Research on 3D-Model Reconstruction and Coordinate-Picking Technology Based on CsGL

Jiang Xu[1],Chen Shizhi[2],Liang Xu[3],Liu Caihui[4]

[1]Beihang University, Beijing, China

[2]Beihang University, Beijing, China

[3]Beihang University, Beijing, China

[4]Beihang University, Beijing, China

Keywords: on-line testing, 3D model reconstruction, Picking-up algorithm CsGL, STL model

Abstract.--3D-model reconstruction and picking-up algorithm for testing points are key technologies for on-line testing. In this paper, based on .NET platform, 3D-model reconstruction for workpiece in STL file format and related transform operation are achieved by the third-party software CsGL. Besides, ray picking-up algorithm for arbitrary point coordinate on the workpiece surface is also explored. Compared to the former, he method for model rendering, the efficiency and accuracy of the picking-up algorithm are all improved in certain degree.

Introduction

Nowadays, advanced manufacturing technology is one of the most important embodiments for comprehensive national strength. On-line testing and compensation technique, which combines processing and testing together well, provide numerous measuring information to manufacturing process. This technology can shorten auxiliary time, improve machining accuracy and reduce production expense. 3D model reconstruction and testing information picking for the workpiece is the critical step to realize on-line testing technology.

There are many recent researches about 3D model reconstruction and testing information picking method, which are proposed in related papers [1-6]. However, most traditional implements are based on MFC Framework, which results in laborious test method, low efficiency and poor accuracy. In this paper, we rebuild 3D model for workpiece model in STL file format using C# language and combining with CsGL, based on VS 2013 platform. Spatial alternation such as rotation, translation and scaling of the 3D model is well performed and circumstance setup of the model is controlled. Besides, a quick coordinate picking-up algorithm for testing point is proposed in this paper. It has significance to a certain to the research on the on-line testing technology during workpiece manufacturing.

3D Model Reconstruction

A. STL Model File Parsing

STL file is a kind of file format applied for triangle mesh representation in graphics application system. It is easily understandable and wildly used. Almost all kinds of 3D graphics processing software can output STL file format. Only closed surface can be expressed in STL file. In STL file with ASCII code, every triangular patch is constituted by 7 rows of code, which is shown as follows:

facet normal $v_x \quad v_y \quad v_z$

outer loop

vertex $p_{1x} \quad p_{1y} \quad p_{1z}$

vertex $p_{2x} \quad p_{2y} \quad p_{2z}$

vertex $p_{3x} \quad p_{3y} \quad p_{3z}$

endloop

endfacet

In this paper, we designed a method called ***GetSTLModelData*** in the custom class STLModel to parse STL files and defined the 3D arrays “stldata[FaceCont,4,3]” used to store related data of model. The parse process is shown in Fig. 1.

Read a line of text

ReadLine()

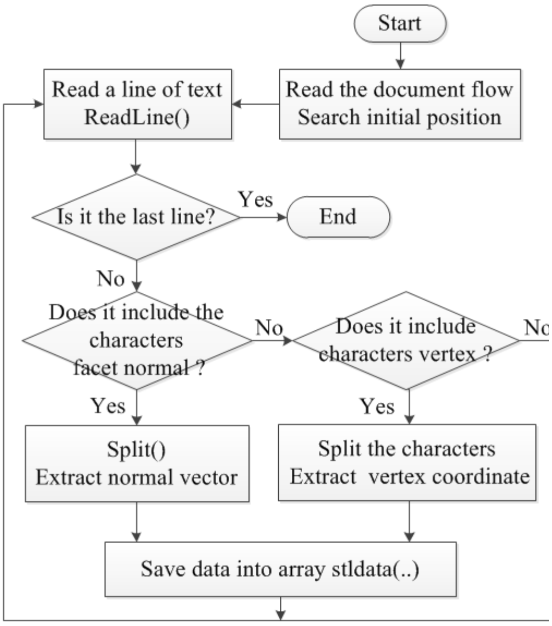

Fig. 1 STL file parse process

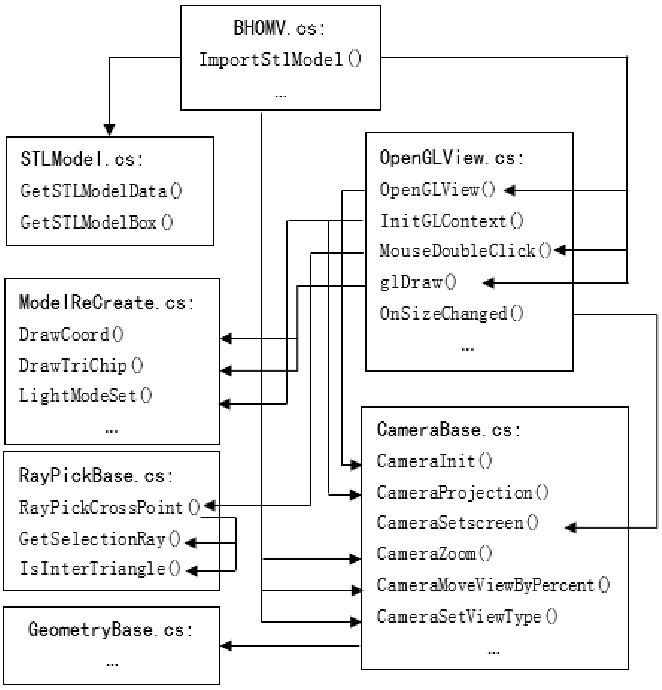

Fig. 2 Program structure

B. Rendering 3D Model with CsGL

OpenGL is a professional cross-language, multi-platform graphic application programming interface for rendering 3D vector graphic models. It is an underlying graphic library with powerful functions and convenient invocation. The implementation technology for OpenGL with C++ language in recent research is proven completely. However, there were not standards and rules for the development of OpenGL with C#. The third-party software CsGL is the OpenGL on the platform of *.NET Framework*, which is operated via two dynamic link libraries, *csgl.dll* and *csgl_native.dll*.

To render 3D model and achieve interactive operation, the program structure design is shown in Fig.2.

Import the STL model menu to BHOMV.cs class in the main interface and trigger the event of ImportStlModel(). Then call the GetSTLModelData() function in the class of STLModel.cs to get model data. Next, we set up the OpenGL graphic environment to render the model. The steps for rendering 3D model with CsGL are as follows:

i) Set up the graphic environment

Create a new class belonging to OpenGLControl and rename it OpenGLView. Then create a new instance under this class.

ii) Initialize the graphic environment

The initialization for the graphic environment includes setups for cache, illumination, material and etc., which can be operated by the function overriding InitGLContext().

iii) Render the model

The model rendering can be operated by the funcation overriding glDraw(). In this paper, axes are drew with DrawCoord() function in the class of ModelReCreat.cs and the model of the workpiece are rendered with DrawTriChip function.

iv) Change the form size

When the form size is changed, redraw of the form can be achieved by rewriting the class with the function OnSizeChanged() in OpenGLControl.

v) Refresh the screen

To create a timer object GLUpdataTimer in constructor function can refresh the screen in the callback function.

C. Model Transformation

To display the model in the window, the process includes view transformation, projection transformation and viewport transformation.

i) View transformation

OpenGL renders the model in the world coordinate system. The eye coordinate system is decided by the viewing position and direction, which is a right-handed coordinate system. Without any view transformation the eye coordinate system is the same with the world coordinate system, as shown in Fig. 3. View transformation is to change the position relationship between the eye and world coordinate system. In the self-defined function CameraInit(), we call the function gluLookAt() in OpenGL to define the viewing position and direction.

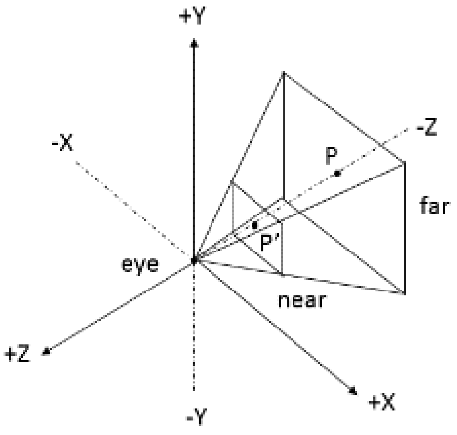

Fig.3 Eye coordinate system

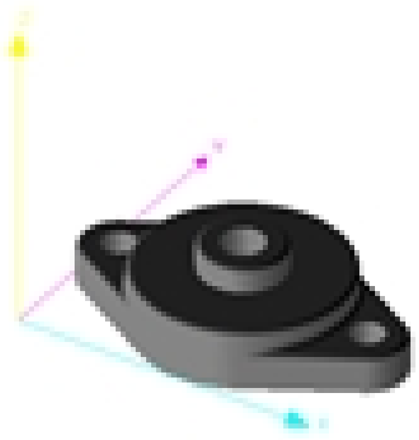

Fig. 4 Model display

ii) Projection transformation

Projection transformation is used to define a view-frustum. In this paper, rectangular projection is used to define the view-frustum of the cuboid. Under the self-defined function

CameraProjection(), function gl0rtho() of OpenGL is called to set up the view frustum.

iii) Viewport transformation

Viewport is the region for graphic displaying in the client area window. Viewport transformation can map the projective plane of view-frustum to the viewport.

The operation of model includes translation, rotation, scaling, perspectives switching and etc. In the class of CameraBase.cs, the function CameraMoveViewByPercent() is defined to operate model translation, which can move the model up, down, right and left or according to the width and height of the viewport; function CameraZoom() is used to scale the model; function CameraZooAll() is used to optimize the display size of the model after using function GetSTLModelBox() to gain the minimum bounding size of the model; function CameraRotate() is used to rotate the model around the horizontal axis or vertical axis, which is based on the rotation matrix generated by the function CreateRotateMatrix() in class GeometryBase.cs; function CameraSetViewTypee() can switch perspective of the model and display six orientation views as Tom, Bottom, Front, Back, Left, Right and four axonometric views as SW, SE, NW, NE.

D. Model Reconstruction

In this paper we read a model in STL file format and reconstruct it. The model is displayed in Fig. 4 after appropriate translation, rotation and scaling.

Coordinate Picking-up Algorithm

A. Picking-up Principle

In OpenGL, the function glPickMatrix() based on former principle is with low efficiency and poor accuracy. If we need the space coordinate of arbitrary point on the workpiece, the algorithm provided by OpenGL cannot meet requirements. The flow of the picking-up algorithm proposed in this paper is shown Fig. 5. The principle is that, the picking-up operation of the point on the STL model surface can be seen as an intersection of the triangular patch of the STL model and the observation line though the cursor position.

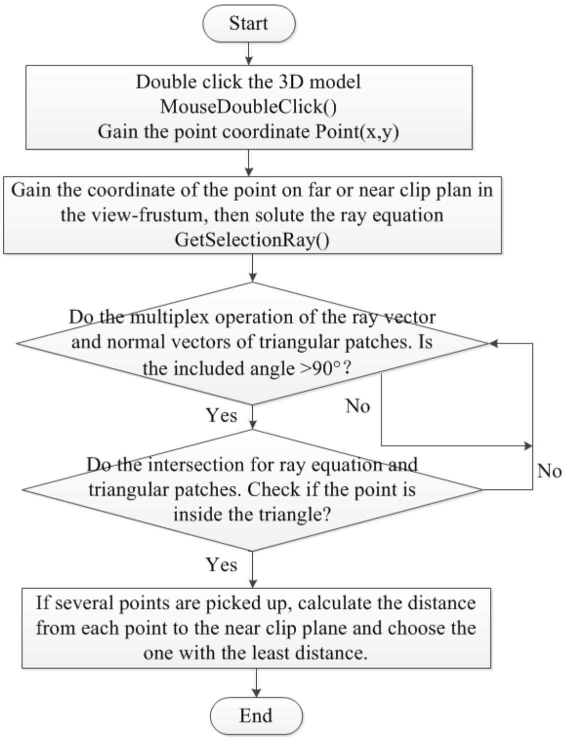
Start
Double click the 3D model
MouseDoubleClick()
Gain the point coordinate Point(x,y)
Gain the coordinate of the point on far or near clip plan in the view-frustum, then solute the ray equation
GetSelectionRay()
Do the multiplex operation of the ray vector and normal vectors of triangular patches. Is the included angle >90°?
No
Yes
Do the intersection for ray equation and triangular patches. Check if the point is inside the triangle?
No
Yes
If several points are picked up, calculate the distance from each point to the near clip plane and choose the one with the least distance.
End

Fig. 5 Picking-up algorithm flow

B. The Picking-up Algorithm Flow

i) Obtain the picking-up ray

Click the screen to gain the 2D coordinate of the selected point $p_{2d}\left(x,y\right)$, which corresponds to p’ on near clip plane and p on far clip plane of the view-frustum. By means of the reverse transformation gluUnProject() in OpenGL the world coordinate of p and p’ can be obtained and then a straight line through p and p’ in the space is determined. The equation of this straight line can be acquired.

ii) Screen patches by vector dot products

Only those patches whose normal direction is opposite to the normal direction of observation line is possibly in demand. To do the dot product operation for the normal vectors of the triangular patch and observation line, only those results less than 0 match the demanded condition.

iii) Calculate intersections of the space line and the triangular patch

To do the intersection operation for the picking-up ray and the triangular patches in demanded by turn, the coordinate of intersection point is gained as shown in Fig. 6.

The point normal form equation of the straight line where the picking-up ray locates can be performed as

$$\begin{aligned} x &= p'[0] + RayVt[0]*t \\ y &= p'[1] + RayVt[1]*t \\ z &= p'[2] + RayVt[2]*t \end{aligned} \tag{1}$$

The point normal form equation of the plane where the triangular patch locates can be performed as

$$TVt[0]*(x - PA[0]) + TVt[1]*(x - PA[1]) + TV[2]*(x - PA[2]) = 0 \tag{2}$$

By the two equations we can calculate that

$$t = \frac{(PA[0] - p'[0])*TVt[0] + (PA[1] - p'[1])*TVt[1] + (PA[2] - p'[2])*TVt[2]}{TVt[0]*RayVt[0] + TVt[1]*RayVt[1] + TVt[2]*RayVt[2]} \tag{3}$$

And then we can obtain the intersection point coordinate *CrossPoint(x,y,z)*.

iv) The position relationship between the point and triangle.

The intersection point calculated must be inside the triangular patch or on its sideline. The mainstream methods to judge include comparing the area and vector cross product. In this paper we apply the algorithm of comparing the area. If

$$S_{OAB} + S_{OAC} + S_{OBC} \le S_{ABC} \tag{4}$$

is true, then the intersection point O is inside the triangle or on its sideline, where we calculate the triangle area by Heron’s formula.

$$S = \sqrt{p(p-a)(p-b)(p-c)} \tag{5}$$

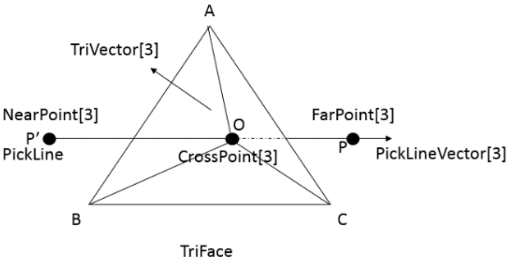

Fig. 6 Intersection operation for the picking-up ray and the triangular patch

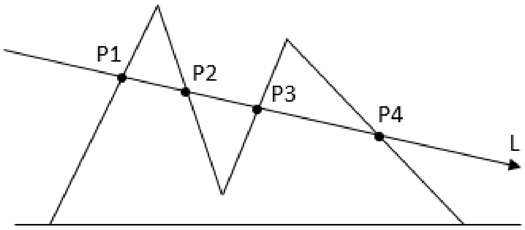

Fig. 7 Multi-point crossover

v) Choose the optimum solution point

If several points in demand is obtained by intersection operations, such as the point p1 and p3 as shown in Fig. 7, we can compare the distance d from these points to the near clip plane and choose the optimum point with the minimum distance.

C. Picking-up Result

In this paper we read a file of Sphere workpiece. After appropriate translation, rotation and scaling, double click the measuring point on the model surface and the information window of the picking-up point pops up. The information includes 3D coordinate, normal vector and serial number of its triangular patch. Users can fill the the test numver and test point serial number and then choose to save or abandon.

Summary

In this paper, based on VS2013 development platform, using C# programming language and the third-party software CsGL, we implement reading and displaying of the workpiece model in STL file format, the setup of the rendering environment and multiple conversion operation of the model. A quick picking-up algorithm for the 3D coordinate information of the testing point on the workpiece surface is also proposed. Besides, via experimental verification, the proposed algorithm is with high efficiency and good accuracy, which is of great significance for the research on on-line test technology.

Reference

[1] Chen Yujun, Zhang Qi, OpenGL simulation and 3D model harvest based on MFC, J. Modern Design Technology,2004.

[2] Xia Ruixue, Chen Lin. The measurement information extraction methods oriented CAD model, J. Journal of Instruments and Meters,2011.

[3] Wang Jian, Lu Guodong. The pick-up method of graphic objects in 3D scene, J. Computer Technology Application,2004.

[4] Zhu Mingliang, Dong Bing. Three-dimensional scene pick-up algorithm based on viewport space, J. Journal of Engineering Graphics,2008.

[5] Zhu Jigui, Zhang Guoquan. A new pick-up method based on OpenGL redraw type object, J. Micro Computer and Application,2005.

[6] Qian Bo, Zhang Lichao, Surface area recursive pick-up algorithm based on STL model, J. Journal of Huazhong University of Science and Technology,2008.

[7] Zhong Sigao, Gao Zhi, The 3D model browsing measurement and management based on OpenGL, J. Computer Engineering,2013.

[8] Lee K H, Woo H. Direct integration of reverse engineering and rapid prototyping, J. Computers

& Industrial Engineering, 2000.
[9] Wang Qinghui, Li Jingrong, CAD application development explanation, M. Electronic Industry Press,2012.