**VISVESWARAIAH TECHNOLOGICAL UNIVERSITY**

**Belgaum-590014**

**A MINI PROJECT REPORT ON**

**INTERNAL WORKING OF HARDDISK**

**Submitted in partial fulfillment of the academic requirements for the award of the degree of**

**BACHELOR OF ENGINEERING**

**In**

**COMPUTER SCIENCE AND ENGINEERING**

**Submitted By**

**Abdul Azeez. C (1CD09CS001)**

**Under the guidance of**

**Dr. Suresh.L**

**Head Of Department, CSE**

****

**Department of Computer Science and Engineering**

**CAMBRIDGE INSTITUTE OF TECHNOLOGY**

**K.R.PURAM,**

**Bangalore-560036**

**2011-2012**

**CAMBRIDGE INSTITUTE OF TECHNOLOGY**

**K.R.PURAM, Bangalore-560036**

Department of Computer Science and Engineering

****

**CERTIFICATE**

Certified that the project work entitled “INTERNAL WORKING OF A HARDISK” is a bonafide work carried out by Abdul Azeez. C (1CD09CS001). In the partial fulfillment for the award of degree of Bachelor of Engineering in Computer Science & Engineering of the Visveswaraya Technological University, Belgaum during the year 2011-2012. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in this Report. The project report has been approved as it satisfies the academic requirements in respect of Project Work prescribed for the Bachelor of Engineering Degree.

Signature of Internal Guide, HOD Of CSE Dept.,

-----------------------------------

Dr. Suresh. L

External Examiners

Name Signature with date

1. …………………. ……………………

1. …………………. …………………….

**ACKNOWLEDGEMENT**

I would like to place on record my deep sense of gratitude to **Shri D. K. Mohan** Chairman, Cambridge Group of Institution, Bangalore, India for providing excellent Infrastructure and Academic Environment at CITECH without which this work would not have been possible.

I am extremely thankful to **Dr. B. V. Ravi Shankar**, Principal, CITECH, Bangalore, for providing me Academic ambience and Laboratory facilities to work in, and everlasting motivation to carry out this work and shaping our careers.

I also wish to extend my thanks to **Dr. Suresh L**, HOD, Dept. of Computer Science and Engineering CITECH, Bangalore, India, for her critical, insightful comments, guidance and constructive suggestions to improve the quality of this research work.

I take this opportunity to thank all my friends, classmates who always stood by me in difficult situations also helped me in some technical aspects and Last but not the least I wish to express deepest sense of gratitude to my parents who were a constant source of encouragement and stood by me as pillar of strength for completing this work and course successfully

**ABDUL AZEEZ. C (1CD09CS001)**

**Table of Contents**

**Chapterspage no’s.**

1. **Introduction**

**1.1 Computer Graphics----------------------------------------5**

* 1. **1.2 Project Description-----------------------------------------5**

1. **Literature Survey------------------------------------------------------6**

**2.1 OpenGL------------------------------------------------------6**

**2.2 How to Implement-----------------------------------------7**

**3. System Requirement--------------------------------------------------14**

**3.1. Hardware requirements---------------------------------14**

**3.2. Softwareware requirements----------------------------14**

1. **Design & Implementation -------------------------------------------15**

**4.1.Code Description------------------------------------------15**

**4.2.Main Modules----------------------------------------------16**

1. **Source Code with Corresponding explaination-----------------**1**7**
2. **Final Output Snaps----------------------------------------------------24**
3. **Conclusion---------------------------------------------------------------27**
4. **Future Enhancement--------------------------------------------------28**

**Bibliography ………………………………………………… 29**

**CHAPTER 1:**

**INTRODUCTION**

* 1. **Computer Graphics:**

Computer graphics is concerned with all aspects of producing pictures or images using a computer. The field began humbly almost 50 years ago, with the display of a few lines on a cathode-ray-tube (CRT). Now we can create images by computers that are indistinguishable from photographs of real objects. One of the softwares used is OpenGL. Computer graphics is concerned with all aspects of producing pictures or images using a computer.

Three-dimensional graphics have many uses in modern computer applications. Applications for real-time 3D graphics range from interactive games and simulations to data visualization for scientific, medical, or business uses. Higher-end 3D graphics find their way into movies and technical and educational publications as well.

* 1. **AIM OF THE PROJECT:**

This project is the field of education, in the field of simulating the manufacturing of a hard disk by various companies. Moreover it might be useful in getting the precision of arranging the highly sophisticated internal parts of hard disk.

The objective of the puzzle is to move the entire stack to another rod, obeying the following rules:

1. Movement of the disks/ platters.
2. Read Write head is activated using the actuator & actuator arm.
3. Viewing the hard disk in a complete 3D view for thorough understanding of Platters spinning around the spindle.

**CHAPTER 2:**

**LITERATURE SURVEY**

The literature survey is where we give our basic concept which we have used in the application program.

**2.1 OpenGL:**

OpenGL graphics library is a standard specification defining a cross platform API for writing applications that produce 2D and 3D computer graphics. The interface consists of over 250 different function calls which can be used to draw complex 3D scenes and simple primitives.

OpenGL is widely used in CAD, vertical reality, scientific visualization, flight simulation and video games. OpenGL specifies a set of commands or immediately executes functions. Each command directs a drawing action or causes a special effect. A list of these commands can be created by repetitive effects.

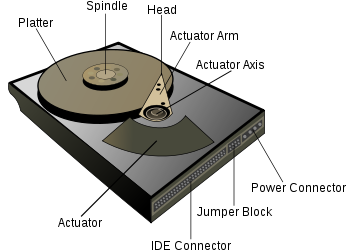
OpenGL is independent of windowing characteristics of each operating system. But provides special GLU routines for each operating system that enables OpenGL to work with the system’s windowing environment.

OpenGL comes with a large number of building capabilities. These include hidden surface removal, alpha blending, ant aliasing, texture mapping, pixel operation, viewing, modeling transformation and atmospheric effects. Functions in the OpenGL library have names that begin with the letters GL and are stored in a library usually referred to as GL. The second is the OpenGL utility library (GLU).this library uses only GL functions in GLU can be created from the core GL library.

To interface with the window system and to get from external devices into our programs, we need at last one more library. Rather than using a different library for each system, we use a readily available library called the OpenGL Utility Toolkit (GLUT), with provides the minimum functionality that should be expected in any modern windowing system. OpenGL makes use of macros to increase code readability.

**OpenGL supports two types of primitives**. Geometric primitives and raster primitives. **Geometric primitives** are specified in the problem domain and includes points , lines segments, polygons curves, and surfaces. They exist in a 2D or 2D space , they can be manipulated by operation such as rotation and translation. Raster primitives, such as arrays of pixels, lack geometric properties and cannot be manipulated in space.

**2.2 HOW TO IMPLEMENT:** Design and Implementation of a Hard Disk Drive System



**H**ARD disk drives provide important data-storage medium for computers and other data-processing

systems. In most hard disk drives, rotating disks coated with a thin magnetic layer or recording medium

are written with data, which are arranged in concentric circles or tracks. Data are read or written with a

read/write (R/W) head, which consists of a small horseshoeshaped electromagnet. Fig. above shows a

simple illustration of a typical hard disk servo system.

**The two main functions of the R/W head positioning servo mechanism in disk drives are**

track seeking and track following. **Track seeking** moves the R/W head from the present track to a

specified destination track in minimum time using a bounded control effort. **Track following** maintains

the head as close as possible to the destination track center while information is being read from or

written to the disk. Track density is the reciprocal of the track width. It is suggested that on a disk

surface, tracks should be written as closely spaced as possible so that we can maximize the usage of the

disk surface.

This means an increase in the track density, which subsequently means a more stringent

requirement on the allowable variations of the position of the heads from the true track center. The

prevalent trend in hard disk design is toward smaller hard disks with increasingly larger capacities. This

implies that the track width has to be smaller leading to lower error tolerance in the positioning of the

head. The controller for track following has to achieve tighter regulation in the control of the

servomechanism. Current hard disk drives use a combination of classical control techniques, such as

lead-lag compensators, PI compensators, and notch filters. These classical methods can no longer meet

the demand for hard disk drives of higher performance.

**Magnetic disk / platter**: A hard disk drive records data by magnetizing a thin film of ferromagnetic material on a disk. User data is encoded into a run-length limitedcode and the encoded data written as a pattern of sequential magnetic transitions on the disk. The data is represented by the time between transitions. The self-clocking nature of the run-length limited codes used enables the clocking of the data during reads. The data is read from the disk by detecting the transitions and then decoding the written run-length limited data back to the user data.

A typical HDD design consists of a **spindle** that holds flat circular disks, also called platters, which hold the recorded data. The platters are made from a non-magnetic material, usually aluminum alloy, glass, or ceramic, and are coated with a shallow layer of magnetic material typically 10–20 nm in depth, with an outer layer of carbon for protection. For reference, a standard piece of copy paper is 0.07–0.18 millimetre (70,000–180,000 nm).

**Form Factor Of Currently available hard disk Drives**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Form factor** | **Width (mm)** | **Height (mm)** | **Largest capacity** | **Platters (max)** | **Per platter (GB)** |
| 3.5″ | 102 | 19 or 25.4 | 4 TB(2011) | 5 | 1000 GB |
| 2.5″ | 69.9 | 7, 9.5, 12.5, or 15 | 2 TB (2012) | 4 | 500 GB |
| 1.8″ | 54 | 5 or 8 | 320 GB(2009) | 3 | 107 GB |

### Access time

The factors that limit the time to access the data on a hard disk drive (Access time) are mostly related to the mechanical nature of the rotating disks and moving heads. Seek time is a measure of how long it takes the head assembly to travel to the track of the disk that contains data. Rotational latency is incurred because the desired disk sector may not be directly under the head when data transfer is requested. These two delays are on the order of milliseconds each. The bit rate or data transfer rate (once the head is in the right position) creates delay which is a function of the number of blocks transferred; typically relatively small, but can be quite long with the transfer of large contiguous files. Delay may also occur if the drive disks are stopped to save energy, see Power management.

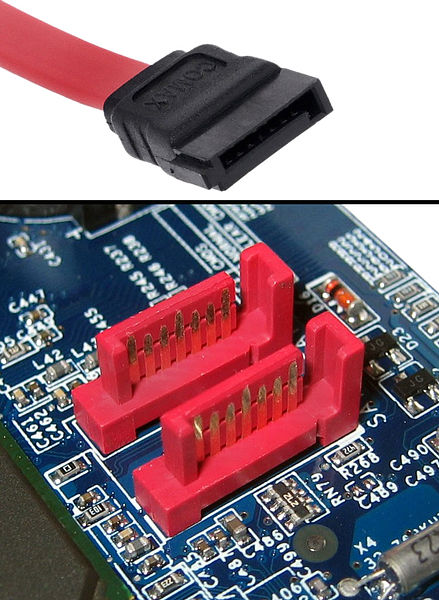
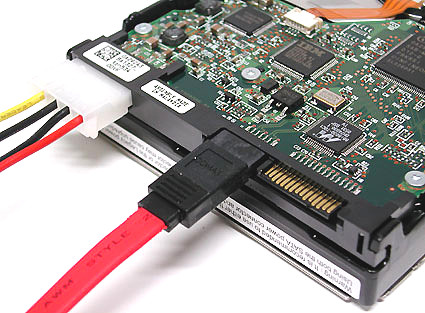
An HDD's **Average Access Time** is its average Seek time which technically is the time to do all possible seeks divided by the number of all possible seeks, but in practice is determined by statistical methods or simply approximated as the time of a seek over one-third of the number of tracks

#### Seek time:

Average seek time ranges from 3 ms for high-end server drives, to 15 ms for mobile drives, with the most common mobile drives at about 12 ms and the most common desktop type typically being around 9 ms. The first HDD had an average seek time of about 600 ms and by the middle 1970s HDDs were available with seek times of about 25 ms. Some early PC drives used a stepper motor to move the heads, and as a result had seek times as slow as 80–120 ms, but this was quickly improved by voice coil type actuation in the 1980s, reducing seek times to around 20 ms. Seek time has continued to improve slowly over time.

#### Rotational latency:

Latency is the delay for the rotation of the disk to bring the required disk sector under the read-write mechanism. It depends on rotational speed of a disk, measured in revolutions per minute (rpm). Average rotational latency is shown in the table below, based on the statistical relation that the average latency in milliseconds for such a drive is one-half the rotational period.

[](http://en.wikipedia.org/wiki/File:Hard_disk_head.jpg)  

**FIGURE(1) FIGURE(2)**

Close-up HDD head resting on disk platter. **Serial ATA** (**SATA** or **Serial Advanced Technology Attachment**)

(Its mirror reflection is visible on the platter surface.) is a computer bus interface for connecting host bus adapters to mass storage devices such as hard disk drives and optical drives. Serial ATA was designed to replace the older parallel ATA (PATA) standard (often called by the old name IDE),

|  |  |
| --- | --- |
| **Rotational speed [rpm]** | **Average latency [ms]** |
| 15,000 | 2 |
| 10,000 | 3 |
| 7,200 | 4.16 |
| 5,400 | 5.55 |
| 4,800 | 6.25 |

#### Data transfer rate

As of 2010, a typical 7,200 rpm desktop hard drive has a sustained "disk-to-buffer" data transfer rate up to 1,030 Mbits/sec.[[88]](http://en.wikipedia.org/wiki/Hard_disk_drive#cite_note-87) This rate depends on the track location, so it will be higher for data on the outer tracks (where there are more data sectors) and lower toward the inner tracks (where there are fewer data sectors); and is generally somewhat higher for 10,000 rpm drives. A current widely used standard for the "buffer-to-computer" interface is 3.0 Gbit/s SATA, which can send about 300 megabyte/s (10-bit encoding) from the buffer to the computer, and thus is still comfortably ahead of today's disk-to-buffer transfer rates. Data transfer rate (read/write) can be measured by writing a large file to disk using special file generator tools, then reading back the file. Transfer rate can be influenced by file system fragmentation and the layout of the files.[

HDD data transfer rate depends upon the rotational speed of the platters and the data recording density. Because heat and vibration limit rotational speed, advancing density becomes the main method to improve sequential transfer rates.[[89]](http://en.wikipedia.org/wiki/Hard_disk_drive#cite_note-88) While areal density advances by increasing both the number of tracks across the disk and the number of sectors per track, only the latter will increase the data transfer rate for a given rpm. Since data transfer rate performance only tracks one of the two components of areal density, its performance improves at a lower rate.

### Power consumption

Power consumption has become increasingly important, not only in mobile devices such as laptops but also in server and desktop markets. Increasing data center machine density has led to problems delivering sufficient power to devices (especially for spin up), and getting rid of the waste heat subsequently produced, as well as environmental and electrical cost concerns. Heat dissipation is tied directly to power consumption, and as drives age, disk failure rates increase at higher drive temperatures.

**S**imilar issues exist for large companies with thousands of desktop PCs. Smaller form factor drives often use less power than larger drives. One interesting development in this area is actively controlling the seek speed so that the head arrives at its destination only just in time to read the sector, rather than arriving as quickly as possible and then having to wait for the sector to come around (i.e. the rotational latency).[[91]](http://en.wikipedia.org/wiki/Hard_disk_drive#cite_note-90) Many of the hard drive companies are now producing Green Drives that require much less power and cooling. Many of these Green Drives spin slower (<5,400 rpm compared to 7,200, 10,000 or 15,000 rpm) thereby generating less heat. Power consumption can also be reduced by parking the drive heads when the disk is not in use reducing friction, adjusting spin speeds,[[92]](http://en.wikipedia.org/wiki/Hard_disk_drive" \l "cite_note-91) and disabling internal components when not in use.

#### Power management

Most hard disk drives today support some form of power management which uses a number of specific power modes that save energy by reducing performance. When implemented an HDD will change between a full power mode to one or more power saving modes as a function of drive usage. Recovery from the deepest mode, typically called Sleep, may take as long as several seconds.

SCSI originally had just one signaling frequency of 5 MHz for a maximum data rate of 5 megabytes/second over 8 parallel conductors, but later this was increased dramatically. The SCSI bus speed had no bearing on the disk's internal speed because of buffering between the SCSI bus and the disk drive's internal data bus; however, many early disk drives had very small buffers, and thus had to be reformatted to a different interleave (just like ST-506 disks) when used on slow computers, such as early Commodore Amiga, IBM PC compatibles and Apple Macintoshes.

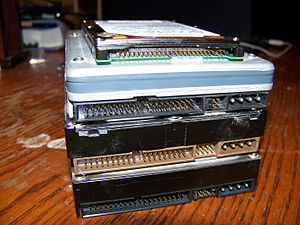
**(Parallel) ATA** interfaces were designed to support two drives on each channel, connected as master and slave on a single cable. Disks typically had no problems with interleave or data rate, due to their controller design, but many early models were incompatible with each other and could not run with two devices on the same physical cable. This was mostly remedied by the mid-1990s, when ATA's specification was standardized and the details began to be cleaned up, but still causes problems occasionally, especially with CD-ROM and DVD-ROM disks, and when mixing Ultra DMA and non-UDMA devices.

PATA is given as in the FIGURE (3), which has much slower data rate & master/ slave redunduncy.

**Serial ATA** supports one drive per channel and per cable, with its own set of I/O ports, avoiding

master/slave problems. Sata hard drives are much faster as given in the FIGURE (2).

Over the Period SATA has taken the Market sharply into the work.



**FIGURE(3)**

Parallel ATA Hard Disk

|  |  |  |
| --- | --- | --- |
| [**Acronym**](http://en.wikipedia.org/wiki/Acronym)**or abbreviation** | **Meaning** | **Description** |
| [SASI](http://en.wikipedia.org/wiki/Shugart_Associates_System_Interface) | Shugart Associates System Interface | Historical predecessor to SCSI. |
| [SCSI](http://en.wikipedia.org/wiki/SCSI) | Small Computer System Interface | [Bus](http://en.wikipedia.org/wiki/Computer_bus) oriented that handles [concurrent](http://en.wikipedia.org/wiki/Concurrency_(computer_science)) operations. |
| [SAS](http://en.wikipedia.org/wiki/Serial_attached_SCSI) | Serial Attached SCSI | Improvement of SCSI, uses serial communication instead of parallel. |
| [ST-506](http://en.wikipedia.org/wiki/ST-506) | Seagate Technology | Historical Seagate interface. |
| [ST-412](http://en.wikipedia.org/wiki/ST-412) | Seagate Technology | Historical Seagate interface (minor improvement over ST-506). |
| [ESDI](http://en.wikipedia.org/wiki/Enhanced_Small_Disk_Interface) | Enhanced Small Disk Interface | Historical; backwards compatible with ST-412/506, but faster and more integrated. |
| [ATA](http://en.wikipedia.org/wiki/Advanced_Technology_Attachment) (PATA) | Advanced Technology Attachment | Successor to ST-412/506/ESDI by integrating the disk controller completely onto the device. Incapable of concurrent operations. |
| [SATA](http://en.wikipedia.org/wiki/Serial_ATA) | Serial ATA | Modification of ATA, uses serial communication instead of parallel. |

### Metrics of failures

Most major hard disk and motherboard vendors now support S.M.A.R.T. (Self-Monitoring, Analysis,

and Reporting Technology), which measures drive characteristics such as operating temperature, spin-

up time, data error rates, etc. Certain trends and sudden changes in these parameters are thought to be

associated with increased likelihood of drive failure and data loss.

However, S.M.A.R.T. parameters alone may not be useful for predicting individual drive

failures. Unpredictable breakdown of the inherently fragile device—and all mechanisms must

eventually fail—may occur at any time in normal use, with potential loss of all data, which is reliably

recoverable only if another copy is stored by using a RAID array or making backup copies (the purpose

of RAID and backup is different in the context of a data storage system, but the distinction is not

relevant to this article). Recovery of some or even all data from a damaged drive is sometimes, but not

always possible, and is normally costly.

#### Recovery of data from failed drive

Data from a failed drive can sometimes be partially or totally recovered if the platters' magnetic

coating is not totally destroyed. Specialised companies carry out data recovery, at significant cost, by

opening the drives in a clean room and using appropriate equipment to read data from the platters

directly. If the electronics have failed, it is sometimes possible to replace the electronics board, though

often drives of nominally exactly the same model manufactured at different times have different,

incompatible, circuit boards.

Sometimes operation can be restored for long enough to recover data. Risky techniques are

justifiable if the drive is otherwise dead. If a drive is started up once it may continue to run for a shorter

or longer time but never start again, so as much data as possible is recovered as soon as the drive starts.

A 1990s drive that does not start due to stiction can sometimes be started by tapping it or rotating the

body of the drive rapidly by hand. Another technique which is sometimes known to work is to cool the

drive, in a waterproof wrapping, in a domestic freezer. There is much useful information about this in

blogs and forums,  but professionals also resort to this method with some success.

**CHAPTER 3:**

**SYSTEM REQUIREMENTS**

**Prerequisites And Familiarity With:**

* Knowledge of the data structures concepts of:
  + - Pointers
    - Stack
    - Structures
* OpenGL basic concepts and knowledge of the functions.
* Creative Imagination Of all Co Ordinate axes.

3.1  **Hardware Requirements:**

* A Pentium IV/AMD Athlon 64 processor or equivalent, less may be used but not

recommended.

* Keyboard and mouse drivers.
* A primary memory of a minimum of 512MB is recommended.
* A hard disk space of minimum 700MB.
* A good high definition graphics card (512 MB upwards).

3.2 **Software requirements:**

* Microsoft Visual Studio 6.0
* Windows Operating System
* OpenGL
* Linux windows enriched with OpenGL can also be used

**CHAPTER 4**

**DESIGN & IMPLEMENTATION**

**4.1 CODE DESCRIPTION:**

This project is the field of education, in the field of simulating the manufacturing of a hard disk by various companies. More over it might be useful in getting the precision of arranging the highly sophisticated internal parts of hard disk.

The objective of the puzzle is to move the entire stack to another rod, obeying the following rules:

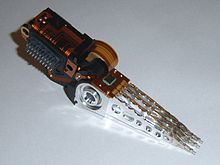
1. Movement of the disks/ platters.
2. Read Write head is activated using the actuator & actuator arm.
3. Viewing the hard disk in a complete 3D view for thorough understanding of Platters spinning around the spindle.

**N**

ow we start with the main function, here, the GLUT library is initialized and the display and keyboard callback are set. Also, the window is created and its name, size and position are specified.

Next is the display function callback in which we call following functions:

1. **display() & rotate():** are the functions used to display all other modules i.e. the base of the hard disk casing, spindle, platters, & to the platters held by spindle.
2. **Draw\_Disks() & Spindle():** are the functions used to display the disks held by the cylindrical spindle.
3. **Motor() & spindle wrap():** are used to draw motor which in turn rotates the spindle, & spindle wrap is used to firmly held the disk movement by the spindle.

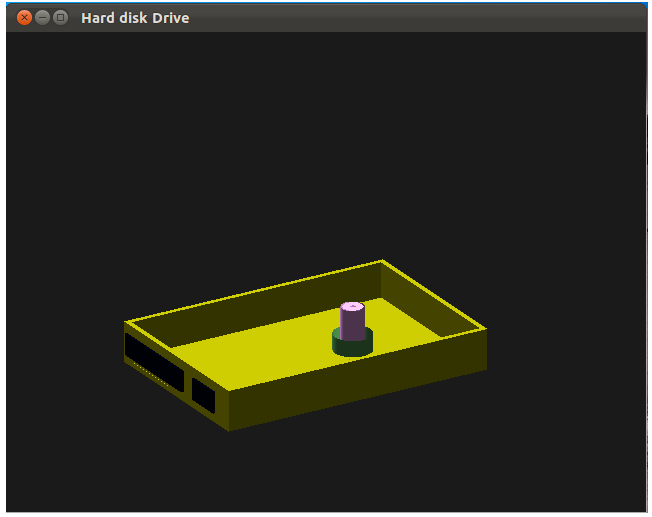
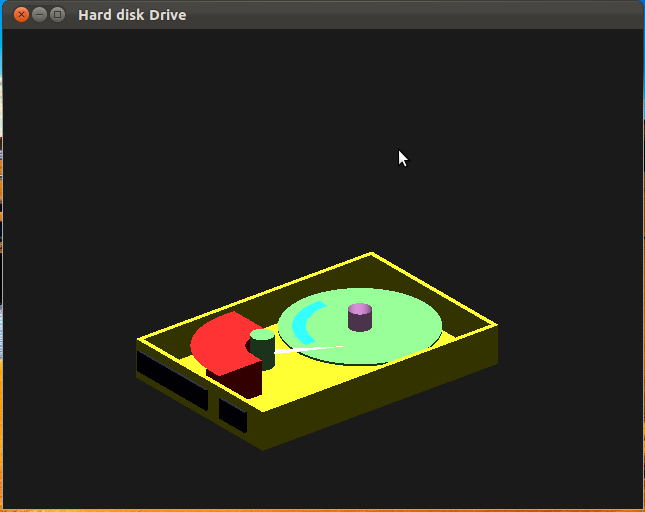


Typical read/ write actuator

**4.2 MAIN FUNCTIONAL MODULES:**

* Void display():

In this function we first clear the color and depth buffer bit, using glClear. Then we call the functions **base\_creation(), draw\_cylinde\_head(), &** **draw\_disks()** Etc**.** This module draws/creates a foundation with desired parameters and performs required affine transformations to draw the posts and the disks. It also calls the subroutines glPushMatrix() and glPopMatrix() to carry out the environmental lighting & shading of the disks.

SNAPSHOT WITH SPINDLE & MOTOR HDD CASING WITH A DISK & ITS ACTUATOR ARM

SNAPSHOT (1) SNAPSHOT (2)

Motor() with spindle() actuator() with an actuator arm

**CHAPTER 5**

**PROGRAM SOURCE CODE**

**PROGRAM CODE FOR DISPLAY FUNCTION CALL BACK**

void display(){ if(s==1)

{

glClear(GL\_COLOR\_BUFFER\_BIT|GL\_DEPTH\_BUFFER\_BIT);

glClearColor(.1f,.1f,.1f,.1f);

glEnable(GL\_COLOR\_MATERIAL);GLfloat lightintensity[]={0.7f,0.7f,0.7f,1.0f}; GLfloat lightposition[]={2.0f,6.0f,3.0f,0.0f}; GLfloat mat\_ambient[]={0.7f,0.7f,0.7f,1.0f}; GLfloat mat\_diffuse[]={0.6f,0.6f,0.6f,1.0f}; GLfloat mat\_specular[]={1.0f,1.0f,1.0f,1.0f}; GLfloat mat\_shininess[]={50.0f}; GLfloat globamb[]={0.5,0.5,0.5,1.0};

glMaterialfv(GL\_FRONT,GL\_AMBIENT,mat\_ambient); glMaterialfv(GL\_FRONT,GL\_DIFFUSE,mat\_diffuse); glMaterialfv(GL\_FRONT,GL\_SPECULAR,mat\_specular); glMaterialfv(GL\_FRONT,GL\_SHININESS,mat\_shininess); glLightfv(GL\_LIGHT0,GL\_POSITION,lightposition); glLightfv(GL\_LIGHT0,GL\_DIFFUSE,lightintensity); glShadeModel(GL\_SMOOTH);

obj = gluNewQuadric(); glMatrixMode(GL\_PROJECTION); glLoadIdentity(); glOrtho(-2.0\*64/48,1.5\*64/48.0,-2,2.,0.1,100.0); glMatrixMode(GL\_MODELVIEW); glLoadIdentity();basecreation(obj);// USED FOR CREATING THE CASING OF THE HDDdraw\_cylinder\_head();// USED TO DESIGN & DRAW THE SPINDLEglRotatef(theta,0.0,1.0,0.0);//ROTATE THE DISKS ALONG THE

SPINDLEdraw\_disk(obj,0.0,0.6,-0.0,0.21);// USED TO DRAW THE DISKSdraw\_disk(obj,0.0,0.6,-0.0,0.13);draw\_disk(obj,0.0,0.6,-0.0,0.05);spindlewrap(0.09,0.2,-0.4,-0.92);//FOR PACKING THE DISKS

WITH SPINDLEmotor(0.5,0.15,0.2);// TO DRAW THE SPINDLEglutPostRedisplay(); /\*TO CONTINUOSLY DISPLAY THE FIGURE

AFTER ROTATE OPERATION\*/

glFlush();glutSwapBuffers();

}

else

welcomscrn();// USED TO CALL INTRODUCTION SCREEN

glFlush();

}- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

**DESCRIPTION OF DISPLAY FUNCTION.**

Note: A display Function calls all the functions like **basecreation()** used to draw the base of the hard disk casing as given in the snapshot(1), which in turn calls **actuator(),** which is used to draw actuator & its arm for read write head as given in the **SNAPSHOT(2)**. Even the functions like **motor()** is called to draw a motor so as to firmly held the spindle along with the disks as given in the **SNAPSHOT(1)**.

It also calls the functions like

* **void base\_creation(obj) :**

This function is used to draw the casing of the hard disk drive, which is specified by using the glTranslatef(x,y,x) function in order to place the object in the right place also by scaling the cube by right dimensions. Here **obj** is passes as parameter i.e object to be drawn inside the hard disk defined by using the function **obj = gluNEWquadric();**

Here the **actuator(obj)** functions is called as follows:

Each actuator arm is drawn using the following **actuator(obj)** function:

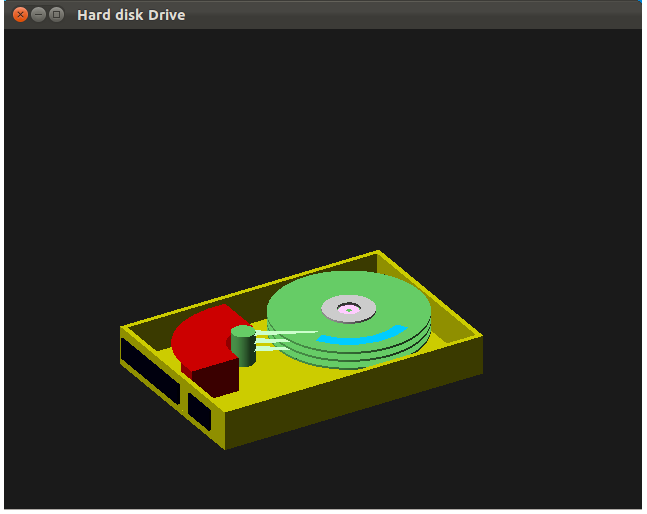
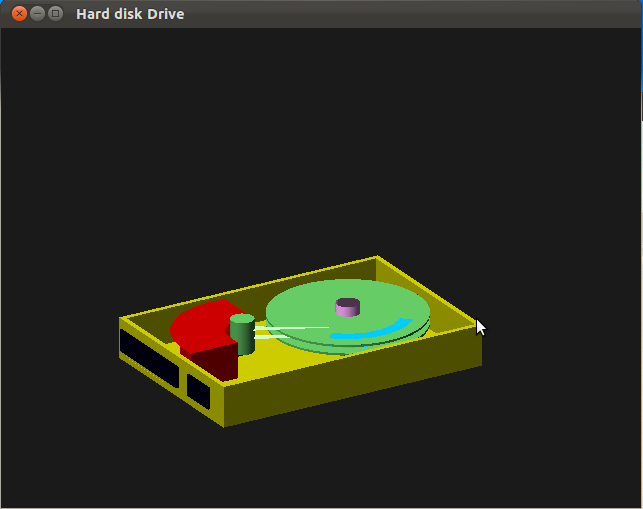
**void actuator(GLUquadricObj \* object)**{ GLdouble i; glPushMatrix(); glTranslatef(-0.02,-0.1,0.8);

glRotatef(90, 1.0, 0.0, 0.0);glRotatef(180,0,1,0);glColor4f(1.0,0.0,0.0,0.5);gluPartialDisk(object, 0.2,0.6,1000,1000,-10,100); //THIS IS USED TO DRAW A PART OF DISK//glPopMatrix();glPushMatrix();glColor4f(0.5, 1.0, 0.5, 0.1);glRotatef(90, 1.0, 0.0, 0.0);glTranslatef(-0.08,0.85, 0.0);spindle(obj, 0.09, 0.3, 0.3);glRotatef(180, 1.0, 0.0, 0.0);gluDisk(obj, 0.0, 0.09, 100000, 1);glPopMatrix();

glPushMatrix();// EACH ACTUATOR ARM IS DRAWN IN THE FOLLOWING MANNER//glColor3f(1.0,10.0,1.0);glRotatef(90,1.0,0.0,0.0);glRotatef(180,0.0,1.0,0.0);glTranslatef(-0.24,0.3,-0.01);gluPartialDisk(object, 0,0.6,5,10,25,8);glPopMatrix();

}-------------------------------------------------------------------------

Similarly 3 disks are also placed one above other by using the function 3 times just by varying the z axis values as given in the figure below.



SNAPSHOT(3) ` SNAPSHOT(4)

Draw\_disk() function is used hereSpindlewrap() function is used here

**Note:** Each disk has its separate read / Write head as given in the SNAPSHOT(3) & SNAPSHOT(4).

**display()** also calls the functions like :

**draw\_disk() :** used to draw the disk as given in the SNAPSHOT(3)

**spindlewrap():** used to firmly held the disks with the spindle as given in the SNAPSHOT(4).

**glortho()** : used to set the viewing volume of the window by co ordinate transformation.

**void spindlewrap(GLfloat inrad, GLfloat outrad, GLfloat ytrans, GLfloat ztrans)**{ glPushMatrix(); glColor4f(1.0, 1.0, 1.0, 0.1); glRotatef(90, 1.0, 0.0, 0.0); glTranslatef(0, 0.0, 0.01); glTranslatef(0, 0,0.01 ); spindle(obj, outrad, inrad, 0.02); glRotatef(180, 1.0, 0.0, 0.0); gluDisk(obj, inrad, outrad, 20000, 1); glPopMatrix();}

**T**his function is particularly used to hole the spindle along with the disks on the top of the spindle as given in the **SNAPSHOT(4)**

Note: The **base\_creation(obj)** also calls a function **casingwalls()** as follows:

**void casingwalls(GLdouble xt, GLdouble yt, GLdouble zt, GLdouble xs, GLdouble ys, GLdouble zs**){ glPushMatrix();

glTranslated(xt,yt,zt); glRotated(20,0,0,0); glScaled(xs,ys,zs); glutSolidCube(1.0); glPopMatrix();}

Here the values of **xt, yt & zt** represents the 3 co – ordinates translations or placements of casing objects. Similarly **xs, ys, & zs** represents the 3 co-ordinates scaling for a cube. As given in the **SNAPSHOT(1).**

**The rotation of disk is given by the following function:**

**void rotate()**{ theta=theta-inc;//TO ROTATE IN ANTI CLOCK WISE DIRECTION glutPostRedisplay();}

* **Void keyboard1(unsigned char key, int x, int y):**

This function provides user interaction and enables movement between windows and gives user options. After pressing any of the mentioned keys the previous window is destroyed and a new window is created and the corresponding display and keyboard functions are called for performing the desired tasks.

**The following keys are pressed for the interactive viewing of the hard disk**

**void key(unsigned char key, int x, int y)**{ if(key=='X') viewer[0]-=1.0; if(key=='x') viewer[0]+=1.0; if(key=='Y') viewer[1]-=1.0; if(key=='y') viewer[1]+=1.0; if(key=='Z') viewer[2]-=1.0; if(key=='z') viewer[2]+=1.0;}

‘x’ indicates –ve movement of viewing Hard disk along x.

‘X’ indicates +ve movement of Viewing Hard disk along X.

‘y’ indicates –ve movement of viewing Hard disk along y.

‘Y’ indicates +ve movement of Viewing Hard disk along y.

‘z’ indicates +ve movement of viewing Hard disk along z

‘Z’ indicates +ve movement of viewing Hard disk along Z

ANOTHER FUNCTION CALLED BY DISPLAY IS MOTOR WHICH IS DRAWN USING **motor()** function as given below.

**void motor(GLfloat inrad, GLfloat outrad, GLfloat ztrans)**{ glPushMatrix(); glColor4f(0.5, 1.0, 0.5, 0.1); glRotatef(90, 1.0, 0.0, 0.0); glTranslatef(0, 0.0, 0.09); glTranslatef(0, 0,ztrans); spindle(obj, outrad, inrad, 0.16);//USED TO CALL SPINDLE FUNCTION TO

DRAW A MOTOR

glRotatef(180, 1.0, 0.0, 0.0); gluDisk(obj, inrad, outrad, 20000, 1); glPopMatrix();}

**Note:** gluDisk() is used to draw a disk like structure using the GLUquadric() functions.

The motor() function displays the figure as in the **snapshot(1)** in page 16.

* **Void welcomscrn() :**

**void welcomscrn()**

{

glEnable(GL\_COLOR\_MATERIAL);

glClearColor(1.0,0.0,.0,0.0);

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glColor3f(1.0,1.0,0.0);

/\* PRINTING OF TEXT ON THE INTRODUCTION WINDOW\*/

glRasterPos2f(-1.3,-1.3);

for( int p=0;p<sizeof(txt1);p++)

glutBitmapCharacter(GLUT\_BITMAP\_TIMES\_ROMAN\_24,txt1[p]);

glRasterPos2f(-1.5,0.65);

for( int p=0;p<sizeof(txt3);p++)

glutBitmapCharacter(GLUT\_BITMAP\_TIMES\_ROMAN\_24,txt3[p]);

glRasterPos2f(-2.5,1.0);

for( int p=0;p<sizeof(txt2);p++)

glutBitmapCharacter(GLUT\_BITMAP\_TIMES\_ROMAN\_24,txt2[p]);

glRasterPos2f(-1.5,0.45);

for( int p=0;p<sizeof(txt4);p++)

glutBitmapCharacter(GLUT\_BITMAP\_TIMES\_ROMAN\_24,txt4[p]);

glRasterPos2f(-2.4,-0.6);

for( int p=0;p<sizeof(txt5);p++)

glutBitmapCharacter(GLUT\_BITMAP\_TIMES\_ROMAN\_24,txt5[p]);

glRasterPos2f(-1.3,-1.5);

for( int p=0;p<sizeof(txt6);p++)

glutBitmapCharacter(GLUT\_BITMAP\_TIMES\_ROMAN\_24,txt6[p]);

glRasterPos2f(-1.3,-1.7);

for( int p=0;p<sizeof(txt7);p++)

glutBitmapCharacter(GLUT\_BITMAP\_TIMES\_ROMAN\_24,txt7[p]);

glRasterPos2f(-1.3,-1.9);

for( int p=0;p<sizeof(txt8);p++)

glutBitmapCharacter(GLUT\_BITMAP\_TIMES\_ROMAN\_24,txt8[p]);

glDisable(GL\_COLOR\_MATERIAL);

glFlush();

glDisable(GL\_COLOR\_MATERIAL);

}

This function is used to print the text on the windows at specified positions and with specified size. It takes five parameters namely x, y, z, scale and the string (which we need to print). x, y, z are used to set the co ordinate values at which the character should be placed. Scale determines the font size of the character. S is the string to be displayed . Also inbuilt function glutStrokeCharacter() is used to print one character at a time, where as glutBitmapCharacter() prints group of charcters/string.

**The following is the main() function for the hard disk drive window display**.

**int main(int argc,char \*\*argv)**{ glutInit(&argc,argv); glutInitDisplayMode(GLUT\_DOUBLE|GLUT\_RGB|GLUT\_DEPTH); glutInitWindowSize(640,480); glutInitWindowPosition(0,0); glutCreateWindow("Hard disk Drive"); glutReshapeFunc(Reshape); glutKeyboardFunc(mykey); glEnable(GL\_LIGHTING); glEnable(GL\_LIGHT0); glEnable(GL\_COLOR\_MATERIAL);//TO ENABLE COLOR APPLYING ON SOLID

MATERIAL glShadeModel(GL\_SMOOTH); glEnable(GL\_DEPTH\_TEST);//USED FOR OVERLAPPED HIDDEN SURFACES glEnable(GL\_NORMALIZE); glClearColor(.1f,.1f,.1f,.1f); glutIdleFunc(rotate); glutDisplayFunc(display); glutMainLoop(); return 0;}

**Note:**  All The initialization functions for openGL functions are called from menu()

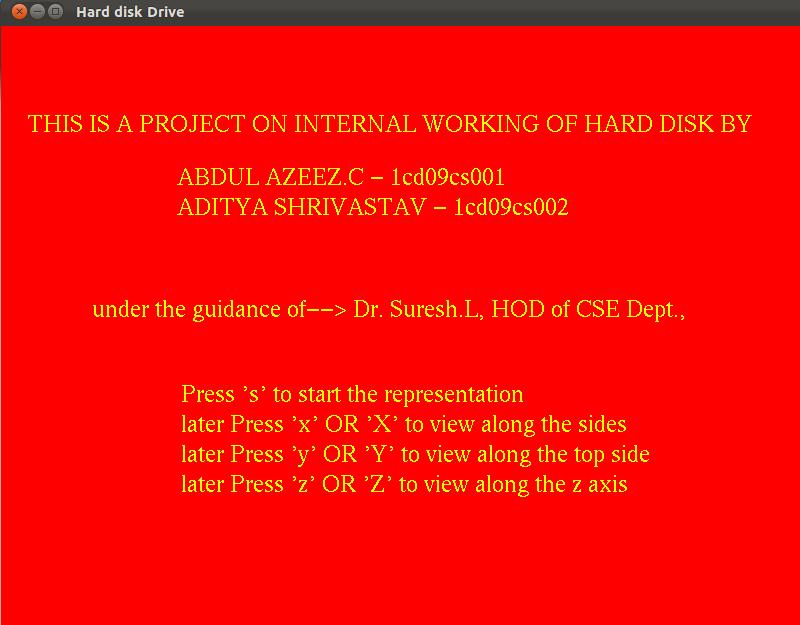
function. Therefore glutMainLoop() is used to call all these funcstions in loop until the

user terminates the process of display operation.

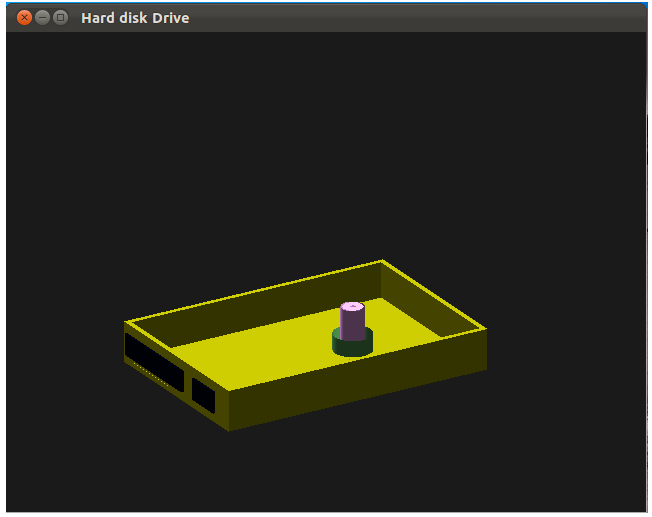
**CHAPTER 6:**

**FINAL OUTPUT**

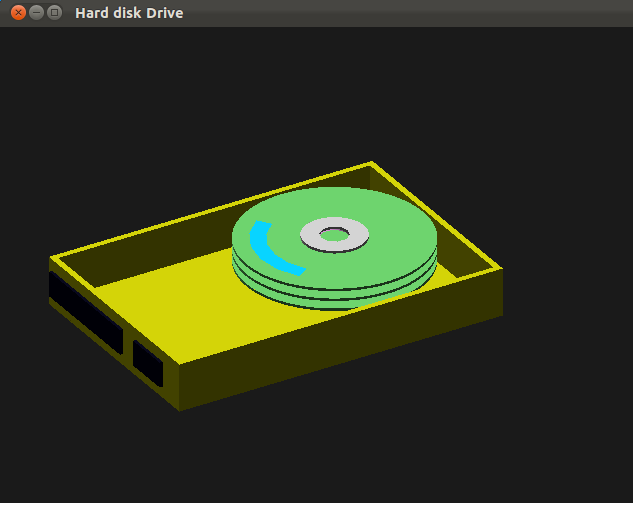
SNAP 1: INTRODUCTION WINDOW WITH OPTIONS FOR USER



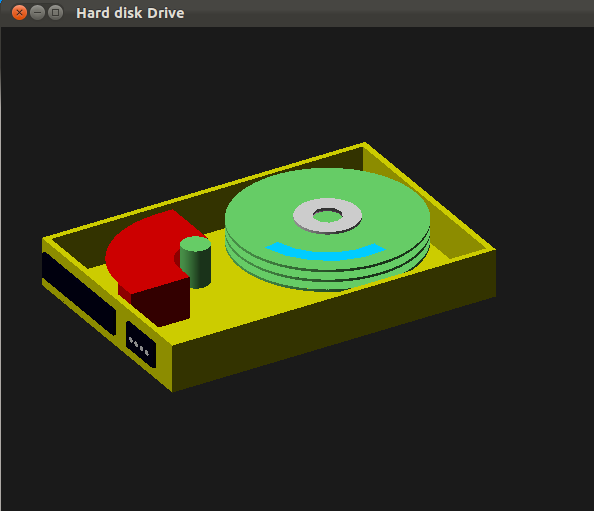
SNAP 2: INITIAL SETUP WITHOUT DISKS



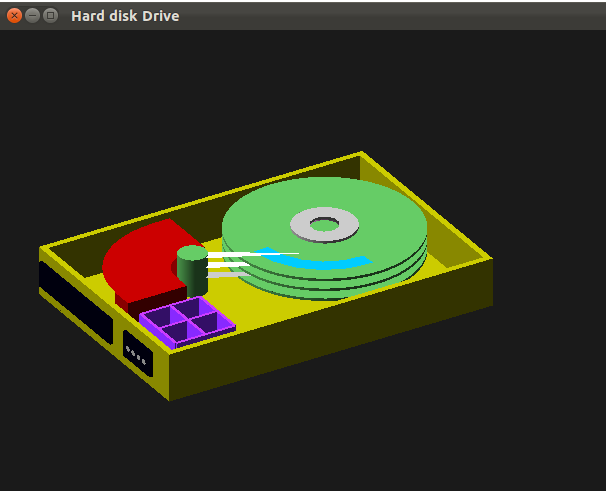
SNAP 3: INITIAL SETUP WITH DISKS



SNAP 4: INTERMEDIATE STEP



SNAP 7: FINAL OUTPUT



NOTE: The above Figure is Purely a Resemblense to actual hard disk structure

Which is given below in the figure.

The above Hard Disk is a Standard IDE Hard Disk, which uses a 40 pin IDE cable & a 4 pin power supply connector. Which is now been over taken by SATA Hard Disk Drives comparatively much faster than other devices with a data transfer rate of 1.5, 3.0, 6.0 Gbit/s

ACTUAL STRUCTURE MAY LOOK LIKE THIS:



**CHAPTER 7**

**CONCLUSION**

Our objective of demonstrating the concept of read/write operation using internal working of a hard disk through opengl representation has been successfully achieved. This mini project gave us a means for better understanding of the OpenGL concept through its practical implementation.

We had a great experience in the course of designing this package. This project on **“Internal Working of Hard Disk”**, which helped us learn and discover many things as pertaining to the topic of OpenGL. We have tried our best potential to incorporate all the basic requirements of normal VC++ for windows OS.

Its user friendly, it enables the user to interact effectively and easily. This package has been implemented using optimized OpenGL techniques. Special attention has been provided to the interfaces that make its use comfortable. We hope this package proves to be flexible in all respect to one and all.

**CHAPTER – 8**

**FUTURE ENHANCEMENTS**

There are a number of future enhancements that we would like to add to our mini project. We have mentioned them here.

* We can even simulate the data track recording & even the read/ write operations happening in the rotation of disk.
* This can be even be improved by Showing the detailed procedure of manufacturing the hard disk drives, irrespective of any hard disk pertaining to any company.
* It can be even used to find the Time required for read/ write operation on requesting an instruction execution from any secondary storage devices.
* Therefore it can act as simulation for increasing the Performance during manufacturing of the hard disk drives.

**BIBLIOGRAPHY**

1. **Interactive Computer Graphics**

A top-down approach using Open GL

--By **Edward Angel**

**2. Computer Graphics Using OpenGl**

2nd Edition ,pearson education,2001.

--By **F.S.Hill ,Jr**

**3. OpenGL Red Book**

2nd Revised Edition ,2005

--By **Jackie.L.Neider, Mark Warhol, Tom.R.Davis**

**Websites referred:**

1. **www.opengl.org**
2. **www.glprogramming.com**
3. **www.redbooksamples.com**