Proceedings of DETC'98
1998 ASME Design Engineering Technical Conferences
September 13-16, 1998 - Atlanta, Georgia

DETC98/CIE-5700

An Architecture for Interoperability of Layered Manufacturing Data

Xiaoping Qian
Research Assistant

Department of Mechanical Engineering
The University of Michigan
Ann Arbor, MI 48109-2125
xpqian@engin.umich.edu

Debasish Dutta
Associate Professor

Department of Mechanical Engineering
The University of Michigan
Ann Arbor, MI 48109-2125
dutta@engin.umich.edu

ABSTRACT

In this paper, we address the lack of interoperability of data in commercial layered manufacturing systems. Since it is a new field, all machines require users to deal with vendor specific softwares and data formats. We propose a simple architecture, one that already exists in the NC machining domain, to address this issue. Analogous to CLData, we propose LMData that is not vendor specific and can be used on a variety of LM machines. Implementation and examples are presented.

Keywords

Layered Manufacturing, NC Machining, CLData

1.0 INTRODUCTION

Layered manufacturing (LM), as a fabrication technology, has gained wide attention in universities and industry. Many layered manufacturing machines based on different layered manufacturing processes have been developed and are available commercially. All LM machines build parts by depositing material, layer by layer, under computer control. While all commercial machines make non-metallic parts, some processes (e.g., Direct Metal Deposition (DMD), Shape Deposition Manufacturing (SDM), Laser Engineered Net Shaping (LENS), etc.,) under development in universities and government laboratories are now focusing on building metallic functional parts. These processes are powerful and can be used to make complex parts with internal heterogeneities and/or multiple materials.

Material removal processes, complementary to layered manufacturing, have matured significantly over the past three decades. Computer Numerically Controlled (CNC) machines have become the industry norm for metal removal (by lathe, mill, drill, EDM, etc.). Since material removal processes "carve out" the part from a workpiece/stock material, any non-geometric complexities in the part (such as heterogeneities or multiple materials) must be appropriately contained in the stock. This is a difficult task. However, there might be situations where (intricate) features on parts from the CNC domain might be easily fabricated by LM. On the other hand, it is unlikely that heterogeneous parts can be (or will be) created by LM alone. Several LM processes, e.g., Sanders, Contour Crafting, SDM, etc., are based on the synthesis of material deposition and removal method. It is, therefore, necessary to consider fabrication environments where both removal and deposition methods will be available and work harmoniously. This synthesis will require a seamless integration of data transfer between the two domains.

While much research has been reported on LM process planning tasks (build orientation, support structures, adaptive slicing, etc.), integration issues are just beginning to be considered. In this paper we investigate a subproblem the interoperability of data between dissimilar layered manufacturing machines. We propose a simple architecture for this task. Within this, common data sets (from CAD systems) can be processed for fabrication by different LM systems.

The remainder of this paper is structured as follows. After a brief overview of two LM systems in Section 2, we outline our architecture for interoperability of layered manufacturing data in Section 3. A detailed description of LMData, an important component of the architecture is presented in Section 4. A short command summary of existing LM machines including Sanders and Stratasys is presented in Section 5, which is followed by the postprocessor working model. In Section 6, we describe our implementation and examples and conclude the paper in Section 7.

2.0 BACKGROUND

2.1 Problem analysis

Layered manufacturing, as a prototyping technology, started in the late eighties. For an overview, refer to Jacobs, 1992 and Marsan and Dutta, 1997. Since we consider two commercial LM systems later in this paper, a brief overview of these two processes is provided here.

- Fused Deposition Modeling (FDM) developed by Stratasys Inc. FDM uses thermoplastic materials or nylon for building parts. In this process, material (in the form of a wire) is pulled into a heating chamber where it is heated to just above its melting point. This semi-liquid material is then deposited on a foam foundation through a pencil-like nozzle by computer controlled X-Y motion of the head. Once a layer is finished, the foam foundation is indexed down and the process is repeated.
- ModelMaker developed by Sanders Prototype Inc. It produces precision parts by combining inkjet plotting with milling. A dual inkjet subsystem rides on a precision X-Y drive carriage and deposits both thermoplastic and wax materials on the build substrate. The X-Y carriage also energizes a flatbed milling subsystem for maintaining precise Z-axis dimensioning of the model by milling off the excess material of the current build layer.

For the users, the basic interactions with these LM machines are through vendor-specific softwares. For example, steps involved in the fabrication a part on FDM are shown in Figure 1 (FDM, 1994). They are in general machine-dependent procedures. (Note, slicing can be done separately.) Each LM machine has its own machine code file (e.g., .sml in FDM, .bin in Sanders, .V in SLA) and build softwares (e.g., QuickSlice in FDM, ModelWorks in Sanders, Maestro/JR in SLA) to generate the propriety machine codes. The processes are illustrated in Figure 2. Due to the propriety nature of the machine codes, and the different layer forming processes, the build softwares are quite different and typically machine-specific. That is, the output from QuickSlice cannot be used to make a part on Sanders and vice versa. In an integrated environment, where several different LM machines exist, this lack of interoperability is a severe bottleneck. This is an issue we address in this paper.

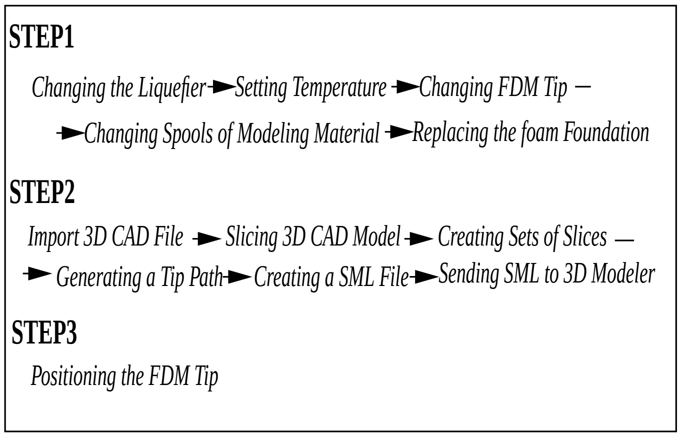

Figure 1. Fabrication Steps in FDM

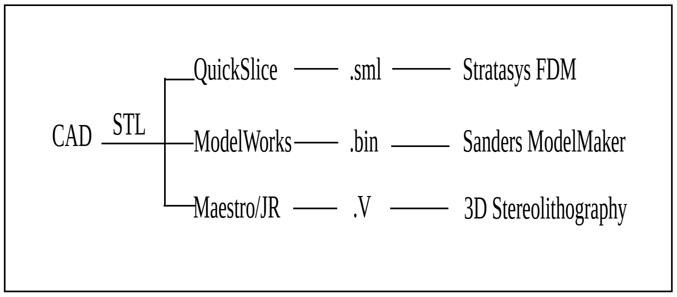

Figure 2. Current Architecture in LM

2.2 Literature review

The focus of this paper is an architecture for data transfer between distinct LM systems. Currently STL file format is the *de facto* industry standard in the LM domain. All commercial systems require the STL model as input which most CAD systems can generate. However, STL has serious drawbacks, e.g., faceted representation, truncation, verbose, lack of topological information, etc. Common sliced data formats for layered manufacturing processes have also been proposed, e.g., Common Layered Interface (CLI) (BRITE-EURIAM, 1994), SLC by 3D Systems and HPGL. Although sliced data format is an attempt to improve upon the deficiencies of STL, it is also process-specific. This is because orientation determination, support design, and layer thickness, etc., depend on the LM process. A Solid Interchange Format (Sequin and McMains, 1995) is being developed for the layer manufacturing domain, but not much is available in the literature yet. For further details on existing data formats in LM and their dis/advantages, refer to Kumar and Dutta, 1997.

Since LM processes can make functional parts, our ultimate goal is the development of an architecture and computational tools for the seamless integration of material deposition (LM) processes and material removal (NC) processes. In such an integrated environment, the product could be fabricated either by NC machining, or by layered manufacturing, or by a combination of both. There are several LM processes which use NC machining during the layer forming processes, e.g., Sanders uses NC milling to trim the layer to get the accurate height dimension (ModelWorks, 1996), SDM uses NC milling machine to make the 3D layer (Weiss, et al., 1997). Some issues in the integration of layered manufacturing and traditional material removal processes in an integrated manufacturing setup have been considered by Kulkarni and Dutta, 1997.

3.0 THE ARCHITECTURE

Before we present the architecture for interoperability of data between LM systems, we define some terms that are used in this paper.

In the NC domain, *Cutter-location data (CLData)* is a description of the tool positions and the desired sequence of operations. It is independent of the particular machine tool and the manufacturer (Chang and Melkanoff, 1989).

For the LM domain, we propose the following concepts:

Layered manufacturing data (LMData): LMData describes the tool head configuration and its movement during the LM process. It is designed to be a neutral data file.

Machine definition file (MDEF): MDEF is the LM machine definition file. In MDEF, machine-dependent commands and parameters are stored to facilitate the conversion of LMData into machine code.

3.1 CLData and its similarity with LMData

Interoperability among CNC machines has been greatly enhanced due to the introduction of cutter location data (CLData) format. Historically, the success of NC can be attributed to two major factors: the improvement of the NC machine tool-controller system and the development of software programming aids (Chang and Melkanoff, 1989). For the first generation NC machines, the inputs were proprietary machine codes. To increase the exchangeability of manufacturing information, the CLData was introduced as an input that could be accepted by all NC machines. The CAD/CAM systems output CLData which is then postprocessed to generate specific machine code. Besides CLData, a 32-bit binary exchange code for CLData, or BCL for short, is provided by EIA RS-94 to allow different machines to operate from the same input data. In this way, the output from commercial CAD/CAM systems are customized to run on every NC machine.

In this paper, we apply the same idea for LM machines. We exploit the geometric similarity between 2.5D NC milling and layered manufacturing.

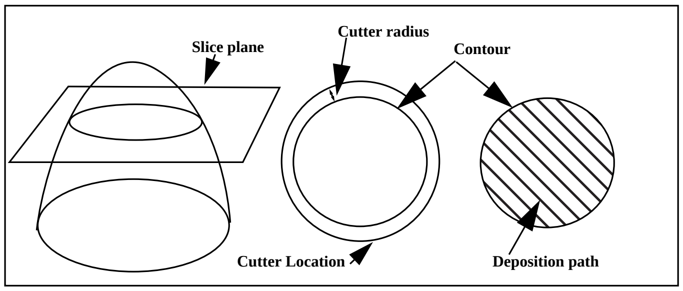

Figure 3. CLData and LMData

In principle, a part with no-undercut features can be NC milled in a way which is similar to layered manufacturing (shown in Figure 3). The CLData for this 2.5D milling is the cutter location data “around” the planar contour. In contrast, LMData includes the volume deposition path information. For a part without undercuts, cutter location data can be easily obtained from contour data (slice data) by an offset of a tool radius. Also, once contour data is known, it is not difficult to get LMData under certain deposition strategies.

3.2 The proposed data transfer architecture

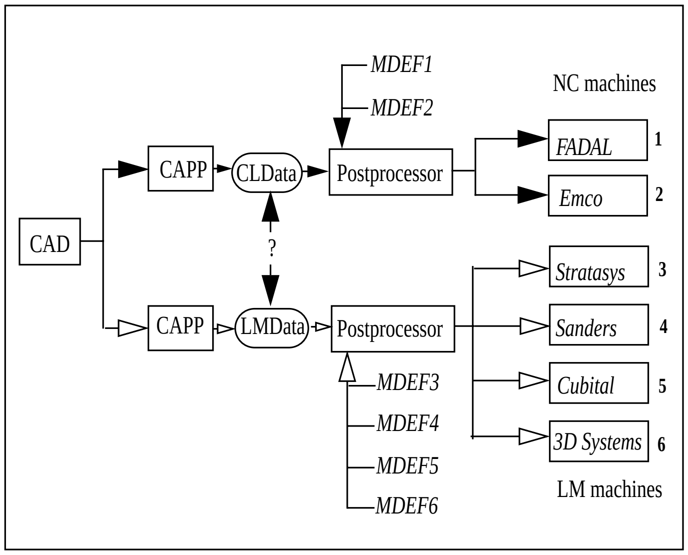

Figure 4. Integrated Data Transfer Architecture for NC/LM

Our main task is to standardize the input for each layered manufacturing hardware. We refer to this standardized input as LMData. It should be able to represent the output of process planning systems for layered manufacturing as CLData does for NC machining. In addition, a postprocessor is required along with the machine-dependent description file (MDEF) for each machine. We first presented this concept at the NIST Workshop on layered manufacturing (Dutta and Kumar, 1997).

Our plan for interoperability of data between LM system is represented in the architecture shown in Figure 4. For reference, Figure 4 also contains the “NC CLData”, a standard practice in industry/NC domain. Furthermore, it outlines our ongoing work (mentioned in Section 1) toward the seamless integration of LM and NC. The symbol ‘?’ on the link between CLData and LMData implies need for further investigation.

3.3 Advantages of this architecture

A distinguishing feature of this architecture is its clear module division, which clarifies the data transfer from design, process planning to final manufacturing. Process planning, identified as a separate module, and the capability of CLData/ LMData to fully represent the output of process planning, provides great convenience and flexibility for all parties (designers, process planners, and NC/LM hardware vendors). This clear module division is a key difference of our integrated standardization model when compared to the other sliced data based standardization approaches. The propriety data of each LM vendors will not be a limitation any more. LM vendors can focus on hardware development and the postprocessor which converts LMData into their own specific LM machine code. In this sense, LM hardware platform and software platform will be separate and both are open platforms. This architecture also liberates designers from the burden of determining (unfamiliar) fabrication processes and constraints. Another benefit of the architecture is that the operation of LM machines is simplified since process specific tasks are now decoupled and handled by the postprocessor and MDEF files.

In summary, the LM process planning system outputs LMData, which is postprocessed to drive the LM machines. The LMData can be directly downloaded to any LM machine of choice.

4.0 LM DATA WORKING MODEL

This section details the generation of LMData and also presents the content of LMData and its similarity with CLData.

4.1 Requirements for LMData

Based on the functional and neutral format requirements, LMData should have the following characteristics.

- It should be part-oriented rather than machine-oriented.
- It is designed to be a neutral LM tool path file.

It should not be specific to any particular process/machine. Actually all LM machine code is machine-oriented and process-oriented. This requires LMData be a good abstract of these LM machine codes.

- LMData should represent the output of process planning system, including specification of build material and support material.
- With a postprocessor plus LM machine definition file, LMData should be able to be converted to actual LM machine code.
- It would be good for the LMData to be verifiable.

Besides being able to be converted into real LM machine codes, LMData could assist in tasks such as

- simulation of LM building process.
- analysis of the surface finish, stiffness and strength of the part after it is built in this virtual LM machine.

4.2 LMData generation

For all LM processes, each layer’s fabrication can be thought of belonging to one of the two categories: (i) those in which the layer is created by incremental deposition along the layer filling path, such as in FDM, SLA, Sanders ModelMaker; (ii) those in which the layer is created as a whole such as in LOM, etc.(Marsan and Dutta, 1997). In this section, we focus on the first category. Its extension to the second category will be discussed later.

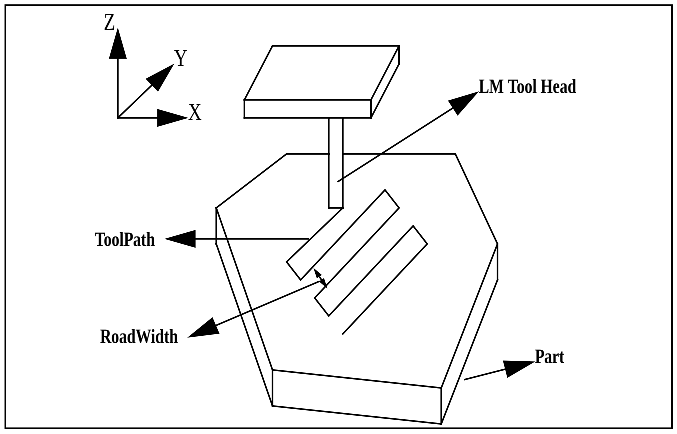

Figure 5. A Generic LM Machine for LMData

Following the layer forming principle of the first category, a generic LM machine is illustrated (in Figure 5) to generate the LMData. In this machine, the layer is formed by incremental deposition along a certain path in the layer. This LM machine has a tool head and can solidify/deposit material in certain thickness and road width.

This generic LM machine’s operations can be simply characterized by tool configuration and tool head movement, which are to be represented by LMData. After the process planning system determines part orientation, support, building time, fill pattern, etc., it will slice the model. Given the width of deposition material, the process planning system can output tool head path and necessary tool configuration commands for each slice according to the chosen deposition strategy (contour offset fill, raster fill or spiral fill, etc.).

Once the LMData is generated, the exact machine code to drive LM machine can be generated by a postprocessor plus a machine definition file. For each specific LM hardware like Stratasys or Sanders, a specific *.MDEF* file will be created, in which the real commands are defined.

4.3 LMData contents

To facilitate interoperability, LMData is designed similar to CLData. In fact we propose parallels to the CLData format as much as possible except the postprocessing commands. The following is a list of LMData commands that we have used.

- Display Commands(PAINT) (same as CLData).
- Tool Setup Commands (CONFIG index). Usually tool setup commands may include feed rate, rapid, material, flowrate, pause/delay, etc. Due to the machine-dependent feature, these commands are stored into a machine-dependent definition file (MDEF). With the index number, the postprocessor can identify the exact tool setup commands.
- Tool Motion Commands (GOTO, GODLTA, CIRCLE). Tool Head Motion command are the same as tool motion commands in NC machining.
- Macro Commands (CALL, MACRO, TERMAC) (same as CLData).
- Control Commands (INDEX, COPY, CYCLE, SYN (synonym)) (same as CLData).
- Comments and Continuations (same as CLData).
- MSYS(MCS) represents a matrix which relates the coordinates of the tool path to the absolute coordinate system. (same as CLData).
- Tool Path Header Command. In CLData, the contents of the tool path header statement differ based on operation type. This can be customized for LM use to include heterogeneous capabilities as well as including LM machines of second category (mentioned in Section 4.2).

As we can see from Table 1, CLData commands and LMData commands are very similar. Towards our long term goal of an integrated NC/LM environment, without much change, CLData and LMData can be represented by one format. This provides great convenience and potential to integrate the heterogeneous NC/LM machines into an integrated manufacturing environment.

Table 1. CLData and LMData commands

Commands	CLData versus LMData
Display Commands	same
Tool Setup Commands	different
Tool Motion Commands	same
Macro Commands	same
Control Commands	same
Comments and Continuations	same
Tool Path Header	different

5.0 POSTPROCESSING: CONVERSION OF LMDATA TO LM MACHINE CODE

After LMData is generated, a postprocessor will convert LMData to specific LM machine code.

5.1 LM machine code analysis

Every LM vendor creates its own LM machine code, like .sml file in StrataSys FDM, .bin file in Sander ModelMaker. To help design LMData and conversion of LMData to LM machine code, we first provide a brief summary of existing LM machine codes from Sanders and Stratasys. Due to their propriety nature, no detail description is given here.

For Sanders ModelMaker, the machine code file is .bin file (ModelWorks, 1996). It includes commands to dictate the inkjet's movement in X, Y, Z direction, commands to specify line type, and commands to describe the overall envelope of the part, and the commands to specify build material and support material. It also includes commands to drive the cutter to trim the surface and commands to temporarily suspend the mode building operation, etc.

For a Stratasys machine, the machine code file is .sml file (FDM, 1994). It includes the commands for tip head's movement in X, Y, Z direction, the commands to control the material flow rate, deposition speed, the commands to start and stop deposition, and commands for initial set-up and final stops.

Since in different LM machines, the part building principles can be quite different as described in Section 2, the machine codes are also quite different. However, these processes still share one common feature that all the models are built layer by layer. After a close analysis of the Stratasys and Sanders LM machine codes, all the commands can be classified into the following categories:

- movement in X, Y, Z direction, along which material is deposited or solidified by the tool head.

- deposition configuration, which specifies the flow rate of material, tool head speed, or droplet size. It is specific for each process.
- processing commands before or after tool movement, which are specific to each process. In FDM, it could be curve length breakup. In Sanders, it could include cutter's trimming of each layer surface.

Therefore, the postprocessor should generate this information from the LMData.

5.2 Postprocessing procedures for LM machine code generation

After a close comparison of the existing LM machine code structures, it is clear that LM machine code usually contains header description, commands for setup/teardown machine, commands before/after each layer's forming, and commands before/after each deposition tool head's movement and final ending marks. Hence, the following generic structured LM machine code, shown in Figure 6, is proposed to contain the differences among all kinds of LM machine codes.

```
Header
    Commands for setup machine
        Commands before each layer
            Commands before tool motion starts
                Commands of tool motion
            Commands after each tool motion
        Commands after each layer
    Commands for tear-down
Tail
```

Figure 6. Generic LM Machine Code Structure

Although the above LM machine code characterization leads to a generic LM machine code structure, the current LM machine code is still process-specific. The process-specific feature is contained in MDEF for each specific process.

Machine definition file (MDEF) is a file which contains all the machine-dependent commands. The machine code can be classified in terms of the above generic LM machine commands category and stored in the MDEF. It can then be loaded by postprocessor easily. The machine definition files that we have developed for Stratasys FDM and Sanders ModelMaker are given in the Appendix.

```
Input:    LMData file and machine definition file MDEF
Output: LM machine code which can drive the LM machine
Procedures:
        Read $MDEF_HEADER
        Read $MDEF_SETUP
        Scan every commands from LMData
            case X,Y movement:
                Read $MDEF_BEFORE_XY_MOVEMENT
                 output XY movement commands according to .mdef
                Read $MDEF_AFTER_XY_MOVEMENT
            case Z movement:
                Read $MDEF_BEFORE_Z_MOVEMENT
                 output Z movement command according to .mdef
                Read $MDEF_AFTER_Z_MOVEMENT
            case CONFIG Index:
                Read Index_th commands from $MDEF_start_deposition
            ...

        Read $MDEF_TEARDOWN
        Read $MDEF_Tail
```

Figure 7. Generic Postprocessing Procedures

Postprocessing is not a difficult task if the LMData is given and the MDEF is specified. It is described in Figure 7. Here, $MDEF_HEADER refers to the LM machine specific commands stored as Header commands in MDEF. Similarly, $MDEF-SETUP refers to the setup commands for a specific machine, which is stored as Setup commands in MDEF. The other categories of commands can be retrieved from MDEF similarly. Therefore, the actual LM machine code can be easily obtained from LMData and the MDEF.

6.0 IMPLEMENTATION AND EXAMPLES

6.1 Process planning system for layered manufacturing

As mentioned in Section 2, a separate process planning system for layered manufacturing is needed. We have developed several process planning modules in our research (Marsan, etc., 1997). Current modules include a solid build module, an orientation module, an adaptive slicing module, a deposition strategies module, etc. Slice data is generated after orientation optimization and support design. With this slice data plus specific machine deposition strategies, our process planning prototype system can output the LMData.

With a postprocessor and machine definition file for Stratasys, the LMData is converted to Stratasys machine code .sml file. This .sml file then is transferred to Stratasys machine to build the part according to the specified deposition strategies. The similar routines can be followed on Sanders. Currently our

LMData can only be converted to machine codes for Stratasys and Sanders Modelmaker, which are in our laboratory.

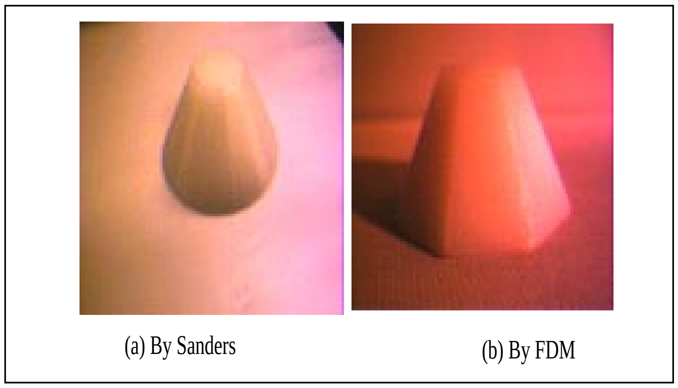

Figure 8. Experiment Part 1

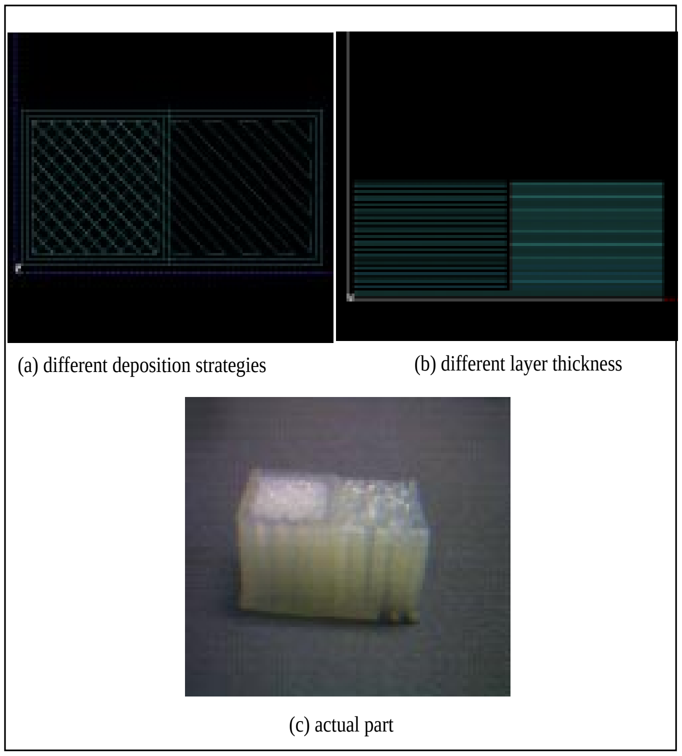
(a) different deposition strategies
(b) different layer thickness
(c) actual part

Figure 9. Experiment Part2

6.2 Examples

To validate our concepts, two experiments were conducted for conversion of LMData to machine code. One was for Sanders and the other was for Stratasys. A CAD model is first imported. The system asks for some fabrication information, like layer thickness range, deposition road width, deposition pattern, etc. After determining optimal orientation, adaptive slicing and support design, the system then outputs the LMData. The postprocessor designed specifically for Sanders and Stratasys reads the LMData plus specific machine definition files, then outputs the LM machine codes for Sanders and Stratasys. These codes then are sent to LM machines, which finally lead to the building of parts as shown in Figure 8, Figure 9.

The part in Figure 8.a was built on Sanders and the part in Figure 8.b was on Stratasys. LMData for the part and the machine definition files for Stratasys and Sanders are shown in Appendix.

The experiment, shown in Figure 9, was conducted to show the flexibility LMData provides. The part in Figure 9 had different deposition strategies in different areas and also had different layer thickness at different heights. The .sml file for this part can not be directly generated using QuickSlice. (We built this part on Stratasys by the conversion from LMData to .sml file.)

6.3 Discussion

As described earlier, LMData is supposed to be a machine-independent data format. However, the field of LM is new and evolving and different LM machines adopt different fabrication technologies. Therefore, some machine dependent configurations and commands are unavoidable at this stage.

Machine-dependent configuration refers to those machine specific configurations which affect the fabrication time, built part material property, etc. These configurations on one machine are different from those on others. For a specified deposition height and width, there could be more than one deposition configuration. For example, in FDM, it refers to different flow rates, speed, pause time, etc. In Sanders, different line types can be defined. The postprocessor doesn't know which line type to use. (Line type defines space between dots, print head movement speed, acceleration, cooling time, etc.) Therefore, it can only output one fixed line type. In our prototype system, CONFIG command is introduced to allow different configurations to be used to fabricate the part.

Machine-dependent commands can be customized by the postprocessor with machine definition file. However, there are some commands which demand specific operations due to the specific LM process. For example, In FDM, deposition has to be paused after certain deposition length even under the same deposition configuration. In Sanders, an overall part envelope is needed in the LM machine code file. This kind of machine-dependent commands can be processed by a dedicated postprocessor.

In principle, the same LMData should be able to drive different LM machines. However, in our experiments, there is no common range of the deposition height and width between FDM and Sanders. Fiber dimension of FDM is larger than the droplet size of Sanders. For material P301 and 0.016in nozzle head in FDM, the layer thickness range is 0.006in ~ 0.014in, and width is 0.016in ~ 0.05in. In Sanders, layer thickness ranges from 0.003in to 0.009in and width from 0.002in to 0.004in. This means that the minimum fiber size is larger than the maximum droplet size. Due to these constraints, in our examples, LMData sets for Sanders and FDM were generated separately. This is an issue that will have to be resolved in the future by LM vendors based on industry-wide standardization efforts (e.g., by NIST).

6.4 Limitation

According to the assumption of Figure 5, there are some other LM processes, like LOM, which are different from the assumed LM machine in terms of layer forming principle. LOM doesn't involve any volume deposition path. The boundary data in LOM is contour data, which is the same as slice data. For this reason, the LMData for LOM only contains the contour information. Therefore, using the LMData for LOM to drive other LM machines is a task that needs further investigation.

7.0 CONCLUSION

This paper addresses a critical problem in layered manufacturing----the lack of interoperability of layered manufacturing data. We presented an integrated standardization architecture in which LMData for layered manufacturing can be used by different LM systems (just as CLData is for NC machining). Experiments based on these ideas were conducted on the LM machines from Stratasys and Sanders. These experiments show that the proposed architecture is an open and effective method for use by designers, process planners, and LM machine vendors. Our ongoing work is focusing on further extensions to include other LM systems, as well as incorporating CNC machines as shown in the Figure 4.

ACKNOWLEDGEMENT

The authors gratefully acknowledge the financial support from ONR (N00014-97-1-0245) and NIST (70NANB 7H0021).

REFERENCES:

BRITE-EURIAM, Common Layer Interface (CLI), Version 1.31, *BRITE-EURAM Rapid Prototyping Techniques project*, Project No. BE5278, 1994.

Chang, Chao-Hwa and Melkanoff, Michel A., *NC Machine Programming and Software Design*, Prentice Hall, 1989.

Dutta, Debasish, Kumar, Vinod, "Towards Standardization of Data Transfer in Layered Manufacturing", NIST *WORKSHOP on Measurement and Standards Issues in Rapid Prototyping*, Oct16-17,97.

FDM System Documentation, Stratasys, 1994.

Jacobs, Paul F. *Rapid Prototyping and Manufacturing*, McGraw Hill, New York, 1992.

Kulkarni, Prashant, Dutta, Debasish, "On the Integration of Layered Manufacturing and Material Removal Processes", submitted to *Journal of Manufacturing Science*, Dec, 1997.

Kumar, Vinod and Dutta, Debasish, "An Assessment of Data Formats for Layered Manufacturing", *Advances in Engineering Software*, Vol.28, No.3, pp.151-164, Apr1997.

Marsan, Anne, Allen, S. W., Kulkarni P.,Kumar, V., and Dutta, D. "An Integrated Software System for Process Planning for Layered Manufacturing", *Proceedings of the Solid Freeform Fabrication Symposium 1997*, Austin, TX, Aug. 1997.

Marsan, Anne and Dutta, Debasish "Survey of Process Planning Techniques for Layered Manufacturing", P*roceedings of DETC'97, 1997 ASME Design Engineering Technical Conferences*, Sacramento, Sep 1997.

ModelWorks Users Manual, Sander Prototype, Inc., 1996.

Sequin, Carlo H. and Sara McMains, NSF Workshop on *Design Methodologies for Solid Freeform Fabrication*, "What can we learn from the VLSI Revolution?", June 5-6, 1995.

Weiss, L.E., Merz, R, Prinz, F.B., "Shape Deposition Manufacturing of Heterogeneous Structures", *Journal of Manufacturing System*, Vol. 16, No.4, pp.239-248, 1997.

APPENDIX

1. Machine definition file for Sanders ModelMaker

```
$mdef_head
ModelWorks V3.3 Bin driver     for SPI BIN 2.0 //header
$mdef_setup
$mdef_start_layer
$mdef_before_Z_movement
$mdef_after_Z_movement
PA 1 0
CU 0 0
$mdef_start_deposition
mdef1:
SP 1 0
LT 0 0
mdef2:
SP 2 0
LT 1 0
$mdef_deposition
PU PD
$mdef_end_deposition
$mdef_start_move
$mdef_move
$mdef_end_move
$mdef_end_layer
$mdef_teardown
$mdef_tail
EN 0 0
```

2. Machine definition file for Stratasys FDM

```
    $mdef_head
    ^[.K;;;;;^[.J;;;;;^[.R
    ^[.I;;17:
    ^[.N10;19:
    ^[.M10;;;;;:
    ^[.E;;;;;;;
    $mdef_setup
    IN;
    CD129;WA1;CD0; # alt-tip off, enable operator alt mat
change
    MZ1200;
    FH;XD210;FZ;
    MZ1200;
    MA1200,500;    # take this out if restarting a model
    MZ0;    # FOAM HEIGHT FACTOR HERE
    PS;WA0;VC102;OA;SO@102,@103;VC104;OZ;MR0,0;
    VS110,@102;VS111,@103;VS112,@104;
    V+110,@121;V+111,@122;
    CD133; #RB RNG
    AM0;
    CD129;WA1; # alt-tip off
    CD133; # NOT ABS
    $mdef_start_layer
    $mdef_before_Z_movement
    MZ204;
    MZ10,10;
    $mdef_after_Z_movement
    MZ-204;
    $mdef_start_deposition
    mdef1:
    PD.23,15;MM;MM0,15;MM6,20;MM-18,   181;   SR500;
WA0.; #ORD:2 Z:0.0100 S:0.01000
    AS1;VM4;BC;
    mdef2:
    PD.23,15;MM;MM0,15;MM6,20;MM-18,   181;   SR700;
WA0.; #ORD:3 Z:0.0100 S:0.01000
    AS1;VM4;BC;
    $mdef_deposition
    MA , MZ
    $mdef_end_deposition
    EC;VM3;
    XD209;#FC
    $mdef_start_move
    $mdef_move
    $mdef_end_move
    $mdef_end_layer
    $mdef_teardown
    VM7;
    MZ400;
    CD129;WA1;CD0; # alt-tip off
    XD211;
    $mdef_tail
```

Postprocessor can identify which deposition configuration to use according to CONFIG Index.

3. An abbreviated part of LMData for cone (shown in Figure 8).

```
LMDATA
LOAD/TOOL,2
RAPID
GODLTA/0,0,204
GOTO/317,317,0
GODLTA/0,0,20
GODLTA/0,0,-204
CONFIG/1
GOTO/317,317,0
GOTO/1016,27,0
GOTO/1716,317,0
GOTO/2005,1016,0
GOTO/1716,1716,0
GOTO/1016,2005,0
GOTO/317,1716,0
GOTO/27,1016,0
GOTO/317,317,0
RAPID
GOTO/356,224,0
GODLTA/0,0,204
GOTO/1688,322,0
GODLTA/0,0,-204
CONFIG/2
GOTO/1688,322,0
...
GOTO/757,1312,0
GOTO/720,1275,0
RAPID
GOTO/649,1204,0
GODLTA/0,0,400
END-OF-PATH
```