Int J Adv Manuf Technol (2007) 34:617–630
DOI 10.1007/s00170-006-0627-y

ORIGINAL ARTICLE

A web-based 3D virtual technologies for developing product information framework

Janus S. Liang

Received: 22 July 2005 / Accepted: 25 March 2006 / Published online: 30 May 2006

Abstract Information technologies and the web are challenging, and changing theway industry works. It is believed that web-based 3D interface can be applied to strengthen the interactive visualization and manipulation of products. In this research, a web-based 3D virtual technologies for developing product information framework is applied in automotive electric system. It is very convenient to share with different partners (e.g., co-designers, manufacturers, customers, etc.) via the web for viewing the aesthetic functions, and verifying its related design functions. The overall procedures are listed: First, build CAD models of automotive electric system (including assemble parts and their components) through CAD system, then store them in a relational database. Second, transforming the CAD models users click into virtual reality modeling language (VRML) objects, it can be visualized graphically and animated interactively using a web browser with a VRML plug-in. Finally, operate the VRML objects by Java Language for the different purposes. Due to the technologies that are applied, many interactive modes are generated in cyberspace. Displaying the exploding process of 3D assemble part on the browser, assigning any component of assemble part through the mouse chick directly to display design details, e. g., dimensions, tolerances, datum planes, material, other references, etc. by retrieving the corresponding data from database, and building 2D projected views automatically follow the 3D viewpoint that users assign.

Keywords CAD models · VRML · Web-based 3D virtual technology

J. S. Liang (✉)
Department of Mechanical Engineering,
Yung-Ta Institute of Technology and Commerce,
Linlo, Ping Tung 909 Taiwan, Republic of China
e-mail: janus@mail.ytit.edu.tw

1 Introduction

Product data management (PDM) systems keep track of the masses of data and information required to design, manufacture and then support and maintain products during the entire product life cycle, and so make it possible for companies to shorten product development time and sales-delivery process. The term sales-delivery process includes all the phases required to sell, design, order, manufacture, and finally deliver an individual product to a customer, as shown in Fig. 1 [1]. In product design field, it is more than production of drawings, there are continuous modifications, often called "design changes" or "engineering changes," during the process of design. These changes might occur during the stage of product development or recur during the manufacturing stage. Each change brings forth new data, such as new design layout, dimension/tolerance modification and redesign of processing. Without a suitable data management system for data storage and maintenance, these new data are subject to impairment and loss. In addition, it might become more difficult to retrieve the data when gradually accumulated in a large volume. Meanwhile, the data are used not only by design and manufacturing departments but also many others, such as marketing and administration departments. Therefore, it is a vital issue for enterprises to establish a data management mechanism for product data, flowcharts and related data in response to future challenges in the increasingly competitive environment.

Furthermore, most of commercial PDM systems have to be installed the specific tools or viewer software on specific platform for browsing or navigating the product information. Additional costs and training fees are not being avoided to promote the company's competition and reduce the lead time of product. Develop the interacting module by open source software, then embedded it into popular

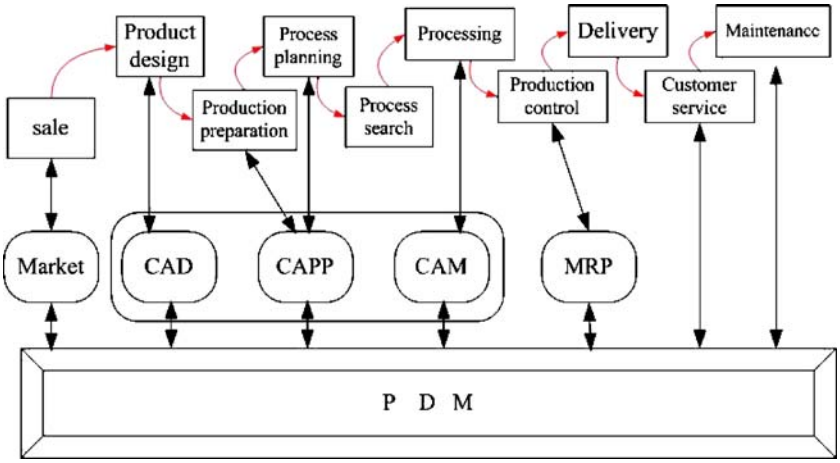


Fig. 1 Flowchart for developing product

browsers (e.g., IE, Netscape, etc.), and a better solution in viewing product information will be achieved. Basically, every CAD system is developed by using different programming language and complier, apply different ways in managing resource, and have different appearance and functions, it is not so easy to navigate the product data, manipulating product models on the internet directly [2, 3]. To visualize and handling of such product data on the World Wide Web (WWW), a human-computer interface must be built. It contains routines for carrying out window management as well as operations on graphics. In applications such as engineering design, modeling, data visualization, product presentation and management, etc, specific graphical interface including specialized interactive dialogues and 3D dynamical scene rendering must be provided, its central work focuses on 3D representation, modeling, object interaction, navigation, etc. Web-based 3D virtual technologies, especially Java gains special attention, as it offers a whole plethora of exciting possibilities for dynamic interaction with the users. One example is the VRML and Java-based visualization tool [4]. Since Java-based application programming interface allows a shared engineering environment, it is possible for users in separate locations to retrieve data simultaneously and to make changes interactively on different platforms over heterogeneous networks. Meanwhile, Java3D API [5] is an application programming interface used for writing 3D applications and applet. It is capable of describing very large virtual worlds. It can provide advanced rendering and interaction capabilities, it can also enable collaborative access of 3D information over the network and has a flexible data input mode.

Besides, virtual reality modeling language (VRML, pronounced as "vermel") is an evolving internet web-based scene description language [6]. Krueger [7] describes VR as "artificial reality". The future of virtual environments and interactivity on the WWW is an exciting and powerful opportunity for communication, collaboration, and participation that will play a significant part of our future. Like its hypertext sibling, (HTML), VRML consists of a script describing the layout or arrangement of objects, plus the data, which in this case are formulas for geometric solids. Unlike HTML, the "page" described is a virtual space or three-dimensional environment in which objects are located and around and through which the user can navigate. The primary advantage of VRML is that it can be transferred and viewed using the internet and free browser plug-ins. VRML can be used to not only create models of worlds, but also to represent data in a three dimensional form.

This research chooses the automotive electric system as the source for developing the product information framework, using CAD/CAM software currently available, such as Master CAM, Solid Edge and Pro/Engineer, to build the CAD models and process data respectively about component attributes and the relationship among components. In the component attributes field, component key features (such as datum planes, dimensions/ tolerances of features, etc.), machining tolerances and materials; in the relationship among components, the allowance of fit, assemble process and assemble orientation can be constructed to produce related design/manufacturing documents and figures about the relationship between a assemble part and its components as well as the relationship among components.

Applying a web-based 3D virtual technologies to establish the product information framework, retrieving the product data with the relational database and interacting with the CAD model in the form of VRML object in the scenes, it is operated by coding object-oriented programming languages. For instance, it can allow users to assign different 3D viewpoints of component and create the 2D projected views (front view, top view, and right view) of component when end-user's assignment is done. The system will support in viewing and interacting with the product, but not modification of objects. The operating flowchart of internet-based virtual environment for product information is shown in Fig. 2. First of all, load the CAD model (assemble part and its components) and translate

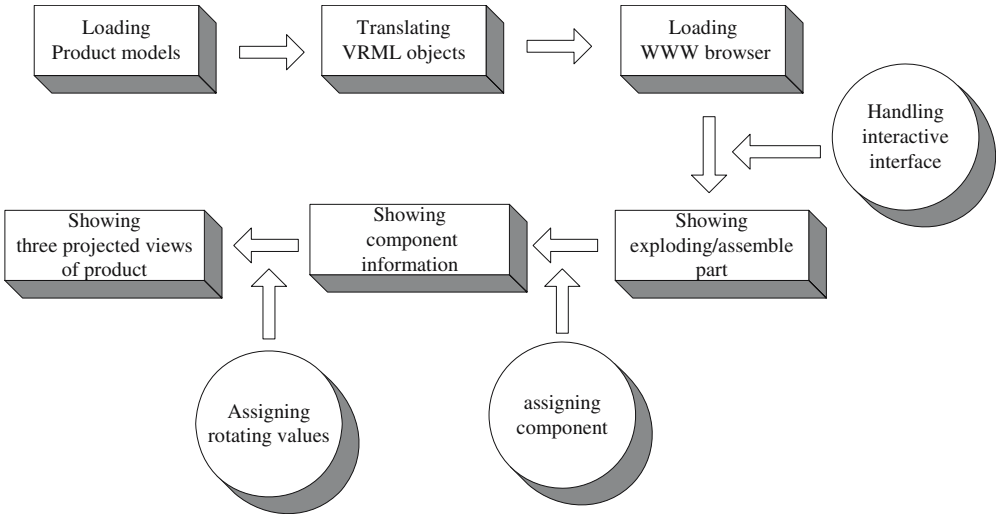


Fig. 2 Operating flowchart of internet-based virtual environment for product information

them into VRML objects, then they can be visualized graphically and animated interactively using a WWW browser with a VRML plug-in, e.g., CosmoPlayer. Secondly, apply the control buttons ("Go" and "STOP") to view the exploding process and break it off (principles and practical procedure are explained in Section 3.1). Thirdly, assign any component directly by mouse and then the component's information is shown in another window, the contents are composed of component name, design data, material and VRML object of component four items (more detail is illustrated in Section 3.2). Finally, show the three projected views (on 2D plane) by means of rotating values of component in three axes (x, y and z direction) are decided. The positive value stands for count clockwise direction and the negative one means clockwise direction (algorithms, formulas and practical procedure are listed in Section 3.3).

Equipped with the above-mentioned functions, this system can help users further understand related information about the product as a whole, important characteristics of each components, and the relationship between the product and its assemble parts.

To explain and comprehend easily about the principles and methodologies in this search, take an example of headlight, an item of headlights system, to demonstrate and illustrate the details on each section of this paper.

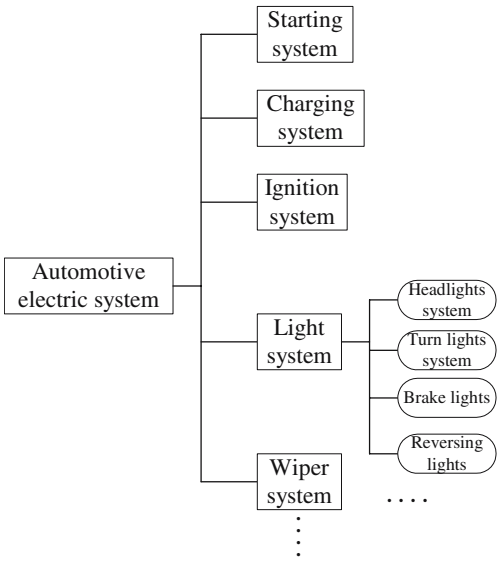


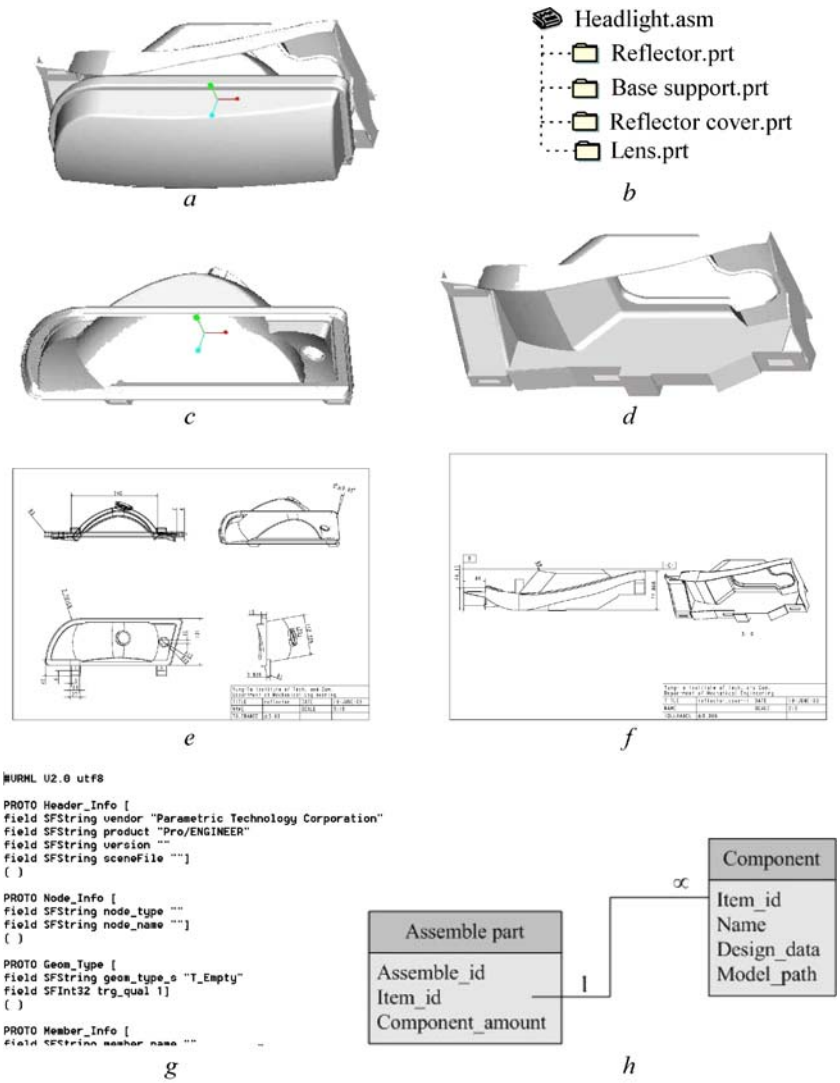
Fig. 3 Tree structure diagram of automotive electric system

2 Building CAD models and storing them in database

An automotive electric system is generally composed of several subsystems such as starting system, charging system, ignition system, light system, and wiper system. Figure 3 shows the tree structure of an automotive electric system. Each subsystem is made up of several important assemble parts. The light system, for instance, consists of headlights system, turn lights system, brake lights, reversing lights and batteries. Each subsystem is responsible for yielding specific function so that the automobile as a whole can work normally.

Following the above-mentioned subsystems to establish data about the design of each assemble part and its components. Using CAD software to build the assemble parts and produce engineering drawings of the automotive electric system. Figure 4 illustrates the headlight of the light system - Fig. 4(a) shows the assemble part, it includes the reflector, base support, reflector cover, and lens, the model tree is shown in Fig. 4(b), while Fig. 4(c and d) respectively represent the reflector and the base support - the components of the headlight. Figure 4(e and f) are the 2D engineering drawings of the two components. During the generation of features, the principal dimensions (including size and location), and engineering drawings of the components are decided according to the functional and aesthetic appeals of the product. Suitable tolerances and materials also are noted.

Since the headlight is an assemble part, the fit among components shall be included as part of the design in



Assemble part			Component	
Filed name	Data Type(Size)		Filed name	Data Type(Size)
Assemble_id	Char(15)		Item_id	Char(15)
Item_id	Char(15)		Name	Char(15)
Component_amount	Number		Design_data	Memo
			Material	Memo
			Model_path	Char(20)

i

Fig. 4 The data models and format of headlight

addition to part design itself. All the dimensions and tolerances of components in this study are set and noted in accordance with the ANSI/ASME Y14.5M-1994 standards.

Transforming the all CAD models into VRML objects by the VRML mechanism, the format is illustrated partly in Fig. 4(g). Then the VRML objects are stored in a MySQL database using PHP (a widely-used general-purpose scripting language), the procedure are: (1) link the database and create the assemble table and component table, the former which is included Assmeble_id, Item_id and Component_amount; the latter that is included Item_id, Name, Design data, Material, and Model_path, the Assemble_id and Item_id will become the primary key in querying assemble part and component of database; (2) store data of CAD model to the corresponding fields of table (as illustrated Fig. 4(h and i), the programs partly are listed:

//Include parameters of database

include("config.php");

```
include("mysql.php");
include_once("./class.upload.php");
//Link database
$dbh=db_connect(array(DB_HOST,DB_USER,
DB_Pass));
if (!$dbh){die(sprintf("Cann't connect to DataBase %s:
%s",db_errno(),db_error())); }
db_select_db(array(DB_NAME));
//Create the table
create assemble_table item (
sn double auto_increment primary key,
Assemble_id char(15),
Item_id char(15),
Component_amount number,
);
create component_table item (
sn double auto_increment primary key,
Item_id char(15),
Name char(15),
Design_data memo,
Material memo,
Model_path char(20)
);
//Store data to the fields of table
$sth_assemble=db_query(array("insert into item values
(",'&Assemble_id','& Item_id','$Component_amount')"));
if (!$sth_assemble){ die(sprintf("Can't execute query %s:
%s",db_errno(),db_error())); }
$sth=db_query(array("insert into item values
(",'&Item_id','&Name','$Design_data','Material','$
Model_path')"));
if (!$sth){ die(sprintf("Can't execute query %s:%s",
db_errno(),db_error())); }
```

3 Building product information framework with web-based 3D virtual technologies

In the past, a huge amount of data has been put on the Internet. To visualize and handle such large scale data on the World Wide Web, web-based 3D visualization techniques, especially Java gains special attention, as it offers exciting possibilities for dynamic interactions with the user. Since the interface is built with Java-based programming and allows a shared engineering environment, it is possible for users in separate locations to browser product information simultaneously and to make changes interactively on different platforms. Based on Java's promise of adapting a user interface to any platform, web-based 3D virtual technologies can bring great portability, flexibility, and provide a good solution to product data.

Literatures are reviewed in the development of product data management system, Chen and Tsao [8] and Miller [9] proposed the establishment of a diversified information network by using web-technology to build the PDM system which can meet the needs of the increasingly complicated product market. Lebovitz [10] and Peltonen et al. [11] suggested the use of the PDM system together with Internet transmission to achieve rapid and real-time information communication among customers, companies, and suppliers. As for the fields of product design and assembly, Vinoid et al. [12] indicated that product design and assembly depend largely on the intended functions of the product as a whole and availability of related information, not solely on the information of each individual component.

Kvan and Candy [13] applied AutoCAD as a platform coupled with SQL database software to establish the smart drawing system that integrates graphic files and design data, providing engineers efficient and instant access to related design graphs and documents via appropriate control interface. Wood and Agogino [14] built the case-based design information system (CDIS), which is implemented using World Wide Web, SQL databases, and CAD packages. Their CDIS enables the designer to effectively manage related data during the process of conceptual design. Its databases store case-based data, such as design drawings, models, graphs and images, in the hyper tex markup language (HTML) format. Dolenc and Makela [15] constructed a design text database which contains CAD files and data about specifications and design. This database is established using the term relevance score (TRS) metric as the base for artificial intelligence. The database can provide designers with needed information and guide user's decision making in a heuristic fashion. Sadd and Maher [16] proposed shared understanding and collaborative design via computers and the Internet. Under the architecture of such a distributive design environment, users can share and use information in the form of texts and graphs. Ginsburg and Kambil [17] established a website-based knowledge management support system, which allows users to store knowledge in texts and compare with knowledge of other users to decide the value of his stored knowledge.

However, for many PDM systems, visualization capability for viewing graphical information is indispensable for managing the product data. Meanwhile, a simulation model should be described and be associated with behavioural descriptions embedded within the model. In this search, a web-based 3D virtual interface can satisfy the needs by constructing scenes of virtue objects according to data description in the product database.

Demonstrating the functions of the web-based 3D virtual interface by using database, where 3D CAD models of automotive electric system are saved. The whole framework is divided into several parts according to the function of each part, which is shown in Fig. 5(a). *Database connection* module is responsible for building a connection to the

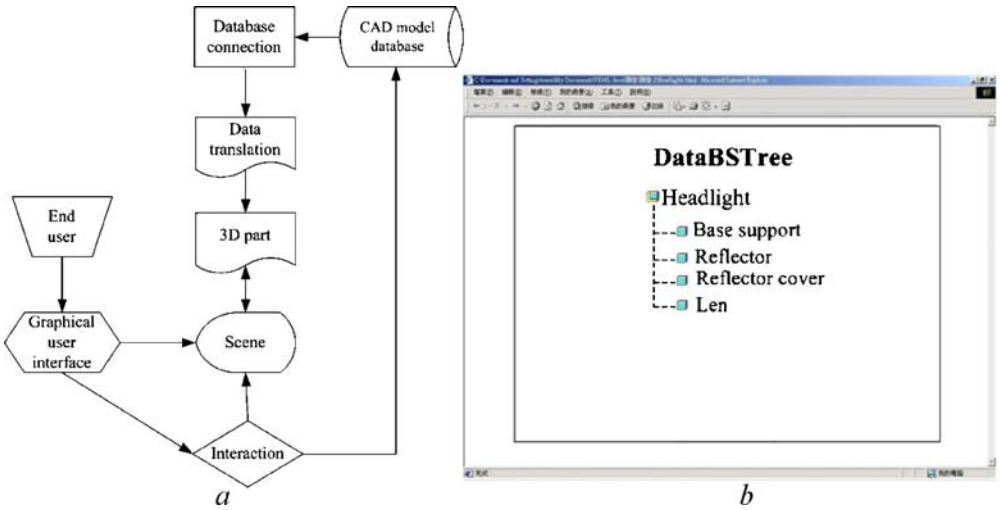


Fig. 5 System overview and database browser ofheadlight

database of CAD models over a network. A PHP application can connect to the database, retrieve or update data from it. Through carrying out SQL queries in PHP program, model data can be read out or into the database.

Due to the difference between the model data format and the Java data structure, a data translation is necessary. Also a Java program treats each field of the tables in database as **string**. Hence the model data should be translated and stored in an **array** before they can be applied by Java program. *Data translation* module will translate the model data into a format suitable for building shape and appearance of the requested objects.

From the transformed data format, each part of the CAD model will be built in the form of VRML object in the *3D part* module. Based on the 3D parts, a scene will be built according to the request of the end user. Interactive request of the end user will be transferred to the *interaction* module by means of graphic user interface, and *interaction* module will inform the *scene* module to update the current display. In this search, the *interaction* module provides three functions. First, it can display exploding process of assembly part. In addition, it provides part information for identifying an object the user has clicked on from the database. Finally, displaying the 2D projected views via assigning the angles of 3D object.

Introduce briefly the two most important ones: 3D scene browser and database browser in *graphical user interface*. The scene browser contains a canvas into which the scene is to be displayed and a menu bar, which action can be triggered on the scene. It is built for communicating with end users. By the way, a VRML plug-in and part information browser will be included in it. The database browser displays all the available objects and parts in a hierarchical tree structure so that user can browser freely within the database (as shown in Fig. 5(b). As for the other functions, e.g., scene representation, visualization, and interaction are displayed individually in the next sections.

In the following, detailed illustration of their functions will be given and explain the algorithms and methods.

3.1 Function for viewing the exploding process

There are two coordinate systems in terms of the object and global coordinate systems are necessary in this function. The object coordinate system is related to a graphics rendering program in *3D part* module, the object coordinate system is consistent with shape definition in VRML 2.0, the centroid of object can be received by Eq. (1):

$$x_{cd} = \frac{\sum_{i=1}^{n} x_i}{3n},\ y_{cd} = \frac{\sum_{j=1}^{n} y_j}{3n},\ z_{cd} = \frac{\sum_{k=1}^{n} z_k}{3n}, \tag{1}$$

where n is the number of triangular-facet of each object, x_i, y_j and z_k the coordinates of three end points of each triangular facet. After obtaining the centroid position, the system continues to conduct a containment test between the centroid point and the closed profile.

To assign a 3D view point randomly, the mechanism has to be built that rotates about an arbitrary axis. The rotation matrices R_x, R_y and R_z (the matrix of rotation about the X, Y, Z axis individually) allow for rotation about the principal axes. These rotations in themselves are not sufficient - for example, unless an object happens to be centered on one of the principal axes, not be able to spin an object around an axis that intersects it. This kind of rotation is important - often in order to get a good understanding of the shape of an object it needs to spin it around an axis that passes through it, or is at least near to it. For instance, construct a composite transformation matrix (R) for rotation by angle α around an axis specified by two points $p_1=(x_1, y_1, z_1)$ and $p_2=(x_2, y_2, z_2)$. R can be constructed as follows:

(1) Translate so that p_1 is at the origin: $T(-p_1)$, T means translation matrix.

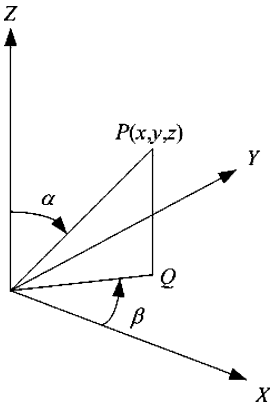


Fig. 6 Diagram of spherical coordinates

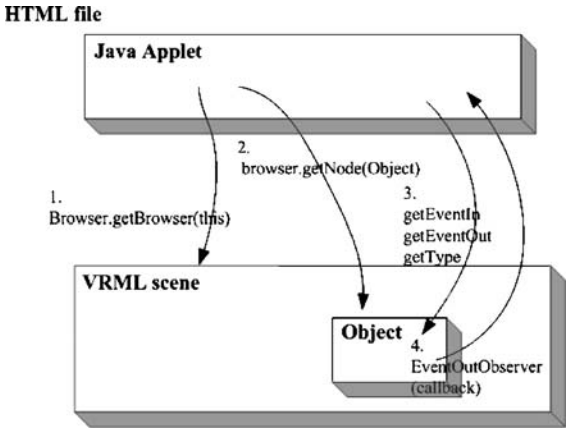


Fig. 7 Diagram for showing browser by Java control VRML objects

(2) Let $(x, y, z)=p_2-p_1$ be the other end of the line segment, and write this in spherical form as (r, φ, θ) (where $\varphi=\alpha$ and $\theta=\beta$ with respect to Fig. 6).
(3) Apply the rotation $-\theta$ about the Z axis to bring Q on the ZX plane: $R_z(-\theta)$.
(4) Apply the rotation $-\varphi$ about the Y axis, so that OP is now coincident with the Z axis: $R_y(-\varphi)$.

Combining these matrixes $M=T(-p_1)\,R_z(-\theta)\,R_y(-\varphi)$ that transforms the axis of rotation to be the Z axis. Rotate by $R_z(\alpha)$ and then apply the inverse transformation M^{-1}. Hence the complete transformation is Eq. (2):

$$R = M\,R_z(\alpha)\,M^{-1} \tag{2}$$

Where:

$$M = T(-p_1)\,R_z(-\theta)\,R_y(-\varphi) \tag{3}$$

$$M^{-1} = R_y(\varphi)\,|R_z(\theta)\,T(p_1)$$

To avoid the collision of objects in exploding process and assign the exploding order, the base-reference object that will be decided follows the real assemble condition of product. In the exploding process, it still keeps the original position, and the other objects will be planned from the moving traces by setting several turning points from different viewpoints. Finally, the above descriptions of each object will be transformed to the global coordinate system, the exploding process of the object is represented in the format of VRML, it includes two parts: translating paths and orientating paths.

Embed the above objects in a web browser with a VRML plug-in, they can be visualized graphically and animated interactively in this cyberspace. Importing EAI (Java using external authoring interface) library and Java library into Java compiling environment, which enables Java to translate and control EAI functions, this type of Java is used mostly in the communication between Applets and VRML scenes when both of them are embedded in HTML pages, the procedure is shown in Fig. 7 take an example of Box. Applets, when embedded in HTML pages, require the browser class packaged in vrml.external, the basic class of browsers is the **browser** class which is embedded in VRML scenes and contains not only all functions which can be used in SAI but also the **getNode** functions which can be used to directly get the DEF-defined node objects in the VRML scenes, using **getEvent-Out**, **getEventIn**, and **getType** functions to obtain the values of event input and output. A change in the value of event input will be responded through **EventOutObserver** to change the scene.

The exploding process of assemble part will be displayed or stopped on the plug-in of web through the user who clicks the "GO" or "STOP" button, and the integrated procedure of handling VRML objects for exploding process is explained as follows:

(1) Retrieve data about the attributes and description of the component from the database through PHP. Then decide and assign the exploding conditions. The following is part of the program content:

```
//Setup related files
$dbh=db_connect(array(DB_HOST,DB_USER,
DB_PASS));
if (!$dbh){
die(sprintf("Cannot connect to DataBase %s:%s",
db_ermo(),db_error()));
}
db_select_db(array(DB_NAME));
$sth=db_query(array("select * from post where
pid='$boardsn'"));
if (!$sth){
die(sprintf("Cannot execute query %s:%s",db_ermo
(),db_error()));
}
```

```
$row=db_fetchrow(array($sth));
…
db_free_result(array($sth));
db_close(array($dbh));
//HTML file
<center>
<embed src="Headlights.wrl" border=0 height="250" width="375">
<applet code="headlights.class" mayscript height="50" width="200">
</applet>
</center>
//Load VRML file
public void init() {
setBackground (Color.Block);
add(grayButton=new Button("Go"));
add(ashenButon=new Button("STOP"));
grayButton.addActionListener(this);
ashenButton.addActionListener(this);
browser=Browser.getBrowser(this);
tss=browser.getNode("caligariCLOCK");
tsloop=(EventnSFBool) tss.getEventIn ("set_loop");
}
public void actionPeformed (ActinEvent event) {
Object clickedButton=event.getSource();
if (clickedButton==grayButton)
{ tsloop.setValue(true); }
else if (clickedButton==ashenButton)
{ tsloop.setVaule(false); }
}
//Setup exploding conditions
DEF Object_1Pos PositionInterpolator {
Key [0 0.448]
keyValue [5.29e–023 −1.7e−007 −3.9, −0.431 3.57e–007 8.17]
ROUTE caligariCLOCK.fraction_changedTOObject_1Pos.set_fraction
ROUTE Object_1Pos.value_changedTOObject_1.set_translation}
}
DEF Object_1Orient OrientationInterpolator {
key [0 0.448]
keyValue [1.33e–015 7.76e–009 2.54e–008 0 .000691,
1.33e–015 7.76e–009 2.54e–008 0.000691]
ROUTE caligariCLOCK.fraction_changedTOObject_1Orient.set_fraction
ROUTE Object_1Orient.value_changedTOObject_1.set_scaleOrientation
}
```

(2) Import EAI library and Java library into Java compiling environment, which enables Java to translate and control EAI functions. The procedure is explained as follows:

(i) Load the following classes that control EAI function into Java:

```
import java.applet.Applet;
import javax.media.j3d.*;
import javax.vecmath.*;
import javax.utils.geometry.*;
import javax.utils.universe.*;
import vrml.external.field.*;
import vrml.external.Node;
import vrml.external.Browser; import vrml.external.exception.*;
```

(ii) Communicate between Java an Web browser. The programs are declared and called as follows:

```
Browser browser;
Browser=(Browser)vrml.external.browser.getBrowser(this);
```

(3) Embed in the HTML file the tag, and then load the VRML plug-in into the web browser to browse the VRML object.

```
<embedsrc="headlight.wrl" border=0 height=100% width=100%>
<applet code="Headlight.class" mayscript height="50" width=200">
</applet>
```

3.2 Function for showing the part information

In order to make it possible for a user to pick a component of the assemble part for detail information, a picking algorithm is developed. The task of picking is to identify whether a location pointed by the mouse can "touch" some part of the scene. As shown in Fig. 8, the virtual viewpoint is positioned at point V_e and the mouse location is at point V_m in the image plane. Since the objects themselves don't lie in the image plane, "touch" means whether the ray passing the two points V_e and V_m can touch some surfaces of the objects in the scene. Solving this problem by projecting the points of the object back onto the image plane, then setting the URL NODE of object.

Figure 9 shows the conceptual model of web-based virtual application, including the web server (server end) and the browser (client end). Through the browser and internet, users can connect to the web server which, after receiving parameters from the client, ascertains requests, and then retrieve corresponding data from the table of Database, i.e., Name, Design_data, Material, and Model_path four fields.

The response data are integrated VRML with PHP, and then sent to the user-end browser to run client programs to browse the VRML object of model and the related model data

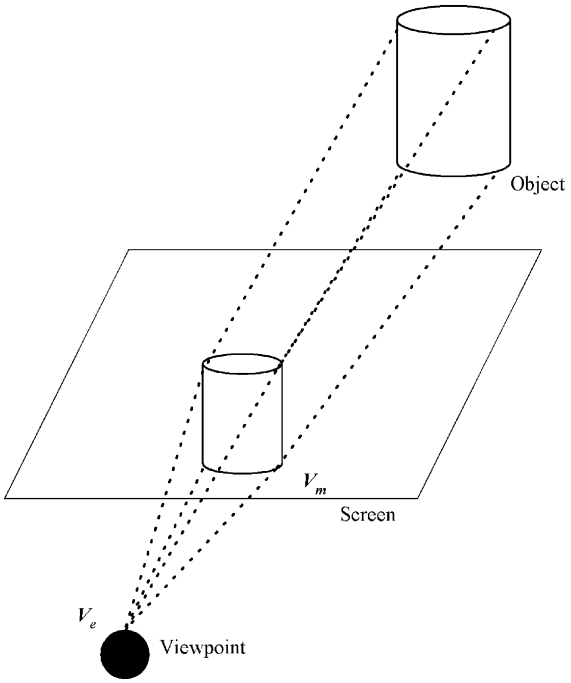


Fig. 8 Concept for imaging projection model

are filled in the form of HTML, the diagram of message flow is displayed Fig. 10(a). The procedure is explained as follows:

(1) Click the object that users assign by the mouse, its URL NODE is triggered and sends the parameter item_id to sever. The program item_object.php gets the argument to link the database MySQL and finds out the database, then query corresponding table and retrieve the data from it. The following is part of the program content:

//Retrieve trigger and setup argument

DEF Link1 Anchor { url "item_product.php?Item_id=reflector"}

//Connect database

$dbh=db_connect(array(DB_HOST,DB_USER, DB_PASS));

//Query data from database

$sth=db_query(array("select * from item where itemid="Item_id")).

(2) Build the templates for data of object, there are four regions generated as shown in Fig. 10(b). The name of object, design data, material, and 3D visual object fill the coincident positions. The following is part of the program content:

//Build templates

$tp1->define(array(board=> "compview.htm"));

//Show data on templates

$tp1->assign(VIEW1, "$temps1");

$tp1->assign(VIEW2, "$temps2");

$tp1->assign(VIEW3, "$temps3");

$tp1->assign(VIEW4, "$temps4");

//Detect template and output

$tp1->parse(MAIN, "board");

$tp1->FastPrint(MAIN).

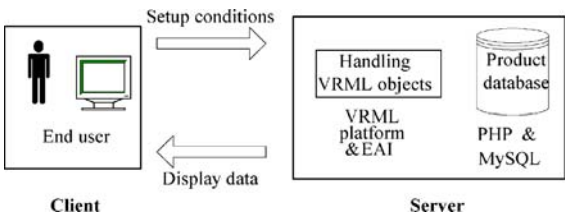


Fig. 9 Concept of web-based virtual application

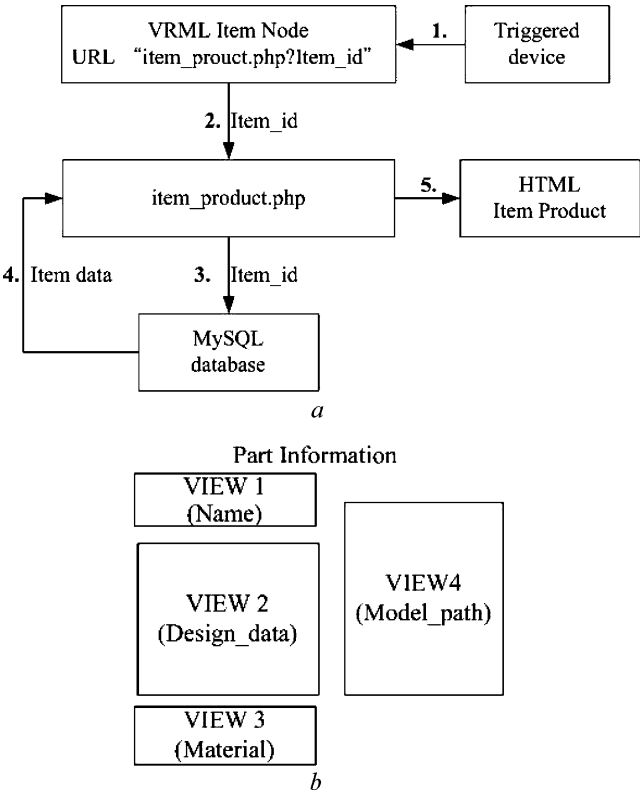


Fig. 10 Diagram of message flow and templates for part information

If no 3D part is located at the position the mouse can "touch", the function just does nothing. Otherwise a part information window will be displayed, which provides detailed information about the found 3D part.

3.3 Function for displaying 2D projected views

Projection is the process of representing a 3D scene on a 2D plane, and it is also a way of associating each point in 3D space with a unique point in the 2D space in which the representation is to appear. There are two fundamentally different ways of doing a projection, the first is perspective and the second is parallel. In this research, the latter is applied to show the 2D views of product. Figure 11(a)

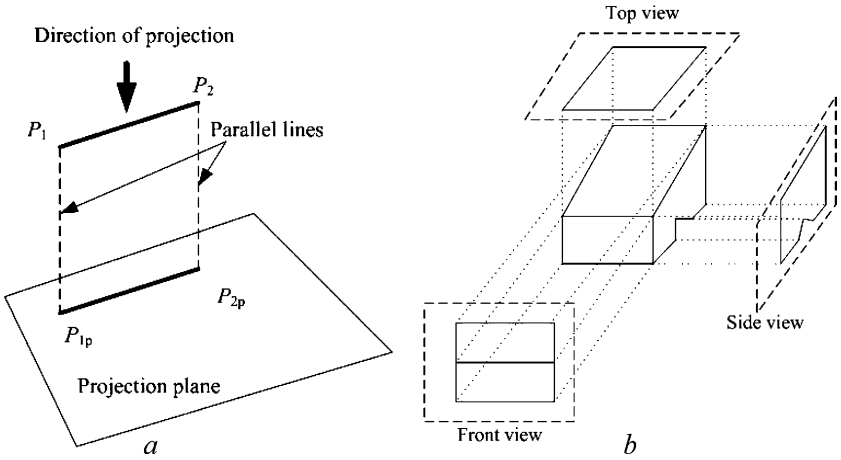


Fig. 11 Diagram of parallel projection and three orthographic views

illustrates projection rays (projectors) emanate from a center of projection (COP) and intersect projection plane (PP). The COP for parallel projectors is at infinity. The length of a line on the projection plane is the same as the true length. There are two different types of parallel projections: If the direction of projection is perpendicular to the projection plane then it is an orthographic projection. If the direction of projection is not perpendicular to the projection plane then it is an oblique projection. Look at orthographic projection: it is simple, just discard the z coordinates. Engineering drawings frequently use front, side, top orthographic views of an object. Here are three orthographic views of an object, as shown in Fig. 11(b). Moreover, the operation of parallel projection is listed below:

//Initial conditions
$Ax+By+Cz+D=0$ **//projection plan**
(u, v, w) **//direction of projection**
(x_0, y_0, z_0) **//a point on the object to be projected**

(1) Start at (x_0, y_0, z_0) and travel along the line in direction (u, v, w) until plane Ax+By+Cz+D=0 is hit, and the parametric equation of the line is

$$x = x_0 + \boldsymbol{ut} \qquad (4-1)$$

$$y = y_0 + vt \qquad (4-2)$$

$$z = z_0 + wt \qquad (4-3)$$

(2) At some value of t, when the plane equation is satisfied, the projection plane is

$$Ax + By + Cz + D = 0 \qquad (5-1)$$

$$A(x_0 + ut) + B(y_0 + vt) + C(z_0 + wt) + D = 0 \qquad (5-2)$$

$$Ax_0 + By_0 + Cz_0 + t(Au + Bv + Cw) + D = 0 \qquad (5-3)$$

(3) Solving for the unknown parameter value from Eq. (5-1), Eq. (5-2), Eq. (5-3)

$$t = -\left[\frac{Ax_0 + By_0 + Cz_0}{Au + Bv + Cw}\right] \qquad (6)$$

(4) Substituting this value of t into the previous line equation from Eq. (4-1), Eq. (4-2), Eq. (4-3) for x, y, and z gives an expression for the projected point (x_p, y_p, z_p).

To realize the above mathematical model in the virtual environment, virtual modules have to be applied. A Java3D [18] scene is created as a tree-like graph structure, which is traversed during rendering, all classes, objects and instances are referenced from it. Figure 12(a) shows the typical structure of Java3D scenegraph - scene branch (left side of locale) and view branch (right side of locale). To develop the platform of product information through the application of 3D virtual technologies, build a new scenegraph for projecting 3D objects into 2D views (top view, front view, and right view), as displayed in Fig. 12(b). The **BasicUniverse** object is the root of our scene graph and provides the overall framework for scene representation. A **locale** object defines a high-resolution position within a BasicUniverse, and serves as a container for a collection of BranchGroup-rooted subgraphs (branch graphs), at that position. Objects within a locale are defined using standard double-precision coordinates, relative to the origin of the locale. On the left side of the scene graph is a content branch, which contains the nodes that describe the actual objects in the scene, and on the right-hand a view branch, which contains nodes that specify viewing related conditions.

In the view branch, **TransformGroup** object called **viewTransform** is established for the orientation of the **ViewPlatform**. The viewTransform controls what the user can observe on the screen. Hence, the end user can change view by changing this transform via the graphical user interface of browser.

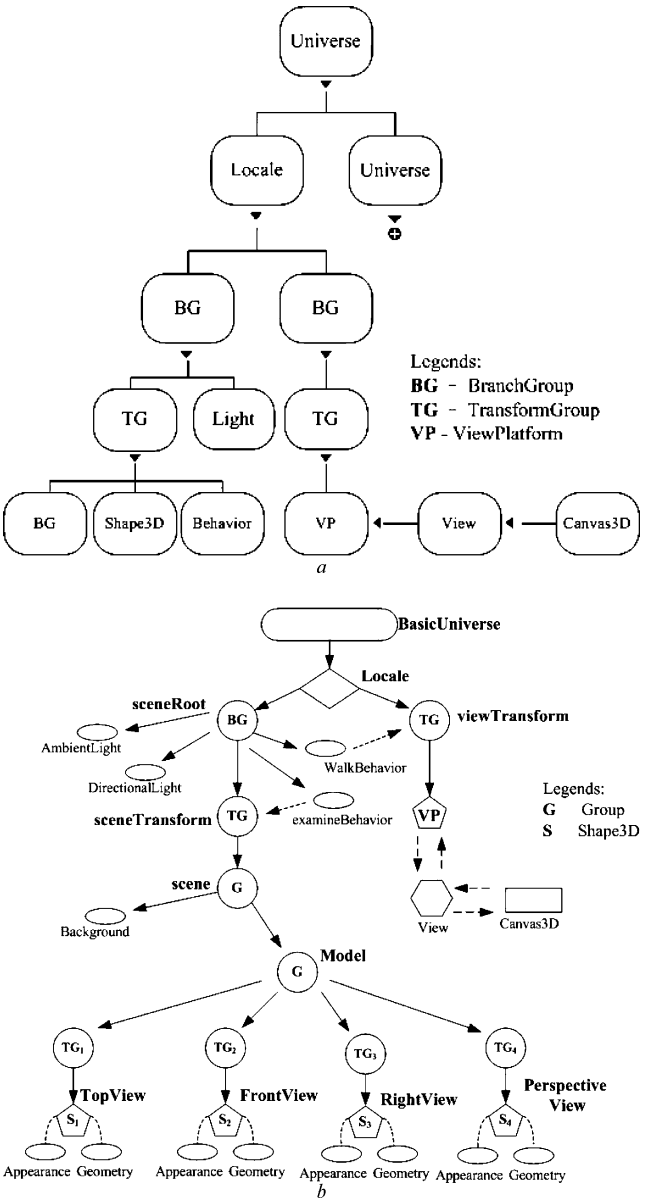


Fig. 12 Typical Java3D scenegraph and revised one for projecting 3D into 2D views

The root of the content branch is called **sceneRoot** which has several sub nodes. Lights are needed for realistic colored shading, using ambient light to illuminates the scene. Directional light defines an oriented light, with which the effect of bright and dark can be observed on the screen. **Behavior** objects can change the position, rotation, and scale of the scene according to the user's request. The **WalkBehavior** can change the viewTransform by acting on the ViewPlatform and changes the orientation of the observer. **examineBehavior** can act on a scene by changing the **sceneTransform**, another child of sceneRoot, through input devices, e.g., mouse, keyboard, and so on.

The **scene** object is made of **background** and **model**. Background defines the background color of the scene.

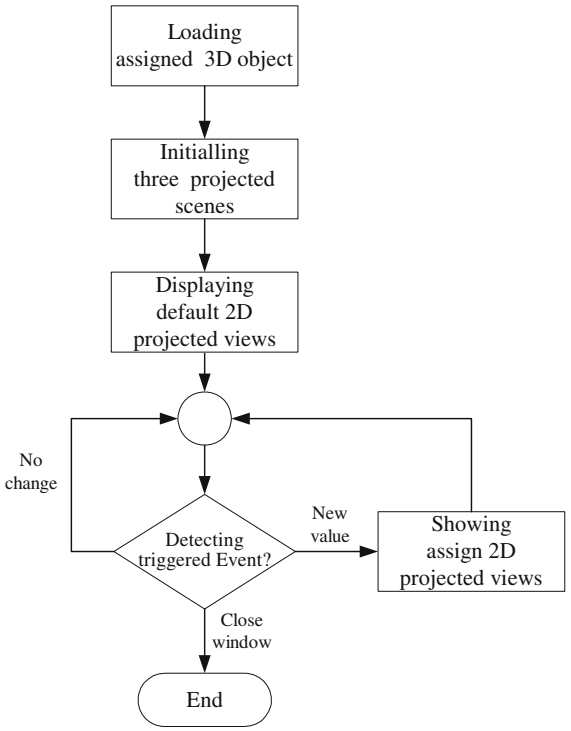


Fig. 13 Operating flowchart for displaying 2D projected views

Model determines how many 3D parts are included for building an object in the scene. Each part of the object is built in the *3D part* module, which models the 3D shape, appearance, and orientation of the geometric part. The class **shape3D** is used to build 3D parts, each **shape3D** object has an **appearnace** node and a **geometry** node. The appearance can be defined using Java3D class **appearance** and **material**, which can be further set by such attributes as color and degree of transparency. Java3D can regulate the orientation automatically, once the default translation, rotation and scale of the object are provided. A **geometry** object can be either primitive object (e.g., cone, box, etc.) or component object which indicates by name that the geometry is composed of a set of 3D points. The **IndexedTriangleArrary** is used to build the 3D geometry.

The main purpose of the function is to provide interactive dialogues so the scene can be visualized dynamically when a user navigates within or interact with the scene. Figure 13 shows the operating flowchart of this function. Firstly, detect and load the 3D object in the perspective view. Secondly, initialing the three projected scenes, they are front view, top view, and right view follow perspective view of 3D object to display the default projected views. Finally, ascertain several kinds of triggered events to response corresponding actions. If detect the event of close window, this function will be closed. The following is part of the main program:

//Create a scene of BasicUniverse and the ViewPlatform BranchGroup

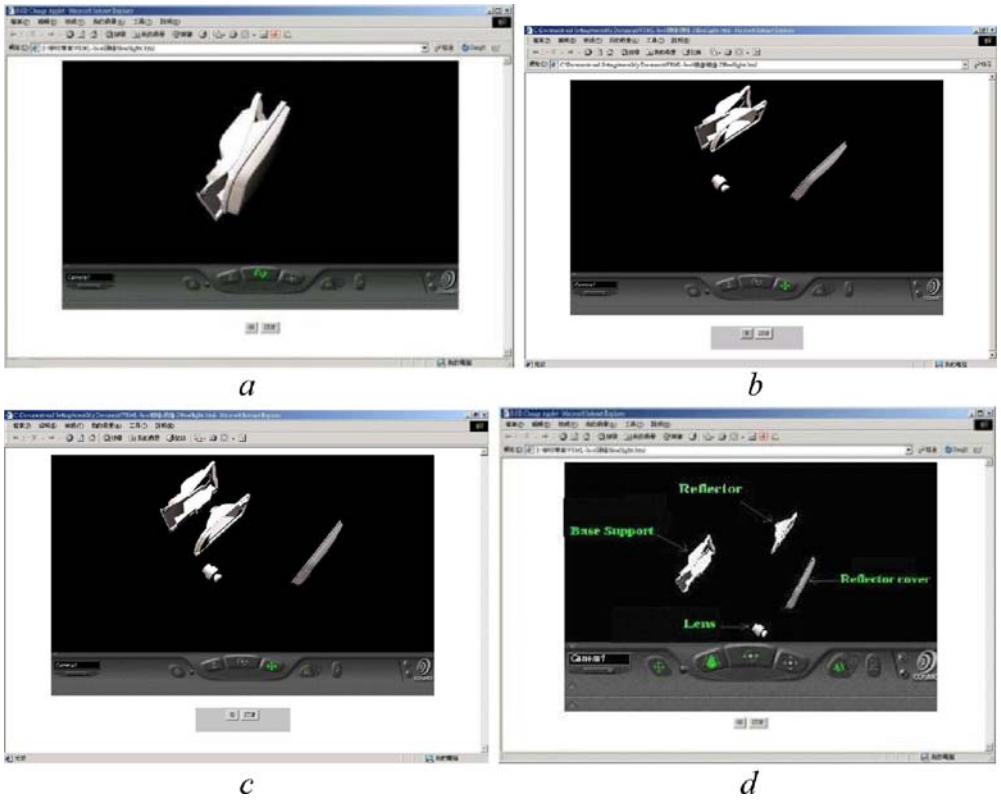


Fig. 14 VRML object and component information of headlight

```
setLayout(new BorderLayout());
GraphicsConfiguration config =
BasicUniverse.getPreferredConfiguration();
Canvas3D c=new Canvas3D(config);
Add("Center" ,c);
BranchGroup objRoot = new BranchGroup();
//Create a Transformgroup to scale all objects so they appear in the scene
TransformGroup objScale = new TransformGroup();
Transform3D t3d = new Transform3D();
T3d.setScale(0.7);
objScale.setTransform(3d);
objRoot.addChild(objScale);
//Load the 3D object what user's assign
ObjectFile f=new Objectfile(flags, (float)(creaseAngle* Math.PI/180.0));
//Default perspective view
objScale.addChild(createObject(f, 1.5f, 1.0f, 0.1));
//Initialing the projected scenes and set default views
BranchGroup b=objFile(filename);
objScale.addChild(creatObject (b, −1.5f, 2.2f, 2.57));
//default top view
objScale.addChild(creatObject (b, −1.5f, 2.2f, −2.0));
//default front view
objScale.addChild(creatObject (b, −1.5f, −1.0f, −1.1));
//default right view
```

To simulate the interactive action of walking of a user in the virtual world, the walk behavior is established. Walk behavior is activated through moving the scroll bar, entering the rotating value, or pressing the UpDown button on the graphical user interface of the browser. Suppose V_x, V_y, and V_z to stand for the rotating angles along x-axis (left toward right), y-axis (outer toward inner), and z-axis (bottom toward top) respectively. The mouse drags, press, or keyboard key in can be mapped to the same transform of the viewTransform. The following is part of the main program:

```
//Read the rotated values and change them into 2D views
TransForm 3D Axis = new Transform3D();
Axis.rot(−Math.PI/2);
RotationInterpolator rotator=new RotationInterpolator (rotationAlpha, spin, Axis, 0.0f,
(float) Math.PI*2.0f);
BoundingSphere bounds=new BoundingSphere(new Point3D(0.0,0.0,0.0), 100.0);
Rotator.setschedulingBounds(bounds);
objTrans.addChild(rotator);
```

4 Implementation

In Section 2, the headlight that an assemble part of lighting system in automotive electric has been illustrated about generation of CAD models and storing them in database for developing virtual design environment of product. First,

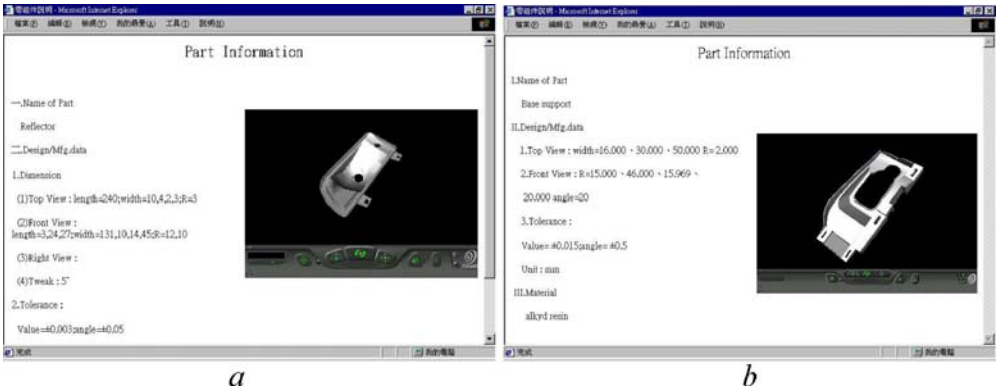


Fig. 15 VRML object and part information of reflector and base support

build assemble part and its components (see Fig. 4(a, c and d)), 2D drawings (as shown Fig. 4(e and f)) by CAD/CAM software. Second, transformall CAD models into VRML objects by the VRML mechanism, the format is illustrated partly in Fig. 4(g). Finally, VRML objects are stored in a MySQL database using PHP application, the data model and corresponding fields of database are displayed in Figs. 4 (h and i).

Follow the theorem and procedure in Section 3 to implement the example. First of all, the VRML object of the headlight assemble part displayed a web browser with a VRML plug-in, as shown in Fig. 14(a). Pressing the left button "GO" by mouse to view the exploding process of assemble part, as illustrated in Fig. 14(b–d), the process will not be interrupted until the right button "STOP" is pressed. Secondly, end users can pick the desired components, such as the reflector and base support, to show 3D object and model design data in the part information browser, as illustrated in Fig. 15(a and b) respectively.

Finally, end users assign the rotating angels by moving the scroll bars, entering rotating values directly or pressing the UpDown buttons on the graphical user interface of the browser. The reflector is shown in Fig. 16(a), and the X, Y, and Z respectively represent the rotating axes of the object, when rotating angles are inputted by UpDown button individually, for example, X=131, Y=−30, and Z=0 in Fig. 16(b), and the three projected views (include front view, top view, and side view) are displayed according to the values (see Fig. 16(b)). By the way, the base support is shown in Fig. 16(c), and the X, Y, and Z respectively represent the rotating axes of the object, when rotating angles are inputted by moving the scroll bars individually, for instance, X=96, Y=−18, and Z=0 in Fig. 16(d), and the three projected views are shown, as illustrated in Fig. 16(d).

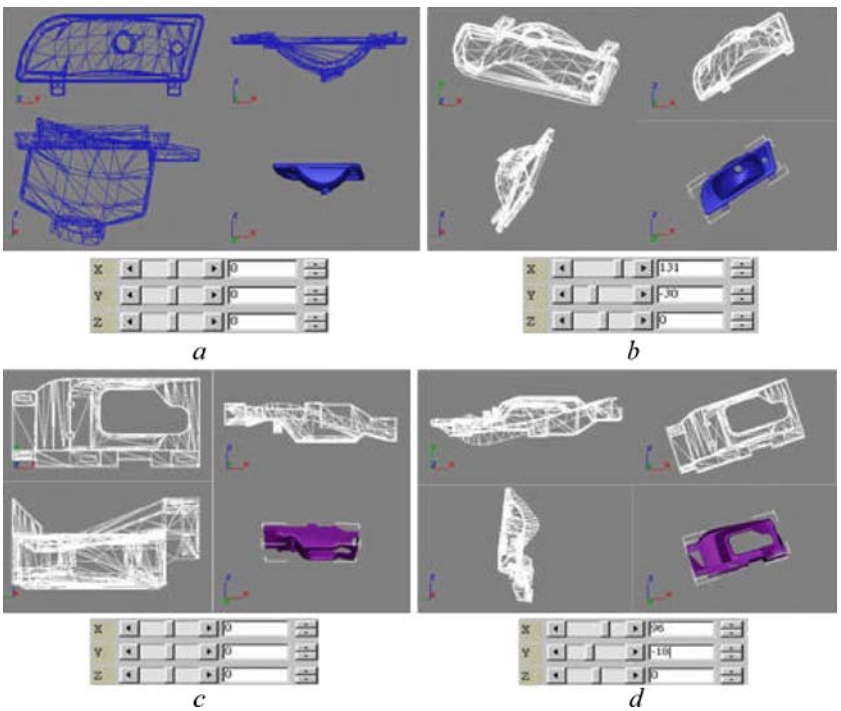


Fig. 16 The projected views by assigning rotation angles of 3D part for reflector and base suppor

5 Conclusion

With the robust growth of information technology, the era of "knowledge economy" has long since started. And the collaborative design technologies in different areas is the inevitable direction of the product development in the 21th century, it is also an important safeguard for improving the capability of competition of enterprise in the global market. According to the actual requirements of the collaborative design system, this research brings forward an architecture based on the Web, namely the architecture is composed of the Web browser, Web collaborative server and shared database, forming a three-layer B/S framework.

The research develops product information framework by applying a web-based 3D virtual technology that provides different ways for visualization, navigation through, and interaction with the product data within the underlying relational database over networks. It facilitates also information exchange between different users, either construction engineer, or manufacturer. Other users such as customers, suppliers, and managers may evaluate the product and quickly give feedback. This communication mechanism may compress the timing for product development and increases the additional-value of the PDM system. Furthermore, the platform also provides the mode of dynamic interactions between a user and the 3D models in the scene. Via the web, a user can examine the scene, walk around within it and interact with it simply.

As described above, the framework can thoroughly develop the possibility of product data sharing and convenience, rapidly understanding product information, using simulation to reduce the number of costly physical prototypes, facilitating communication so less time is spent in face to face meetings, it has the potential to save time and money. It can also be easily adapted for other human-machine interaction environment such as multimedia-based learning and training, computer supported cooperate work, virtual and augment reality applications.

References

1. CIMdata (1997) Product data management: the definition, an introduction to concepts benefits, and terminology, 4th edn. CIMdata Inc, Ann Arbor, MI
2. Rezayat M (2000) The enterprise-web portal for life-cycle support. Comput Aided Des 32:85–96
3. Bender M, Klein R, Disch A, Ebert A (2000) A functional framework for web-based information visualization systems. IEEE Tans Visual Comput Graph 6(1):8–23
4. Hendin O, John N, Shochet O (1997) Medical volume rendering over the WWW using VRML and JAVA. Proc MMVR
5. Sowizral HA, Deering MF (1999) The java 3D API and virtual reality. IEEE Comput Graph Appl 19(3):12–15
6. The ecma international, standards@internet speed, available from http://www.ecma-international.org, Site visited in July 2004
7. Krueger M (1991) Artificial reality II. Addison-Wesley, Reading, MA
8. Chen YM, Tsao TH (1998) A structure methodology for implementing engineering data managemet. Robot Comput Integr Manuf 14(3):275–296
9. Miller E (1998) PDM in the forefront. Comput Aided Eng 17 (3):30–42
10. Lebovitz PP (1997) Design data management for the small engineering shop. Mach Des, pp 20–26
11. Peltonen H, Pitkanen O, Sulonen R (1996) Process-based view of product data management. Comput Ind 31(2):195–203
12. Vinoid B, Gevins J, Baudin C, Mabogunje A, Toye G, Leifer L (1992) An experimental study of design information reuse. Proc 4th International Conference on Design Theory and Methodology. ASME, 13–16 September, Scotts-dale, AZ, pp 141–147
13. Kvan T, Candy L (2000) Designing collaborative environment for strategic knowledge in design. Knowl-based Syst 9(2):429–438
14. Wood WH III, Agogino AM (1996) Case-based conceptual design information server for concurrent engineering. Comput Aided Des 28(5):361–369
15. Dolenc A, Makela I (1997) Text analysis for constructing design representations. Artif Intell Eng 11(1):65–75
16. Saad M, Maher ML (1996) Shared understanding in computer-supported collaborative design. Comput Aided Des 28(1):183–192
17. Ginsburg M, Kambil A (1999) A web-based knowledge management support system for document collections. Proc 32nd Hawaii Conference on system Sciences, pp 1–10
18. Sun Developer Network-Java 3D API, available from: http://java.sun.com/products/java-media/3D/>, Site visited in July 2004