**MATHEMATICS IN**

**GAMING INDUSTRY**

*Seminar Report* **on**

***submitted in partial fulfillment for the Learning Objectives of Outcome Based Education***

**Paradigm in**

**(1st Sem. B.E)**

by,

Under the guidance of **Mrs. RAKHI POTE**



KLS Gogte Institute of Technology, Belgaum Department of Computer Science and Engineering

**Udyambag, Belgaum - 590008, Karnataka**

**2020-2021**

# CERTIFICATE



*This is to certify that the Seminar entitled “OPTICAL STORAGE DEVICES” is a bona fide record of the Seminar work done*

*under my*

*supervision and guidance, in partial fulfillment of the requirements for the*

*Outcome Based Education Paradigm in MATHEMATICS from*

*Gogte Institute of Technology for the academic year* ***2020-2021***

## Ms. Rakhi Pote

Asst.Professor

### Dr. Prabhakar Hampiholi

*Professor & Head*

***Dept. Of COMPUTER SCIENCE AND ENGINEERING***

Place: KLS Gogte Institute of Technology

Belgaum. Date:18-03-2021

## COURSE REPORT

**Marks allocation:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Batch No. : B1 | |  | | | | |
| 1. | Title:  MATHEMATICS IN GAMING INDUSTRY | Marks  Range | USN | | | | |
|  |  |  |  |  |
| 2. | Abstract (PO2) | 0-2 |  |  |  |  |  |
| 3. | Application of topic to the course(PO2) | 0-3 |  |  |  |  |  |
| 4. | Literature survey and its findings (PO2) | 0-4 |  |  |  |  |  |
| 5. | Methodology, Results and Conclusion (PO1,PO3,PO4) | 0-6 |  |  |  |  |  |
| 6. | Report and Oral presentation skill (PO9,PO10) | 0-5 |  |  |  |  |  |
|  | Total | 20 |  |  |  |  |  |

### \* 20 marks is converted to 10 marks for CGPA calculation

1. **Engineering Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.
2. **Problem Analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences and Engineering sciences.
3. **Design/Development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

### ACKNOWLEDGEMENT

This group feels greatly indebted to Computer Science and Engineering Department, for the opportunity given us to undertake this “MATHEMATICS IN GAMING INDUSTRY ” Seminar. This Seminar includes thoughts and contribution of many individuals. And we wish to express our sincere appreciation and gratitude to them.

First and foremost we want to extend entirely our gratitude to our

lecturer Mrs. Sudha for sharing his knowledge and profound wisdom with us. We appreciate all his comments and suggestions, which are incorporated into this project.

We would also like to express our gratitude towards and group

members. Without their help, support, and encouragement, this project would never had been completed.

In our respect, this project is an outcome of the learning experience we have Shared with our fellow students. We dedicate this Seminar to all our fellow engineering students.

**SAHANA TAVARI**

**SANAT K.S**

**NAVANIDHI NAYAK**

**PARISHKAR SINGH**

## CONTENTS

### Sl.No. TITLE

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

1. INTRODUCTION
2. HISTORY
3. WORKING
4. TYPES OF OPTICAL STORAGE DEVICES
5. WORKING OF CD ROM
6. ADVANTAGES

# MATHEMATICS IN GAMING INDUSTRY

# INTRODUCTION

Math = The Foundation of Game Design

In the same way that math doesn’t work unless you learn and apply the rules, a video game can’t have rules without math. When you think about it, video games are essentially virtual worlds with lots of rules that keep everything working as intended.

No math means Mario keeps floating up after jumping, bullets in Call of Duty shoot in random directions, and even your favorite character in Angry Birds move in inconsistent ways if it moves at all.

Most of the time the math you learned in high school and college is no different than what was used to design a game.

To name a few, some of the common branches of math utilized in game development include:

* Algebra
* Trigonometry
* Calculus
* Linear Algebra
* Discrete Mathematics
* Applied Mathematics

More specific elements of math almost always used in games include:

* Matrices
* Delta time
* Unit and scaling vectors
* Dot and cross products
* And scalar manipulation

The purpose of this article is to have a look at how mathematics is used in computer games. I'll use examples from computer games you've probably already played. There are lots of different types of computer games, and I'll talk about how maths is used in some of the following examples:

The *First Person Shooter* (FPS) is a type of game where you run around 3D levels carrying a big gun shooting stuff. Examples of this sort of game include *Doom* , *Quake* , *Half Life* , *Unreal* or *Goldeneye* . There are other games that look very similar, but aren't first person shooters, for instance *Zelda: Ocarina of Time* or *Mario 64* .

The *Strategy* games are divided into two main types, *Real Time Strategy* (RTS), and *Turn Based Strategy* (not usually called TBS for some reason). These games usually involve building and managing a city or civilization and also fighting wars by controlling troops. Examples of real time strategy games are *Age of Empires* , *Command &; Conquer* , *Tiberian Sun* . Examples of turn based strategy games are *Civilization* and *Alpha Centauri* .

*Simulation* games are games that try to make something as realistic as possible. For instance, *Flight Sims* are computer games which try to realistically simulate flying an aeroplane or helicopter. Two games of this sort are *Microsoft Flight Simulator* and *Red Baron* . *Space sims* are like flight sims, but with spaceships instead of planes. For instance, *Wing Commander* or *X-Wing vs. Tie Fighter* . *Racing* games are games which simulate driving different sort of cars. For instance, *Need for Speed* , *NASCAR Racing* , *Gran Turismo* or *Driver* .

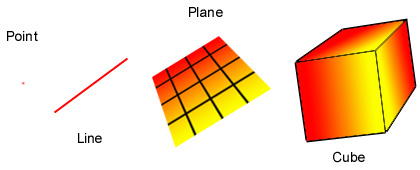
There are some exercises which you can do (if you want) throughout this article. The answers are at the end of the article, but do have a go at solving them on your own first.

.

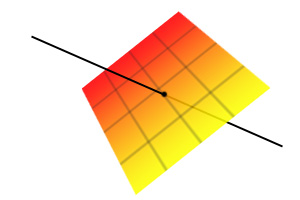
If you find any of the article patronising, I apologise, my excuse is that the article is aimed at people of many different ages so there might well be bits you already know. If you already understand one bit, you can just skip through until you get to something more interesting.

#### Geometry, Vectors and Transformations

Geometry is the study of shapes of various sort. The simplest shape is the *point* . (It's quite difficult to explain what a point is, it is basically just a position, for instance, the very end of your nose is a point). Another simple shape is a *straight line* . A straight line is just the simplest shape joining two points together. A *plane* is a more complicated shape, it is a flat sheet, like a piece of paper or a wall. There are more complicated shapes, called *solids* , like a cube or a sphere. Here are some pictures of these things.



Simple geometric figures  
  
If you have a line and a plane, you can find the point where the line cuts through the plane. In fact, sometimes you can't find the intersection, because they don't meet and sometimes the line is inside the plane so they meet at every point on the line , but this doesn't happen in the cases we're interested in. We call this the *intersection* of the line and the plane. Here is a picture of what this looks like.



Intersection of a line and a plane

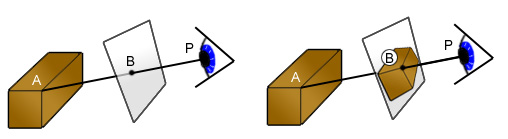
A *vector* is a mathematical way of representing a point. A vector is 3 numbers, usually called *x*, *y* and *z*. You can think of these numbers as how far you have to go in 3 different directions to get to a point. For instance, put one arm out pointing to the right, and the other pointing straight forward. I can now give you a vector and you'll be able to find the point I'm talking about. For instance, if I say *x*=3, *y*=1, *z*=5, you find the point by walking 3 metres in the direction of your right hand, then 1 metre in the direction of your left hand, and then getting ladder and climbing up 5 metres. Here is a picture of a vector.

#### 

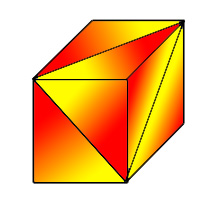
#### 3D Graphics

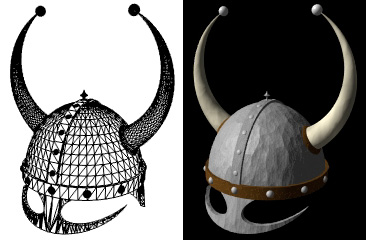
The basic idea of 3D graphics is to turn a mathematical description of a world into a picture of what that world would look like to someone inside the world. The mathematical description could be in the form of a list, for instance: *there is a box with centre*(2,4,7)*and sides of length 3, the colour of the box is a bluish grey* . To turn this into a picture, we also need to describe where the person is and what direction they are looking, for instance: *there is a person at*(10,10,10)*looking directly at the centre of the box* . From this we can construct what the world would look like to that person.

Imagine there is a painter whose eyes are at the point *P*. Imagine that he has a glass sheet which he is about to paint on. In the room he is painting, there is a wooden chest. One of the corners of the chest is at point *A*, and the painter wants to know where that corner of the chest should be on his glass sheet. The way he works it out is to draw a line *L* from his eyes (*P*) to the corner of the chest (*A*), then he works out where this line goes through the canvas, *B*. He can do this, because the glass sheet is a plane, and I mentioned that you can find the intersection of a line and a plane above. This point *B* is where the corner of the chest should be in his painting. He follows this rule for every bit of the chest, and ends up with a picture which looks exactly like the chest. Here are two pictures, the first one shows the painting when he has only painted the one corner of the chest, the second one shows what it looks like when he has painted the entire chest.



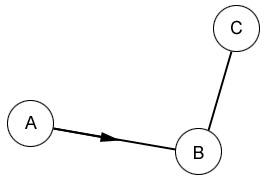
Projection on to a plane  
  
What I just described above is similar to what the computer is doing (50 times a second!) every time you run around shooting hideous monsters in *Quake* , although the details are slightly different. In computer games (at the moment) the description of the world is just a list of triangles and colours. The newest computer games are using more complicated descriptions of the world, using curved surfaces, NURBS and other strange sounding things, however in the end it always reduces to triangles. For instance, a box can be made using triangles as illustrated below.



Box made from triangles  
  
Here is a much more complicated example, using thousands of triangles. The first picture shows the triangles used, the second picture is what it looks like with colours put in. 

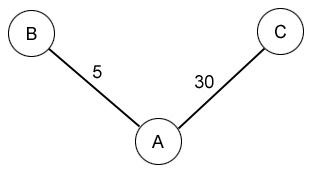
**Nodes, Edges and Graphs**

To explain how the computer works out the best route, you need to know what nodes , edges and graphs are. You may have heard of graphs before in maths, but they mean something slightly different here. The simplest example of nodes and graphs is a map of some cities, and the roads between them (or an underground map). Each city is a node, usually drawn as a circular blob. Each road is an edge, and connects two nodes (cities), these are usually drawn as straight lines. The whole collection of nodes and edges (cities and roads) is called a graph. Sometimes there is a one way road, called a directed edge , and we draw an arrow on it to show which way you can travel along it. For instance, if there are two cities A and B, and a line with an arrow from A to B, then we can travel from A to B, but not from B to A. Here is an example of a graph, you can't travel from B to A, but you can travel from A to B. You can't travel from C to A or from A to C, but you can travel from B to C and from C to B.



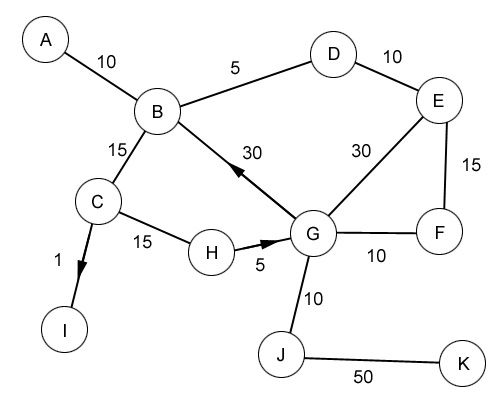
Graph with directed edges

To complicate things even further, we sometimes want to add something called a cost to each edge. The idea of a cost is that it indicates how much it would cost to travel down that edge. A simple example of this is shown below.



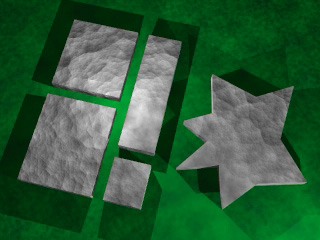
Graph with costs

In this graph, most of the people in city A want to get to city C, whereas only a few want to get to city B. Unfortunately, both roads are the same size, this means that there are long traffic jams on the road from A to C, it takes about 30 minutes to get there. To get from A to B is much easier as most people are going to C, so it only takes 5 minutes. The numbers written next to the edges indicate how long it takes to travel along that edge. Here is another (much more complicated) example.



More complicated graph

**Path finding**

Now you know all you need to know to be able to understand path finding. If you did the last exercise, you'll have already solved one example of the sort of problem computers have to solve to guide troops through complicated maps.****

How does all this stuff about graphs help the computer guide troops around levels? It makes a graph where every interesting point is a node on the graph, and every way of walking from one node to another is an edge, then it solves the problem you solved above to guide the troops. There are some complications. For starters, what are the interesting points? You might think that every position on the entire level is interesting, but for most games this would lead to hundreds of thousands of interesting points, and finding the path would take years. Instead, the people making the game decide where the interesting points are. For instance, if there is a wide open expanse (a big field perhaps), you don't need a node at every point on the field, because the troops can walk in a straight line across the field. Basically, you only need nodes around obstacles. Here is an example of a map of a level seen from above.

A map of a level in a game, the green areas are the ground, the grey areas are the buildings

Once you have chosen your interesting points, you need to connect them up with edges, you can only connect up those nodes which don't have an obstacle between them.