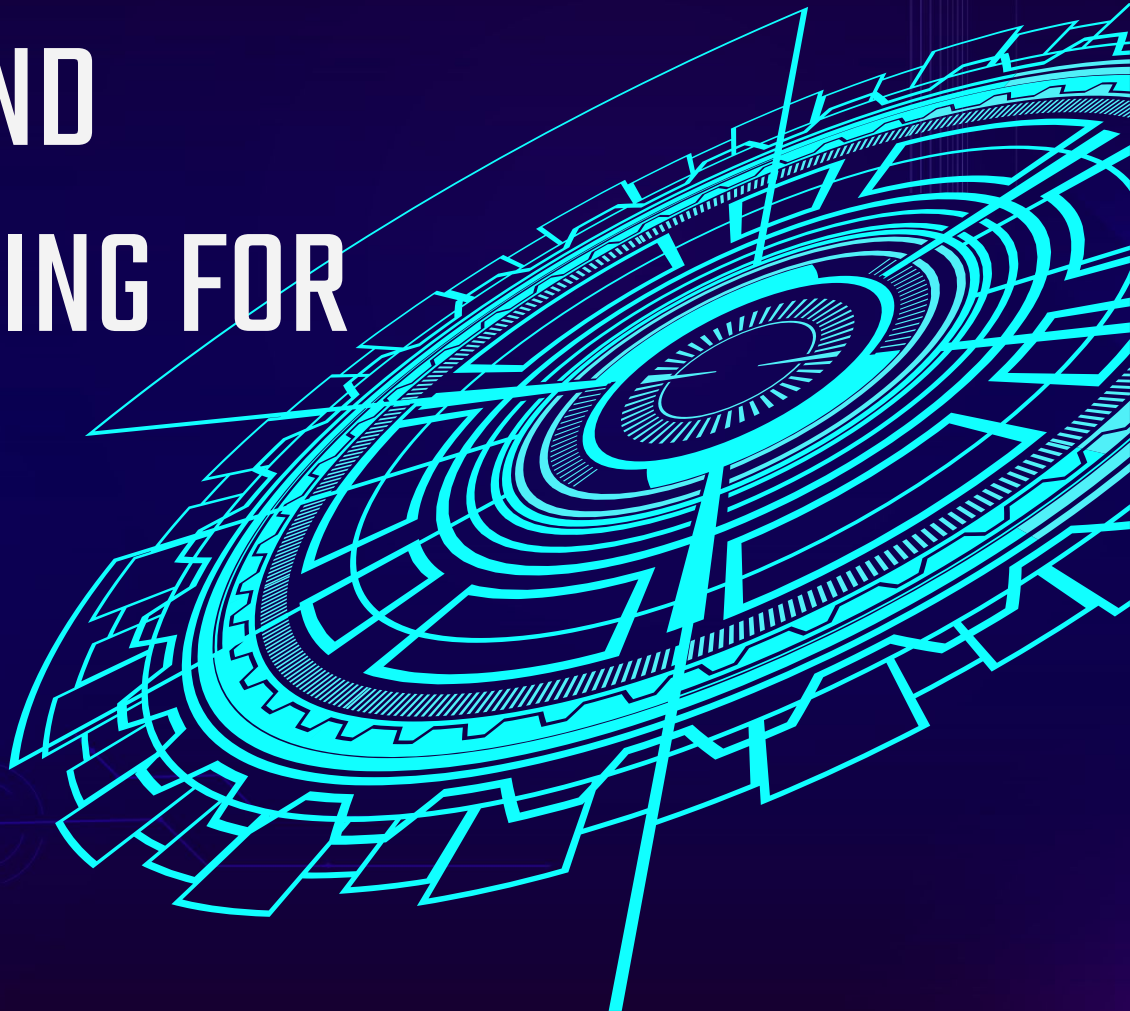
3D INDUSTRY AND PROBLEM-SOLVING FOR PROGRAMMERS

A (not very) gentle introduction



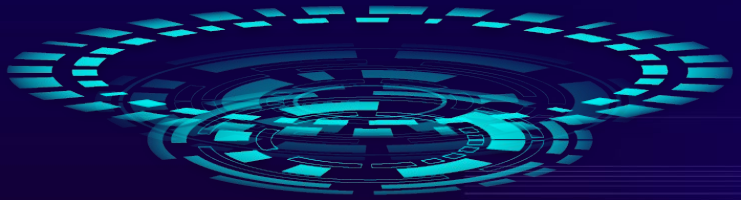
$whoami

Konstantinos Benos

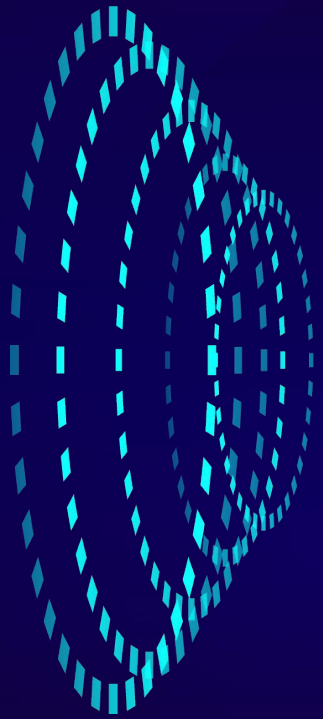
- Gameplay & Graphics programmer @ Supercourse ELT (2018-19)
- Security Vulnerability Researcher @ Census Labs (2017-18)
- System Administrator @ IT AUTh (2015-16)
- CS Undergraduate @ AUTH (2013-??)

TABLE OF CONTENTS

INTRODUCTION



WHAT DO WE MEAN WHEN WE REFER TO THE TERM “ 3D INDUSTRY ” ?

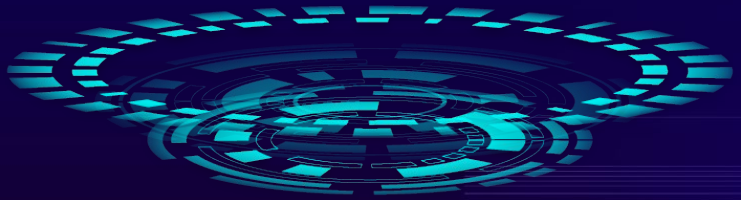


3D INDUSTRY

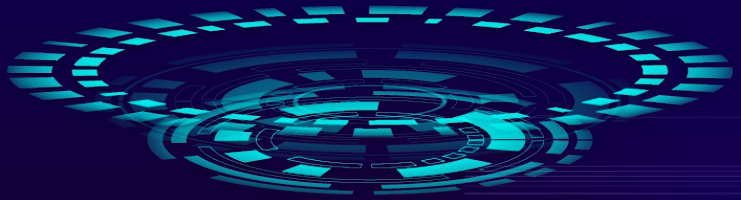
Every field with an immediate or indirect goal of visualizing a multi-dimensional world scenario:

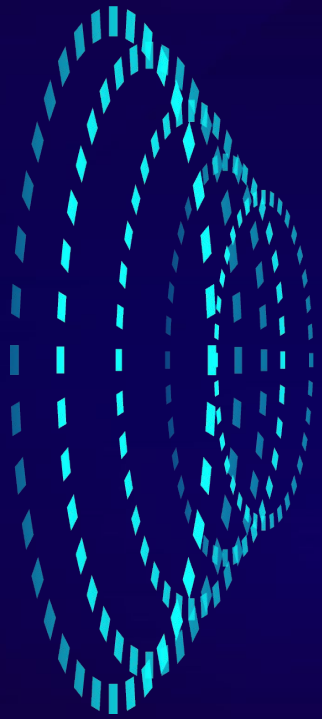
- Video Games
- Animation Movies
- Robotics
- Automobiles
- Manufacturing
- ...

PROGRAMMING AND 3D



WHAT IS THE ROLE OF A PROGRAMMER IN 3D FIELDS ?

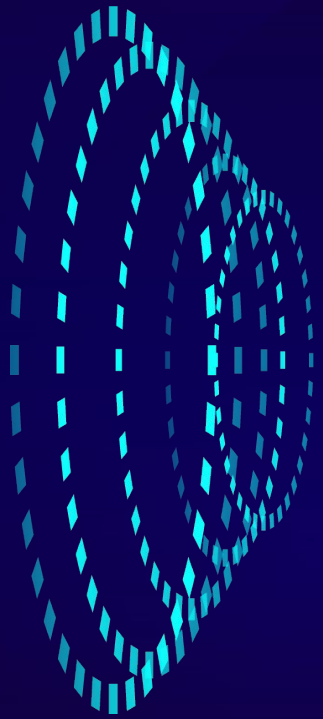




PROGRAMMING

During the early days of 3D applications being a programmer for 3D mostly meant that you would be defined as a **graphics programmer**.

However, as 3D tools became more accessible and the industries grew in size, a variety of disciplines have emerged.

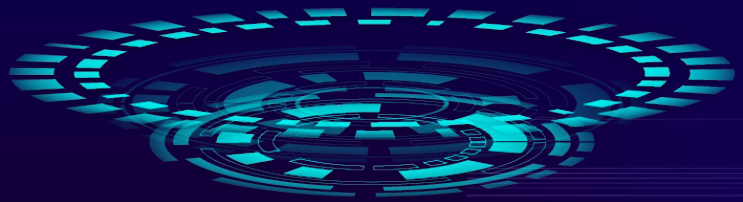


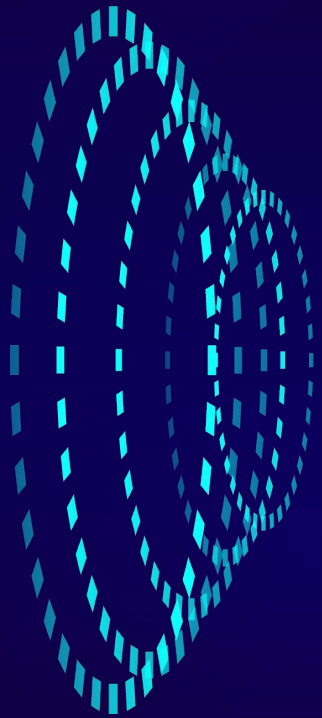
PROGRAMMING

- **Gameplay** Programmer
- **Animation** Programmer
- **Graphics/Engine** Programmer
- **Camera** Programmer
- **Networking** Programmer
- **Physics** Programmer
- **AI** Programmer
- **Tools** Programmer
- **UI** Programmer
- **Graphics/VFX** Programmer
- **Audio** Programmer
- **Technical Artist** (⋆)
- ...

QUESTION

What is the required skillset for a 3D programmer ?

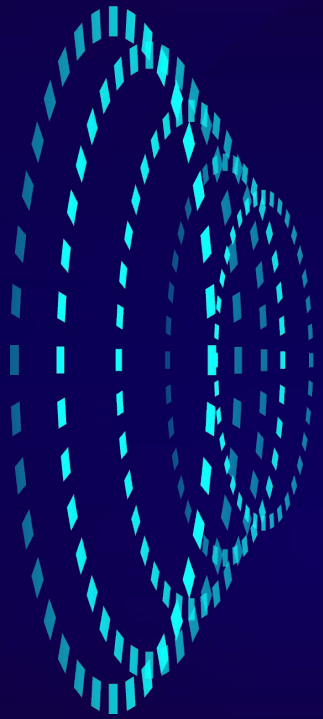




SKILLSET

It really depends on specific discipline and company size/perspective.

In general, and for big sized companies, most of the aforementioned positions have a few things in common:

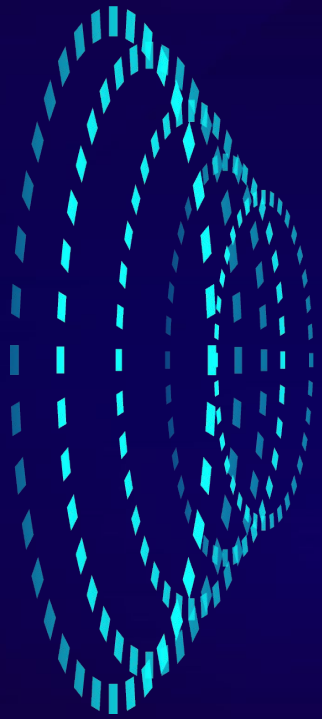


SKILLSET

It really depends on specific discipline and company size/perspective.

In general, and for big sized companies, most of the aforementioned positions have a few things in common:

1. **3D Math,** especially **Linear Algebra (!!!)**

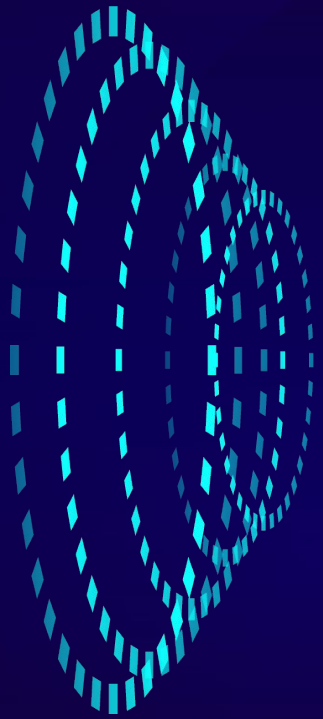


SKILLSET

It really depends on specific discipline and company size/perspective.

In general, and for big sized companies, most of the aforementioned positions have a few things in common:

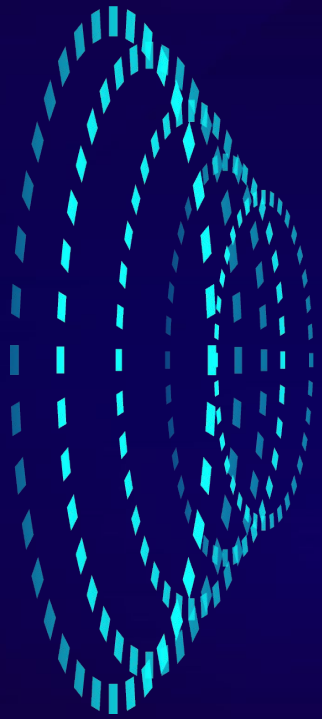
1. **3D Math,** especially **Linear Algebra (!!!)**
2. **Programming (wow!)**



SKILLSET

Additionally, some other requirements depending on programming position:

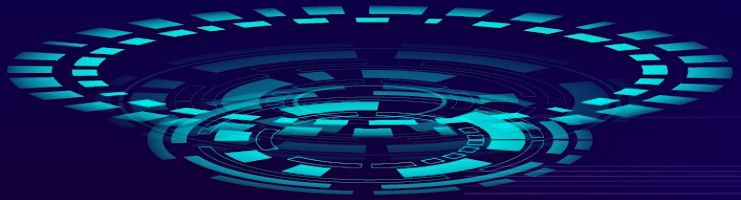
- **Graphics**: Modern Graphics APIs (OpenGL, DirectX, Vulkan) and techniques
- **Networking**: Sockets and understanding of networking concepts in general
- **AI**: State machines, Pathfinding, Learning techniques
- **Tools**: Knowledge of production software development tools and best practices
- **Camera**: Knowledge of camera techniques used in cinema and the ability to understand the reasoning behind specific camera motions

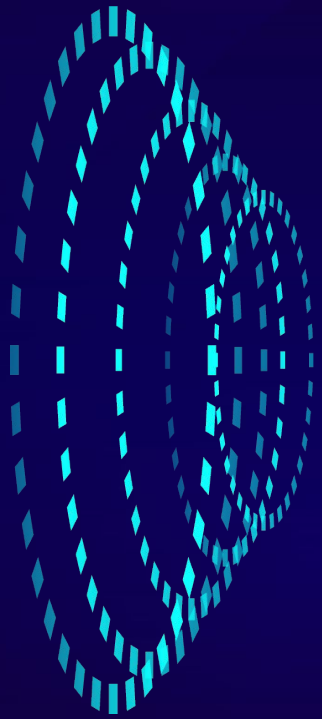


SKILLSET

You can find more about the requirements of a specific position by comparing the “career” sections of different companies.

What are the programming languages used ?



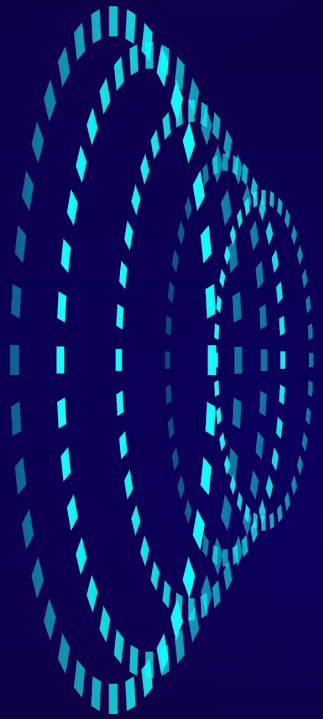


LANGUAGES

Again, depends on company and role.

Commonly used ones are:

- **OOP**: C++, C#
- **Scripting**: Python, Company-Custom



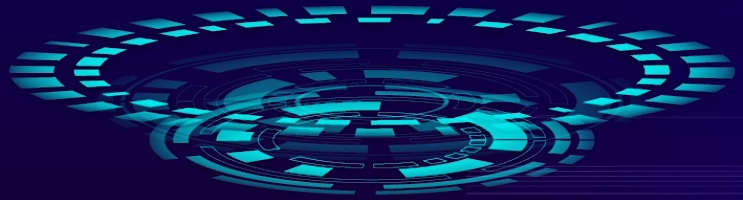
LANGUAGES

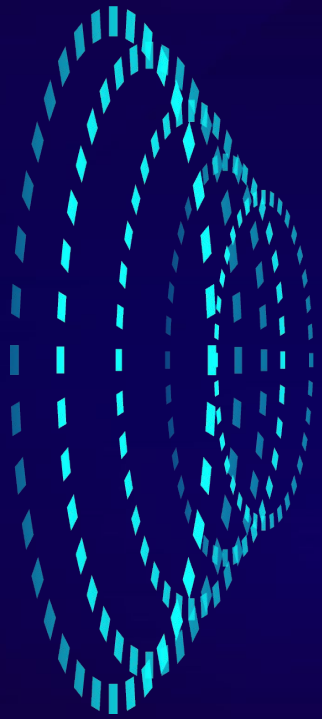
It’s worth noting that many visual coding tools have emerged in the last few years.

Some examples:

- Blueprints, Material Editor (Unreal Engine)
- Bolt, Shader Graph (Unity)

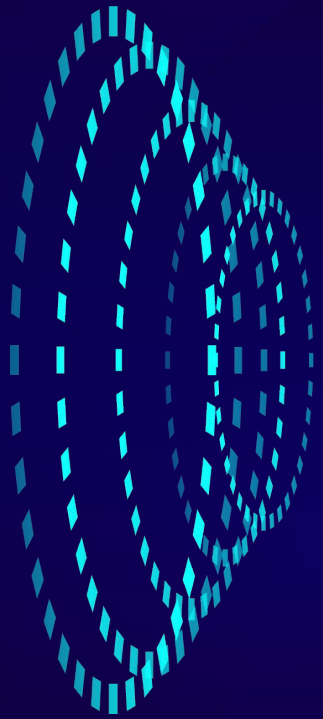
What are some popular 3D tools ?





3D TOOLS

- **Modelling**: 3DSMax, Maya, Cinema4D, Blender
- **Animations**: Maya, Cinema4D, Blender
- **Sculpting**: Zbrush, Blender
- **Game Engines**: Unreal, Unity, CryEngine, GoDot, GameMaker Studio
- **Image Editing**: Photoshop
- **VFX**: Houdini



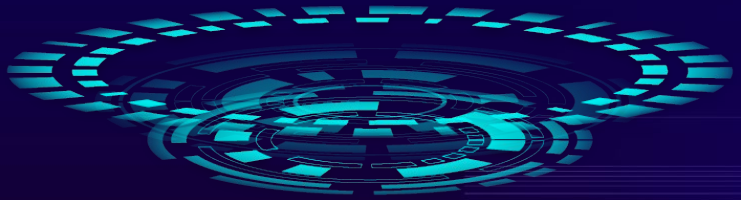
3D TOOLS

Keep in mind that as a programmer you will, most probably, not need to interact with the aforementioned artist tools unless you are a tools or, in some cases, graphics programmer.

It’s also worth noting that, if you are aiming for a big gaming company, you are not required to master any of the commercial engines (most big companies use in-house developed engines).

However, generic knowledge of how an engine operates is always important.

THE GAMES INDUSTRY

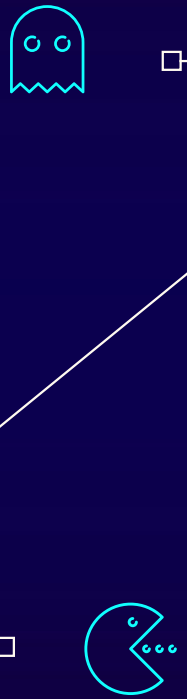


GOOD

Fast growing sector that attracts more and more people each year, both developers and gamers.

Games are considered the storytelling medium of our age.

Games are awesome!



BAD

Hard to get in.

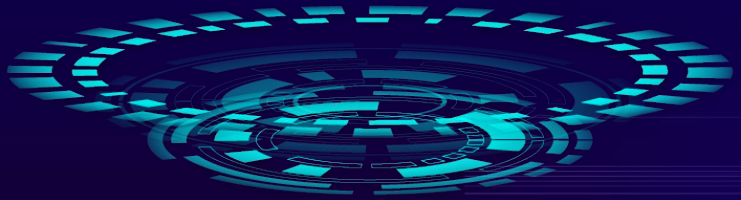
Difficult time-planning.

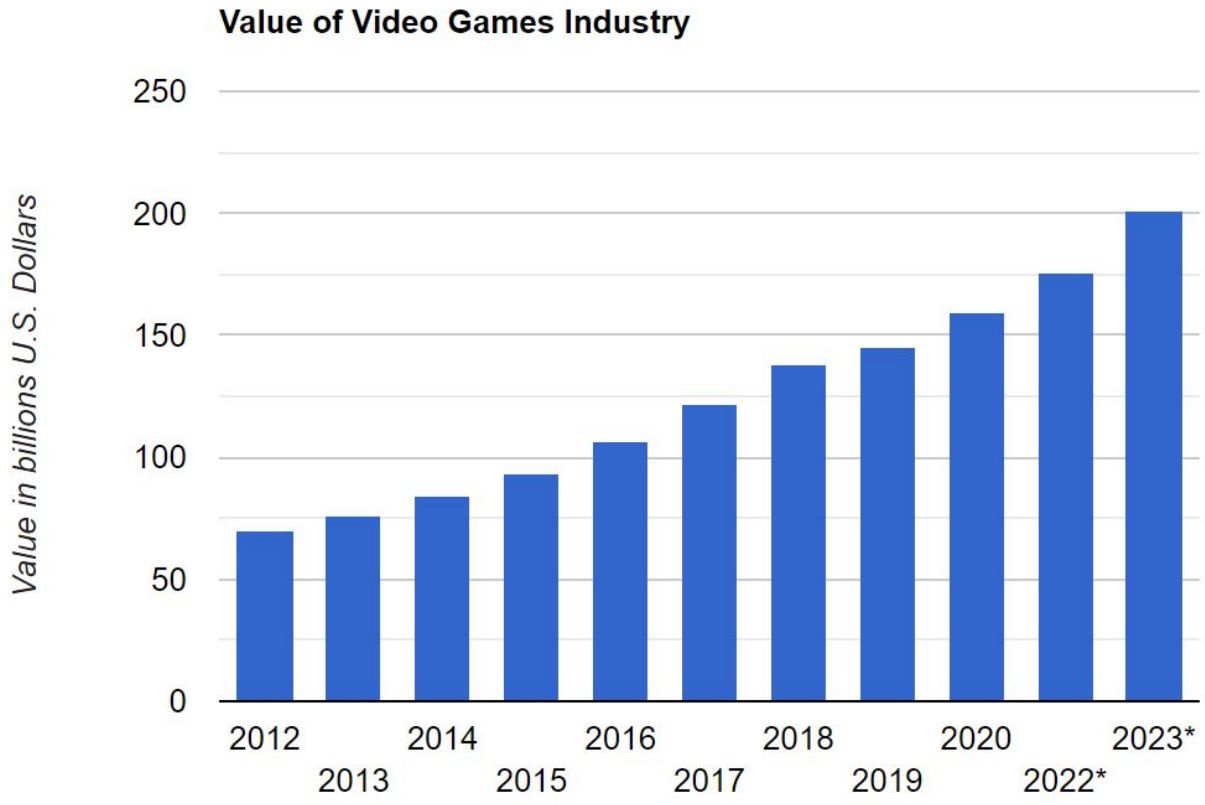
Big competition.

Crunch culture.

Risky investment.

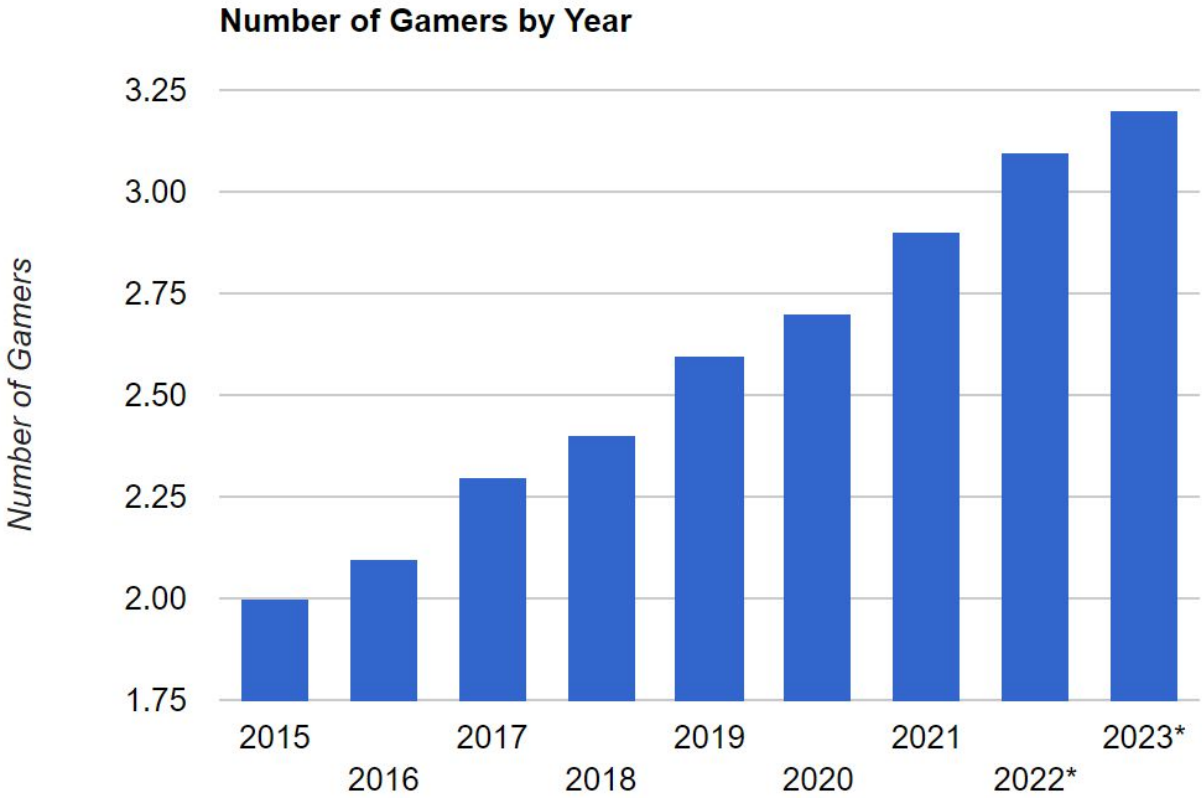
Some statistics





sources:

Statista, NewZoo

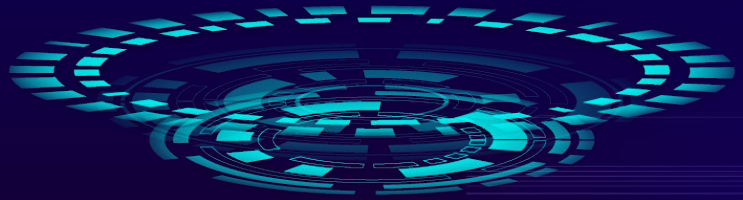


sources:

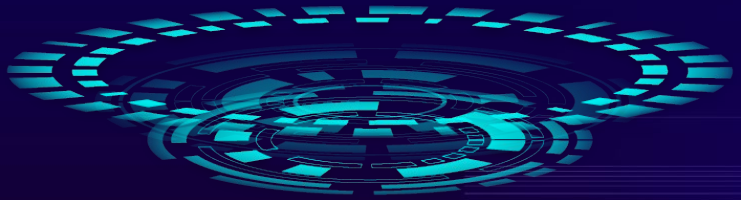
Statista, NewZoo

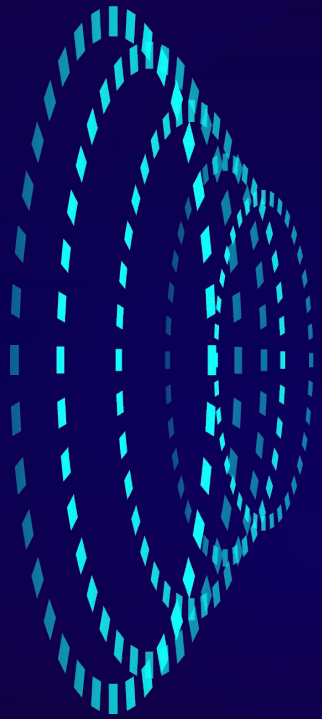
DISCUSSION

How do u explain the growth in gamers ?



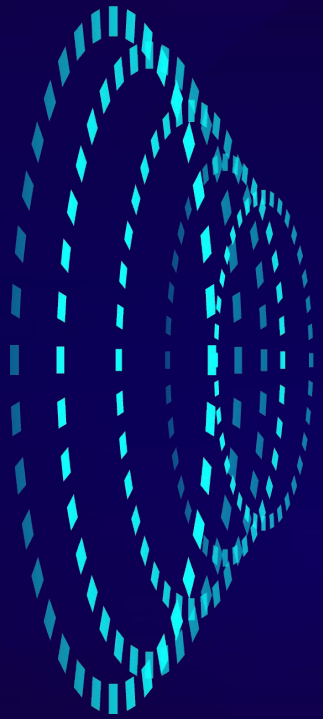
REAL WORLD PROBLEMS





PROBLEMS

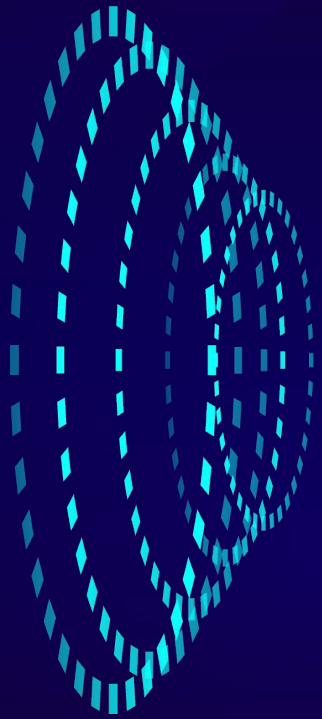
In the following slides we will discuss some real world problems, commonly encountered in game programming.



PROBLEMS

Some things to keep in mind:

1. No programming knowledge is required.
2. A single solution is provided but most problems can be solved in a number of ways.
3. This type of problem solving is required by other 3D programming disciplines as well and is not limited to game programming.

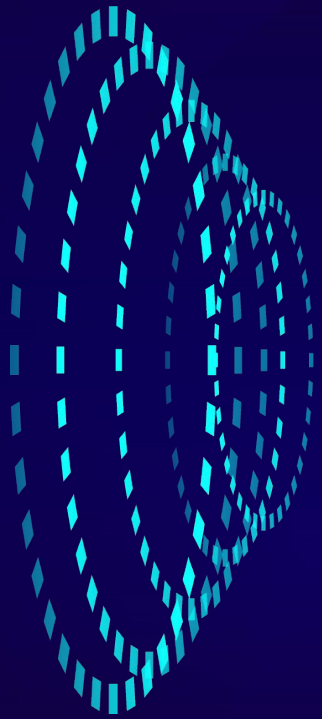


PROBLEMS

Most importantly:

Try to participate!

Don’t be afraid of failure.
We are all here to learn.



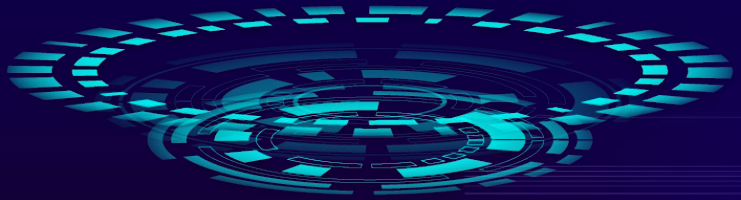
PROBLEMS

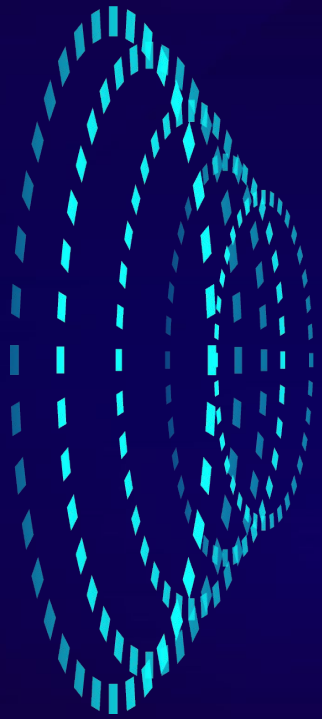
Some definitions we need:

- Every object/model in a game stores some information regarding its position, rotation, scale and more…
- The information needed to solve each problem is provided accordingly.
- If you feel like you need some extra declarations let me know.

PROBLEM #1

Strategy game unit movement

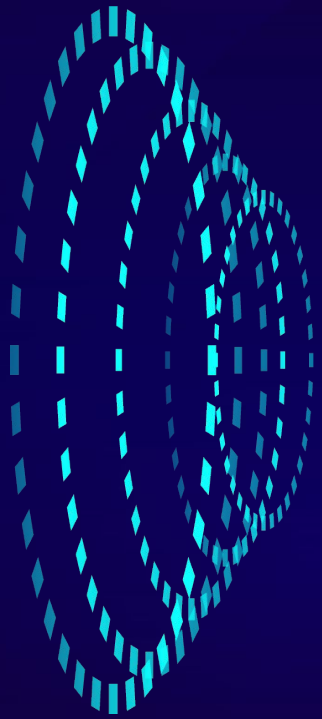




PROBLEM #1

It’s your first day as an intern gameplay programmer at BestGamesEverMade studio which is currently working on a real time strategy title.

Your first task has arrived.



PROBLEM #1

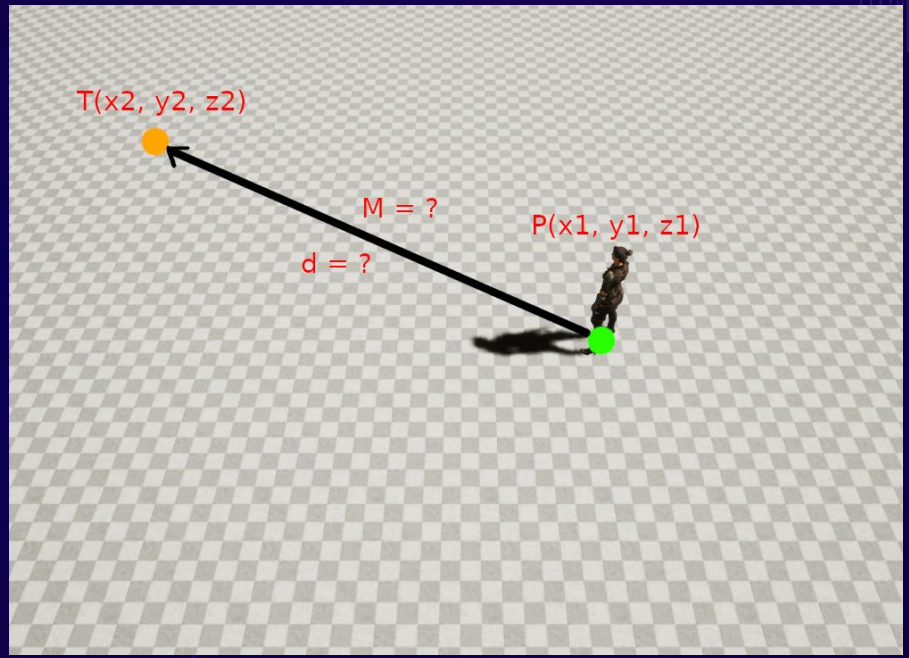
When a player selects a unit he can right click anywhere in the world to make the unit move to that position. Given:

A units position vector:
P(x1, y1, z1)

The right mouse button position vector:
T(x2, y2, z2)

Find the distance the unit has to travel and the direction vector of the movement: **d = ?, M = ?**

PROBLEM #1



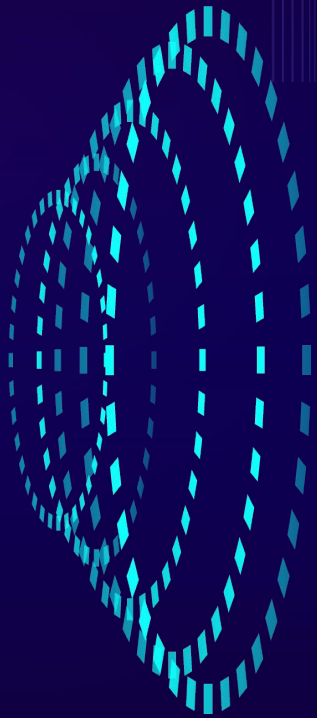
SOLUTION

We calculate the euclidean distance between the two positions:

$$d = \text{sqrt}((x2-x1)^2 + (y2-y1)^2 + (z2-z1)^2)$$

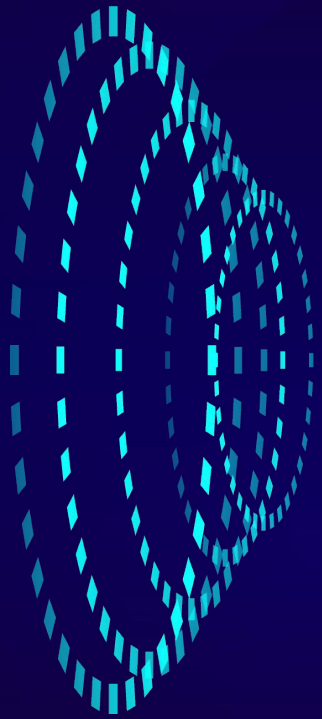
For the vector M:

$$M = T - P = (x2-x1, y2-y1, z2-z1)$$



PROBLEM #2

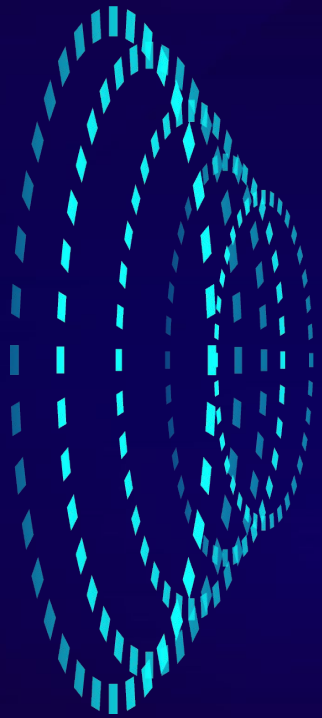
Environment Damage Detection



PROBLEM #2

You walk up to your manager to showcase the amazing work you've done with the unit movement.

Unfortunately he was too busy to evaluate your work so he gives you the next task instead.



PROBLEM #2

In some areas of the world map there exist circular pools of lava. If a unit stands in the pool it takes damage. The closer it is to the center of the pool the more damage it takes.
Given:

The position of a unit in the world:

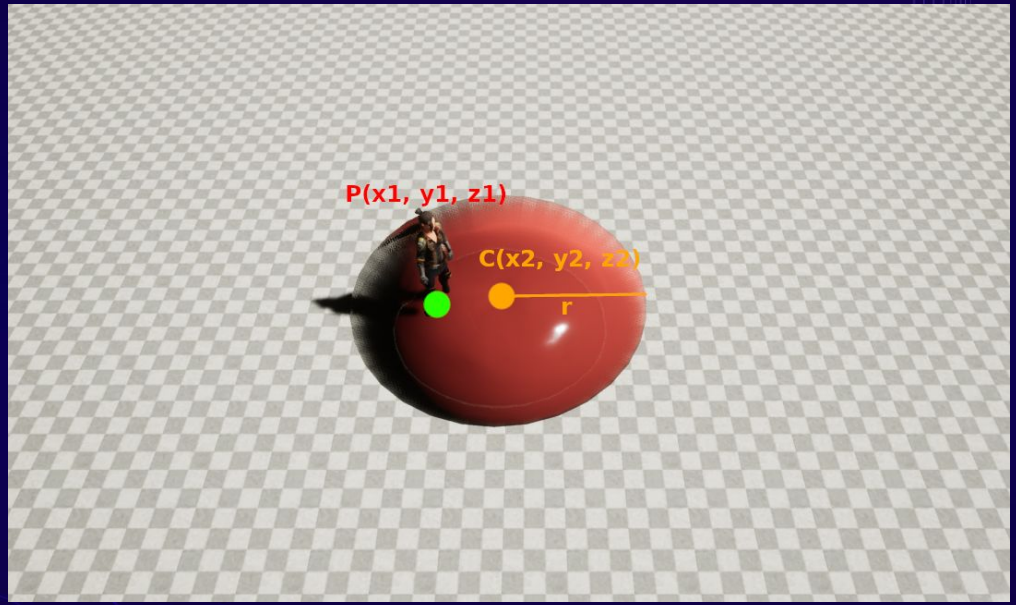
U(x1, y1, z1)

The center and the radius of the circle of a lava pool:

C(x2, y2, z2), r

Find: a) if the unit is in the pool
b) how close it stands to its center

PROBLEM #2



SOLUTION

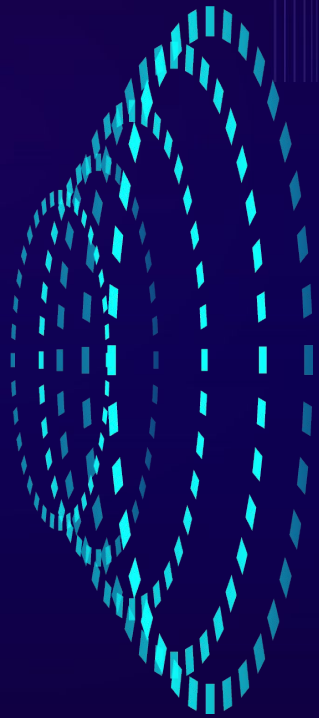
Similarly as before, we calculate the vector from the center of the pool to the player:

CP = (x1-x2, y1-y2, z1-z2)

Its magnitude:

|CP| = sqrt((x1-x2)² + (y1-y2)² + (z1-z2)²)

If **|CP| > r** then the unit stands outside the pool, otherwise it stands inside and has to take damage.

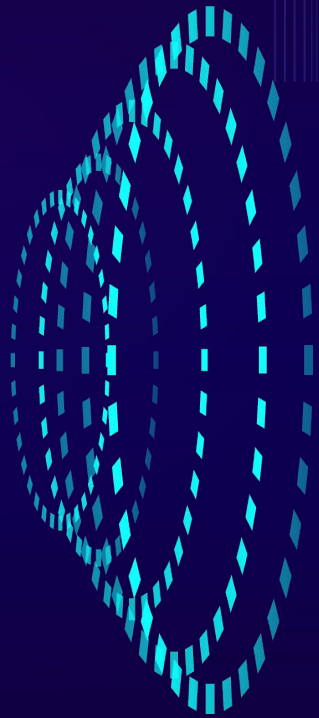


SOLUTION

To calculate how far in the pool it stands we subtract the radius from the vectors magnitude:

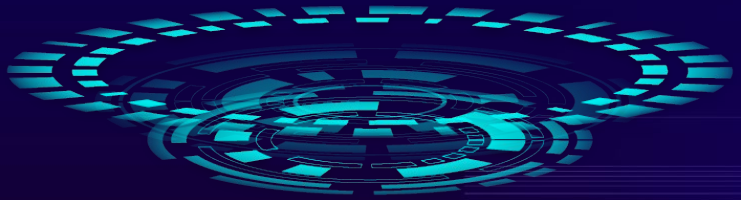
$$s = |CP| - r$$

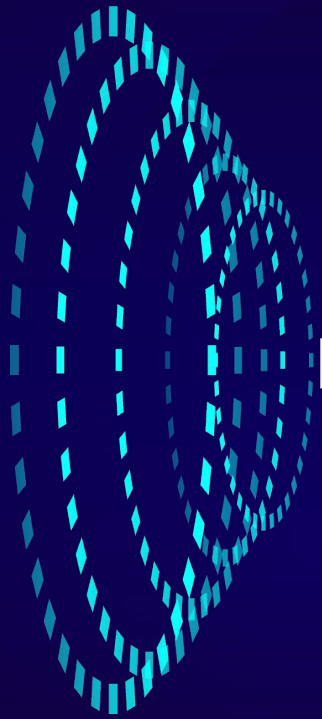
The value of s is $-r \leq s \leq 0$ if the unit is in the pool, $s > 0$ otherwise. Note that this formula is called the **Signed Distance Field** between a circle and a point. SDF functions exist for many geometric primitives and are, in general, very useful.



PROBLEM #3

Enemy AI detection

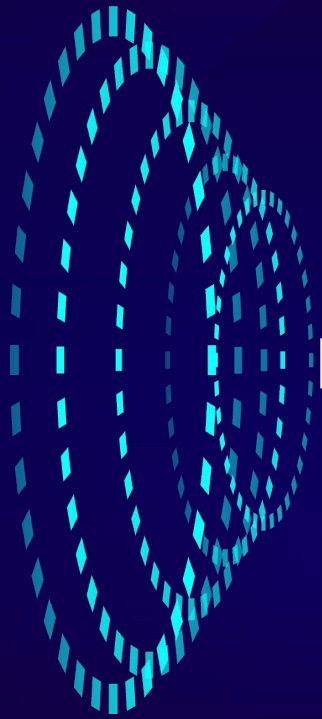




PROBLEM #3

After solving the lava pool problem you feel ready to step up your game.

You walk up to your manager and ask for a more complicated task. He happily grants you your wish.



PROBLEM #3

The enemy AI works with a detection mechanism. When one of your units gets detected by an enemy unit, the enemy unit has to turn and look at yours before engaging in combat. Given:

Your units **position vector** the moment of detection:

P = (x1, y1, z1)

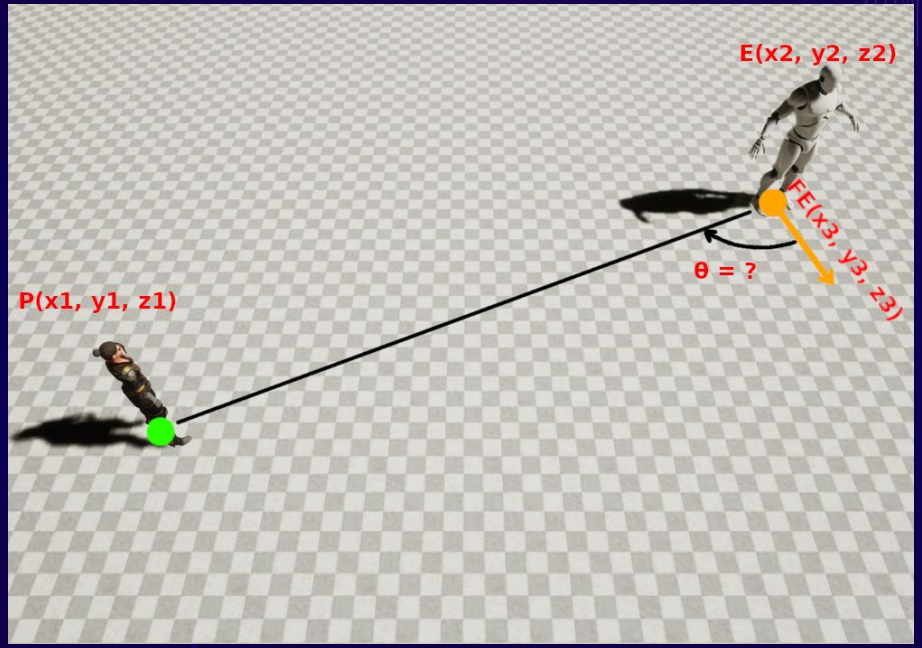
The enemy units **position** and **forward vector** at the moment of detection:

E = (x2, y2, z2), FE = (x3, y3, z3)

Assume that **z1 = z2 = z3 = 0**

Find the angle that the enemy has to turn to look at your unit: **θ = ?**

PROBLEM #3

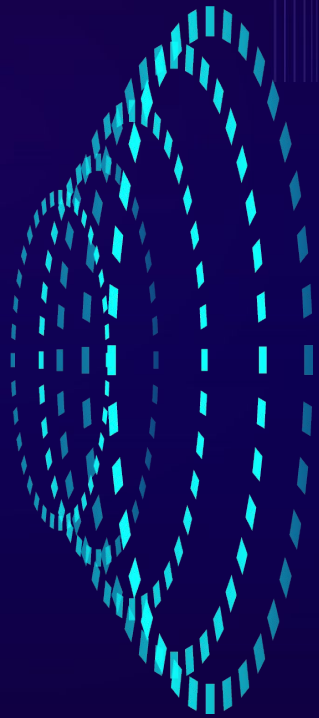


SOLUTION

First we need the enemy-to-player vector. This is the target forward vector for the enemy:

FT = P - E = (x1-x2, y1-y2, z1-z2) = (x4, y4, z4)

Now we need to calculate the angle between the current and the target forward vector.



SOLUTION

The formula for the dot product of two vectors:

A = (x, y, z), B = (a, b, c)

A•B = xa + yb + zc

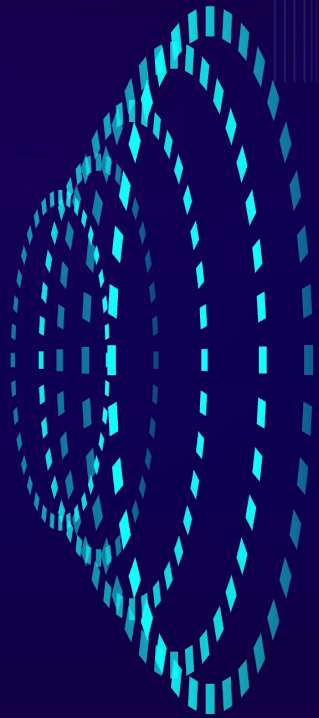
Also, the following property holds:

A•B = |A||B|cosθ => cosθ = (A•B)/(|A||B|)

In our case:

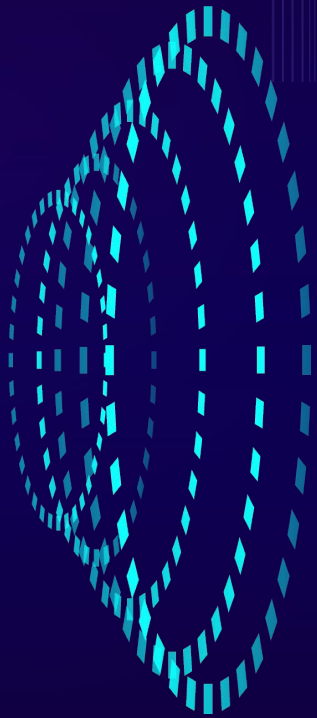
cosθ = (FE•FT)/(|FE||FT|) = …

To derive the angle from the cos we can use some approximation for the arccos (Taylor polynomial, Ronald Doelfler formula). The result is in radians.



One thing remains…

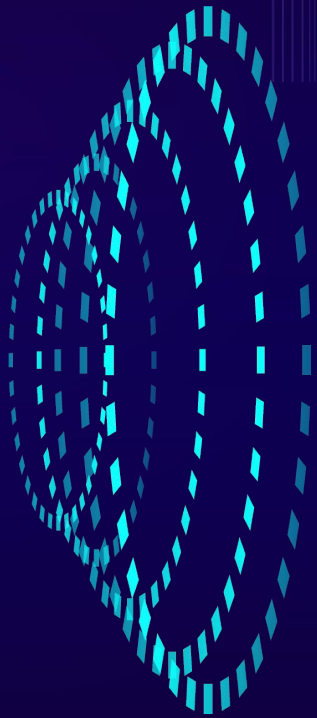
SOLUTION



SOLUTION

One thing remains…

Direction to rotate?



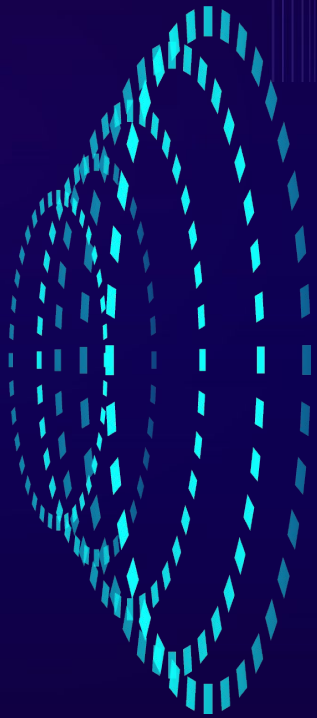
SOLUTION

We calculate the cross product:

$$FE \times FT = \begin{vmatrix} i & j & k \\ x3 & y3 & z3 \\ x4 & y4 & z4 \end{vmatrix}$$

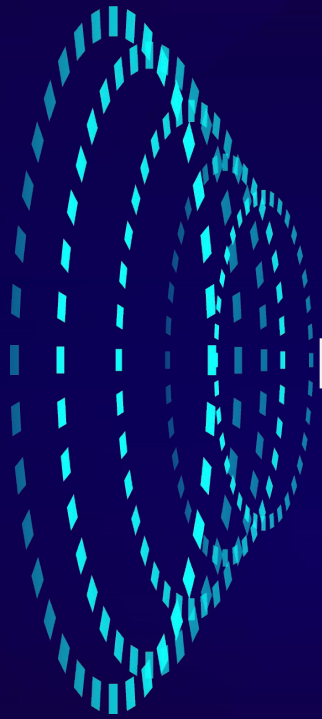
$$= (y3z4-y4z3,\ x4z3-x3z4,\ x3y4-x4y3)$$

Given that the vectors FE, FT are coplanar with the xy defined world plane (z3 = z4 = 0) we can derive that if **x3y4-x4y3 > 0** then we rotate positively, otherwise negatively. This method is also Hand System independent.



PROBLEM #4

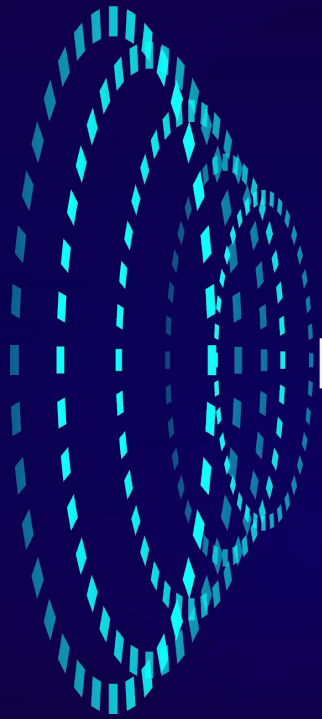
Linear Projectile Prediction



PROBLEM #4

Time passes by. The strategy title of BestGamesEverMade studio gets released and it’s a big success. You get promoted to a senior gameplay programmer.

The studio moves on to its next project. A top-down rpg. You are in charge of the projectile system and your first task is…



PROBLEM #4

In this game there are many types of enemies. One of them is the caster/mage. The caster throws a fireball in constant time intervals that tries to hit the player by predicting his position. Given:

The players current **position and velocity** vectors at the moment of a fireball launch:

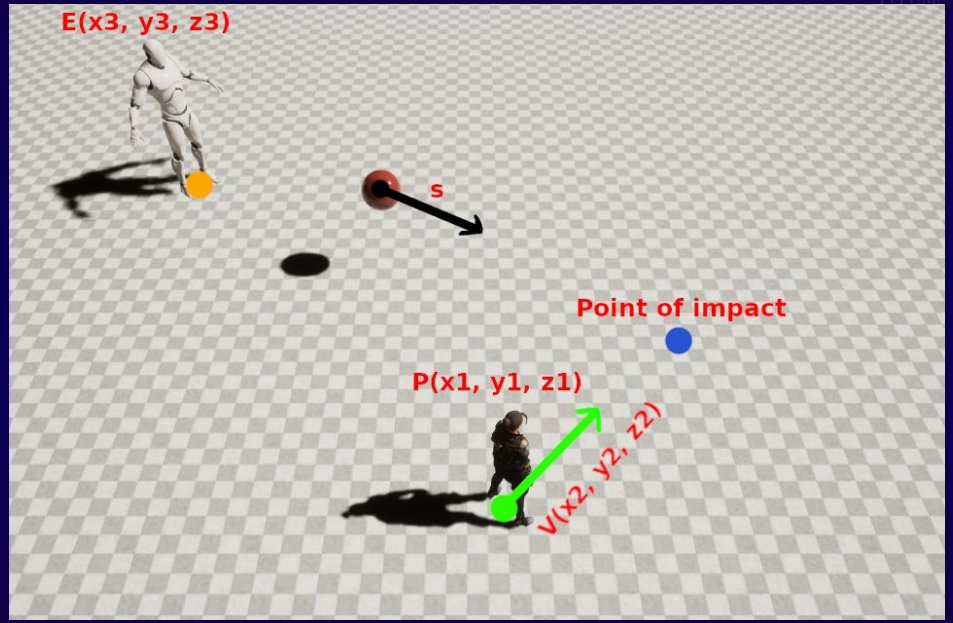
P(x1, y1, z1), V(x2, y2, z2)

The casters **position** and the **fireball speed**:

E(x3, y3, z3), s

Find the position needed to launch the fireball towards to in order to hit the player if his velocity vector doesn’t change. Also find the fireball direction vector.

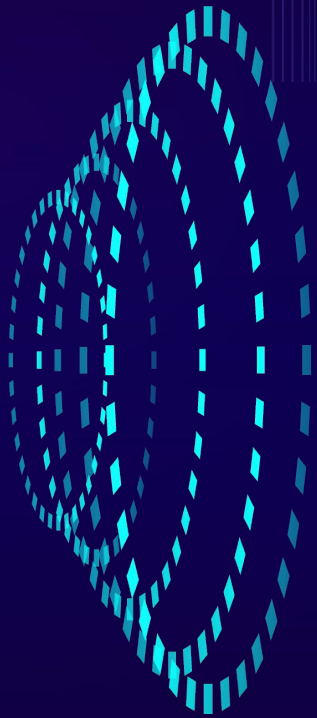
PROBLEM #4



SOLUTION

First we need to find the line equation of the players movement.

P’ = P + Vt, where **t** is the time



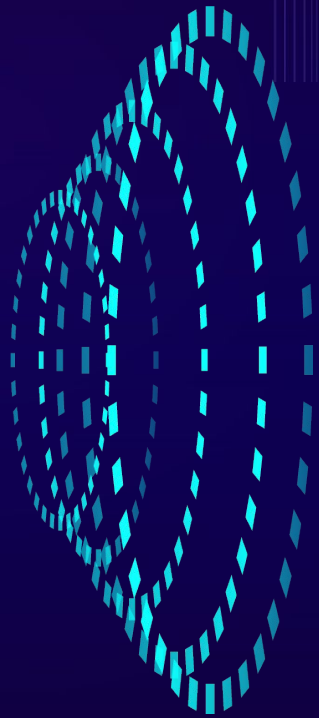
SOLUTION

First we need to find the line equation of the players movement.

P’ = P + Vt, where **t** is the time

Then we need to find a system of circles for the fireball position (we can launch the fireball in any direction, hence circles and not line).

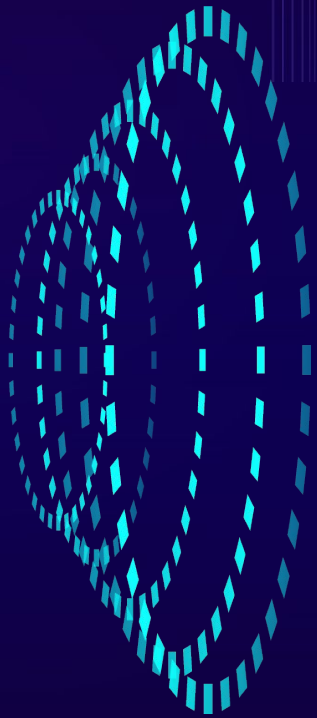
$(F - E)^2 = (st)^2$, where **t** is the time and **F** is the fireball position.



SOLUTION

(1): $P' = P + Vt$
(2): $(F - E)^2 = (st)^2$

We need **(3): $F = P'$**.

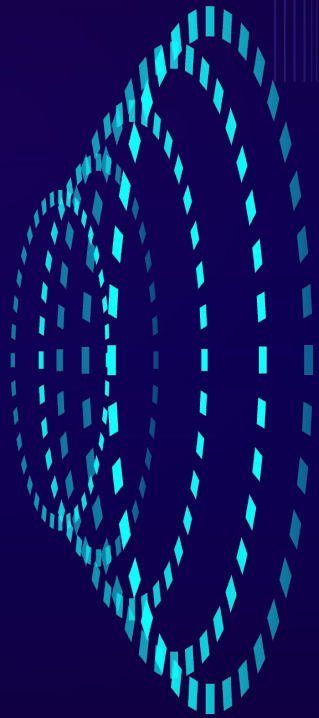


SOLUTION

(1): $P' = P + Vt$
(2): $(F - E)^2 = (st)^2$

We need **(3): $F = P'$**.
Replace **P'** with **F** and solve for **t**:

(4): (1),(3): $F = P + Vt$
(5): $F^2 - 2FE + E^2 = (st)^2$
(4),(5): $(P+Vt)^2 - s^2t^2 - 2(P+Vt)E + E^2 = 0$
$\Rightarrow P^2 + 2PVt + V^2t^2 - s^2t^2 - 2PE - 2VtE + E^2 = 0$
$\Rightarrow (V^2-s^2)t^2 + 2(PV-VE)t + (E^2-2PE+P^2) = 0$



SOLUTION

(1): P’ = P + Vt
(2): $(F - E)^2 = (st)^2$

We need **(3): F = P’**.
Replace **P’** with **F** and solve for **t**:

(4): (1),(3): F = P + Vt
(5): $F^2 - 2FE + E^2 = (st)^2$
(4),(5): $(P+Vt)^2 - s^2t^2 - 2(P+Vt)E + E^2 = 0$
$\Rightarrow P^2 + 2PVt + V^2t^2 - s^2t^2 - 2PE - 2VtE + E^2 = 0$
$\Rightarrow (V^2-s^2)t^2 + 2(PV-VE)t + (E^2-2PE+P^2) = 0$

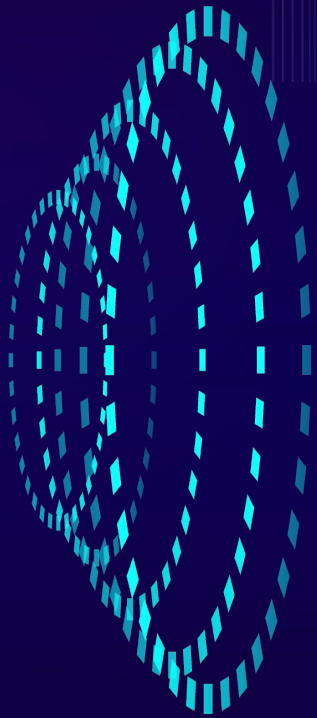
Quadratic equation with:
$a = (V^2-s^2)$
$b = 2V(P-E)$
$c = (E-P)^2$

We calculate **Δ** and then we find **t**.
(Any problems here?)

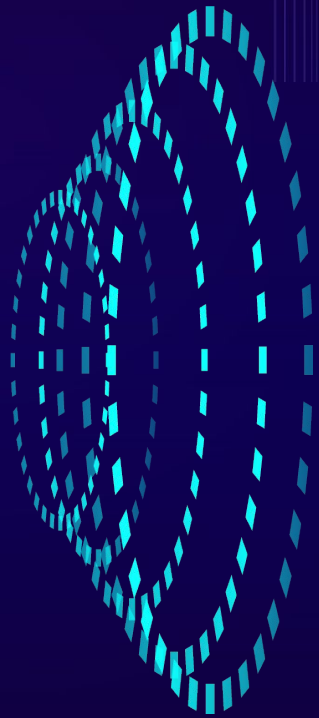
Q: How do we solve equations of vectors?

Q: What happens for different values of **Δ**?

SOLUTION



SOLUTION



Q: How do we solve equations of vectors?
A: **a)** We treat multiplications as dot products
or
b) We represent them in a basis and we solve for each basis component individually

Q: What happens for different values of **Δ**?
A: **Δ<0:** That means we are never going to hit (the velocity of the player is sufficient to never get hit by the fireball)
Δ=0: Single or zero chances of hit (*)
Δ>0: Two,one or zero chances to hit (*)

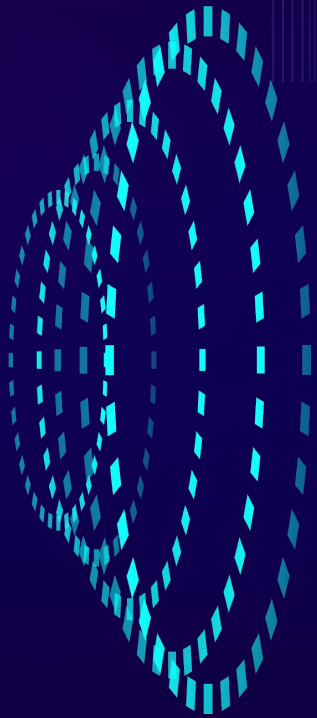
(*): Depends on velocity and current position of player

SOLUTION

After we found **t** we can replace it in the player line equation to find the predicted point of impact:

P’ = P + Vt

All that’s left is to find the direction to throw the projectile at:



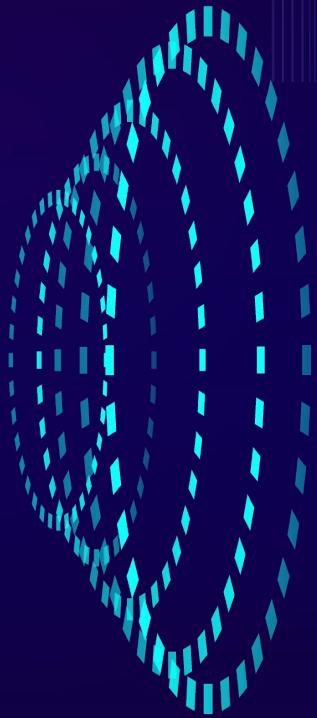
SOLUTION

After we found **t** we can replace it in the player line equation to find the predicted point of impact:

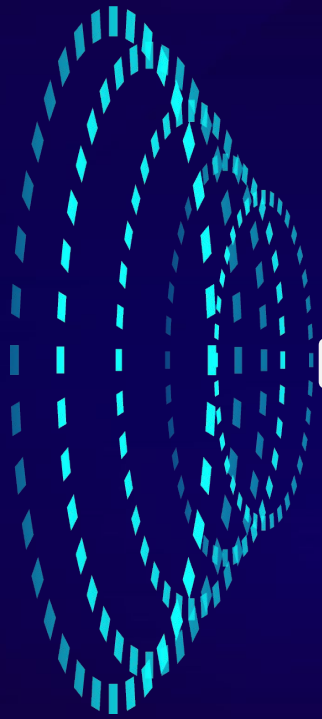
P’ = P + Vt

All that’s left is to find the direction to throw the projectile at:

D = P’ – E



CONCLUSIONS



CONCLUSIONS

We talked about the 3D industry, as seen from a programmers perspective, with a greater focus on game development.

We’ve seen the importance of math and dealt with some real world encounters.

If you find this type of problem solving interesting and you ever thought of becoming a games/3D programmer I hope that this presentation helped you find a direction to move towards to in the future.

QUESTIONS AND CONVERSATION

RESOURCES

There are endless resources out there to learn about 3D programming. Some I personally used and found invaluable:

- GDC (Game Developers Conference):
  https://www.gdcvault.com/
  (also many talks on youtube by acknowledged developers)
- SigGraph (Graphics Conference):
  https://www.siggraph.org/
- LearnOpenGL (by Joey de Vries):
  https://learnopengl.com/
- Mathematics for 3D Game Programming and Computer Graphics (book by Eric Lengyel):
  https://www.mathfor3dgameprogramming.com/
- Graphics tutorials by Inigo Quilez:
  https://iquilezles.org/
- Unity programming tutorials (by Catlike Coding):
  https://catlikecoding.com/unity/tutorials/
- Unreal programming tutorials (by Tom Looman):
  https://www.tomlooman.com/

SKG GameDevs

We are managing an active community of Game Developers of Thessaloniki on discord and facebook.

It is a place for everyone interested in game dev (programmers, artists, sound engineers, etc) to share their knowledge and opinions.

We were also holding meetups regularly, prior to covid-19. Currently we are doing some virtual meetings through discord occasionally.

If you are interested in joining contact me and I'll send you the invites.

THANKS

If you have further questions or you want to share your thoughts, about this presentation or anything 3D related, don’t hesitate to contact me.


Konstantinos Benos


@deg3x


mpenos.ks@gmail.com

CREDITS

- Presentation template by Slidesgo
- Icons by Flaticon
- Infographics by Freepik
- Author introduction slide photo created by Freepik
- Text & Image slide photo created by Freepik.com