

Rendering System Design of S-CAD

SEAH BOON KEONG & ABDUL RAHMAN ABDULLAH

ABSTRACT

In this paper, the authors describe the S-CAD Rendering System that the authors have developed. S-CAD is a CAD system specifically tailored to the needs of the student. This system allows interactive editing and positioning of objects. The basic approach that S-CAD takes is to provide both graphical and dialog editing tools simultaneously. Numerous CAD functions are provided

The Rendering System that the authors have developed handles the rendering of a model. This system is designed to exist concurrently with the S-CAD Editor. Hence, any invoking of rendering function on S-CAD Editor will automatically communicate with the S-CAD Rendering System. Here the authors have designed the system architecture for the communication between the S-CAD Editor and S-CAD Rendering System, and data communications. The link of this is through COM (Component Object Model).

Besides rendering, this system also allows user to manipulate the model interactively. User can rotate, scale or translate the model interactively by using mouse. It can also export the model as VRML vi.O or V2.0 format or as an image format such as JPEG, and BMP. This system also incorporates numerous added junctions such as texture mapping, background image, transparency and many more.

Keywords: Rendering, System Design, Modelling, VRML, COM

ABSTRAK

Dalam kertas ini penulis membincangkan pembangunan Sistem Pewarnaan Bagi S-CAD. s-CAD ialah satu sistem rekabentuk berbantuan komputer (RBK) yang dibangunkan khusus untuk keperluan pelajar. Sistem ini membolehkan penyuntingan dan perletakan objek secara interaktif. Pendekatan asas yang diambil oleh S-CAD ialah dengan menyediakan alat suntingan dialog dan grafik secara serentak. Pelbagai fungsi RBK disediakan.

Sistem Pewarnaan yang dibangunkan menyediakan kemudahan mewarnakan sesuatu model. Sistem direkabentuk supaya wujud secara serentak dengan Penyunting S-CAD. Dengan demikian, sebarang usaha penggunaan

fungsi dalam Penyunting S-CAD secara otomatis akan dihubungkan kepada Sistem Pewarnaan S-CAD. Dalam hal ini, penulis telah merencanakan sistem bagi komunikasi dan komunikasi data antara Penyunting S-CAD dan Sistem Pewarnaan S-CAD. Ini dilakukan melalui Model Objek Komponen.

Di samping pewarnaan, sistem ini juga membolehkan pengguna memanipulasi model secara interaktif. Pengguna boleh memutar, menskala dan menganjak model secara interaktif menggunakan tetikus. Model juga boleh diekspor dengan format VRML V1.0 atau V2.0 atau sebagai format imej seperti JPEG atau BMP. Sistem ini juga memasukkan pelbagai fungsi tambahan seperti pemetaan tekstur, imej latar belakang, imej lutsinar dan sebagainya.

INTRODUCTION

Visualization of an object or model is an important feature of a CAD system. It gives user the ability to present the model in a more effective way. In view of this, a rendering application is integrated into S-CAD system. This paper will first describe the goals in designing this system. It is then followed by a brief overview of this system. Next a brief mathematical description of the rendering techniques is given. Then it is followed by a description of RENSYS system user interface and software architecture. In this section, some of the important formats supported are also discussed. Later in this paper, some examples of model created in S-CAD editor and rendered in RENSYS system are illustrated. Lastly, a summary of this paper is given.

DESIGN GOALS

The main goal of RENSYS system is to support S-CAD editor in rendering a model. This will undoubtedly provide a more realistic method of representing a model. The system must also provide interactive editing methods which will enable a user to visualize a model in various different angles. In order to make RENSYS more applicable, important data formats for model exportation must be supported. Data format especially for Internet usage such as VRML (Pesce 1995) must be supported. Undoubtedly, this will attract user to use S-CAD editor. Besides all the above goals, another important area is the method in developing this application. As this system is to be used very often with each rendering command applied in the S-CAD editor section, RENSYS must not be generated or executed multiple times in doing this rendering task but rather this system should stay transparent. One of the main advantages using this technique is that it can prevent the operating system from crashing when RENSYS is to be invoked many times.

SYSTEM OVERVIEW

The main purpose of this stand alone rendering application is to further support the S-CAD editor system. It uses OpenGL (Fosner 1997, Akeley et al. 1992, Woo et al. 1997, Wernecke 1994a; Wernecke 1994b; Richard & Sweet 1996) as the graphics platform to render the CAD data. OpenGL supports a wide range of 3D graphic functions. Presently, this system provides important functions in the area of editing, rendering, and exportation of the documents. These functions are classified important as it signifies the ability of this application to further manipulate and export the data out into other useful formats, which can be integrated into other well-established applications. In the area of editing, real time translating, rotation, and scaling are implemented. The interaction at this level uses mouse to control the relevant parameters of these editing functions. It can be done by just dragging the mouse to the necessary locations and at the same time visualize its new generated view. In the area of rendering, flat surface, smooth surface and wire frames are supported. Since the main idea of having this application, is to render the model of s-CAD, supporting S-CAD object and colour representations is of great importance. Object in this system uses NURBS (Piegl 1991) and polygon to represent. In future we can easily incorporate more OpenGL attributes in order to make this system more attractive.

In the area of exporting data, one of the major features of this rendering system is the support of VRML V1.0 and V2.0 (Pesce 1995). Both of these formats support 3D data. In the advent of advancement in the Internet, the enhanced support of this format will further strengthen the use of s-CAD not only in the area of CAD but also in graphics presentation. Besides this format, image file formats such as JPEG, and BMP are supported.

Another important area that this rendering system is being implemented is in the controlling section. This system is to coexist with S-CAD editor and thus the linkage of these two systems must be implemented. In the area of rendering in S-CAD editor, this function will be used very frequently and hence a number of this rendering system will be generated every time the model is rendered. Hence, in order to overcome this, COM (Component Object Model) or OLE (Object Linking Embedded) (William & Chris 1995; Chappell 1996; Fronckowiak 1997; Microsoft 1997; Brockschmidt 1995) technique of system coexisting is used.

BRIEF OVERVIEW OF RENDERING TECHNIQUE

In this section, we present the basic geometric concept of rendering a model.

SHADING

A shaded model or a rendered model will give a user a better viewing of the model as only the visible area towards the viewpoint are rendered or shaded. A shaded model is illuminated by three different kinds of light: ambient, specular and diffuse. The intensity of each point on the surface of the model is determined as follows:

$$I_p = I_d + I_a + I_s$$

where I_d is the intensity due to diffuse reflection, I_a is the intensity due to ambient light, I_s is the intensity due to specular reflection, and finally I is the resultant intensity of the three light components. Detail explanation regarding this can be found in any optics reference text (Ditchburn 1976; Eugene & Zajac 1979; Jenkins & Harvey 1976).

A. Diffuse Reflection

In RENSYS, all objects do not emit light. These objects will absorb light coming from one particular direction and reflect part of it. The surface of the object scatters the light equally in all direction, and it does not depend on the viewer's position. However, the diffuse light depends on the direction of the surface with regards to the direction of the source light. Hence, the object will be brighter if the light source is pointed directly at the surface. The diffuse reflected light formula is given by Lambert's Law:

$$I_d = I_s k_d \cos \theta \quad 0 \leq \theta \leq \pi/2$$

where I_s is the intensity of the light sources, θ is the angle surface normal and the direction of the light source, and finally k_d , which is a constant value which indicates how much light is diffuse from the surface. This value depends on the material of the object. The k_d value varies from 0 to 1.

B. Ambient Light

Ambient light is light that does not come from any particular direction. It is the result of multiple reflections around the scene and objects. Hence, it is a light with uniform brightness and thus its intensity is constant. By combining the diffuse and ambient light, we can write the new light formula out as:

$$I = I_a k_a + I_s k_d \cos \theta$$

where I_a is the intensity of the ambient light, and k_a is a constant which indicates how much ambient light is reflected from the surface.

C. Specular Reflection

like diffuse reflection, specular reflection is directional, but is reflected sharply and in a particular direction. However, the sharp reflection can be applied only to glossy material. Bui-Tuong Phong (Phong 1975) modelled this behaviour and developed a formula to calculate the specular reflection that can be applied onto all material. The formula is given below:

$$I_s = I_l k_s \cos^n \varphi \quad 0 \leq \varphi \leq \pi/2$$

where φ is the angle between the viewing vector and the reflection vector, n is a value that determines the highlight area, k_s is a constant value of the specular reflection. L is the light source intensity and I_s is the resultant specular intensity. We can achieve a glossy surface, by applying a large n value.

By combining the three light components, we can find the intensity of the material as follow:

$$I = I_a k_a + I_l [k_d \cos \theta + k_s \cos^n \varphi] \quad [1.1]$$

SHADING SURFACE

In this section, we will discuss three of the basic techniques used in shading of a surface.

A. Constant Shading or Flat Shading

This is the simplest technique to shade a surface. For example, if each polygon lies on a particular plane, we can then compute each polygon and obtained one normal vector. We can then apply this normal vector to [1.1] and calculate the intensity of the light on the whole polygon area. The angle φ can be calculated using the dot product between the normal vector and direction vector from the light source. In Figure 1.3, a model using flat shading is illustrated.

B. Smooth Shading

In order to overcome the unsmooth shading, two techniques are used in particular to achieve this: Gouraud and Phong techniques.

a. Gouraud Shading

Gouraud (Gouraud 1971) shading computes the average of the surface normals and produces a new vertex normal. This normal vertex can be used

in [1.1] to compute the vertex intensities. After that we then can use it to compute the shade of each polygon by linear interpolation of vertex intensities.

b. Phong Shading

This techniques, interpolates surface normal vectors across polygons and uses these interpolated vectors into [1.1] and produces the intensities. In Gouraud (Gouraud 1971) shading, we use the averaged normal vector to calculate the vertex intensities and then interpolate these vertex intensities to produce each intensity on the surface. Whereby in Phong (Phong 1975) shading, the normal vector of each surface vertex are interpolated to produce the interpolated normal vector which are then used to calculate the intensity.

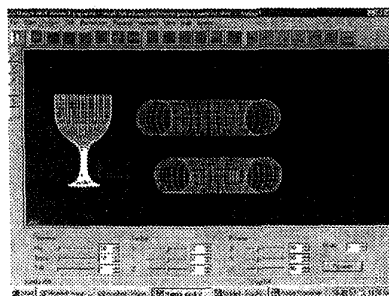


Figure 1.1: Model created in S-CAD editor. This model will be rendered in RENYS when user entered the command *Render* in S-CAD editor.

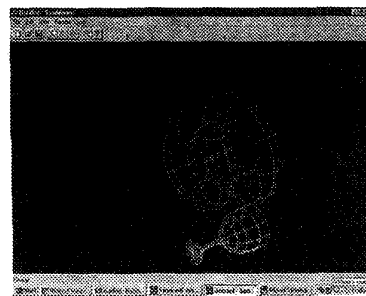


Figure 1.2: The model in wireframe mode.

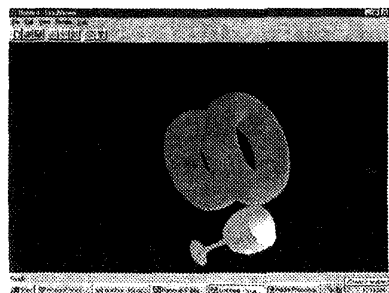


Figure 1.3: The model in flat shading mode.

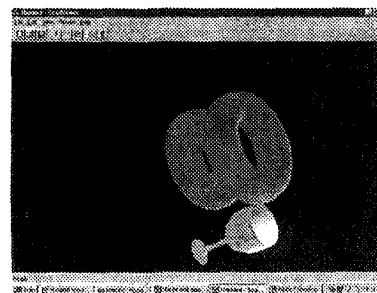


Figure 1.4: The model in smooth shade mode. The difference between Figure 1.3 and this figure can be clearly seen in the shading of the surface.

RENSYS DESIGN

In this section, the RENSYS system is described in detail. Firstly, we will describe the RENSYS system user interface. Then the design of the RENSYS system is discussed. It is then followed by a discussion on the geometric representation and data communications.

THE RENSYS USER INTERFACE

The RENSYS window is divided into four different areas: menu bar, display area, status bar and tool bar. The menu bar supports the full functionality of RENSYS. Tool bar provides basic functions that are to be accessed most frequently.

The manipulation control section can be grouped into three categories. All these categories performed directly onto the display area. The first is the viewing rotation control. This control uses mouse to control *it* and hence it does not require any additional control widgets outside the display area. Rotation of the view is performed by dragging the mouse cursor to a new location in the display area. The second is the scaling control. It uses the same mouse technique applied in the rotation control. We can zoom in or zoom out of a viewing display. The third control is the translation control. It too uses the mouse technique to position the view.

In the rendering operations, three modes of rendering are offered. The wire frames mode as implied will display the lines representing the edges of every polygon or polygon mesh. The flat surface mode, will display the surface of the model using flat shading technique. While in the smooth surface mode, the surface of the model will be rendered using Gouraud (Gouraud 1971) technique. These modes are illustrated in the previous section.

In the File section, export of data into various formats is provided. This will enable user to use the model created in S-CAD to be use in other 3-D applications. The formats offered are 3D and 2D file formats. In the 3D file formats, VRML format is offered in the RENSYS system while DXF file format is in the S-CAD editor. In the 2D file formats, well established image formats such as BMP, GIF and JPEG are provided.

RENSYS SOFTWARE ARCHITECTURE

This application is developed on the Window95 platform. It is implemented in Visual C++ based on the Windows Single Document Interface (SDI) and the Microsoft Foundation Classes (MFC) (Toth 1996; Holzner 1996). In order to integrate with the S-CAD editor this application is developed using OLE technique.

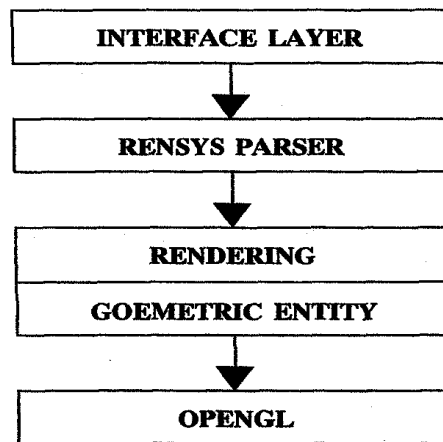


Figure 1.5: The basic software architecture of RENSYS.

In Figure 1.5, the basic software architecture of RENSYS is illustrated. In order to link S-CAD editor and the RENSYS application, a new format which is transparent to the user is developed. S-CAD editor will generate the model into this format before RENSYS application is invoked. This section is the Interface Layer. In the parser section shown in Figure 1.5, this format will be parsed and an internal data structure representing this model will be constructed.

The following section is the display or rendering section. This section will render the model in the display area. The rendering layer consists of graphics routines that handle the basic entity of a model, rendering mode and the viewing transformation.

In RENSYS system, the entities supported are: close and open surface, close and open curve, and finally the polygon entity. Interpretation of these entities are handled by graphics routines developed on top of OpenGL 3D graphics API (Application Programmers Interface). Presently, a few modes in the OpenGL are used. The attributes in OpenGL such as lighting, material, and various different kinds of modes can be easily incorporated into this graphics routines section.

GEOMETRIC REPRESENTATION IN RENSYS

Since RENSYS system is to support S-CAD editor, all geometric data in S-CAD editor must be represented correctly in RENSYS system. However, since RENSYS is developed on top of OpenGL, we have to develop a new set of geometric entity to represent the data in S-CAD editor. In Figure 1.6, we illustrate the steps to process the geometric data conversion.

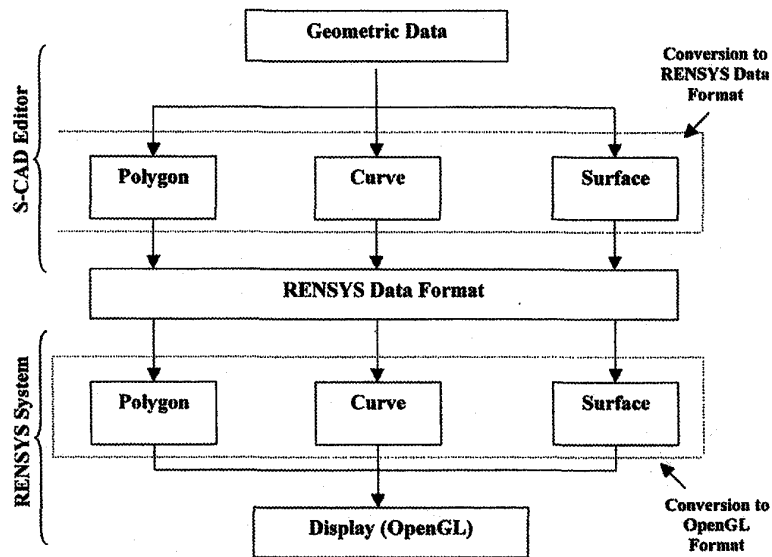


Figure 1.6: Geometric Conversion Architecture.

As shown in Figure 1.6, geometric data is converted into RENSYS format before RENSYS application is invoke. In the curve geometry, data needs to be organized into open or close loop type, while in the surface geometry, data is converted into four types of options: *open m direction and open n direction*, *open m direction and close n direction*, *close m direction and open n direction*, and finally *close m direction and close n direction*.

In RENSYS, this data will be converted into internal data entity format which will use OpenGL as the basic 3D rendering library. In the surface generation, NURBS data structured is used. In order to calculate the four options of surface generation, the knot vector of these m and n directions must be determined. In Figure 1.7,1.8,1.9, and 1.10, shows the four different surface generation options drawn in RENSYS system.

DATA COMMUNICATIONS

In this section, we will discuss two types of data communications that RENSYS handles. One data format is to handle communications between RENSYS system and S-CAD editor while the other is to enable communications in established formats between RENSYS system and other applications. We will first describe the established file formats handle by RENSYS system and then followed by the RENSYS Internal File Format.

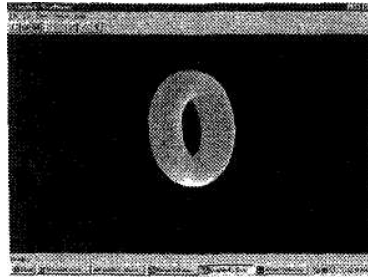


Figure 1.7: A surface generated with close m direction and close n direction.

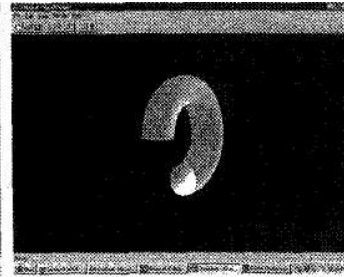


Figure 1.8: A surface generated with close m direction and open n direction.

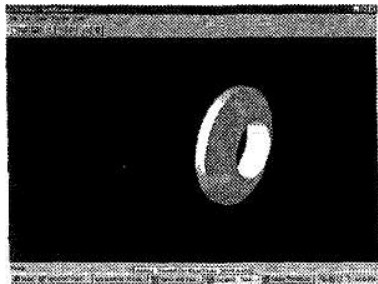


Figure 1.9: A surface generated with open m direction and close n direction.



Figure 1.10: A surface generated with open m and open n direction.

A. Established File Format

The Virtual Reality Modelling Language (Pesce 1995) or known as VRML, is evolving as a standard data format for the storage and interchange of 3D models in the Internet. Currently, this format is being used to represent 3D models in the Web (Berners-Lee et al. 1994) browser. The VRML plug-in basically comes together with the Web browser. In VRML V1.0 animation is not supported but in the VRML V2.0 animation is supported.

Hence the support of this format will encourage user to use S-CAD to develop a model and publish it into the Internet. Moreover, since the browser exist also in other platforms, the VRML format is thus able to be transferred from one platform to the other platform.

Besides the 3D format, 2D image formats are also supported such as Joint Photographic Experts Group (JPEG) (Peenebaker & Mitchell 1992; Wallace 1991; Cook 1990) which is a standard for the compression of single images, and Bitmaps (BMP) graphic format which is device-independent bitmap (Dm) format that allows Microsoft Windows to display the bitmap on any type of display device.

B. RENSYS Internal File Format

This internal file format is used to interact between S-CAD editor and RENSYS system. Before invoking RENSYS system, S-CAD editor will generate this data format. The format being use consist of the following:

- Geometric Flag - 1 for polygon, 2 for curve, 4 for surface.
- Geometric Parameter - such as total vertices for polygon and curve or totalm and n direction vertices
- Close or open loop flag for surface : 1 - for open, 2 - for close
- Geometric Colour - 24 bits colours information - red, green, and blue colour data.
- Data of vertices to be followed - x, y, z

The data format above has sufficient information to represent the geometric data in S-CAD editor to be use in RENSYS. For example, the following is a model which consist of a surface and a polygon:

EXAMPLES OF RENDERED MODELS

In this section, we illustrate few models created in S-CAD editor and rendered using RENSYS system. These rendered models can be exported as image file formats such as JPEG, and BMP or as 3D models such as VRML V1.0 or VRML 2.0.

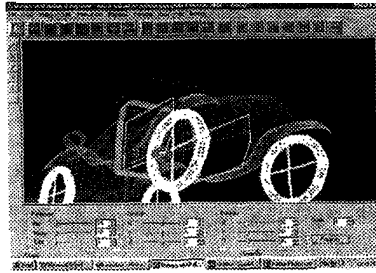


Figure 1.11: A car created in S-CAD Editor.

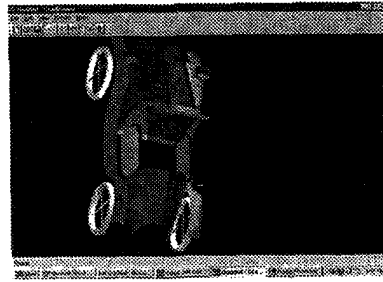


Figure 1.12: The rendered image of the car.

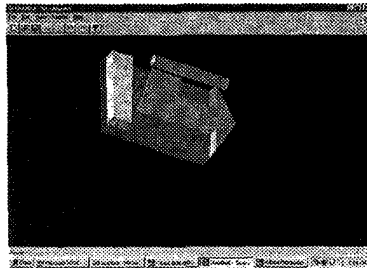


Figure 1.14: Model exported into VRML format and displayed in a browser.

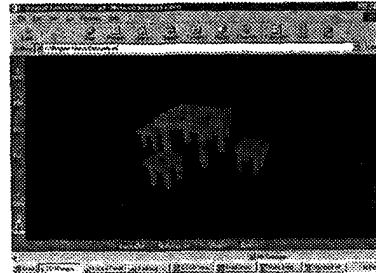


Figure 1.13: The rendered image of a portable iron. This iron is created in S-CAD Editor.

SUMMARY

In this paper the authors discussed the RENSYS application that the authors have developed. This system is an OLE based application which exist along with the s-CAD Editor. The main aim of having this system as OLE is to avoid duplication of this application every time user invoke the render function in S-CAD Editor. With RENSYS system, model can be rendered, manipulated and exported into VRML format, or 2D graphics format such as JPEG, and BMP. Along with the development the authors also have developed the system architecture, the geometric representation, manipulation and the file format

for data communication with the S-CAD Editor. The system architecture that the authors have developed consists of four layers: interface layer, parsing layer, rendering layer, and finally OpenGL layer. The interface layer that the authors have designed, is to define the meaning of the necessary flag for the attributes, data, and close/open loop flag. This layer provides the communication between S-CAD Editor and RENSYS system. The parsing layer that the authors have developed is to parse the interface layer, and later to be use by the rendering layer. Presently, the authors only read from the interface layer. However if there is a need to have two way communication this layer can be altered. In the rendering layer that the authors have developed, the authors divided it into three sections: the geometry section, the rendering mode section and the viewing transformation section. As the representation in S-CAD Editor and RENSYS in OpenGL are different, the authors have designed the geometry section to handle the conversion of the geometry entity so that the entity drawn in RENSYS still maintain the same structure as in S-CAD Editor. In the viewing transformation section, it permit us to manipulate the camera viewing from various angle. The OpenGL layer, however are provided by the OpenGL system. The rendering layer, will make use of the OpenGL layer to draw in the necessary mode.

Besides the internal file format use in the communication with the S-CAD Editor, the authors also have developed external file format communication such as VRML V1.0 and V2.0. VRML model can be use by any Internet browser to walkthrough a 3D model. Hence, the model can also be treated as a transferable 3D file in the Internet which does not limit itself to the Windows platform but any browser platform that support VRML. The authors hope will further propel the interest of student in designing using S-CAD.

REFERENCES

- Akeley, K.[et al]. 1992. *OpenGL Reference Manual*. Menlo Park: Addison-Wesley.
- Berners-Lee, T. [et al]. 1994. The World-Wide Web. *Communications of the ACM* 37(8): 76 - 82
- Chappell, D. 1996. *Understanding Active X and OLE (Strategic Technological Series)*. Microsoft Press.
- Cook, R. 1990. Image Compression Squeezes to the Fore. *Computer Graphics World*, vol 13(9): p.p. 70-76.
- Ditchburn, R.W. 1976. *Light 12*. London: Academic Press.
- Eugene, H & Zajac, A. 1979. *Optics*. Menlo Park: Addison-Wesley.
- Fosner, R. 1997. *OpenGL Programming for Windows 95 and Window NT*. Menlo Park: Addison-Wesley.
- Gouraud, H. 1971. Continuous Shading of Curved Surfaces. *IEEE Transactions on Computers*, C-20 (6): p.p. 623 - 629.
- Holzner, S. 1996. *Advanced VC++ 4*. [s.l.]:Hungry Minds Inc.
- Jenkins, F.A. & Harvey E.W. 1976. *Fundamentals of Optics 4E*. New York: McGraw-HM.

Microsoft OLE DB 1.1 *Programmers Reference and Software Development Kit*. 1997.[s.l.]: Microsoft Press.

Peenebaker, W.B. & Mitchell, J.L. 1992. *Jpeg: Still Image Data Compression Standard*. New York: Van Nostrand Reinhold.

Pesce, M. 1995. *VRML: Browsing and Building Cyberspace*. [s.l.]: New Riders/Macmillan Publishing.

Phong, Bui Toung. 1975. Illumination for Computer Generated Pictures. *Comm. ACM* 18(8): 311 - 317.

Piegl, L. 1991. On NURBS: a survey. *IEEE Computer Graphics and Applications* 11(1): 55 - 71.

Richard, S.W.J. & Sweet, M. 1996. *Open GL Superbible*. [s.l.]: Waite Group Press.

Toth, V. 1996. *Visual C++ 4 Unleashed*, [si]: SAMS Publishing

Wallace, G. 1991. The JPEG Still Picture Compression Standard, *Comm. ACM* 34(4): 30 - 44.

Wernecke, J. 1994. *Open Inventor Architecture Group. The Inventor Mentor*. Menlo Park: Addison-Wesley.

. 1994. *The Inventor Toolmaker*. Menlo Park: Addison-Wesley.

William H.M.IDL & Chris H. Pappas, 1995. *OLE Wizardy: Programming OLE Applications and Custom Controls Using Wizards*. Osborne: McGraw-Hill.

Woo, Neider, J., Davis, T. & OpenGL Architecture Board. 1997. *OpenGL Programming Guide: The Official Guide to Learning OpenGL*. Menlo Park: Addison-Wesley.

Seah Boon Keong & Abdul Rahman Abdullah Multi Media Synergy Corporation Berhad, Level 3, Incubator 3, Technology Park Malaysia, 57000 Kuala Lumpur, Malaysia. ara@mmsc.com.my.