NISTIR 6076

Translating IGRIP Workcells into VRML2

Qiming Wang
Sandy Ressler

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards
and Technology
Information Technology Laboratory
Gaithersburg, MD 20899-0001

NIST

NISTIR 6076

Translating IGRIP Workcells into VRML2

Qiming Wang
Sandy Ressler

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards
and Technology
Information Technology Laboratory
Gaithersburg, MD 20899-0001

September 1997

Translating IGRIP Workcells into VRML2

Qiming Wang and Sandy Ressler

Information Technology Laboratory
National Institute of Standards and Technology

Abstract

IGRIP has been widely used for the simulation of off-line robot programming and manufacturing processes in various industries. The Virtual Reality Modeling Language version 2 (VRML2) enables the creation of three dimensional (3D) interactive worlds incorporating behaviors and animations which can be operated over the Internet. VRML2 models can be automatically generated directly from existing IGRIP workcells through the use of an efficient translator written at NIST. This work benefits users wishing to share their manufacturing simulations on the World Wide Web. VRML2 IGRIP workcells can also be easily integrated with models generated using other software products.

A translator for an IGRIP workcell to VRML2 written in C, the Igrip2Vrml translator, has been implemented for Unix systems. This software can be used for translating IGRIP devices, workcells, and their behaviors to VRML2 files. The translator converts the Deneb Parts file format, IGRIP device, workcell, and recording file formats into VRML2. The IGRIP device, workcell, and recording file formats were reverse engineered in order to accomplish this. Technical issues of translation, such as the conversions of the IGRIP transform matrix to a VRML2 Transform node, the IGRIP device hierarchy structure to the VRML2 hierarchy, and workcell recordings to VRML2 key frame animations, are presented in this paper. The resulting VRML2 worlds contain complex Deneb workcells with devices that operate according to the Deneb simulation. The advantages and limitations of the translator are discussed. Several examples of translated IGRIP workcells from different projects are presented.

Introduction

Deneb IGRIP is an interactive, three dimensional (3D) graphics-based simulation software package which provides tools for evaluating and verifying production concepts, workcell designs, off-line robot programming, and manufacturing processes. It has been extensively used in a variety of industries and research institutions. Typical of IGRIP workcells that have been developed are simulations of robot motion, ship movement, machinery operation, and human workers.

Virtual Reality Modeling Language (VRML) is a file format for describing 3D worlds over the Internet. VRML Version 1.0 (VRML1) does not support behavior in the 3D world. VRML Version 2 (VRML2) describes 3D interactive worlds incorporating behaviors and animations which can be operated over the Internet. VRML2 allows sharing of 3D worlds over the Internet. It can also act as a Web publishing media to integrate the 3D models from different software products. 3D manufacturing workcell models and simulations of their behavior is an important application area of VRML2.

Building 3D VRML worlds from scratch is time-consuming work. Even with a good VRML authoring tool, it can take several days to build a VRML model. On the other hand, there are lots of existing 3D models generated by other 3D modeling packages. Translating those 3D models with different formats to VRML files is an ideal way to generate VRML worlds. Various translators for VRML1, such as DXF to VRML1, IGES to VRML1, have already been developed by others. The translation of the Deneb Part and QUEST model files without behaviors to VRML1 was a precursor to the work described here. VRML1 translators consist primarily of straightforward file format translation.

We developed a simple translator of IGRIP workcells to VRML2, the Igrip2Vrml translator , which translates not only the 3D geometry, but also the animation. Currently, this translator does the geometry translation of parts, devices, and workcells. It also translates the simple kinematic motions of devices, and workcell simulations recorded in the IGRIP recording file. The translator does not translate the IGRIP Graphic Simulation Language (GSL) programs. In the following sections, we describe the functionality of the translator, technical issues, limitations of the translator, and present example VRML2 files translated from IGRIP.

Overview of The Translator

The translator is implemented in C under the UNIX operating system environment. It can be called with three options, -d for only translating an IGRIP device, -w for translating an IGRIP workcell without a recording file, and -p for translating an IGRIP workcell with a recording file. For options -d and -w, the user provides the IGRIP device or workcell file name and generated VRML2 file name. For option -p, the user inputs two additional parameters: the recording file name, and animation cycle time.

For option -d or -w, the generated VRML2 device or workcell file includes two sections. The first is a nested Transform node which contains the geometry of the device or workcell. This node's children may have other nested Transform nodes. The hierarchy structure of the Transform node is in accordance with the hierarchy structure of the device and workcell. Each joint of a device is translated to one named Transform node and one related TouchSensor node. The second section of the VRML2 file consists of all the key frame nodes which show the joint motions of the devices, and related ROUTE nodes. When navigating a translated VRML2 model on a browser, a mouse click on the part associated with a certain joint causes the part and its children parts to move according to the joint's initial value, lower limit value, and upper limit value. For option -p, there is an additional section in the generated VRML2 file. This is a group of key frame nodes translated from the recording file, and related ROUTE nodes. A click on the part without any joints, usually the base of one device, starts the animation.

An IGRIP workcell is made up of several devices, and a device is made up of several parts. The IGRIP workcell file has three sections, head, device, and path. All device information is provided in the device section. The IGRIP part files are distributed in several directories which are specified in the configuration file `.telpthfig` in the HOME directory. The devices in the device section refer to their part files with the file names. Figure 1 illustrates the IGRIP files structure and generated VRML files structure.

IGRIP Parts translation is based on the file format specification given by Deneb, the translator reads the point coordinates and the indices of the polygons from IGRIP part file, and outputs VRML IndexFaceSet nodes in the generated VRML2 part file.

IGRIP device translation is the heart of the translator. The IGRIP device file format was reverse engineered due to the lack of a specification. The essential information for translation extracted from the IGRIP device file or workcell file includes device joint information, part file paths, part transform matrices, and the attachment information of the parts. The translator creates one sub directory for each device in the working directory, the name of the directory is the device name with the prefix d_. For example, if the device name is DB100, the device directory is named d_DB100. Each generated VRML2 part file is saved in the device directory to which the part belongs. The VRML2 device uses Inline nodes to connect to the translated VRML2 parts. Details about the translation of the device hierarchy structure are discussed in the next section.

IGRIP workcell and recording file formats were also reverse engineered. Basically, the workcell translation is made of the device translations described in the previous paragraph. In addition, it deals with the workcell hierarchy structure because of the attachments of the devices. The resulting VRML workcell file is a nested Transform node. The IGRIP recording file consists of the information of all device locations, joint motions, and device attachments with time. The translator reads the information from the recording file and generates the related VRML Interpolator nodes for all devices and joints in the resulting VRML file.

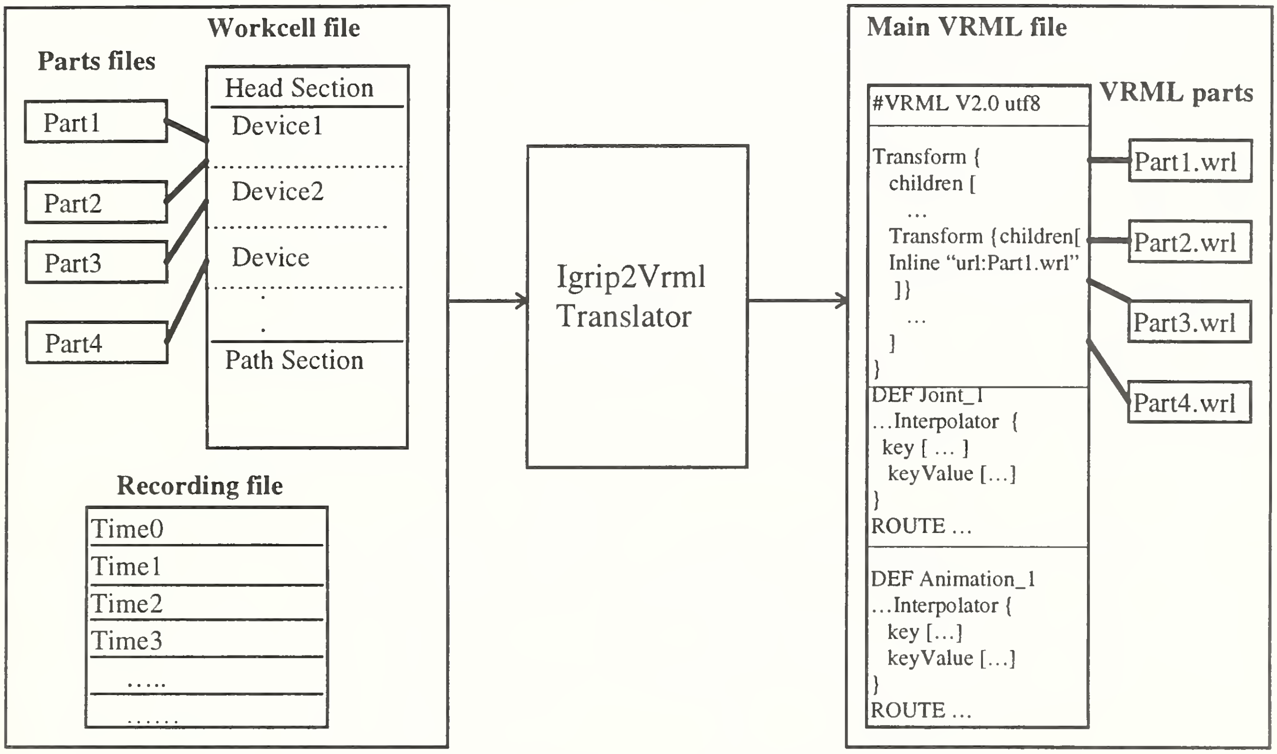

Figure 1: Illustration of IGRIP files and VRML files

Technical Issues

Translating the transform matrix of IGRIP to a Transform node in VRML2

The VRML2 specification uses the Transform nodes with fields: center, translation, rotation ... to represent the coordinate systems instead of using a transform matrix concept. The rotation in VRML2 is defined as the rotation around a specified axis, the rotation field in the Transform node has four values, x, y, z, r, where x, y, z are the unit vector of the axis, r is the rotation angle. The rotation in the transform matrix is composed of the rotations around X, Y, and Z according to certain rules. Deneb software uses a normalized transform matrix to represent the coordinate system. In order to translate the transform matrix of IGRIP to the Transform node in VRML2, some calculation is required. The scale values and translation values can be obtained from the transform matrix easily according to the definition of the matrix. For the rotation components, we adopted the method described by Michael E. Pique[1] for converting between the matrix and axis-amount representation. See the Appendix for more detail about the mathematics of the translation.

Hierarchy of coordinate systems

To build an IGRIP device after creating all needed parts, a user retrieves a base part first, then attaches all parts together in specified local coordinate systems. To build an IGRIP workcell, a user attaches the devices to certain parts of other devices. The attachments of the parts construct the device's hierarchy structure, and the attachments of the devices construct the workcell's hierarchy structure. Translating the attachments of the parts and devices is the core of the translator.

The part attachment information within a device can be found in the device file or the device section of the workcell file. There is a set of bits called the attach flag for each part representing the position of the part in the hierarchy structure which is related to the attachment between the part and another part in the same device. A '1' means opening one new layer in the device structure, '0' means closing the current layer. For example, if the flag is '01', it means closing current layer after putting this part in the structure, then opening another new layer.

The device attachment information in an IGRIP workcell can be found at the end of the device section. There is an integer for each device to represent the device attachment, its value is -1 or a positive integer. If the value is -1, then the device is not attached to any other part. If the value is positive, the device is attached to a part. The positive value tells which part the device is attached to. For example, if the value for a device attachment is '3', then that device is attached to part number 3. All part numbers in a workcell are sequentially numbered from zero and are unique.

The translator creates an integer sequence to represent the layer structure of the workcell after processing all the part attachments and device attachments. In the sequence, integer 999 indicates opening a layer, -999 indicates closing a layer, any other integer represents the part number. The sequence is used to create the nested VRML Transform node which has the same hierarchy structure with the IGRIP workcell. For example , an IGRIP workcell has the following structure:

```
Device 1 :  part 0  attach flag = 0
Device 2 :  part 1   attach flag = 0
Device 3 :  part 2   attach flag = 1
             part 3   attach flag = 00
Device attachments:  -1, -1, 1
```

The generated integer sequence is {999, 0, -999, 999, 1, 999, 2, 999, 3, -999, -999, -999}. The translated VRML world has the hierarchy structure :

```
Transform {  children [
        Transform { children [ part 0] }
        Transform  { children [ part 1
                        Transform { children [ part 2
                                        Transform { children [ part 3 ] }
                                ] }
                ] }
]}
```

Recording file translation

When running a workcell simulation in IGRIP, a user can turn the "recording" on, and save the simulation in a recording file. This function provides an opportunity to play the simulation of a generated VRML2 workcell in a VRML environment. The recording file records the transform matrix and all joint data for each device at each time step. It also records the attachments of the device to the part of another device which change during the simulation. The translator generates one Interpolator node and related ROUTE nodes for each joint of each device. The type of Interpolator node, which is PositionInterpolator, OrientationInterpolator, or ScaleInterpolator, depends on the kinematics information of the joint.

The Grab function of GSL programs is often used in workcell simulations. The device grabbed by other device changes its attachment during a Grab function. For example, a workcell has three devices: `device_1`, `device_2`, and `device_3`. Each device has only one part. The `device_1`'s part is `part0`, the `device_2`'s part is `part1`, and the `device_3`'s part is `part2`. The `device_2` is attached to `device_1`, and `device_3` will grab `device_2` during the simulation. The grab happens at the time 'time_grab'. Therefore, the `device_2` is attached to `device_1` before the grabbing, and is attached to `device_3` after the grabbing. There are two problems for the translation. First, the geometry structure changes because of the change of attachment. Second, the attachment change should happen at a certain time. In this translator, the VRML Switch nodes are used to deal with the geometry structure problem, and ScalarInterpolator nodes with associated Script nodes are used to deal with the time problem. Following is the simplified VRML file generated for above example:

```
DEF TIME TimeSensor {...}
Transform { children [part0
                    DEF Switch_1 Switch { whichChoice 0
                                          choice [ part1 ] }
                    Transform { childern [ part2
                              DEF Switch_2 Switch { whichChoice -1
                                                    choice [ part1 ] }
                              ] }
          ]
}
DEF Script_1 Script {  eventIn SFFloat switch_time
                       eventOut SFInt32 sw
                        url "javascript:
                        function switch_time(value) {
                           if ( value < 0.5) sw = 0;
                           else sw = -1;
                       }"
}
DEF Script_2 Scirpt {  eventIn SFFloat switch_time
                      eventOut Int32 sw
                       url "javascript:
                      function switch_time(value) {
                          if ( value < 1.) sw = -1;
                          else sw = 0;
                      }"
}
DEF Scale ScalarInterpolator {
      key [0, time_grab, time_grab,1]
      keyValue [0,0,1,1]
}
ROUTE TIME.fraction_changed TO Scale.set_fraction
```

```
ROUTE Scale.value_changed TO Script_1.switch_time
ROUTE Script_1.sw TO Switch_1.whichChoice
ROUTE Scale.value_changed TO Script_2.switch_time
ROUTE Script_2.sw TO Switch_2.whichChoice
```

Limitations

Because Deneb's IGRIP is a powerful 3D graphics simulation software package, there are many functions that could not be translated directly to VRML2. In addition, the file format of the device, workcell and recording files were partially reverse engineered. Therefore, this initial translator has many limitations, such as:

(1) The translator must be modified if Deneb IGRIP file format changes.

(2) The curves, surfaces, texts, and textures are not transferred.

(3) The device joint related information represented as functions of the degree of freedom, such as =...dof(1), are ignored. The function is replaced by zero.

(4) Command Line Interpreter (CLI) commands embedded in GSL programs are ignored. They are not shown when playing the VRML2 recording file

(5) The color seen in the VRML environment is the color defined in the part file. If the color of a part is redefined in the IGRIP device, it is ignored.

(6) Every device can only grab/release one other device and it can only be grabbed/released by one device during a simulation.

Due to these limitations, some translated VRML2 workcell models may need additional authoring work. Even in this case, the translator still provides a good initial model for further improvement. Some examples shown in this paper were modified using a text editor.

Examples

In this section, three example workcells are described. More examples can be found on the Web page:

http://www.nist.gov/itl/div894/ovrt/OVRThome.html

<u>PCAR model</u>

This IGRIP Car Welding model was developed by Deneb and the workcell file is one of the IGRIP Demos. It was translated to VRML2 file using this translator without any additional efforts. It has two welding guns which are grabbed and released by two Deneb Robots. The welding guns have sparks that appear when welding the car. Fig.2(a) is the IGRIP model and Fig. 2(b) is the VRML2 model.

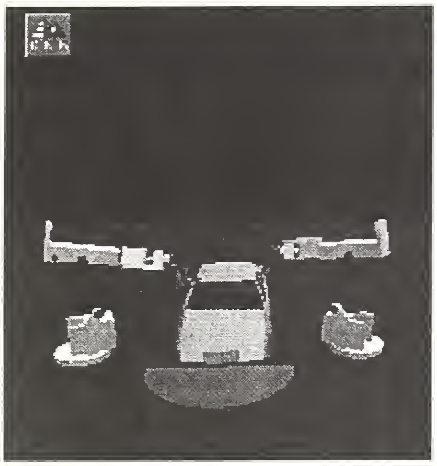

Figure 2(a): IGRIP Car Welding model from Deneb Demos

Figure 2(b): VRML2 Car Welding model translated from IGRIP

NIST Hexapod model

The IGRIP Hexapod model in Fig. 3(a) was developed by Nickolas Dagalakis of the Manufacturing Engineering Lab, National Institute of Standards and Technology (NIST). The Octahedral Hexapod is a machine tool for the production of the parts, which combines speed, accuracy, stiffness, and multiaxis versatility. CLI commands were embedded in IGRIP GSL programs to draw lines which visualize the error and sensor vectors (whisker lines) during the simulation, the coordinates of the lines were calculated and saved in a file.

An initial VRML2 Hexapod generated using the translator includes the geometry and the animation of the robot without drawing the whisker lines. It was enhanced with hand-coded Script nodes for drawing the lines. Fig. 3(b) shows the final model.

Figure 3(a): IGRIP Hexapod model from MEL,NIST

Figure 3(b): Generated VRML2 Hexapod model

Assembly model

The IGRIP assembly model in Fig. 4(a) was developed by Xiangyu Zhou of the Manufacturing Engineering Lab, NIST. This model illustrates one work station in the Black &

Decker miter saw assembly line[2]. The VRML2 model is shown in Fig. 4(b). Because it has complex Grab/Release actions of the devices during the simulation, additional authoring work has been done.

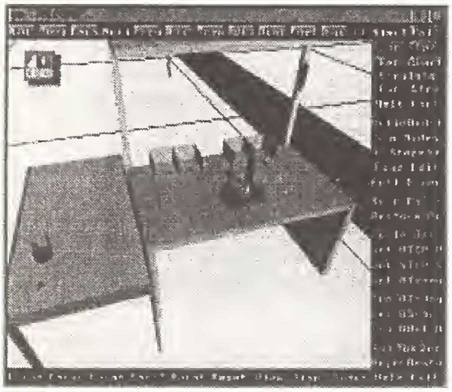

Figure 4(a): IGRIP Assembly model from MEL, NIST

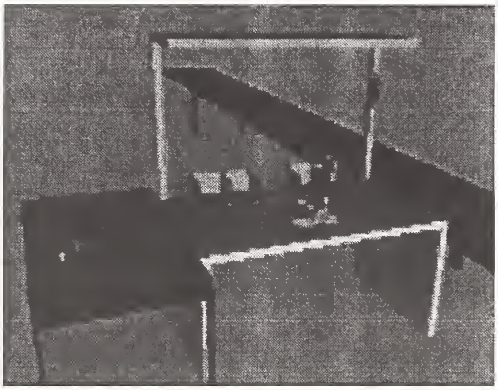

Figure 4(b): Generated VRML2 Assembly model

Summary

The Igrip2Vrml translator will generate 3D VRML models from existing IGRIP workcells which include the kinematic joint motions and workcell simulations. Some IGRIP models can be translated to VRML2 models without any further work, some may need more authoring to the generated VRML file due to limitations of this version of the translator. The IGRIP software is not necessary to translate the IGRIP workcell if all related files are available. The source code of the translator are available on the Internet at the web page:

http://www.nist.gov/itl/div894/ovrt/OVRThome.html.

Acknowledgments

The authors thank Nickolas Dagalakis of MEL, NIST and Xiangyu Zhou from Technological University for providing the IGRIP models.

Appendix Converting between matrix and axis-amount representation

Let matrix R[3,3] to present the rotation about an arbitrary axis through the origin:

$$R = \begin{pmatrix} tx^2+c & txy+sz & txz-sy \\ txy-sz & ty^2+c & tyz+sx \\ txz+sy & tyz-sx & tz^2+c \end{pmatrix}$$

where x, y, z are the components of a unit-vector along the axis, θ is the angular amount of rotation, and $s = \sin(\theta)$, $c = \cos(\theta)$, $t = 1-\cos(\theta)$. Given the matrix R, θ can be calculated from

$$\cos(\theta) = (R[0,0]+R[1,1]+R[2,2]-1) / 2.$$

Providing $\sin(\theta) \neq 0$,

$$x = (R[1,2]-R[2,1]) / (2\sin(\theta))$$

$y = (R[2,0]-R[0,2]) / (2\sin(\theta))$
$z = (R[0,1]-R[1,0]) / (2\sin(\theta))$

There are two possibilities to make $\sin(\theta) = 0$. First, $\theta = 0$, the axis is undefined. Since this has no meaning for the translation, the result of the rotation will be (0,0,0,0). Second, $\theta = \pi$, the R becomes

$$R = \begin{pmatrix} 2x^2 -1 & 2xy & 2xz \\ 2xy & 2y^2 -1 & 2yz \\ 2xz & 2yz & 2z^2-1 \end{pmatrix}$$

From it, abs(x), abs(y) and abs(z) can be computed as

abs(x) = sqrt((R[0,0]+1)/2)
abs(y) = sqrt((R[1,1]+1)/2)
abs(z) = sqrt((R[2,2]+1)/2)

The signs of x, y, and z are determined by the signs of xy = R[0,1], xz = R[0,2], yz = R[1,2], and the values of abs(x), abs(y) and abs(z).

References

[1]Andrew S. Glassner, “Graphics Gems”, 1990, p466.
[2]Ressler S. , Wang Q., Bodarky S., Sheppard C., Seidman G. “Using VRML to Access Ma nufacturing Data”, Proceedings of VRML 1997, p109-116.