RAPID PROTOTYPING USING ROBOT WELDING - SLICING SYSTEM DEVELOPMENT

Fernando Ribeiro and Prof. John Norrish
Universidade do Minho University of Wollongong - Faculty of Engineering
Electronica Industrial Department Northfields Avenue
4800 Guimarães Wollongong NSW 2522
PORTUGAL AUSTRALIA

ABSTRACT

With direct deposition of metal a new Rapid Prototyping process had been developed at Cranfield University in the last couple of years. The process entails the use of a Gas Metal Arc fusion welding robot which deposits successive layers of metal in such way that it forms a 3D solid component. First, a solid model is drawn using a CAD system, then data indicating the kind of layers and dimension is incorporated and the solid is automatically sliced. This slicing routine also generates reports on the welding time and conditions for the production of the component and automatically generates the robot program.

The slicing routines used in traditional rapid prototyping processes did not cope with this welding deposition technique and therefore a new slicing process was developed specifically for this process. A few attempts had been made before getting to the final solution. All these attempts are described in this paper as well as the final solution.

INTRODUCTION TO RAPID PROTOTYPING USING FUSION WELDING

The current process was first described by Ribeiro in and therefore only a brief description of the process will be given here [1]. To reduce prototyping time, it has often been stated that there was a need to automate the production of ‘one off’ components for development and evaluation. Casting, which may be used for components of this type is an expensive and time consuming process especially if used to make only ONE component. Most of the rapid prototyping processes evolved in response to this requirement use resin based materials which are not always suitable for the purpose of testing. Metal based prototypes are often required and additional processing is necessary to convert resin based materials to a useful form.

In the process used here the component is formed by melting and depositing the metal using the GMA welding process. A CAD drawing system is used to create the initial solid shape and a welding robot is used to manipulate the welding torch.

After having drawn the component in a CAD, a slicing ‘add-on’ of the CAD program is implemented to produce the desired layers which will form the robot program. It is also necessary to enter additional data to indicate the bead geometry and the material used. The welding parameters are automatically generated by the program in order to achieve the required bead geometry and stable operating parameters. These parameters were derived from welding studies carried out by Norrish and parametric equations generated by Ogunbiyi and Norrish [2] [3]. The robot program is automatically generated and can be simulated with the use of a robot simulation program to check for collisions or other problems such as access. The robot program may be modified if necessary then compiled and downloaded to the robot.

CONCEPT OF SLICING

To build up a component using the gas metal arc welding process, the robot has to deposit several layers of metal on top of each other. The location of these layers have to be mathematically defined to allow the robot to follow them during the building process. These weld layers need to be drawn in a CAD program in order to visualise them. The weld deposited has some thickness and the path the robot has to follow is the centre line of the weld. The individual weld runs are considered to be ‘slices’ of the initial CAD solid.

The main objective of this paper is to explain the different stages or attempts to generate the slices. The slices are 2D mathematical entities like lines, arcs or polylines (a polyline is a group of one or more ARCS and/or LINES). These polylines represent the welding trajectory to build up the desired component or in other words the centre line of the weld bead. Following are described the attempts taken their problems and advantages/disadvantages.

MANUAL PROGRAMMING

The first attempt was to prove that it was possible to make a component using this metal deposition
technique, to prove that this system would work even if manually programmed. Therefore, the CAD package was used to draw the first component as LINES/ARCS instead of drawing it as a real solid. These LINES/ARCS represent the path that the robot would have to follow (welding trajectory) to make the component.

It was decided that a very simple shape would be used to test this technique. A 200 mm height ‘VASE’ as shown in Fig. 1 was chosen.

This shape was fairly easy to draw as a solid model but as a set of geometrical arcs it was more difficult. Firstly, it took a very long time to draw because every ARC has a different radius from the previous one and a different Z co-ordinate (if a simple cylinder was to be drawn this way, an array instruction could be used). This method of drawing was time consuming and repetitive and considered not to be suitable for the final implementation of the technique. It was however pursued in order to fully test the welding techniques.

After the shape (several ARCS) was drawn it was saved as a DXF file format and then imported into a robot simulation package (WorkSpace 3.3) to create the robot program.

The only import file format that the simulation package accepts is DXF. This simulation program was then used to create the robot program to follow these geometrical ARCS. The robot language used was ARLA (since the robot used was an ASEA IRb 2000). A robot simulation sample can be seen in Figure 2.

This was again a very time consuming task because the instruction 'FOLLOW POLYLINE' was repeated 'N' times where 'N' was the number of slices. For the VASE described and considering a layer height of around about 1 mm there were at least 200 slices.

Each time a FOLLOW POLYLINE command is to be used, the user has to select the polyline (with ARCS/LINES) by clicking on it with the mouse and then selecting the command on a sub-menu. Considering the VASE above described with his 200 arcs the image seen on the screen is very confusing due to having so many arcs on it and therefore it is very difficult to click on the correct ARC. The order in which the arcs are selected is also very important and has to be followed from the bottom to the top respectively.

Although very boring, repetitive, tiring and difficult to perform, this technique proved possible to draw this kind of component. The next task was to automate it.

**EXCEL SPREADSHEET INTERFACE**

As expected, the first components to be built using the manual approach needed several tests in order to perfect the welding parameters. These tests required some adjustments of the shape drawn most commonly in the bead height (or distance between ARCS). Because the shape was made up of around about 200 ARCS, when the layer height had to be changed this entailed redrawning the remaining 199 ARCS manually.

This task was so impractical that it forced an attempt to automate the adjustment of the co-ordinates and radius of the arcs, by using a spreadsheet. Several parameters were defined in cells which when edited would change the co-ordinates and radius of the arcs automatically. This technique facilitated the creation of the model but the shape was not stored in a CAD program which meant that the available CAD tools could not be used.

An advantage of using a spreadsheet however is that some useful values can be calculated from the cells. Arc Length can be extracted from the radius of each arc, total welding length can be calculated, timing for the operation can be calculated if another cell is created containing the welding speed, etc. A spreadsheet also allows the user to create a report which generates informative values relative to the component.

The problem with using a spreadsheet is that no complex shapes can be 'drawn'. Only regular or extruded shapes can practically be placed in a spreadsheet. Although being easy to use, this option limits the complexity of components to be programmed and therefore it is not a very good choice for production.

The spreadsheet proved to be a useful tool for developing the system but could not be accepted as a final option.

**C PROGRAMMING IMPLEMENTATION**

The spreadsheet option had another very important limitation which was that it did not create the robot program. There was no point in making the spreadsheet with all the desired co-ordinates if it could not generate the robot program automatically. The only way to overcome this problem was by creating a C Language program which could read the co-
ordinates and radius of the arcs from the spreadsheet and write them down as a text file in the form of a robot program. This was undertaken as the third attempt for automating the creation of slices and the robot program.

Since the user would have to run the C program, the use of the spreadsheet could be avoided and everything could be done in one program with a ‘user friendly’ menu driven interface. A C program was therefore written in such a way that the user only had to create a text file in which the required shape was designed and variables such as the welding speed, the position of the table in relation to the robot, the orientation of the welding torch and the height of the layers are specified. An example of a typical text file is shown below.

```plaintext
// extra variables (welding)
LAYER_HEIGHT  =  2.0
SPEED         = 350.0
ORIENTATION   = ORIENTATION1

// extra variables (ARLA program)
WELDING       = OFF
CREATE_REPORT = ON
USE_TABLE     = OFF
TABLE_INDEX   = ON

By using a text editor to write a text file which contains generic and pre-defined entities like CONES, CYLINDERS or CUBES, the use of a CAD program is avoided but it was not possible to see the shape of the component on the computer screen. It was easier to ‘type in’ the shape as text rather than creating cells and imagining the component but the user had to draw the component from memory and had to calculate the centre point of every shape as well as every dimension. The user was also limited by the number of different regular shapes which were just 4 or 5.

The first section of this file sample ([GLOBAL] section) allows the user to define some global variables like the location of the turn table in relation to the robot, co-ordinates of the centre of the table when the table is tilted, orientation of the welding torch, geometry of the bead (such as thickness and height) as well as Boolean variables to define if the user wants the welding instructions to be incorporated in the ARLA program or not, if the user wants to generate the BUILD UP report, etc.

In this example the first part built is a cone called CONEBFLN and the robot ARLA program is number 80. Then, the definition of the cone with values for its location, rotation in relation to the table, radius of the base, radius of the top, height of the cone, etc. If the thickness or height of the bead are to be different from those defined in the global section, these can be defined again in this section otherwise the global values are used. Boolean variables can also be used in this component section. Examples may be; if the user wants to use the table or the robot to make a circle or the table is to be indexed before starting each circle or if a build up report for the part is to be generated.

More than one entity can be written in this text file and this enables the user to create more complex parts.

When this file had been written, a post-processor program (written in C language by the author) was used to transform this text into one or more ARLA programs which would later be downloaded to the
robot to make the component. It would also create a report containing timing for the operation, welding length, etc. In other words it would produce the same results as the spreadsheet but with an application specially built for the process which should be easier to work with.

Another option which was considered was to create a small dedicated CAD program which would allow the user to draw simple shapes or solids. This program could then be expanded to accept more solids according to the needs of the system. This idea was rejected due to the complexity of creating a powerful enough CAD program to be able to generate solids. In addition, there are many CAD packages on the market and the author saw no need to ‘re-invent the wheel’.

**FINAL SLICING ROUTINE**

The final concept considered was the use of a well known and powerful CAD package which had its own programming language to manoeuvre its CAD data. For economic and portability reasons the CAD package chosen was AutoCAD from AutoDesk. It has its own language; AutoLisp, and this language is very powerful. In addition the whole package is user friendly, easy to learn, very compatible with most other CAD systems and has excellent data portability.

In the technique which was devised, the user draws the component in separate solid shapes, to simplify the process, and gives to each part of the component a sequential order number. This number represents the order in which the robot will build up the component. The only limitation in solid shape complexity is dictated by AutoCAD. When the user finishes the drawing, he then calls the add-on (within AutoCAD created by the author of this work) which comprises all the commands for automating the slicing relative to this rapid prototyping process. Most of these tasks will be automatic but the user still has to define some variables for example to inform the program which welding parameters to use, height of each layer, etc.

There are Global variables called PREFERENCES and also variables for each part of the component which have to be input although default values are always assumed. All the interface to input these values are AutoCAD like and the interface is very simple to use.

Once the user has defined all the variables, he has an option to slice the parts of the component. Once the slices are created it is then possible to generate the robot ARLA programs as well as reports with information relative to the build up. This build up plan can then be used during the construction of the component to check and control its production.

When the robot programs are generated these can be tested in a Robot simulation package to check integrity and to detect collisions. Then the complete program can be downloaded to the robot which is then ready to start building the component.

**TENTATIVE CONCLUSIONS**

This slicing process was deeply tested and used and several components were built. All these components were made completely automatic, from the 3D drawing up to the final component.

The main advantages of the slicing program used was that the slices were automatically created, the ARLA robot program was generated completely automatically and it was not essential to use a robot simulation package to test it, although simulation can be used to save on line time.

These slicing routines are independent of the robot used (or robot language). The ‘moving’ instructions and the welding start/stop instructions are parametrised which means that should any of the hardware change, the routines still work perfectly.

For these reasons, this process is a very flexible rapid prototyping system and has been very successful for all the components made so far. As bottom line, a few samples produced with these slicing routines and this rapid prototyping technique can be seen in Figure 3 and Figure 4.

**ACKNOWLEDGEMENTS**

Mr. Ribeiro would like to thank John Norrish for his support in this project and his expertise in the welding field and also Dr. McMaster for his academic help. My very special thanks go to John Savill for his strong support in welding all of the components and for solving in an efficient way the practical problems which arose as well as encouraging me to new ideas. A special thank you for my wife is also deserved for her understanding during the difficult periods of this project as well as for my all family for supporting all my decisions.

**REFERENCES**


Fig. 1 - Vase drawn as a series of geometrical arcs

Figure 2 - Robot Simulation of the Vase build up
Fig. 3 - 'Portuguese' Vase

Fig. 4 - Pint Glass

