Proceedings of DETC99:
1999 ASME Design Engineering Technical Conference
September 12-15, 1999, Las Vegas, NV

DETC99/DFM-8910

LAYERED MANUFACTURING WITH EMBEDDED COMPONENTS: PROCESS PLANNING CONSIDERATIONS

Jorge G. Cham[1] Beth L. Pruitt[1] Mark R. Cutkosky[1]

Mike Binnard[1] Lee E. Weiss[2] Gennady Neplotnik[2]

[1]Center for Design Research
Dept. of Mechanical Engineering
Stanford University

[2]The Robotics Institute
Carnegie Mellon University

ABSTRACT

This paper addresses the design and manufacturing of products with embedded components through layered manufacturing processes such as Shape Deposition Manufacturing (SDM). Embedding components allows the creation of novel designs such as "smart" products and integrated assemblies of sensors, actuators and other mechanical components. We present prototypes to illustrate the possibilities for such devices and we address the issues that constrain their process planning. Next, we present a combination of process planning algorithms and manufacturing methods that we have developed to support the design of layered products with embedded components.

1. INTRODUCTION

Layered manufacturing processes such as Shape Deposition Manufacturing [Merz 94] provide a number of capabilities not easily achieved with conventional manufacturing. Examples include the ability to create complex three-dimensional shapes with internal voids and passages, the ability to produce parts with continuously varying material properties and the ability to produce parts with embedded mechanical and electrical components.

The last of these capabilities is perhaps the least explored but also the most useful, particularly when working with plastic materials which can encapsulate components without subjecting them to high temperatures or pressures. The possibilities include embedding sensors and microprocessors to create "smart" products and embedding bearings, shafts and other items that have materials properties, tolerances, or finishes that would be difficult to obtain with the layered manufacturing process. Indeed, as long as issues such as inter-layer bond strength, control of material properties and surface finish during deposition, and materials shrinkage remain incompletely solved, the ability to embed discrete components is essential for achieving high-quality products. Embedding components is especially advantageous for creating small electromechanical systems, where the size and weight of the design are constrained by the need to provide sufficient material for screw holes and fasteners and where reliability is often limited by the integrity of the connections between components.

To our knowledge, there are few examples of layered products with embedded electrical and mechanical components. We present example artifacts created at Carnegie Mellon University and Stanford to provide an idea of the possibilities. We then examine the process planning requirements associated with embedding components. Fundamentally, the desire to embed components requires that the designer assume more control over the manufacturing process than is usually done when making homogeneous layered parts from a CAD model. We present a combination of process planning algorithms and manufacturing methods that we have developed to support the design of layered products with embedded components. We conclude with a look at future directions for heterogeneous products with embedded components and a design system that facilitates their creation.

2. LAYERED PROTOTYPES WITH EMBEDDED COMPONENTS

The development of prototypes with embedded electronics began at Carnegie Mellon University with the fabrication of an electronic game [Beck 92]. Subsequent products have included a series of compact wearable computers [Weiss *et al* 96]. The latest in the series is Frogman (Figure 2.1) a waterproof, wearable computer that can store maps for navigational aids, or detailed assembly drawings for service, maintenance, or field

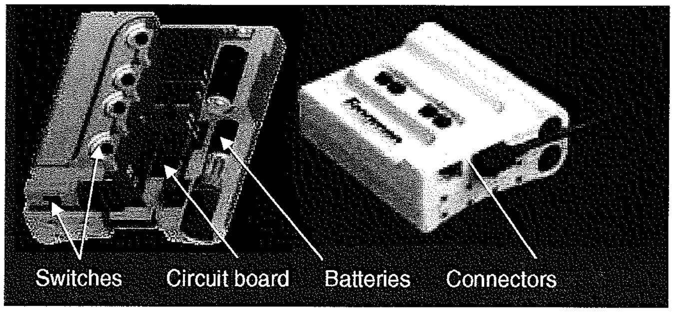


Figure 2.1. The Frogman waterproof computer developed at Carnegie Mellon University. A CAD model is shown on the left and the artifact is shown on the right.

operations. The device is built in layers of castable polyurethane and sacrificial wax. No custom tooling is required. Embedding the electronic components and interconnects directly in the urethane structure increases product durability and facilitates waterproofing. With the approach used to create Frogman, it becomes feasible to fabricate rugged, customized computer modules in small lot sizes.

Figure 2.2 shows another example of a device with embedded components. The device is a small robot limb with an embedded pneumatic cylinder and valves, a pressure transducer, and associated circuitry for signal amplification. There is also an embedded steel leaf spring at the joint. By embedding the components in a solid structure it is possible to locate the valves and pressure transducer immediately adjacent to the cylinder, avoiding some of the compliance and transmission delays typically associated with pneumatics connected through hoses or tubing. As with Frogman, encapsulating the electronics also helps increase durability.

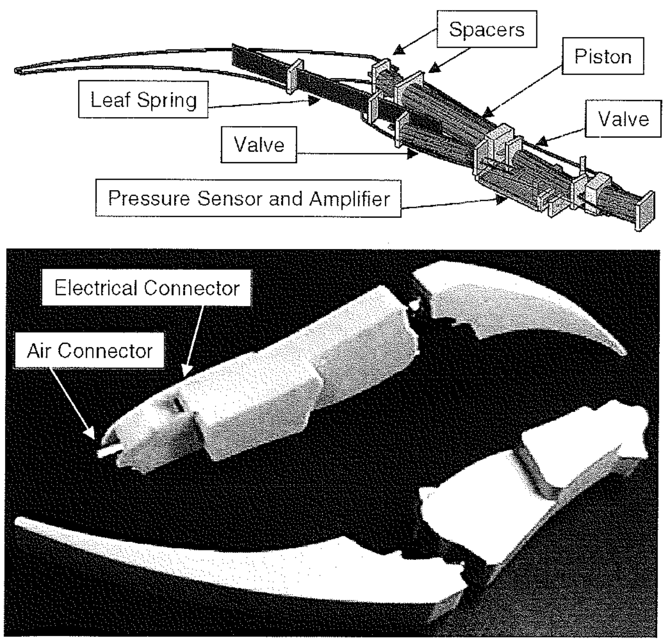


Figure 2.2. A prototype linkage with embedded pneumatic components and a flexible joint developed at Stanford.

Designs such as those in Figures 2.1 and 2.2 show some of the potential for creating complex products using layered manufacturing with embedded components. However, neither of these designs would have been possible without special attention to the manufacturing process, including the addition of extra processing steps. In the following sections we briefly describe the basic shape deposition process with embedded components and then discuss processing issues that arise with embedded parts. Finally we present algorithms that automate the process planning for layered manufacturing with embedded components.

3. SHAPE DEPOSITION MANUFACTURING WITH EMBEDDED COMPONENTS

The process on which we will focus our discussion is Shape Deposition Manufacturing. However, similar issues are likely to arise for other rapid prototyping processes when discrete components are inserted during the fabrication cycle. The characteristics of SDM manufacturing have been covered elsewhere (e.g., [Merz 94; Ramaswami 97]) and will not be repeated here. Figure 3.1 shows how discrete components can be added during the processing cycle.

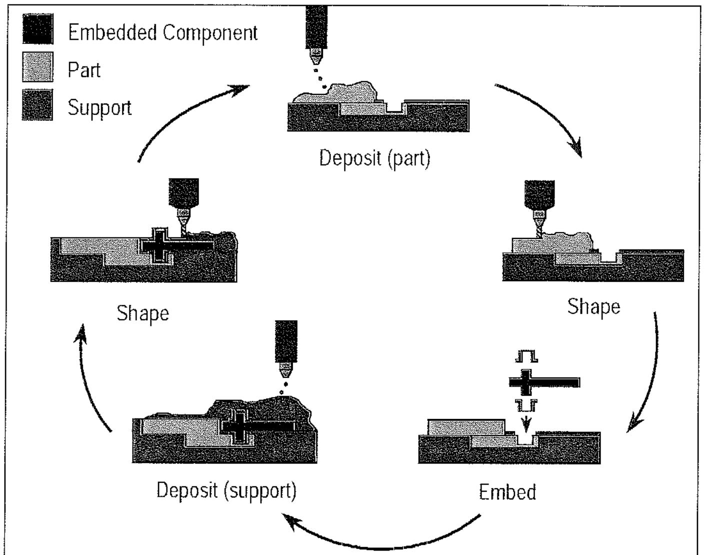


Figure 3.1. Shape Deposition Manufacturing Cycle. The figure shows a prefabricated link and bearings being embedded between shaping and deposition operations.

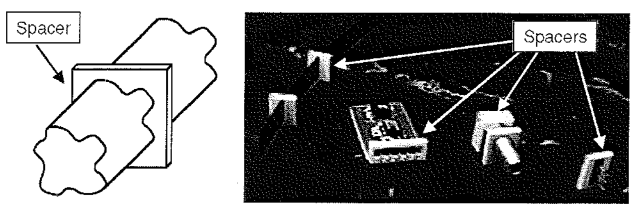
In this example, the existing support material is machined to create a mold to shape the underside of the part, which is deposited using casting, micro-casting, laser welding or other deposition processes. More support and part material are deposited and machined to continue the part build-up and to create support material features for the embedded components. The shaping process, performed after each layer of part or support material is deposited, is done by CNC milling to obtain the desired geometry, accuracy and surface finish. Prefabricated components can be embedded after any shaping step. In this case a prefabricated assembly of bearings and a linkage with an integrated shaft are inserted part way through the build cycle. The embedded component is then encased in the next layer of material (in this case more support) which is subsequently shaped. Additional processing steps such as cleaning, stress relief or surface preparation may also be inserted at any point in the SDM cycle. The key feature of the process is complete access to the interior of parts as they are being created.

Additional advantages of embedding components with SDM include:

- Deposition can be tailored to obtain the best material properties because the removal process is responsible for achieving net shape.
- Discrete components can be used where local geometry is critical. Gap pieces can also be created explicitly and embedded to provide critical alignment or spacing
- Fully functional components like sensors, motors, or bearings with known performance specifications can be included in an SDM part or assembly rather than trying to build them in place or duplicate off-the-shelf items

4. MANUFACTURING TECHNIQUES FOR SDM PARTS WITH EMBEDDED COMPONENTS

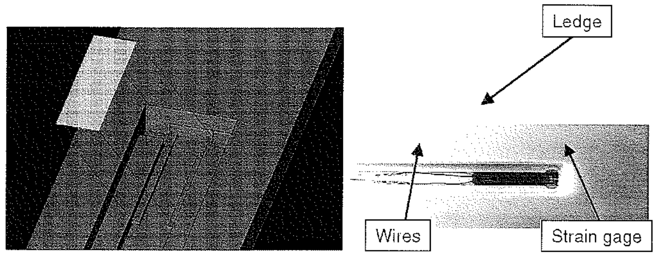
Embedding components is easy in principle, but each kind of component presents practical challenges in positioning and fixturing, maintaining functionality, and dealing with tolerances. Techniques used to solve these problems involve the addition of extra steps applied to the component before it is embedded or between subsequent shaping and deposition steps. These steps include for example: pre-treating the embedded component by adding sacrificial or temporary material (for instance, to preserve air space around a motor armature), and adding alignment features to the surrounding material design or the embedded component. In addition, the processes used to deposit, shape and remove the part and support materials must be thermally, chemically and mechanically compatible with the embedded components. We now present some of the practical challenges that arise in embedding components and examples of techniques used to address them.



(a) (b)

Figure 4.1. (a) Example of using additional parts, or spacers, for fixturing of irregularly shaped components. The figure in (b) shows a leaf spring, circuit board, tubing fitting and electrical connectors with their spacers. These spacers were manufactured in a separate production run using the same design and manufacturing interface.

4.1 Positioning and Fixturing. Locating embedded components is particularly a problem when the component has an irregular shape or is flexible, as is the case in embedding wires or tubing. A solution is to add locating features to the embedded component by creating additional parts made of part or support material (see Figure 4.1a). The additional parts are then embedded with the component. An example is the use of spacers for connectors, valves, pistons and motors, as shown in Figures 4.1b, 4.4 and 4.5.

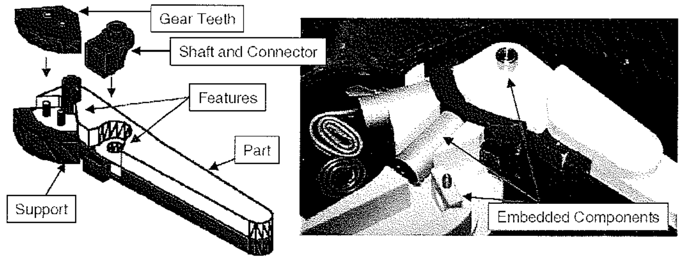


(a) (b)

Figure 4.2. Example of creating features for fixturing. Here the features help position a strain gage and wires in a urethane specimen. Placement of the gage on the smallest ledges also allows the urethane to contact both sides of the gage for complete encapsulation. (a) CAD model and (b) strain gage aligned in pocket.

Another general solution is to create locating features, like pockets or posts, in the preceding part geometry that the embedded component can be aligned or affixed to. An example of this is the use of pockets and "shelves" to position a strain gage and its wires in a urethane tensile specimen (see Figure 4.2). The gage is supported on a pair of ledges that allow the liquid urethane to flow to the front and back of the gage for good adhesion. The channel is sized to the width of the gage to provide alignment and runs the length of the component to position the wires and guide them to the edge of the part. Another example is shown in Figure 4.3, where posts and surrounding geometry help position the embedded components.

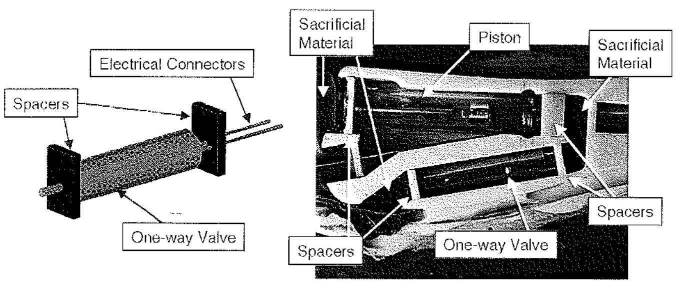
4.2 Functionality. Successful embedding of a component requires that its function be preserved in subsequent processing steps. Of course, techniques to achieve this will vary not only from component to component, but also according to the build direction and surrounding geometry. In general, the types of challenges encountered thus far have required the selective use of sacrificial and temporary materials and additional parts like spacers as demonstrated by the following examples.



(a) *(b)*

Figure 4.3. Embedded components can be positioned by creating locating features in the preceding geometry. (a) Model of gear teeth and a shaft with a connector being embedded in a link. (b) This link is subsequently embedded in the construction of the second link. Magnetic sensors detect the gear teeth and provide position sensing.

Moving parts: Bearings, actuators and other pre-fabricated mechanisms must be protected from infiltration by part material. One example is a pneumatic actuator that was embedded in the prototype urethane robot leg. Here, the inlet port and the piston were encased in sacrificial materials (glycerin soap and wax) that were easily removed once the part was finished (see Figure 4.4b). Another example shown in Figure 4.5 is the use of spacers to define barriers between part and support.



(a) (b)

Figure 4.4. (a) Model of an air valve and spacers. The combined component is then placed in a pocket as shown in (b). (b) Example of using sacrificial material to encase critical parts of the embedded component. The figure shows a pneumatic actuator and an air valve encased on either side with red wax. The spacers are used as dams here when pouring different materials on either side of the spacer.

Flow channels: Similar to moving parts, fluid-system components such as pressure sensors and valves require that the airspace and connectors be protected. In the case of the valves, the inlet and output ports were encased in sacrificial material to protect them during subsequent deposition steps (see Figure 4.4). The sacrificial material used must be easily and completely removable in order to prevent clogging or restricted flow. For example, glycerin soap, which is soluble in water, was used to protect the valve ports.

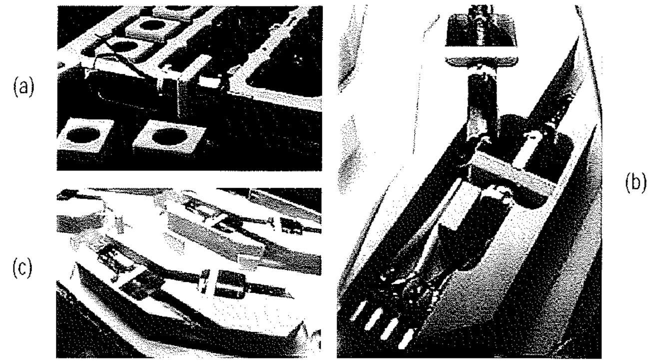


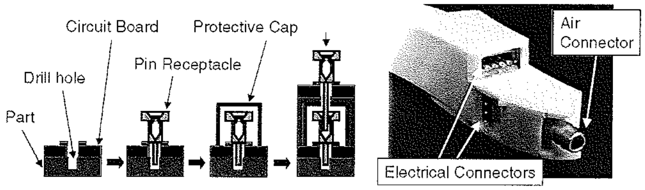
Figure 4.5. Embedded components like these electric motors must often be embedded half in part material and half in support material. The spacers shown in (a) are assembled with the motors before they are encapsulated (b). The photograph in (c) shows the motors with the front end encased in soap support material. Urethane then encapsulates the exposed sections.

Contact and adhesion: Mechanical components that transmit forces such as motor shafts or springs need strong adhesion between the part and the embedded component. Similarly, the performance of embedded strain gages depends on the quality of the bond with the part material (Figure 4.2). The design of the mechanical interface and surface preparation must provide features that allow sufficient access of part material for good mechanical contact and strength. For example, the embedded leaf springs shown in Figure 4.1 were abraded and altered to include gripping teeth at the interface with part material. In contrast to embedded components that need good adhesion, the Frogman computer purposefully embedded a removable teflon-coated aluminum mandrel with poor adhesion in order to create cavities for batteries.

4.3 Multilayer Interfaces: Components that cross the junctions of two or more layers require special consideration. For example, it was necessary in the case of the Frogman computer to connect a circuit board embedded in a layer of urethane to another circuit board embedded in a subsequent layer (see Figure 4.6a). Other examples are components between part and support material such as connectors for electrical and fluidic systems that must be accessible to the outside of the part (see Figure 4.6b).

Accessibility: Connectors such as the ones mentioned above must often be made accessible between SDM steps. In the Frogman computer, for example, a protective cap was used to prevent part material from completely encasing the electrical connectors as shown in Figure 4.6a. Spacers and sacrificial material were used to partially encase the electrical and pneumatic connectors of the prototype leg shown in Figure 4.1b and 4.6b.

Stress and Abuse: Embedded components experience stress concentrations at sharp transitions between materials; provisions must be made for protection and stress relief at the exit of items like wires or connectors. For example, silicone was applied to the circuit components in the Frogman computer in order to protect them from the stresses resulting from shrinkage.



(a) (b)

Figure 4.6. (a) Electrical connection between different SDM steps. Here, a pin receptacle is first soldered to a circuit board. A protective cap covers the receptacle to avoid complete encapsulation by the next deposition step. A drill hole then makes the receptacle accessible to the circuit board in the next layer. (b) Example of connectors that join components inside the part to the exterior of the part.

Alignment and Tolerances: Dimensional and locational tolerances of the embedded components are particularly an issue when the component lies at the boundary of two layers since subsequent shaping steps can damage the part. Factors such as poor alignment, dimensional inaccuracy, and part shrinkage can stack-up unfavorably to place the embedded component in a toolpath. Components that protrude from one layer into the next, such as shafts and bearings, need to be protected during machining from damage by tools or debris. Such embedded components can be defined with a dimensional buffer zone and a sacrificial material can be used to temporarily encapsulate and protect them during processing. Figure 4.7 shows a modified build cycle for the example first presented in Figure 3.1.

4.4 Integrated Analysis. The design of parts with embedded components requires integrated analysis. In addition to design rules that ensure manufacturability, analysis tools are needed to allow the designer to verify functionality, mechanical behavior of composite structures (e.g., deflections and stress gradients) and thermal profiles during operation. For example, embedding electrical components affects their heat dissipation and embedding wires affects the mechanical integrity of the part by introducing stress concentrations as well as varying material properties. Integrated thermal analysis for cooling of embedded electronic components is under investigation at Carnegie Mellon [Egan 96].

5. AN ALGORITHM FOR DESIGN AND PROCESS PLANNING WITH EMBEDDED COMPONENTS

The techniques described in the previous section were performed manually for prototypes such as the Frogman computer and the robotic limbs in Figures 2.2 and 4.3a. As we have seen, the inclusion of embedded components requires that designers have more control over the manufacturing process than is usually the case when sending a CAD model out for layered manufacturing using a homogeneous material. Presently, the designer must understand the SDM process and understand how parts with embedded components will be processed. Indeed, the success of the prototypes shown was largely due to the fact that the designers were also the manufacturers.

Furthermore, it does not appear, at least for SDM processes, that completely automated design decomposition and process planning are feasible in the immediate future. To this end, we are developing a semi-automated approach that will make it easier for designers to build heterogeneous products created by SDM.

Our solution is an extension of the Design-by-Composition approach described in [Binnard and Cutkosky 98]. In this system, designers use a library of *primitives* as design building blocks (see Figure 5.1). Each primitive has an associated manufacturing plan, in the form of part and support material *compacts* and a precedence graph. A compact is a volume of material that can be manufactured in one cycle of shaping and deposition, as described in [Merz 94]. When the user creates a new design from two primitives, the CAD system automatically combines the two manufacturing plans to create a plan for the new design. The merging algorithm detailed in [Binnard and Cutkosky 98] allows designers to merge primitives and automatically computes the resulting compacts and process ordering constraints.

In order to free the designer from manually planning all the special techniques described in Section 4, we create pre-defined libraries of generic embedded components. The embedded components, as defined in the library, include the necessary

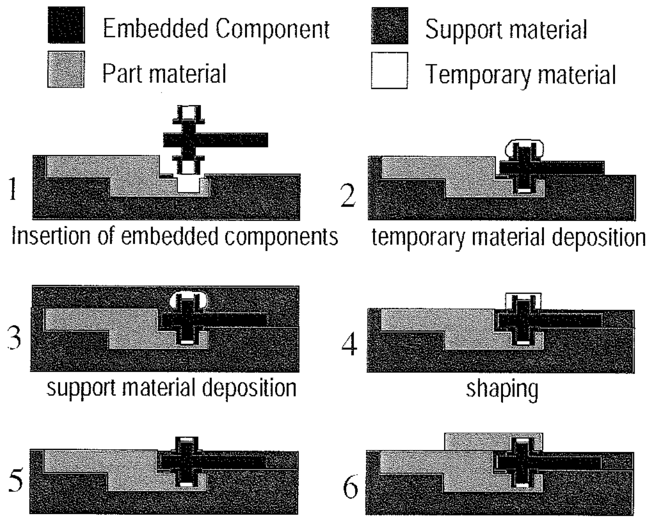


Figure 4.7 Embedded components that are exposed between SDM steps can be damaged by shaping operations such as machining. Here, the embedded bearing is first modeled with a buffer zone around it. A temporary material is deposited on the embedded component to provide this buffer zone for the shaping operation.

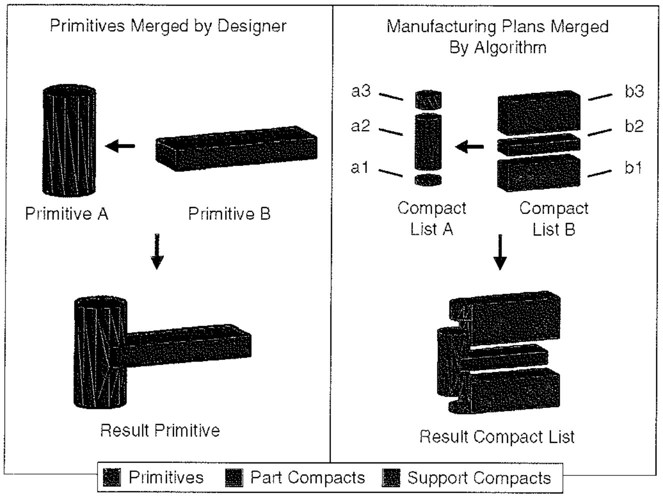


Figure 5.1 Design-by-Composition: The designer builds designs by combining primitives with Boolean operations. Each primitive contains its own high-level manufacturing plan.

process modifications, such as spacers, channels for wiring and connectors, fixturing features for accurate placement, and standoffs for flow access of the next layer poured. An example is shown in Figure 5.2.

The challenge is to find ways to represent these specific manufacturing techniques, which include geometric and process ordering constraints, in the library and to ensure that library components remain valid after merging with other primitives. We believe that a solution lies in the following two directions:

1. Expand the list of properties that compacts can have. This list already includes material type and geometry, but can be expanded to include material-specific Boolean operations and compact-list merging constraints.
2. Encapsulate the manufacturing techniques discussed in Section 4 by constructing library elements from a collection of compacts of part, support, and embedded-component materials that have special material properties and ordering constraints.

As a starting point, we have expanded the previous merging algorithm to support primitives that are of "embedded" material. The designer can now create embedded components from libraries of simple shapes and merge them with other primitives. For example, the ledges and channels needed for embedding the strain gage shown in Figure 4.1 were created by specifying an embedded component with the shape of the ledges and channels.

Because embedded components such as shafts, bearings, sensors, and circuit boards cannot be divided, the compacts that represent them must be handled differently from ordinary part and support compacts. Thus, the geometric merging procedures must be modified as well as the material merging and ordering rules. We now present a simplified example to illustrate the three general parts of the original algorithm described in [Binnard and Cutkosky 98] and the modifications made to them.

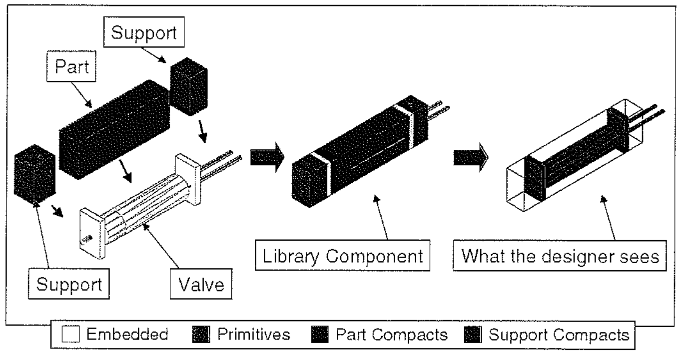


Figure 5.2 Library Components: The designer places embedded components from a predefined library into an emerging design. These library components contain the extra steps needed to successfully embed them. In this example, the definition of an embedded valve already contains the necessary part, support and spacer compacts.

In the example, we wish to merge the two simple primitives A and B with a chosen operation, as shown in Figure 5.3. Both primitives contain the list of part and support compacts that represent their manufacturing plan. For example, compact list A is shown in Figure 5.1, and it consists of compacts $\{a_1, a_2, a_3\}$. When primitive B is an embedded component, the Result primitive from the operation will be different, since compact b_2 is of "embedded" material. The Appendix contains a listing of the algorithm, discussed below.

Part 1. Generate Intersection List C. Part 1 of the algorithm generates a third list of compacts, C. This list is generated first by finding the geometric intersection between pairs of compacts from the parent sets A and B. Thus, the list C will contain $\{a_1b_1, a_1b_2, \ldots a_2b_1, \ldots\}$, where a_ib_j is the geometric intersection between compacts a_i and b_j. In the original algorithm, these intersections are then removed from their parent compacts. For example, this results in the adjusted lists

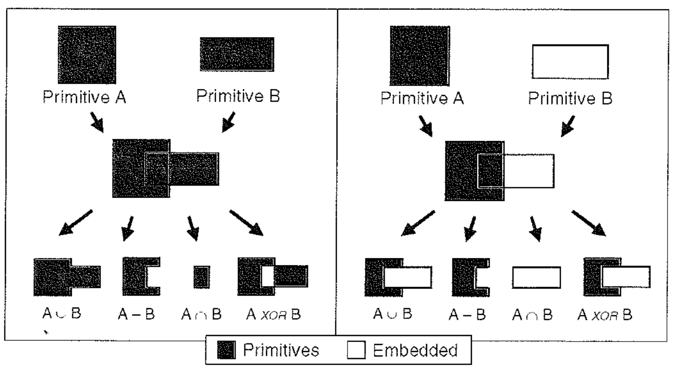


Figure 5.3 Simplified example of primitive operations. The figure on the left shows the operation results of two primitives of "part" material. The figure on the right shows the results from a "part" and an "embedded" material primitive.

A' and B', shown in the left side of Figure 5.4. Finally, each compact in C is assigned a material type based on the materials of the parent compacts and the rules shown in the material-merging truth table of Figure 5.5.

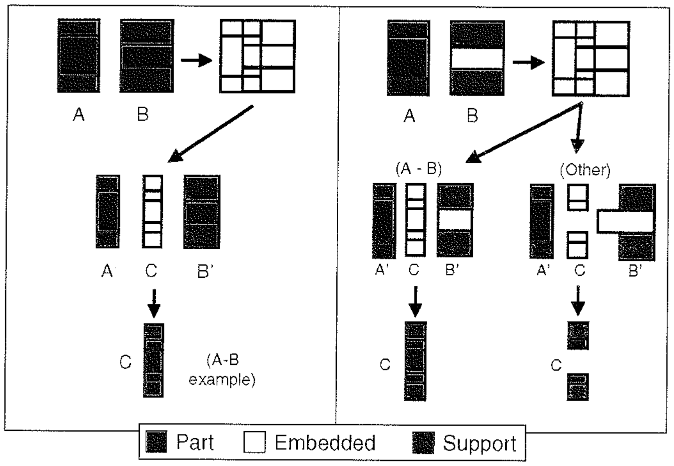


Figure 5.4 Generation of intersection list C. First, the geometric intersections are generated and removed from the parent compacts. The material types of these intersection compacts are determined by the material merging truth tables.

In the modified algorithm, we must first check for the geometric intersection between compacts of embedded material, which will return an error. We then proceed as in the original algorithm, except for geometric intersections between an embedded compact and a non-embedded compact. Since we cannot divide the embedded compact, we leave it intact and remove the intersection from the non-embedded compact. One exception is for the (A - B) operation, where we *can* divide the embedded compact since it will not be part of the final Result set. The final modification to Part 1 of the algorithm is the extension of the truth table to include material of type "embedded," as shown in bold in Figure 5.5.

Add
$A \cup B$

a	*b*	*c*
P	P	P
P	S	P
S	P	P
S	S	S

Subtract
$A - B$

a	*b*	*c*
P	P	S
P	S	P
P	**E**	**S**
S	P	S
S	S	S
S	**E**	**S**

Intersect
$A \cap B$

a	*b*	*c*
P	P	P
P	S	S
S	P	S
S	S	S

Exclusive or
$(A+B)-(A \cap B)$

a	*b*	*c*
P	P	S
P	S	P
S	P	P
S	S	S

Figure 5.5 Modified material-merging truth tables. P = Part, S = Support, E = Embedded. The only modification is to the case of (A - B), since it is the only case where we divide the embedded compact.

Part 2. Find the Result List. The second part of the algorithm simply chooses the Result list among the lists A', B' and C, based on the desired operation. For example, Figure 5.6 shows that for the case of (A - B), the Result set is the union of lists A' and C.

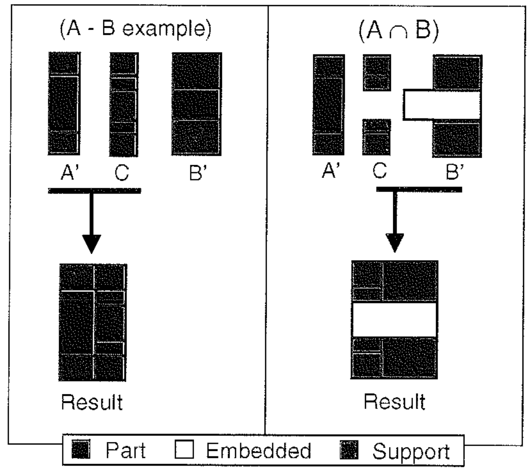


Figure 5.6 Find the Result List. The final Result compact list is selected among lists A', B' and C based on the operation.

In the modified algorithm, we must make a special case for the (A ∩ B) operation. Part 1 leaves all embedded compacts in their parent lists, A' or B'. If there are any embedded compacts in either list A' or B', we want to include them in the Result list. For example, as shown in the right side of Figure 5.6, in order to include the embedded compact in the Result list, we have to convert all non-embedded compacts in B' to support material. This prevents the embedded compact from "sticking out" of the Result list. Thus, the final Result list is the union of lists C and the modified list B'.

Part 3. Establish Precedences. A precedence graph contains the necessary manufacturing order of the compacts in a compact list [Pinilla *et al*, 97]. An example is the compact list A shown in Figure 5.1. Compact a_1 must be manufactured before compact a_2, so a_1 *precedes* a_2, or $a_1 \rightarrow a_2$. Since we are merging two compact lists, we must establish the precedences of the Result compact list. Precedences between compacts within the parents lists A' and B' remain the same as before the merging operation. This is because all intersecting surfaces between the two lists are parallel to the build direction. Precedences between compacts within the intersection list C are constructed based on the precedences of their parent compacts and a set of rules (listed in the Appendix), as shown in the left side of Figure 5.7. In the original algorithm, we did not need to establish precedences between compacts in C and compacts in A' or B'.

In the modified algorithm, precedences are established the same way as in the original algorithm. However, Part 1 left the embedded compacts in their parent lists in order to avoid

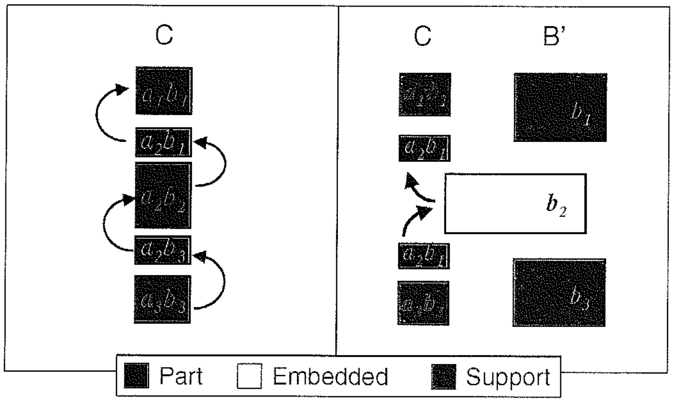


Figure 5.7 Establish Precedences. As shown at left, the manufacturing order of the new compacts in the intersecting list C depends on the precedences of their parent compacts. In the modified algorithm, we must also establish precedences between these compacts and embedded compacts, as shown at right.

dividing them. Therefore, we must establish precedences between these compacts and the new compacts in the intersection list C, as shown in the right side of Figure 5.7. The modified precedence rules are also listed in the Appendix.

6. DISCUSSION AND FUTURE WORK

Unique mechatronic designs that would be difficult or impossible to make by conventional means can be created using layered manufacturing processes with embedded components. However, constraints on the design are still necessary to ensure manufacturability. If designers are going to exploit the capabilities of layered processes, it will be necessary to provide a design interface that allows them to create parts and embed components without laborious process planning.

The algorithm presented above is part of an effort to develop such a design interface. The interface will include libraries of previous designs that can be integrated with any new design. The library should include standard components such as shafts, motors, bearings, batteries and circuits for embedding. We have described an approach in which these components are defined in terms of process-related geometries and associated ordering constraints. Components are defined with features that facilitate placement of parts that are difficult to locate accurately or have special alignment and encapsulation requirements (e.g., embedded sensors and motors).

The embedded component library, when combined with the merging algorithm, is analogous to feature-based CAD. The features contain fragments of predefined process plans, with the result that manufacturing planning and analysis can be performed incrementally, as the design evolves [Kambhampati *et al*, 93]. In the present case, we have an important advantage: for processes like SDM, in which parts are built in layers and in which the support material eliminates traditional fixtures, there are relatively few global feature interactions of that kind that complicate traditional feature-based CAD/CAM.

Future improvements to the system will allow user interaction in the selection of the build order to obtain optimal surfaces and protection of embedded components. An open question is whether the library definitions can be made sufficiently flexible that they will work with many parametric variations, or whether a large number of special-cases will be needed for particular geometries, orientations and material combinations.

Investigations are also underway to characterize the performance and reliability of embedded components like sensors and actuators. Future research will address the development of an integrated design interface that includes design rules (analogous to those developed to ensure the manufacturability of VLSI circuits) and tools for mechanical and thermal analysis and wiring layout.

ACKNOWLEDGMENTS

The authors thank the members of the Stanford CDR and RPL teams for their contributions to the work documented in this paper. This work is supported by the National Science Foundation under grant MIP9617994 and by the Office of Naval Research under N00014-98-1-0669.

REFERENCES

Beck, J.E., Prinz, F.B., Siewiorek, D.P., and Weiss, L.E., **"Manufacturing Mechatronics Using Thermal Spray Shape Deposition,"** *Solid Freeform Fabrication Symposium*, The University of Texas At Austin, August, 1992.

Binnard, M., **"Design by Composition for Rapid Prototyping,"** *PhD Dissertation*, Stanford University, Stanford, CA, 1999.

Binnard, M., Cutkosky, M.R., **"Building Block Design for Layered Shape Manufacturing,"** *Proceedings of the ASME Design Engineering Technical Conference*, Atlanta, GA, September 13-16, 1998.

Egan, E., Amon, C.H., **"Cooling Strategies for Embedded Electronic Components Fabricated by Shape Deposition Manufacturing,"** IEEE Intersociety Conferences on Thermal Phenomena

Finger, S., Terk, M., Subrahmanian, E., Kasabach, C., Prinz, F., Slewiorek, D.P., Smailagic, A., Stivoric, J., Weiss, L., **"Rapid Design and Manufacture of Wearable Computers,"** Communications of the ACM, Vol. 39, No. 2, February, 1996.

Kambhampati, S., Cutkosky, M. R., Tenenbaum J. M. and Lee, S-H, **"Integrating General Purpose Planners and Specialized Reasoners: Case Study of a Hybrid Planning Architecture,"** *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 23, No. 6, November/December, 1993, pp. 1503-1518.

Merz, R., Prinz, F.B., Ramaswami, K., Terk, M., Weiss, L., "**Shape Deposition Manufacturing,**" *Proceedings of the Solid Freeform Fabrication Symposium*, University of Texas at Austin, August 8-10, 1994.

Pinilla, J. M., Kao, J. H., Binnard, M., Prinz, F. B., **"The Compact Graph Format: an Interchange Standard for Solid Freeform Fabrication,"** *NIST Measurement and Standards Issues in Rapid Prototyping Workshop*, Gaithersburg, MD, 1997.

Ramaswami, K., "**Process Planning for Shape Deposition Manufacturing,**" *PhD Dissertation*, Stanford University, Stanford, California, 1997.

Weiss, L.E., Prinz, F.B., Neplotnik, G., Padmanabhan, P., Schultz, L., Merz, R., "**Shape Deposition Manufacturing of Wearable Computers,**" *Proceedings of the Solid Freeform Fabrication Symposium*, The University of Texas at Austin, August 10-12, 1996.

Weiss, L.E., Merz, R., Prinz, F.B., Neplotnik, G., Padmanabhan, P., Schultz, L., Ramaswami, K., **"Shape Deposition Manufacturing of Heterogeneous Structures,"** *SME Journal of Manufacturing Systems*, Vol. 16, No. 4, 1997.

APPENDIX

For a more detailed description of the algorithm and the precedence rules, including proofs of completeness and an analysis of complexity, see [Binnard 99].

Parts 1 and 2 of the compact merging algorithm are shown Figure A.1. For Part 3 of the algorithm, the precedence rules between compacts in the intersecting list C, $a_x b_y$ and $a_i b_j$, depend on the precedences of their parents:

$$a_x b_y \rightarrow a_i b_j \text{ if } (a_x = a_i \text{ and } b_y \rightarrow b_j) \text{ or } (b_y = b_j \text{ and } a_x \rightarrow a_i) \text{ or } (a_x \rightarrow a_i \text{ and } b_y \rightarrow b_j)$$

The precedence rules between an embedded compact, b_y, and compacts in the intersection list C, $b_j a_i$ and $b_j a_k$, also depend on the precedences of their parents:

if $(b_y \cap a_i) \,!= \varnothing$ **and material**(b_y) = **EMBED**
- **if** $(b_y \rightarrow b_j)$ **then** $b_y \rightarrow b_j a_i$
- **if** $(b_j \rightarrow b_y)$ **then** $b_j a_i \rightarrow b_y$
- **if** $(b_y \rightarrow b_j)$ **and** $(a_i \rightarrow a_k)$ **then** $b_y \rightarrow b_j a_k$
- **if** $(b_j \rightarrow b_y)$ **and** $(a_k \rightarrow a_i)$ **then** $b_j a_k \rightarrow b_y$

```
∀ b ∈ B                                        // PART 1: loop through all
  ∀ a ∈ A                                      // A and B compacts.
    if matl(a)=EMBED ∧ matl(b)=EMBED
        error: EMBED components intersecting.
    if (matl(a)≠EMBED ∧ matl(b)≠EMBED)
        ∨ (matl(a)≠EMBED ∧ matl(b)=EMBED ∧ oper=SUBTRACT)
        new(c ∈ C) = a ∩ b                     // create intersections per
        matl(c) = f(matl(a), matl(b), oper)    // truth table
        a = a - c                              // subtract from source
        b = b - c                              // compact sets
    if matl(a)=EMBED ∧ matl(b)≠EMBED           // embedded compacts in A
        b = b - (a ∩ b)
    if matl(a)≠EMBED ∧ matl(b)=EMBED           // embedded compacts in B
        a = a - (a ∩ b)

if (oper = ADD) ∨ (oper = XOR)                 // PART 2: copy non-
  C = C ∪ A ∪ B                                // intersecting compacts
if oper = SUBTRACT
  C = C ∪ A
if oper = INTERSECT                            // make sure embedded
  if ∃a[a∈A ∧ matl(a)=EMBED]                   // compacts and support get
    ∀a[a∈A ∧ matl(a)≠EMBED]                    // copied
      matl(a)=SUPPORT
    C = C ∪ A
  if ∃b[b∈B ∧ matl(b)=EMBED]
    ∀b[b ∈ B ∧ matl(b)≠EMBED]
      matl(b)=SUPPORT
    C = C ∪ B
```

Figure A.1. Embedded component merging algorithm