Constructing and managing complex virtual worlds for manufacturing applications

Zhengxu Zhao

School of Engineering
University of Derby, Kedleston Road, Derby DE22 1GB, UK

Tel: 01332-622222 ext.1177
Fax: 01332-294861
e-mail: Z.Zhao@derby.ac.uk

Abstract: Virtual reality is becoming a real business-oriented technology and holds great potentials especially in computerised manufacturing applications. Technical problems however have to be resolved before it can be practically employed. This paper reports the research into one of the crucial and challenging problems -- the pain-taking and tedious process of creating and managing virtual manufacturing environments. The paper presents a method of managing virtual worlds, attempting to provide an effective approach to building complex virtual environments for manufacturing with less effort and shorter time. First the paper presents the concept behind the approach. Then it describes the computing algorithms for implementing the approach and presents a system of managing the virtual environments. In the end the paper discusses the applications of the approach with an emphasis on manufacturing operation planning.

Keywords: virtual reality, virtual world, manufacturing environment, variant approach

1. Introduction

Virtual reality (VR) is an emerging technology with potential manufacturing applications in areas such as product design and modelling, process simulation, manufacturing planning, testing and verification, real-time shop floor controls, training and maintenance (Blanchard, 1995, Boman, 1995). One speculation on VR's development in manufacturing industry is a computing architecture that could provide virtual production environments within concurrent engineering context to achieve zero-defect and non-risk production (Sherard and Narayanan**,** 1994, Kreitler *et al*, 1995). In fact, VR has already been applied to a wide range of problems associated with manufacturing, industrial maintenance, post-production training and customer services in the areas such as visualisation of complex data, robot control and remote operation of equipment (Neugebauer, 1992), communication, training and planning (Carpenter *et al*, 1996), and virtual prototyping and design -- to name a few. The success of those applications has mostly relied on a realistic virtual environment.

For manufacturing in particular, a virtual environment is often a complex digital world made of a large number of different 3D computerised models. Anyone who has spent time wrestling with the task of mastering a conventional CAD software knows that even building a 3D graphic representation of a single object can only be made easy up to a point. To create a complex manufacturing environment, even the design and planning process at the beginning still takes enormous thought and effort -- not mentioning the actual construction process. While creating a new environment is a tedious and pain-taking process, deriving the environment also lacks of the support of an efficient management of existing virtual worlds. Relating to this problem, research is being conducted in IMRC centre at University of Derby to investigate the effective methodology of constructing and managing virtual manufacturing environments based on the commercial VR system Superscape VRT (Superscape VR plc. UK., 1996). This paper reports the research findings with the emphasis on the virtual world management. The paper presents the method of classifying virtual manufacturing environments and shows how it can be implemented to manage the environments and to simplify the construction process.

2. Variant approach to constructing virtual manufacturing environments

2.1 The concept

The concept of the variant approach can be described as this. Suppose virtual manufacturing environments are created for manufacturing tasks, both allowing themselves to be described. The environments are called master environments and the tasks are master tasks. To accomplish a specific master task, there should exist at least one master environment that is related to the master task in the data base. Therefore, to perform a given task (for instance, to manufacture a mechanical part), the mast environment is retrieved from the data base based on the similarity between the given task and the master task. If the given task is the same as the master task, then the retrieved mast environment is the environment for the given task. If the task is similar to the master task, then the retrieved mast environment may only need to be modified into the required environment, thus avoiding the creation of the environment from the scratch. The amount of the modification depends on the similarity between the given task and the master task. The more similarity there is, the less modification is needed. Only in the worst case where the given task bears little similarity to any one of the master tasks in the data base and no environment can be retrieved, does a new environment needs to be created. This newly created environment and the given task are then related with each other and added to the data base to become a new set of master task and master environment.

2.2 The computing architecture

To create a virtual manufacturing environment based on the above variant concept, a VR based system was developed whose architecture is shown in Figure 1. The system consists of five basic computing modules (1) the dedicated data base, (2) the task description and classification programme, (3) the Superscape VRT and SDK system, (4) the peripheral interfaces and (5) the virtual environment manager.

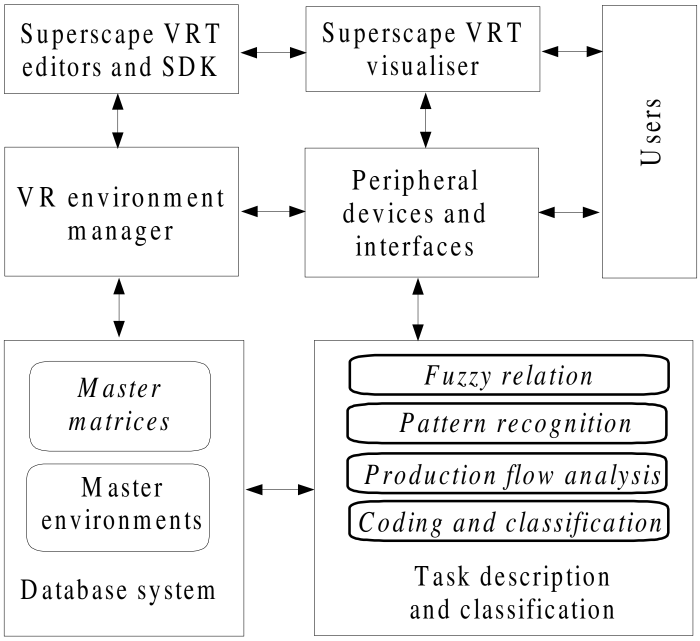

Figure 1. General structure of the variant VR system

The data base is divided into four sectors, each supporting different applications and accordingly being associated with four parallel task description and classification modules. Those modules are implemented on four different clustering techniques to accommodate different manufacturing applications (see Section 4). The DBMS of the data base is a dedicated programme integrated with those modules and the Superscape VRT system.

The task description and classification modules describe a number of selected tasks in codes and cluster them into task families. Each family is represented by a master task and defined within the data base in a format called master matrix. The master matrix is attached with at least one master environment together with the related manufacturing information including manufacturing methods, machine and tool specifications, operations instructions and shop floor control data.

The Superscape VRT editors and visualiser are from Superscape VRT5 system provided by Superscape VR plc. Superscape VRT5 provides seven types of editors that can be used to

create a master environment and the visualiser is a VR animation engine for animating the manufacturing reality within the environment.
The peripheral interfaces are control protocols, mostly provided by Superscape VRT5. They support the peripheral hardware devices such as head mounted display (HMD), electric gloves, 3D-mice and tracking devices in an almost plug-and-play working mode. Other interfaces also provided are for data transfer and data exchange. For example, the Data Converter is such an interface programme to transfer AutoCAD's DXF data into 3D VR data. The interfaces that were developed for the system are links between the VR environments to real manufacturing facilities with numerical controlled devices, for instance, the interface that enables the data to communicate between a virtual robot and its reality counterpart for remote robot mission control. Since Superscape SDK has provided Superscape Control Language (SCL) utility which is mostly compatible with common C/C++ code, interfaces between the Superscape's editors and the other programmes in shown Figure 1 were also developed in a similar style.

The virtual environment manager is a programme whose main function is to co-ordinate the other programme modules within the systems and to manage the communication between the users and the system (see Section 5).

2.3 The implementation methodology

The implementation of the system shown in Figure 1 takes six stages: (1) designing the task description system and describing tasks, (2) forming task families and defining master tasks and master matrices, (3) creating the master environments for each task family, (4) setting up the data base and store the master families and master environments, (5) designing the virtual environment manager and (6) designing peripheral tools and control interfaces. The sections to follow will only explain stages (1), (2) and (5).

3. Designing the task description system

The task description system was programmed (1) for system developers to describe tasks when tasks are to be clustered and (1) for users to describe a given task when virtual environments are to be retrieved. The system describes tasks in GT-like code (British Institute of Management, 1971, Burbidge, 1971) with on a generic internal representation of the coding methods. It is thus able to describe tasks based on any coding schemes (both well-established industrial coding standards and user specified coding schemes).

A coding scheme or coding standard specifies the number of digits of the code, the structure of the code, the information represented by the code. The coding programme can work on different coding schemes simply by representing different coding schemes into a generic text file format. The file consists of three sections, each being made of several blocks. Below is an example of the file, representing a five-digit coding scheme that describes mechanical parts.

```
// ----- First section: Header (comments)
#TASKCODE;
#digits 5 12;              //Number of digits and maximum number of code values
#points 10 10 120 120      //Position of the graphic display
#pen&brush 1 0 100         //Painting context: pen width, pen colour and brush colour
```

```
// ----- Second section: For digit values
d1 n6 d1c0 d1c1 d1c2 d1c3 d1c4 d1c5;
d2 n12 d2c0 d2c1 d2c2 d2c3 d2c4 d2c5 d2c6 d2c7 d2c8 d2c9 d2ca d2cb;
d3 n4 d3c0 d3c1 d3c2 d3c3; d4 n5 d4c0 d4c1 d4c2 d4c3 d4c4; d5 n2 d5c0 d5c1;
// ----- Third Section: For description of digits
d1c0 Diameter < 10 mm;   d1c1 Diameter < 100 mm; d1c2 Diameter <500 mm;
d1c3 Diameter <800 mm;   d1c4 Diameter <1400 mm;d1c5 Diameter >=1400 mm;
d2c0 Length < 40mm;              d2c1 Length < 80mm;              d2c2 Length < 140mm;
d2c3 Length < 180mm;     d2c4 Length < 200mm;     d2c5 Length < 250mm;
d2c6 Length < 500mm;     d2c7 Length < 800mm;     d2c8 Length < 1000mm;
d2c9 Length < 1300mm;    d2ca Length < 2300mm;    d2cb Length < 2900mm;
d3c0 Length/Diameter (L/D) <= 1;  d3c1 Length/Diameter (L/D) <= 15;
d3c2 Length/Diameter (L/D) <= 50;          d3c3 Length/Diameter (L/D) >50;
d4c0 Without feature presence;      d4c1 Step shaft d/D < 0.8;
d4c2 Step shaft d/D >= 0.8;                d4c3 Pocket presence;      d4c4 Hole presence;
d5c0 Metal materials;                      d5c1 Non-metal matetrials;
```

The file starts with the header section where each block is started with a key word preceded by a pound symbol ***#***. The next section defines the code digits. The first word of each block, with a letter ***d*** followed by a number, defines the serial number of a digit. The second word, with a letter ***n*** followed by a number, defines the number of code values. In the last section each block describes the meaning of each digit value with a string of text.

The programme works like this. It first loads a coding scheme text file and graphically displays it on the screen. Users interact with the programme through the graphic coding tools and interpret tasks to the programme. The code is generated through the interrogation to the coding scheme simply by clicking on the relevant code digit icons. The programme has been integrated with the data base and can be used as a data base utility tool.

4. Forming the task families and the master tasks

After the tasks are coded, task families are formed automatically. Because a task may need to be coded differently for different virtual environments (for instance, different manufacturing systems or manufacturing companies), task families should be formed accordingly with different clustering techniques, which makes the system as shown in Figure 1 have four clustering modules. To limit the length of the text, only the fuzzy relation clustering technique is explained. The rest are based on the same idea.

The fuzzy relation clustering (Lou *et al* 1983) is a synthetical technique. It describes tasks with a number of given attributes. Task families are formed using the fuzzy relationship of the attribute values. Suppose there are ten parts $\{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$ to be manufactured, each being described by five attributes $\{A_1, A_2, A_3, A_4, A_5\}$, where A_1 for geometry complexity, A_2 for tolerance level, A_3 for machinability, A_4 for the flexibility of the required manufacturing equipment and A_5 for manufacturing cost. Each attribute is evaluated by a value between a range from 0 to 100, where 0 means the part does not has that attribute at all and 100 means it has the most outstanding attribute. By using the coding programme just described, the ten parts can be quantitatively described as in Table 1.

To calculate how close each part to its best suitability for manufacturing, the following formula is used:

$$C(i, j) = (A(i, j) - A(i, jmin)) / (A(i, jmax) - A(i, jmin)) \quad (1)$$

where i = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 and j = 1, 2, 3, 4, 5; $A_{(i,j)}$ is the jth attribute of the ith part; $A_{(i, jmax)}$ and $A_{(i, jmin)}$ are the maximum and the minimum values of the jth attribute of the ith part. The result of the calculation is called clustering degree $C_{(i,j)}$ whose values are listed in Table 2.

Table 1. Attribute values describing tasks

	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}
A_1	90	80	75	70	85	95	90	90	65	60
A_2	85	95	80	65	70	90	75	90	55	50
A_3	75	85	80	95	60	90	50	80	70	65
A_4	60	80	95	80	80	85	75	55	90	60
A_5	80	50	65	70	80	95	90	60	90	55

Table 2. Clustering degree of tasks

	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}
C_1	0.86	0.57	0.43	0.29	0.71	1.00	0.86	0.86	0.14	0.00
C_2	0.78	1.00	0.67	0.33	0.44	0.89	0.56	0.89	0.11	0.00
C_3	0.56	0.78	0.67	1.00	0.22	0.89	0.00	0.67	0.44	0.33
C_4	0.13	0.63	1.00	0.63	0.63	0.75	0.50	0.00	0.88	0.13
C_5	0.67	0.00	0.33	0.44	0.67	1.00	0.89	0.22	0.78	0.11

Next the fuzzy relation of every two parts is defined based on the maximum and the minimum clustering degree, i.e., the degree of the similarity between every two parts is calculated by:

$$r(i, j) = \sum_{j=1}^{5} min(A(i, j), A(k, j)) / \sum_{j=1}^{5} max(A(i, j), min(A(k, j))) \quad (2)$$

where i , k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 and j = 1, 2, 3, 4, 5. The fuzzy relation, $r_{(j,k)}$ among the tasks described in Table 1 is represented in the relation matrix:

$$r(i, j) = \begin{bmatrix} 1.00 & 0.52 & 0.53 & 0.42 & 0.62 & 0.66 & 0.62 & 0.75 & 0.39 & 0.19 \\ 0.52 & 1.00 & 0.65 & 0.56 & 0.49 & 0.62 & 0.39 & 0.61 & 0.33 & 0.15 \\ 0.53 & 0.65 & 1.00 & 0.64 & 0.55 & 0.60 & 0.44 & 0.53 & 0.54 & 0.18 \\ 0.42 & 0.56 & 0.64 & 1.00 & 0.55 & 0.56 & 0.40 & 0.40 & 0.54 & 0.21 \\ 0.62 & 0.49 & 0.55 & 0.55 & 1.00 & 0.59 & 0.73 & 0.42 & 0.54 & 0.17 \\ 0.66 & 0.62 & 0.60 & 0.56 & 0.59 & 1.00 & 0.62 & 0.58 & 0.48 & 0.13 \\ 0.62 & 0.39 & 0.44 & 0.40 & 0.73 & 0.62 & 1.00 & 0.13 & 0.42 & 0.08 \\ 0.75 & 0.61 & 0.53 & 0.40 & 0.42 & 0.58 & 0.13 & 1.00 & 0.22 & 0.16 \\ 0.39 & 0.33 & 0.54 & 0.54 & 0.54 & 0.48 & 0.42 & 0.22 & 1.00 & 0.24 \\ 0.19 & 0.15 & 0.18 & 0.21 & 0.17 & 0.13 & 0.08 & 0.16 & 0.24 & 1.00 \end{bmatrix} \quad (3)$$

Now a threshold value, λ, is required to enforce $r_{(j,k)}$ equal 0 if it is less than λ and 1 if it is greater than and equal to λ. For instance, if the threshold value λ is set as 0.65, a similar clustering matrix designated as $r_{(i,j)\lambda=0.65}$ can be derived from relation matrix (3). Therefore the task families are formed by extracting the sub-sets of the elements in this clustering matrix. In this case with $\lambda = 0.65$, six task families, $\{P_1, P_6, P_8\}$, $\{P_2, P_3\}$, $\{P_5, P_7\}$, $\{P_4\}$, $\{P_9\}$, and $\{P_{10}\}$, are clustered. Different task families can be clustered if a lower threshold value, for instance $\lambda = 0.62$, is used.

Another technique adopts the production flow analysis (PFA) (Burbidge 1971) to define task families in terms of manufacturing methods. This technique is used when manufacturing tasks are assigned individually to different specific manufacturing processes or machines rather than to the whole environment. The tool used is a task-environment matrix formed with bit-wise ranking code, having a value of 1 if tasks are accomplished by an environment and of value 0 otherwise. A clustering programme was written based on the rank order clustering (ROC) algorithm (Gallagher and Knight 1986) to deal with large-size task-environment matrix (otherwise its computation needs a specified array or memory). In this was the clustering programme is able to work directly on data base files instead of buffer memory so that the length of the ranking code can be extended to as long as 2^{32} (number of data records a data file can hold in the data base).

The clustering is done by a file rotating mechanism which uses four data files ***f1***, ***f2***, ***f3*** and ***f4***.
File ***f1*** holds the original codes that are initially stored, each code being a row of the ranking matrix and occupying one data record. The codes are automatically sorted in ascending order with the last data record holding the code of the greatest value. From ***f1***, new codes are automatically created, each being the column of the ranking matrix and occupying one data record in file ***f2*** which obtained by "rotating" file ***f1*** 90 degrees, both files together defining the ranking matrix. Next rotating ***f2*** in the similar way to form file ***f3*** which is in turn rotated to form file ***f4***. The two files ***f3*** and ***f4*** define another ranking matrix which is better clustered than the previous ones. Rotating ***f4*** to form ***f1*** and continue the process until the matrix defined by files ***f1*** and ***f2*** and the matrix defined by files ***f3*** and ***f4*** are both ranked and are the same to each other, which means that all tasks are clustered in families.

5. Virtual environment manager

This is a programme specially developed to co-ordinate Superscape VRT5's editors with other programme modules within the system shown in Figure 1. The programme run on IBM-compatible PC's under Windows NT (Networks based version is under development). The entire system can be operated with ordinary hardware without extra peripherals, but the minimum hardware requirements must be at lest a 486 IBM-compatible PC with a hard disk of 80MB space and 640 RAM normal memory, a colour monitor with VGA card, a two button mouse, and the I/O interface with two serial ports (one for mouse, the other for optional device). If it works in MS Windows, the supporting software is DOS 5.0 or higher and the extended memory manager (HIMEM.SYS and EMM386.SYS). For a true VR effect, the system should be equipped with VR devices such as stereo viewing glasses (position tracker HMDs), 3D controller, sound blaster sound board, electronic gloves, joysticks, and tracking device. Virtual environment manager has following main functions.

5.1 Creating master environments

It provides an integrated computing environs that enables users to switch between the 3D graphic interface editing tools within Superscape VRT5 and the data base and the task description modules. This function provides considerable convenience and flexibility during the creation of master environments. In addition the data format of the created worlds can be converted in ASCII text to allow users to create or edit the virtual world by writing text scripts. Alternative, a master environment can also be formed by importing 3D data partially or completely from a CAD package. A typical example is using VRML to down load AutoCAD data to a virtual world. The system allows importing 3D data in DXF or IGES format and then converting the data to their individual private formats.

5.2 Storing master environments in data base

The main problem of storing the master environments in the data base is to assign the environments to the right master matrices. Virtual environment manager ensures the correct association between the tasks and the environment, particularly after the association is changed or updated by, for instance, the deletion and insertion of a task or environment.

Depending on the clustering method used during formatting the master matrix, a master environment can be assigned to a master matrix in one of four ways as one-to-one, one-to-many, many-to-one and many-to-many. If however rank order clustering is used, the virtual environment manager automatically clusters both the tasks and the environments around a master matrix in a many-to-many way.

5.3 Setting up the system

The virtual environment manager is used to initialise the data base and set up the coding scheme in a ASCII file in a format described in Section 3. It supervises the coding programme in the task description and classification modules to create code for tasks. Next it selects a clustering module to form task families and represents them with master matrices. If the PFA clustering method is used to form the task families, virtual environment manager will identify the machines, tools, operations or processes and assign them to the corresponding tasks to form a machine flow, tool flow, operation flow or process flow. Finally, for each master matrix, the virtual environment manager helps users to create a representative machine flow, tool flow, operation flow or process flow, from which, the master environment can be planned and finally created and verified using Superscape's world editor and visualiser. In the end, it saves the master environments in the data base, each being indexed by a master matrix.

5.4 Construction of virtual manufacturing environments

With the virtual environment manager, the construction of a virtual manufacturing environment becomes the process of retrieving a master environment for a given task. For example, if a mechanical part is to be manufactured, a virtual manufacturing environment for carrying out the machining operations can be retrieved. First the part is described by the task description and classification programme into code. Then the virtual environment manager compares the code with each of the master matrices in the data base to search for the one that can accommodate the code, which means the given task is the most likely a member of the task family represented by the master matrix. The virtual environment manager will retrieve from the data base the environment indexed by that matrix for manufacturing the part. The virtual environment manager will activate the Superscape VRT's visualiser to animate the environment or make further change by using one of the Superscape VRT's editors. Figure 2 shows the retrieved environment for machining the rotational mechanical parts. Figure 3 shows the environment of a robot control system constructed with the same method.

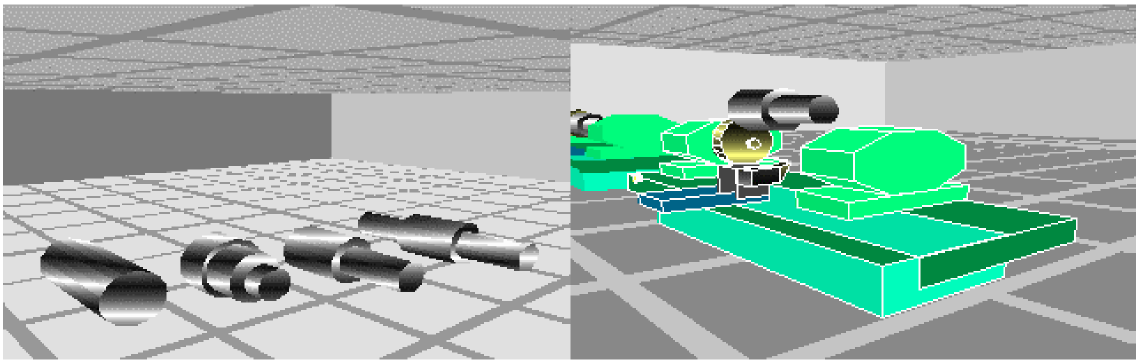

Figure 2. A virtual machining environment

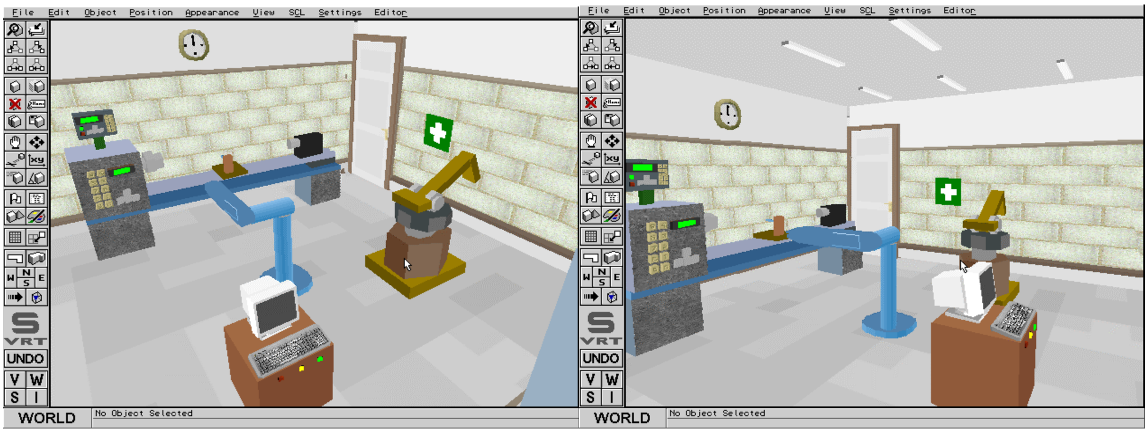

Figure 3. A virtual robot control envoronment

6. Conclusions

The variant approach and the system reported in this paper can reduce the effort in constructing virtual manufacturing environments by avoiding the creation of similar environments from scratch. The master environments in the data base provide building blocks for constructing large and complex virtual worlds and lead to a potential development for a standard VR based library.

The system can be used for not only constructing VR manufacturing environments, but also product validation, process planning, manufacturing verification and shop floor control simulation. It can also be used for planning manufacturing operations, validating product design and verifying shop floor control processes. It can provide a synthetic scenario showing how an entire manufacturing process is to be carried out, eliminating the elements of engineering risks, revealing the potential courses of failure and suggesting better solutions.

However the system needs a large complex data base which is expensive to set up at the beginning. It also takes effort and time to establish the master matrices and the master environments stored in the data base. The clustering algorithms for making the master tasks in the data base must be selected based on human judgement. Because the number of the task families can be limited, the amount of modification on a retrieved master environment depends on the size of the task family. The fewer tasks a task family has, the more task families are needed in the data base and the less modification is required. In an ideal situation where no modification is need, each task family should has only one task, but for each task a master environment has to be prepared.

The potential of the techniques described in this paper is yet to be fully assessed against a real industrial manufacturing background. Little literature of the similar investigation is available. Certain issues addressed in this paper may be debatable while others will need further research. Nevertheless, with the development of virtual reality technology and its fast permeating into manufacturing applications, effective techniques for a fast and economic creation of large and complex virtual environments are no doubt a demand in the future.

References

Blanchard, D. "The reality of virtual manufacturing", *Intelligent Manufacturing, June* 1995 <URL: http://lionhrtpub.com/IM6-Reality.html>

Boman, D. K. "International survey: virtual-environment research", *Computer, IEEE Computer Society*, 1995, 28(6), pp57-65

British Institute of Management, *Classification and Coding*, BSI, London, 1971.

Burbidge, J. L., *Production Planning*, Heinemann, London, 1971.

Carpenter, I. D., Dewar, R. G., Ritchie, J. M. and Simmons, J. E. L. "Enhancing a virtual environment for manual assembly", Proceedings of *The 12th International Conference on CAD/CAM Robotics and Factories of The Future* (Eds. R. Gill and C. S. Syan), Middlesex University, UK, pp1067-1072, Middlesex University Press, 1996.

Kreitler, M., Heim, J. and Smith, R. "Virtual environments for design and analysis of production facilities", IFIP WG5.7 Working Conference on *Managing Concurrent Manufacturing to Improve Industrial Performance*, Seattle, Washington, USA, September 1995. <URL: http: //weber.u.washington.edu /~jheim/VirtualManufacturing/ vrPaperIFIPS. html>

Lou, S. B., Sun, Z., and Chen, H. C., *Fuzzy Mathematics*, Science and Technology Publisher, China, 1983.

Neugebauer, J. G. Virtual reality - more than just simulation, *Industrial Robot*, 19(3), 1992, pp30-34

Sherard, D. C. and Narayanan, M., "The aid of virtual reality in the industry", Conference Record on *Wescon/94, Idea/Microelectronics*, Anaheim CA, USA, September 1994, IEEE Press, USA, pp374-377.

Superscape VR plc., *Superscape VRT for Windows: User Guide*, Superscape VR plc. UK, 1996.