International Journal of Shape Modeling, Vol. 2 No. 4 (1996) 215–226

A VIRTUAL FACTORY

ADRIAN BOWYER
School of Mechanical Engineering, University of Bath, Bath BA2 7AY, UK

GARY BAYLISS[a]
School of Mechanical Engineering, University of Bath, Bath BA2 7AY, UK

RICHARD TAYLOR[b]
School of Mathematical Sciences, University of Bath, Bath BA2 7AY, UK

and

PHILIP WILLIS
School of Mathematical Sciences, University of Bath, Bath BA2 7AY, UK

Received 9 August 1994
Revised 27 June 1996
Communicated by V. Savchenko and A. Pasko

This paper describes a virtual factory in which almost any manufacturing process can be defined, modelled, and then carried out. The results of this are designs of the components manufactured (in the form of geometric models) and part programs and process plans for their manufacture. Both of these can subsequently be re-loaded into the virtual factory and amended and enhanced, or—in the case of the part programs and process plans—can be downloaded into the real factory and used directly for production.

Keywords: Virtual reality, virtual manufacturing, virtual factory, process planning, CIPP, geometric modelling, solid modelling, concurrent engineering.

1. Introduction

People often work best when they feel that they are not really working at all. Common examples are when a designer is doodling ideas on a scrap of paper, or when a musical composer is improvising. Tools to help people to work are often most effective when they allow the work to become more like improvisation, and this is especially so when the tools take the drudgery out of the work and leave the person doing it free to be creative. This particularly applies to computational tools such

[a]Now at Radan Computational Ltd., Ensleigh House, Granville Road, Bath, BA1 9BE, UK
[b]Now at Canon Research Centre Europe Ltd., 20 Alan Turing Road, Surrey Research Park, Guildford, Surrey GU2 5YF, UK

as spreadsheets, word processors, CAD systems, systems for musical composition, systems for creating artwork and layouts, and so on.

This paper is about the University of Bath *Virtual Manufacturing System.* This system has been put together specifically to remove the drudgery from design and manufacture, and also has the added luxury of cost-free second thoughts. The system is a virtual world representing factories in which engineering components can be made and, almost as importantly, unmade. This is to say that time can be reversed to obliterate mistakes, with material reappearing to fill erroneous cavities unlike the way it so inconveniently does not in real life. Many people have produced simulation systems that will replay a pre-determined sequence of machining operations inside a computer (for a review see the report by one of us—Bayliss [1]), but we are not aware of any other fully-interactive system that is intended to be a source of such operations, and which is intended to be used by designers as a design system and by process planners for planning production. The closest system is perhaps Denford's Superscape [2], but this does not allow interactive modification of the production process; it will only replay simulations.

2. The Virtual Manufacturing System

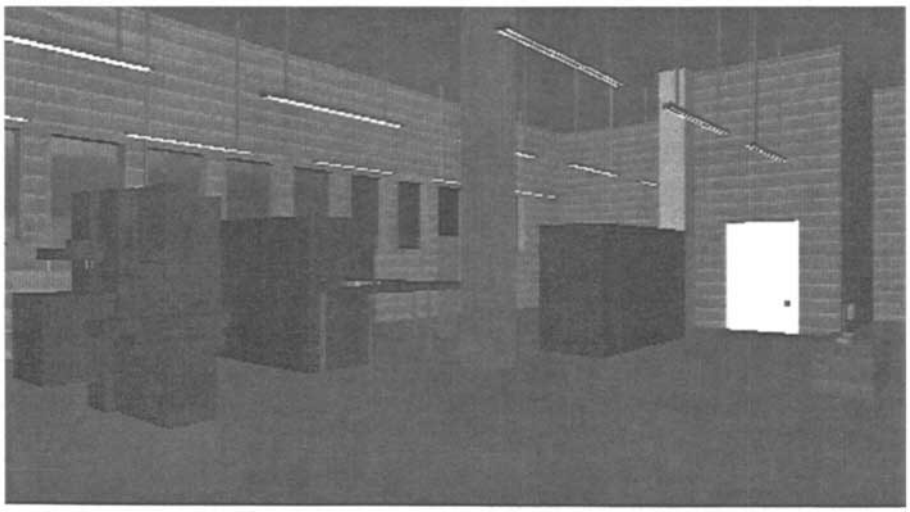

Figure 1: Part of a virtual factory.

The Virtual Manufacturing System records all the actions taken by those using it. This record can be rewound and edited to correct mistakes, as mentioned above, and then saved. The saved record can then be sent in the form of part programs to the numerically-controlled machine tools in the real factory that is modelled in the virtual one. The machines can then produce the real components automatically.

Figure 1 shows part of a factory modelled in the Virtual Manufacturing System.

It is, in fact, our own Manufacturing Laboratory at the University of Bath. On the left is a three-axis NC machining centre, and on the right is a small spray-painting robot (the spray screens have been omitted for clarity; they could of course be included if desired). Left of centre is a table with a number of workpieces and blanks placed upon it. The red pillar behind that is not part of the factory, but is a set eye position: by clicking on this the user can immediately move there and look at the machine nearby. In this case the machine is a small desktop mill (Figure 2). Other machines are behind the viewer.

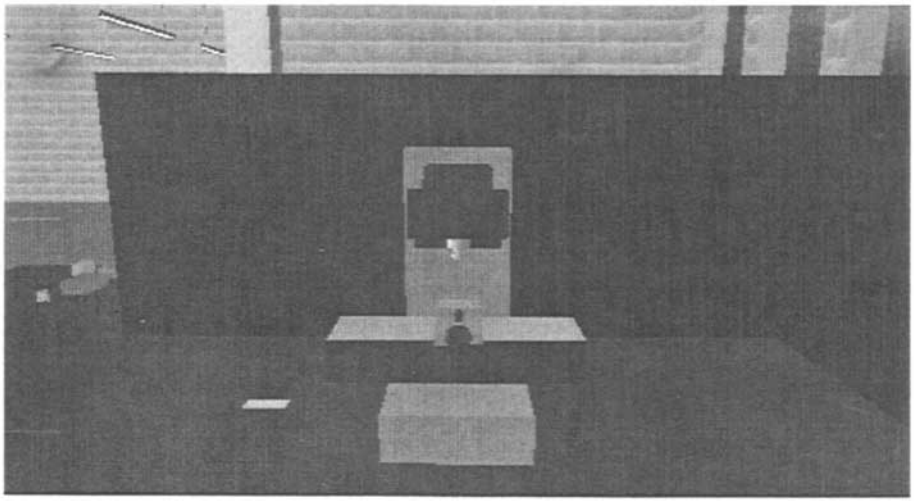

Figure 2: Desktop milling machine.

3. The system in use

Workpieces to be processed can be picked up with a virtual hand and placed in a vice, chuck, or fixture on the bed of a machine tool or other device such as a coordinate measuring machine. Alternatively, a robot can be moved to pick them up. At this point the geometric model representing the workpiece is appended to a tree of movement dependencies for the machine tool or robot so that it moves appropriately when the tool or robot do.

Suppose that the manufacturing process being used is cutting on the machining centre in Figure 1. Once the workpiece has been placed on the machining centre, the designer selects a cutting tool (from the carousel by clicking on its image, or by selecting it from a menu of tools in the machining centre's control panel—see below), and it is loaded into the machine. As the machine, the tool, and the workpiece all move, the relative movements between the tool and the workpiece are calculated and fed to a geometric modeller which, given the shape of the tool, is required to compute the swept volume of the movement, to subtract that from the model of the workpiece, and to provide the display with a faceted version of the resulting new

shape of the workpiece. Note that the underlying geometric model is not faceted—it is as accurate as the modeller's representation will allow.

Even though the model of the workpiece may become very complicated, the incremental changes to it that need to be made fast are usually both simple in shape and occupy but a small volume compared with the whole workpiece; this is very convenient for incremental re-faceting. The SVLIS modeller [3]—which is one of the modellers that we can use with the system—can take advantage of this spatial locality, and only re-facets in the small volumes where it detects that changes have actually taken place.

The virtual machining centre has a control panel associated with it, just like a real one. The designer using the system can load part-programs into the machine tool and execute them, or cause the machine tool to move and cut under manual control. At all times, the system maintains a complete representation of the currently-defined cutter paths, tool selections, and so on. As the designer works, this part-program record is automatically modified. Figure 3 shows a simple example of cutting and editing. This can be seen happening in the view of the factory, of course. But it is also possible to point at a workpiece and thereby to launch a window in which just it alone is depicted. This was done to generate the stills for Figure 3. These workpiece views make it much clearer to see detail of what is going on; also the workpiece can be rotated about any axis in this window so that it can be seen from any angle.

Similar procedures are followed for non-machining manufacturing processes modelled in the system.

In addition to being able to machine components in the Virtual Manufacturing System, its users can also check the components' dimensions using virtual metrological tools [4]. When a surface is pointed at using the mouse, a ray is traced into the model to find the actual surface hit. This surface then becomes a datum for measurements such as distances to other surfaces, diameters (in the case of cylinders), depths, heights, and so on. It is thus possible to check dimensions as manufacture progresses, and to set datums relative to which new machining operations can be carried out. As the cut in the part program which created each surface is recorded as an attribute of the swept volume that makes that part of the workpiece model, it is also easy to identify which cuts created which parts of the workpiece, then to go back and edit them to fix mistakes.

4. Design decisions

A number of significant design and software engineering decisions were taken during the course of the research project to create the Virtual Manufacturing System. This section lists the important decisions, and gives our reasons for taking them.

1. *Modeller independence.* The Virtual Manufacturing System represents machines, workpieces, and the whole of the rest of the virtual factory using geometric models. It thus obviously requires a kernel geometric modeller to

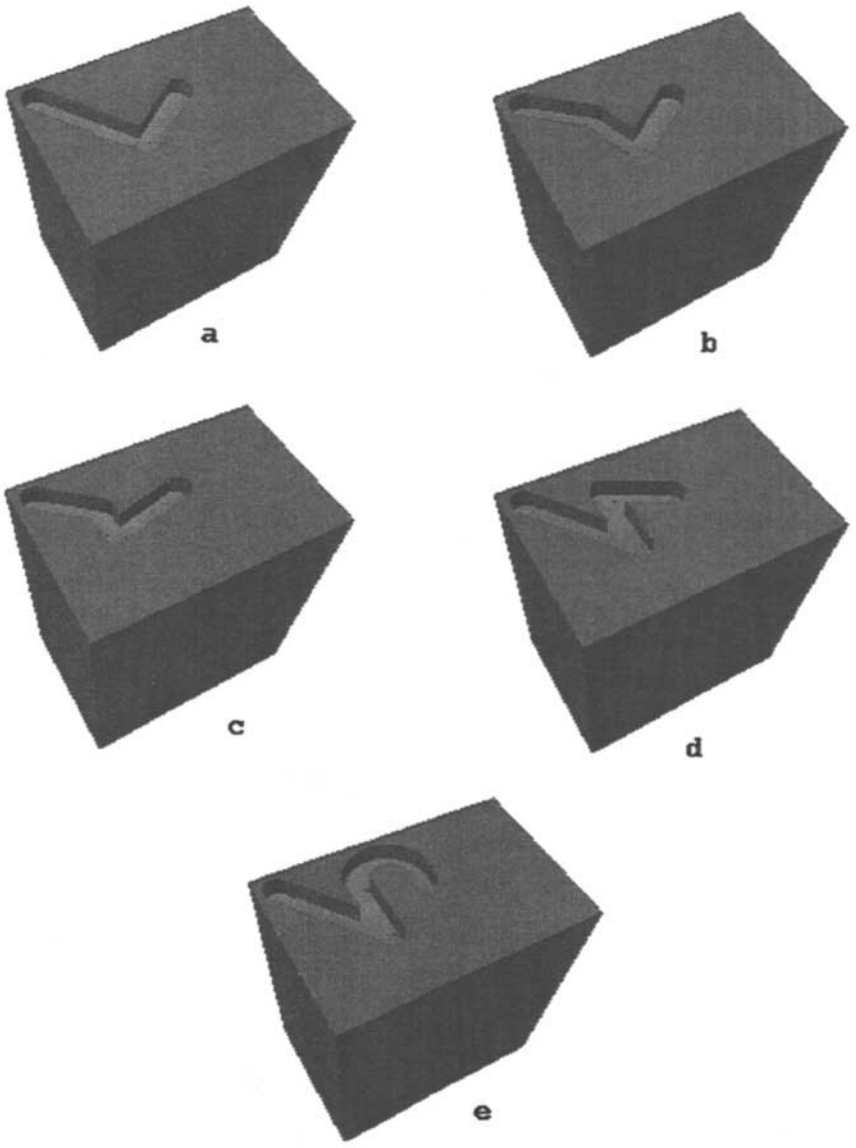

Figure 3: A workpiece being cut by the machining centre from Figure 1. Two cuts are made (a); a third is inserted (b); the position of the end of two of the cuts is changed (c); a cut is deleted and another inserted (d); the final cut is changed from straight to circular (e).

which it can make procedure calls. The interface to the modeller was deliberately kept short (it is about 500 lines of code), and all communication with the modeller is handled by that interface. In fact, the modeller interface antici-

pates the Djinn API project [5], which is an attempt to specify a call interface to geometric modellers that is completely independent of the representation of solids that those modellers use. This makes it relatively straightforward to change the modeller which the Virtual Manufacturing System is employing, allowing it easily to be utilized in an industrial organization which already has geometric models of products, fixtures and tooling. To demonstrate modeller independence, the Virtual Manufacturing System is currently working with two completely different modellers: the ACIS b-rep modeller [6], and the SVLIS CSG modeller [3].

2. *Graphics.* The Virtual Manufacturing System uses the OpenInventor rendering system [7]. This was originally a Silicon Graphics product, but it is now available on Sun, Hewlett Packard and many other Unix workstations, and also on PCs. The Virtual Manufacturing System is thus not tied to any one type of computer or any one graphics hardware system. The Virtual Manufacturing System uses texturing on all graphics (though this can be switched off to improve response when the System is running on a machine that does not support hardware texturing). We have found that texturing gives users important cues about scale and distance, and thus helps them to use the System efficiently and accurately.

3. *Implementation language.* We decided to use C++. This gives good compatibility with the OpenInventor graphics system and both the ACIS and SVLIS modellers. Also, the modular object-oriented approach is particularly well-suited to the Virtual Manufacturing System, with its large number of separate but similar components.

4. *Immersive or desktop VR.* The problem with immersive VR systems is that they are very tiring to use for extended periods of time and have poor visual resolution, as Stone says in his 1993 paper [8]. This makes them unsuitable (at present) for a system that may need to be used for hours at a time and in which subtle graphical cues are important. For this reason desktop VR was chosen for the Virtual Manufacturing System, though this choice is not final: it would be a simple matter to change the input mechanism from a mouse to a data glove, and the display system (which already offers a stereo option via LCD spectacles) would remain unaltered. In fact a mouse is not a particularly useful way of interacting with machines such as the five-degrees-of-freedom robot spray-painter in Figure 4, so even in the present system we also have a small dummy robot arm with potentiometers at the joints which is used to drive this in a way that is very similar to a teach-session for a real spray robot.

5. *Computer power.* Perhaps surprisingly, the need for real-time moving graphics is not the heaviest load on the computer in the Virtual Manufacturing System. This is because all practical use of it is on systems with Z-buffer rendering

built into hardware—we use a Silicon Graphics Onyx Reality Engine II. The principal computing load comes from the need to modify and to evaluate geometric models of changing workpieces in real time. The Onyx used by the system has twin 70MHz processors and is just fast enough to do this with the two geometric modellers that we currently use. Less efficient modellers, or slower processors, would make the system unresponsive and thus difficult and frustrating to work with.

6. *Representations.* Representations are covered in more detail in the next section. However, to give an overview, all aspects of the virtual factory are specified either by simple easily-edited text files or by geometric models in whatever file format the modeller being used by the system employs. This makes the system fully user-configurable. To specify a new factory would take the user about a day for every ten machines that it contained ***plus*** the time needed to create geometric models of them. This latter time obviously depends upon the facilities provided by the modeller. As an example, with SVLIS the machining centre took an experienced user about three hours to create.

7. *Documentation.* All the documentation for the Virtual Manufacturing System is written in HTML [9]. When the user of the system clicks on the HELP button, a standard browser—Netscape (see [10]), though any other one would work equally well—is forked with its home page set to the root of the system's documentation. This allows:

   (a) The documentation to be in a standard and easily-extended form;

   (b) Straightforward hypertext links and included graphics;

   (c) There to be just one set of documents for the system on the Web for use all over the world[c].

5. Representations

In the Virtual Manufacturing System, machine tools, robots, and other machines are represented by geometric models of their moving parts arranged in a tree to represent movement dependencies. Thus the machining centre on the left in Figure 1 consists of a model of its base and rear column which is located immovably on the machine shop floor. The bed is made up from two models: one for the y movement (front/back), the value of y being related to the base; and one for the x movement (left/right), the value of x being related to the y-bed. The chuck of the machine which holds the cutting tool can move in the z direction (up/down), and z is again related to the base. The limits on the values of x, y, and z are recorded in the

[c]Though for commercial reasons we have not yet put these documents on the Web site for the Virtual Manufacturing System.

structure as well, of course, as is a default initial value for each. The machining centre also has its carousel holding cutting tools.

What follows is the text-file description of the machining centre in Figure 1.

```
# list of machine axes - name, linear?, range, quantum, initial value

Axis: X T [-1.200 1.200] 0.00001  0.0
Axis: Y T [-1.000 0.200] 0.00001  0.0
Axis: Z T [-0.750 1.000] 0.00001  1.0

# list of control points - name, starting_place
# first one is where tool gets attached

Point: tool_end (0 0 1)
Point: bedX     (0 0 0)
Point: sliderY  (0 0 0)
Point: carousel (-1 0 0)

# list of machine parts - name, filename, offset

Part: base     Machines/matchBase.pt     (0 0 0)
Part: carousel Machines/matchCarousel.pt (-0.3 0.8 1.3)
Part: bed      Machines/matchBed.pt      (0 0 0)
Part: slider   Machines/matchSlider.pt   (0 0 0)
Part: head     Machines/matchTool.pt     (0 0 0)
Part: chuck    Machines/matchChuck.pt    (0 0 1.1)

# list of movements - axis, point, motion

Move: X bedX     t(-1 0 0)
Move: Y sliderY  t(0 -1 0)
Move: Z tool_end t(0 0 1)

# list of tools - name, filename, offset

# centre of carousel is -0.48,0.80 and radius is 0.25

CutterTool: tool1 Toolpieces/tool1.tp (-0.38 0.63 1.2) (1 0 0 0)
CutterTool: tool2 Toolpieces/tool2.tp (-0.43 0.61 1.2) (1 0 0 0)
CutterTool: tool3 Toolpieces/tool3.tp (-0.48 0.60 1.2) (1 0 0 0)
CutterTool: tool4 Toolpieces/tool4.tp (-0.53 0.61 1.2) (1 0 0 0)
CutterTool: tool5 Toolpieces/tool5.tp (-0.58 0.63 1.2) (1 0 0 0)
CutterTool: tool6 Toolpieces/tool6.tp (-0.62 0.66 1.2) (1 0 0 0)
CutterTool: tool7 Toolpieces/tool7.tp (-0.65 0.70 1.2) (1 0 0 0)
```

```
CutterTool: tool8 Toolpieces/tool8.tp (-0.67 0.75 1.2) (1 0 0 0)

# list of initial attachments - part/tool, point

Attach: head   tool_end
Attach: chuck  tool_end
Attach: bed    bedX
Attach: bed    sliderY
Attach: slider sliderY

# end of specifications
```

The machine parts list contains pointers to the geometric models that define the shapes, colours, and textures of the individual components of the machining centre, and the tool list contains similar information about the tooling. Apart from them, the above information is all that is needed completely to define the machine. There are similar specification files from the whole virtual factory at the top level, down to individual workpieces and furniture at the bottom.

The user can specify any tool and process for the Virtual Manufacturing System in this way, as long as the actions of the tool on the geometric models of the workpieces can be defined. Broadly, this covers all non-random manufacturing processes, but tends to exclude things like shotblasting, which are hard to represent deterministically (though even there a statistical approach, like the Gaussian jet of the paint spray gun being wielded by the robot in Figure 4, could be used).

The component parts of a machine can move, but none ever changes its shape (unless a cutting tool collides with part of the machine that is using it...). As was mentioned in the previous section, the display hardware which we are using for the Virtual Manufacturing System is a texture-mapped depth buffer (at present a Silicon Graphics Onyx) which requires the objects being depicted to be represented as polygons. These polygons can be pre-computed for the whole machine tool as none ever changes—they only move about. The solid modeller is therefore required only to facet each model for each part of a machine tool once, which reduces the load on the modelling software considerably. This task could even be done off-line, though this is not really necessary; in fact one of the modellers that we are currently using (SVLIS [3]) can facet a model more quickly than pre-computed facets can be read from the disc.

Workpieces are also represented by solid models, but, as they are machined, they do change shape. As the aim of the Virtual Manufacturing System is to allow machining operations to be performed on workpieces in real (or faster than real) time, this places a serious load on the geometric modeller being used to represent the workpieces. It has to be able to re-facet the models of workpieces as holes are carved in them faster than the frame refresh rate being used for the display. As was mentioned in the previous section, this task is currently just within the limits

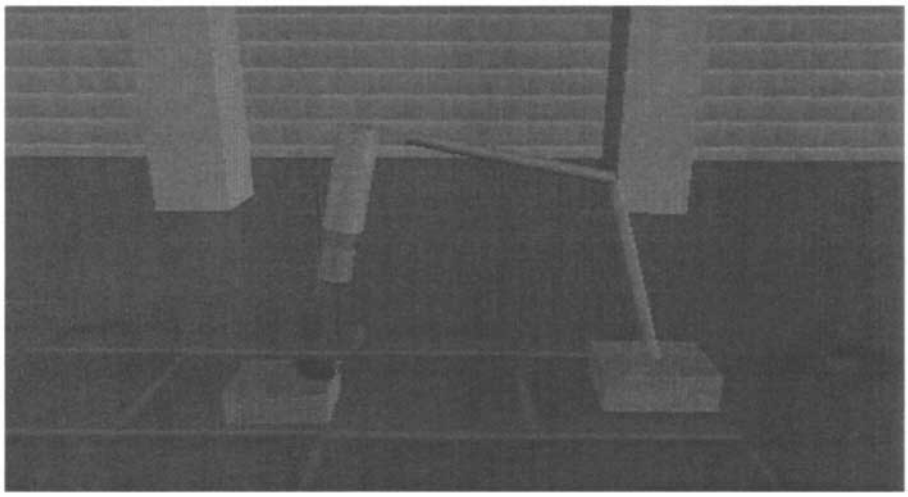

Figure 4: The small paint-spraying robot on the right in Figure 1 in the process of spraying a workpiece with black paint.

of the technology. However, not all manufacturing processes change the shape of workpieces. For example, the robot spray painting (Figure 4) only changes the colour of the surface of the component, and the planning of coordinate measuring machine inspection cycles using the Virtual Manufacturing System does not affect workpieces at all.

This distinction between objects in a virtual world that change their shape and ones that merely move about is an important one. Indeed, many commercial VR systems only support the latter sort of object. This is clearly unacceptable for any system that is to model manufacturing, and (as might be expected) VR system vendors are often rather coy about the limitation. As far as we have been able to ascertain our Virtual Manufacturing System was the first[d] virtual world to allow real-time shape modification.

Visual realism was one of our main aims. A high degree of realism is achieved (with minimum performance cost) by taking advantage of hardware real-time texture mapping as much as possible. Typically, decorative objects (such as the factory walls) are modelled very simply and made to look real by full-colour texture mapping[e]—whereas critical objects like the actual workpieces are modelled very accurately and only use simple texturing to indicate things like surface finish.

For flexibility, we begin with a completely general factory and add objects specified in a configuration file. This file is then 'edited' by the designer through his or her interactions with the virtual factory. Since each object (for example, a work-

[d]The initial version was working in 1993.

[e]Though the exterior scenery outside the windows is not just painted on; it is a large texture map at a great distance which gives the correct perspective effects as one moves round the interior of the factory.

piece, fixture, or machine tool) is also specified by a configuration file, objects can be freely shared amongst designers—much as files can be shared in a UNIX file-system.

By making the virtual factory as dynamic as possible, we have enabled designers to design new machines and processes as well as to produce components using existing techniques.

6. Conclusions

The great potential of this virtual factory is that it provides a safe environment in which designers can experiment with different manufacturing techniques. Experimentation is encouraged by equipping the virtual factory with mechanisms which circumvent some of the more annoying features of a real factory, such as the inability to run backwards in time. Therefore, rather than being just a *simulation* of a real factory, the virtual factory can provide much more by effectively integrating design tools and manufacturing processes.

To gain the benefits of a designer's familiarity with the real factory, the virtual factory must be visually realistic. In addition, we require sufficiently accurate geometric models of objects in the factory, since it would be pointless to create a component on a virtual machine only to discover later that the resultant NC program cut into a piece of the real machine that was modelled too crudely.

Designers do not design in real time but manufacturing does occur in real time. It is therefore necessary for a design *by* manufacture system to be able to relax and to tighten the applied constraints as required by the designer. Additionally, multiple levels of constraints may be applied in different circumstances; for example there are several possible ways of dealing with feed-rates in the Virtual Manufacturing System:

1. No constraint (pure design, don't care about feed rates);
2. Feedback constraint (design with manufacturing in mind) use colour, sound, and labels to indicate physical quantities such as rate of metal removal;
3. Full constraint (manufacturing conditions) don't allow constraints to be broken.

The first stage of development of the Virtual Manufacturing System is now complete, but from the outset it was planned to be user-extendible. There is no non-random manufacturing process which could not, in principle, be modeled within it, and no design activity that could not have its results virtually realized within it. By designing by making, the whole of part-programming and process-planning is taken care of implicitly, and the time taken to get a design into production is much reduced. In addition, the ability of all such computer-based editing procedures to allow variants to be easily made is inherited by our system; once a design is working it can be improved at will and, equally importantly, with little expense and effort.

Throughout this paper we have deliberately referred to the user of the Virtual Manufacturing System as a designer. This is because concurrent engineering is

inherent in the system—components and assemblies are designed as they are virtually made, and it is not possible to create anything within the system without also devising a way to manufacture that thing.

Acknowledgements

We would like to thank the Design and Integrated Production Group of the United Kingdom's Engineering and Physical Sciences Research Council [11] for awarding the grant which supported the project described here.

References

1. G.M. Bayliss, *Real-time geometric modelling for manufacture*, University of Bath School of Mechanical Engineering Technical Report 001/1994.
2. Denford Machine Tools Ltd *Superscape VR CIM System*, Denford Machine Tools Ltd, Birds Royd, Brighouse, West Yorkshire HD6 1NB, UK.
3. A. Bowyer, SVLIS - *Introduction and User Manual*, ISBN 1-874728-06-2, 116pp, Information Geometers Ltd., Winchester, UK, 1994; http://www.bath.ac.uk/~ensab/G_mod/Svlis/svlis.html
4. A.F. Wallis, *Toolpath verification using set-theoretic solid modelling*, PhD thesis, University of Bath, 1991.
5. A. Bowyer, S.A. Cameron, G. Jared, R. Martin, A.E. Middleditch, M.A. Sabin, and J.R. Woodwark, *Introducing* Djinn - *A geometric interface for solid modelling*, ISBN 1-874728-08-9, 24pp, Information Geometers Ltd., Winchester, UK, 1995; http://www.bath.ac.uk/~ensab/GMS/Djinn/djinn.html
6. I.C. Braid, *Improving product models and kernel modellers*, Organization of Engineering Knowledge for Product Modelling in Computer-Integrated Manufacturing, ed. T. Sata, pp 275-299, Elsevier, 1989; http://www.csn.net/spatial/
7. J. Wernecke, *The Inventor Mentor*, ISBN 0-201-62495-8, 514pp, Addison Wesley, 1994.
8. R.V. Stone, *Reality - Who needs it?* IEE Review, V 39, No. 6, November 1993 pp 243-246
9. Hypertext Mark-up Language, http://www.w3.org/hypertext/WWW/MarkUp/MarkUp.html
10. Netscape, http://www.netscape.com/
11. Engineering and Physical Sciences Research Council, http://www.epsrc.ac.uk