyp80, lu86]

In the feature extraction systems, features are extracted from pure CAD model data based on the predefined feature descriptions. However, it is very difficult, if not impossible, to implement a universal algorithm that extracts all types of features that are required by various applications. Another disadvantage is that most of the feature extraction algorithms can only find the features that exactly match the feature descriptions topologically and geometrically. The algorithms are not generally extendible to find general-shape features such as thin-walls, slender features and lateral protrusions. Most of the features that are required by the SFF processes are such general features in contrast to the machining features that have relatively concrete forms.

In the design-with SFF feature systems, a designer builds models using the features specific to the SFF process so that the features required by SFF software are embedded in the CAD model. As the primary usage of the SFF systems is for fabricating prototypes, however, this method is not practical because designers will focus on the final manufacturing processes such as injection molding and machining. This approach is not adequate for the CE system where multiple manufacturing systems should be supported.

As both traditional approaches are not suitable to the SFF process, a hybrid system design is desired to integrate the process into a feature-based system. The new system must support multiple feature representations, because the feature representation for the SFF processing will be different from the designer's feature representation. Therefore, the system software must provide a mechanism to extract customized features (SFF features) from the part model described by design features. Even though each user works on different views (different feature representations), any information provided by one of the users (such as dimensional constraints specified by a designer) must be shared among the users through features.[bro93]

1.3 Requirements for the feature-based system for the SFF process

The new system must satisfy the following functional requirements to support the SFF process in the CE environment.

- New features must be easily defined and extracted without system modification.
- General-shaped features must be defined and extracted.
- Design information such as dimensional constraints and surface finish must be supported.

A flexible feature-based system that supports the above requirements is designed and implemented. In section 2, this system architecture will be briefly discussed. In section 3, we will discuss a feature-based approach to the selection of the optimal part orientation.

2 SYSTEM ARCHITECTURE

The detail description of the system design will not be discussed in this paper. Instead, the basic idea of the system design and its features will be described very briefly.

2.1 Overview of the system

A part described by design features are converted to an internal feature representation in the system. A finite number of *fundamental features* (FF) and *fundamental spatial relations* (FSR) are defined, and the internal feature structure is formed by using them as building blocks. This is based on the assumption that any form feature can be decomposed to the FF's, and again, any form features can be represented by rearranging those FF's. Therefore, the application form features are extracted from the intermediate feature description converted from the design features using the FF's. The FSR's specify the spatial relationship between the FF's. Therefore, any design constraints specified by the designer can be applied to the application feature model. Using the FF's and FSR's, algorithms can be standardized without regard to the details of user-customized features.

2.2 Advantages over traditional feature extraction systems

A part with one feature representation can be transformed (extracted) to another through the two-step algorithm -- *Feature Shape Code Matching* step and *Feature extraction* step. Through these steps, designer's feature information is used when it's possible. The term, *"feature extraction"* will be used in this context in this paper. As the detail of the method is beyond the scope of this paper, only the advantages of this approach over traditional feature extraction systems are highlighted below.

- Design feature information can be used from other feature views.
- Features can be dynamically defined.
- A newly defined feature can be immediately extracted from the model without implementing the extraction algorithm specific to the feature.
- General-shape feature can be defined and extracted.

2.3 An example of feature extraction

In this system, features are defined in a declarative way as shown in Figure 2.1 in which a *through-slot* machining feature is defined. Writing the feature description in a text file using the predefined system keywords, the feature can be immediately extracted without coding additional program. Figure 2.2 shows a part described by several design features such as ribs, protrusions and depressions, and Figure 2.3 shows one of the three through-slot features extracted from the part as described in Figure 2.1.

```
face id 0 name wall-1 type HB-planar const { ^3 ^4 };
face id 1 name wall-2 type HB-planar const { ^3 ^5 };
face id 2 name bottom type HB-planar const { ^4 ^5 } open;
confront-paral id 3 entity ^0 ^1 hint value 15 +0.01 -0.01;
confront-angle id 4 entity ^0 ^2 value 90;
confront-angle id 5 entity ^1 ^2 value 90;
profile-open ^0 ^2 ^1 both-sides;
```

Figure 2.1 Declarative through-slot feature description

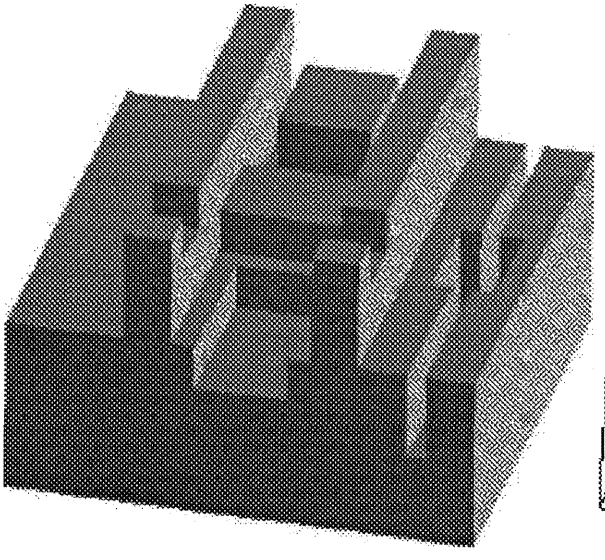


Figure 2.2 Example part

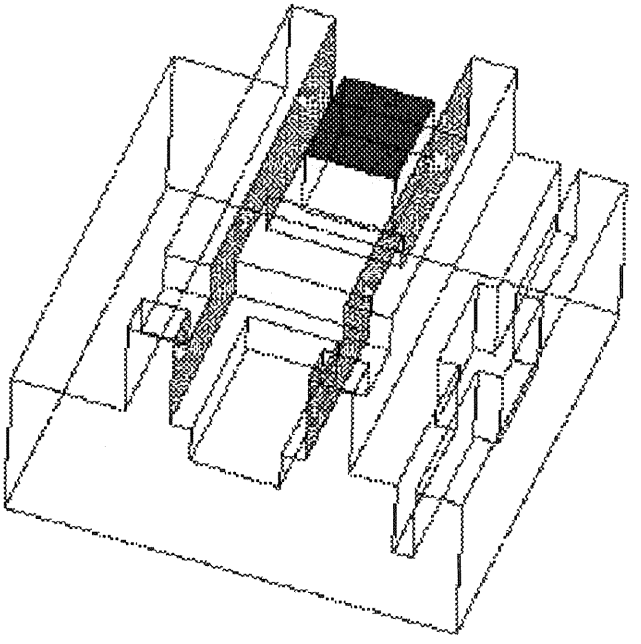


Figure 2.3 Extraction of a thru-slot feature

3 FEATURES FOR SOLID FREEFORM FABRICATION PROCESSES

The proposed system enables the SFF process to exploit the advantages of the feature-based CAD system in the concurrent engineering environment. An SFF process operator can define any features of their interests based on the characteristics of his/her SFF machine. The defined features are extracted from the designer's CAD model. In this section, we will discuss a feature-based approach to evaluation the optimal part building orientation as an example.

3.1 Features for determining the optimal part orientation

The part orientation within the building chamber of an SFF machine has a primary impact on the building time, part resolution and surface finish[jac92]. Comparing with the low-level geometric reasoning processes for evaluating the part orientation taken by several researchers[set94, kim94, sre94], the feature-based approach has the following advantages.

- **Faster processing**: As the features are directly extracted from the design features, many of the time-consuming geometric reasoning steps are reduced.
- **Intelligent processing**: The information supplied by designers can be used.
- **Flexibility**: The features can be easily added and removed for better results, customizing to the needs of a company.

In the following subsections, we will discuss the criteria for determining the optimal part orientation with the brief descriptions of relevant features. For the brevity of this paper, we will not discuss the detail of the algorithm for processing each type of features.

3.1.1 Trapped liquid

In the liquid-based SFF processes such as the SLA, trapped liquid is one of the major concerns in determining the part orientation. The trapped volume is the space (formed during the building process) within a part that hold the liquid isolated from the liquid in the vat.[1]

In this section, the features for the trapped volume are described borrowing the terminology of the machining features to represent the shapes of the features, as follows.

- Pocket
- Blind step
- Blind slot
- Hole
- General depressions

Once these trapped liquid features are extracted, the vertical height of the trapped liquid can be evaluated easily without extensive geometric reasoning. For each feature type, an algorithm for calculating the liquid height, $\mathcal{H}_i(\Theta)$, at an arbitrary orientation, $\Theta = \theta(\alpha,\beta,\gamma)$[2], can be defined easily. An example is shown in Figure 3.1 that shows a pocket feature at an arbitrary orientation. A local coordinate frame is attached to each feature, and its principal axes are represented as X',Y' and Z', and $\hat{X}', \hat{Y}', \hat{Z}'$ denote the unit vectors of the local coordinate system with respect to the global coordinate system.

The liquid height, $\mathcal{H}_{pocket}$, of the pocket is calculated with a simple algorithm as follows.

$$\text{if } \hat{Z}' \bullet \hat{Z} \geq 0.0 \text{ then } \mathcal{H}_{pocket} = h_{\min}(\{\mathcal{V}_i^{top}\}) - h_{\min}(\{\mathcal{V}_j^{bottom}\}) \qquad (3.1)$$

where

$h_{\min}(\{\mathcal{V}\})$: the minimum Z component value of the set of vertices, $\mathcal{V}$.

---

[1] In the powder-based processes such as SLS, the effect of the trapped volume is on the warpage of the part. If the open end of the trapped volume is downward, the heat is built up inside the trapped volume, which causes uneven temperature distribution.

[2] To specify an orientation of a solid in 3D space, three independent variables are required. The roll,pitch and yaw angles are used in this implementation.

V_i^{top}, V_j^{bottom} : the top and bottom vertices of the pocket respectively (i,j=1...4)

Similar algorithms can be defined for each feature type. A general algorithm has also been developed for evaluating the trapped liquid height of a general arbitrary depression feature. After evaluating each feature, a cost function, $\mathcal{F}(\Theta)$, can be calculated at a certain part orientation, Θ, as in Eq.(3.2).

$$\mathcal{F}(\Theta) = \sum_{i=1}^{n} \mathcal{H}_i(\Theta) / \mathcal{D}_{\max} \tag{3.2}$$

where

n : number of the relevant features in the part

$\mathcal{H}_i(\Theta)$: sum of the trapped liquid height of every feature at Θ

$\mathcal{D}_{\max}$: maximum diagonal size of the bounding box of the part at initial orientation

The cost function will be used in the nonlinear optimization procedure described in section 3.2.

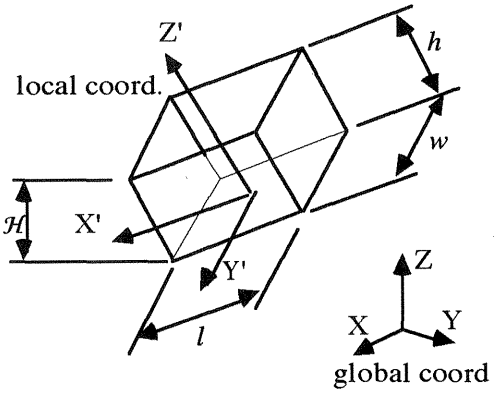


Figure 3.1 Evaluating trapped liquid height

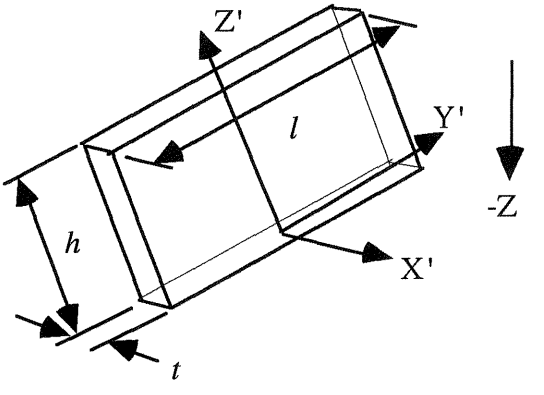


Figure 3.2 Evaluating support structure

3.1.2 Support structure

Support structures must be generated to hold the part during the fabrication in several processes. The complexity of the support structure of a part depends on the part orientation, and it must be as simple as possible so that they can be removed easily at the post processing step. The following features are extracted from the part to evaluate the support structure complexity.[3]

- Rib
- Boss
- Land
- Hole
- Pocket
- Slot
- Step
- General protrusion
- General depression

Given a part orientation at Θ, the area of the surfaces that require the support structure can be calculated according to the feature types. For example, as shown in Figure 3.2, the area, $\mathcal{A}_{rib}$, of the surfaces that need support structure can be calculated as follows for a rib feature.

$$\text{if } \hat{X}' \bullet \hat{Z} \neq 0.0 \text{ then } \mathcal{A}_x = l * h \tag{3.3}$$

$$\text{if } \hat{Y}' \bullet \hat{Z} \neq 0.0 \text{ then } \mathcal{A}_y = t * h \tag{3.4}$$

[3]The features can also be used in the automatic generation of the support structures. For example, a gusset-type support structure can be generated for a rib feature, while a honeycomb-shaped support structure is appropriate for a flat protrusion feature.

$$\text{if } \hat{Z}' \bullet \hat{Z} < 0.0 \text{ then } \mathcal{A}_z = t * l \tag{3.5}$$

$$\mathcal{A}_{nb} = \mathcal{A}_x + \mathcal{A}_y + \mathcal{A}_z \tag{3.6}$$

A general algorithm for an arbitrary feature has also been developed.

The cost function, $\mathcal{G}(\Theta)$, is calculated as follows.

$$\mathcal{G}(\Theta) = \sum_{i=1}^{n} \mathcal{A}_i(\Theta) \Big/ \mathcal{A}_{part} \tag{3.7}$$

where

n : the number of the relevant features in the part

$\mathcal{A}_i(\Theta)$: the area of the surfaces that requires surface finish in the i-th feature

$\mathcal{A}_{part}$: the overall area of the surfaces of the part

3.1.3 Part strength

In some processes, some features need to be appropriately oriented because of the strength problems.[sub94] An example of the features is a small diameter pin pointing up in the Z direction. The following features are identified:

- Thin wall
- Slender protrusion

The cost function, $\mathcal{J}(\Theta)$, returns either 1 or 0 as in Eq.(3.7).

$$\mathcal{J}(\Theta) = \begin{cases} 1 & \text{if properly oriented} \\ 0 & \text{otherwise} \end{cases} \tag{3.8}$$

3.1.4 Surface finish

The *'stair-stepping'* effect that is the trace of the layer can be minimized by properly orienting the part, focusing on the following features.

- Cylindrical hole
- Cylindrical boss
- Curved features

The cost function, $\mathcal{L}(\Theta)$, is defined as in Eq.(3.8)

$$\mathcal{L}(\Theta) = \sum_{i=1}^{n} \mathcal{A}_i(\Theta) / \mathcal{A}_{part} \tag{3.9}$$

where

$\mathcal{A}_i(\Theta)$: the area of the surfaces which are subject to the stair-stepping effect

$\mathcal{A}_{part}$: the overall surface area of the part

3.1.5 Dimensional accuracy

Most of the SFF processes have the characteristic that the dimensional accuracy in the vertical direction is inferior to that of the horizontal directions. Therefore, the part should be oriented so that the feature with a critical dimension lies on the horizontal X-Y plane. To build an accurate functional part, we must consider the tolerance information specified on the design features. The tolerance information can be easily extracted from the model using the proposed system, even though the design features are not explicitly visible from the SFF application side. The features with critical tolerance are extracted, and the part is oriented so that the feature lies on the X-Y plane.

The cost function, $\mathcal{I}(\Theta)$, is described in Eq.(3.10).

$$\mathcal{I}(\Theta) = \begin{cases} 1 & \text{if the feature is oriented appropriately} \\ 0 & \text{otherwise} \end{cases} \quad (3.10)$$

3.1.6 Build time

The build time can be roughly estimated by the height of the part to be built. The cost function, $\mathcal{H}(\Theta)$, returns the normalized height of the part from the bounding box as follows.

$$\mathcal{H}(\Theta) = h(\Theta) / \mathcal{D}_{\max} \quad (3.11)$$

where

$h(\Theta)$: the height of the part at the orientation Θ

$\mathcal{D}_{\max}$: the maximum diagonal dimension of the bounding box of the part

3.2 Evaluation of the optimal part orientation

The optimal part orientation is evaluated by setting up an equation as in Eq.(3.12), and solving it using a nonlinear optimization procedure.

$$Min\ \mathcal{O}(\Theta) = w_1\mathcal{F}(\Theta) + w_2\mathcal{G}(\Theta) + w_3\mathcal{H}(\Theta) + w_4\mathcal{I}(\Theta) + w_5\mathcal{J}(\Theta) + w_6\mathcal{L}(\Theta) \quad (3.12)$$

where

$w_1, w_2, \ldots, w_6$: user defined weighting factors ($0.0 \le w \le 1.0$)

$\mathcal{F}(\Theta), \mathcal{G}(\Theta), \mathcal{H}(\Theta), \mathcal{I}(\Theta), \mathcal{J}(\Theta), \mathcal{L}(\Theta)$: the cost functions of the criteria

The convergence to the solution depends on the initial orientation of the part. Therefore, the procedure is repeated several times with different initial orientations. Among the converged solutions, the orientation with the minimum design function value, $\mathcal{O}(\Theta)_{\min}$, is chosen as the optimal orientation of the part.

3.3 Examples

Two simple cases are tested as shown in Figure 3.3. For the simplicity and to verify the result intuitively, only the trapped volume, support structure and build time criteria are considered in the example. The figure shows the configurations at the optimal orientations found for the weighting factors as represented in the figure respectively. The configurations are selected from the converged orientations with six different initial orientations for each case.

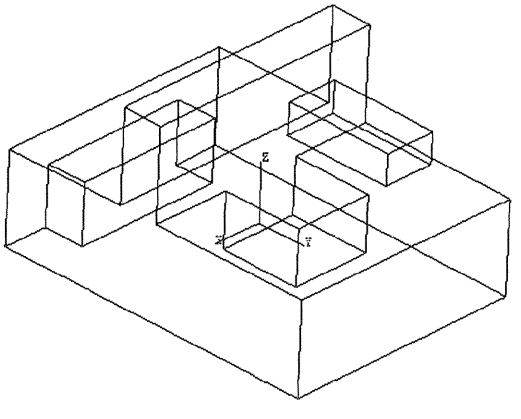


Figure 3.3 $w_1 = 0.3$ $w_2 = 0.6$ $w_3 = 0.1$
(Support > Trapped liquid > Build time)

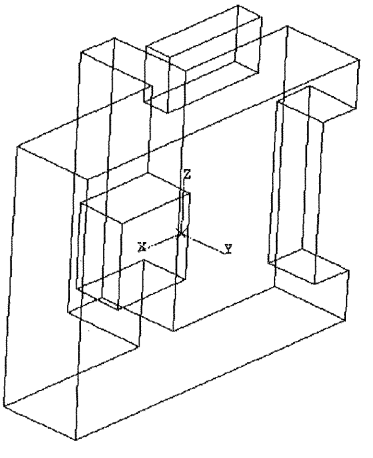


Figure 3.4 $w_1 = 0.6$ $w_2 = 0.3$ $w_3 = 0.1$
(Trapped liquid > Support > Build time)

4 SUMMARY

A feature-based approach to the SFF process software is discussed. In order for a SFF process to be integrated into a feature-based CAD system environment, the CAD system must provide a mechanism to map design features to the features that are meaningful to the SFF process. The feature-based approach simulates the behavior of human operators. In determining a part orientation, a human operator investigates the part and focuses on several outstanding features that are relevant to guidelines at a global point of view. By using the features, a faster and more reliable result is expected than traditional approaches based on geometric reasoning. The proposed system software architecture is implemented, and the SFF process software is integrated to the system to build a design-manufacturing system. The system is applied to the automatic determination of optimal part orientation process.

5 REFERENCES

[bro93] Willem F. Bronsvoort and Frederik W. Jansen, "Feature modelling and conversion - key concepts to concurrent engineering," Computers in Industry, 21:61-86, 1993

[dol94] A. Dolenc and I. Makela, "Slicing procedures for layered manufacturing techniques," Computer-Aided Design, 26(2):119-126, February 1994

[hen94] Mark R. Henderson, Gopal Srinath, Roger Stage, Kim Walker, and William Regli, "Boundary representation-based feature identification," Advances in Feature Based Manufacturing, chapter 2, pp.15-38, Elsevier Science B.V., 1994

[hou93] F.J.A.M. van Houten, O.W. Salomons and H.J.J. Kals, "Review of research in feature-based design," Journal of Manufacturing Systems, 12(2):113-132, 1993

[jac92] Paul F. Jacobs, "Rapid Prototyping & Manufacturing, Fundamentals of StereoLithography," Society of Manufacturing Engineers, 1992

[kim94] Kim, J. Y., Lee, K., and Park, J.C.,"Determination of optimal part orientation in stereolithographic rapid prototyping," preprint, July 1994

[kyp80] L. K. Kyprianou, "Shape classification in computer aided design," PhD thesis, Computer Science Laboratory, Cambridge University, 1980

[lub86] S. C. Luby, J. R. Dixon, and M. K. Simmons, "Creating and using a features data base," Computers in Mechanical Engineering," 25-33, November 1986

[set94] Seth Allen and Deba Dutta, "On the computation of part orientation using support structures in layered manufacturing," Proceedings of Solid Freeform Fabrication Symposium, pp.259-269,Austin, TX, August 1994

[sre94] Sreeram, P., and Dutta, D., "Determination of Optimal orientation based on variable slicing thickness in layered manufacturing," Technical Report UM-MEAM-TR-94-14, Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI, July 1994

[sub94] P. Kamatchi Subramanian, N. K. Vail, J.W. Barlow and H.L. Marcus, "Anisotropy in alumina processed by SLS", Proceedings of Solid Freeform Fabrication Symposium, pp.330-338, Austin, TX, August 1994

[suh94] Yong Seok Suh and Michael J. Wozny, "Adaptive slicing of solid freeform fabrication processes," Proceedings of Solid Freeform Fabrication Symposium, pp.404-411, Austin, TX, August 1994