# Computación Gráfica

Ing. Gabriel Ávila, MSc.

# Competencia

Domina técnicas de:

- **Visualización 3D**
- **Modelado de superficies**
- **Detección de superficies ocultas**

Enriqueciendo los entornos con la implementación de:

- **Modelos iluminación**
- **Representación superficial**

Aplicando conceptos computacionales, tendrá la **capacidad de generar aplicaciones gráficas** a partir de una fundamentación teórica y no como un usuario.

# ¿De qué trata?

Conocer y solucionar problemáticas relacionadas con el desarrollo de proyectos en computación gráfica, partiendo de los conceptos teóricos y aplicándolos de manera práctica utilizando diversas herramientas.

# Herramientas



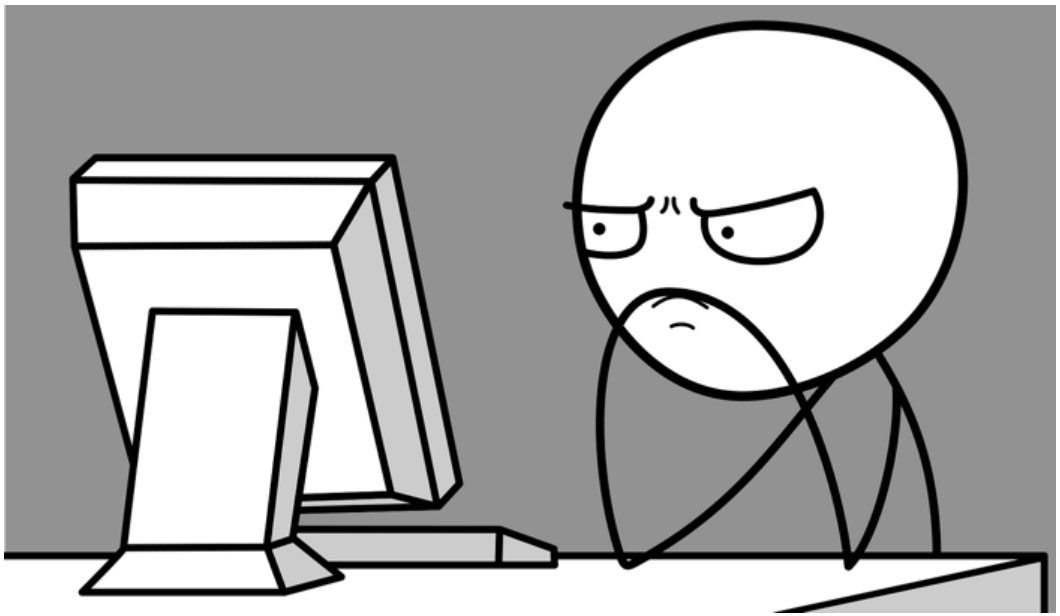
three.js

# Bibliografía recomendada


- ***Interactive Computer Graphics: A Top-Down Approach with Shader-Based OpenGL.*** *Edward Angel, Sexta Edición, 2011. (biblioteca UMNG).*
- ***Computer Graphics with Openg GL.*** *Donal Hearn, M. Pauline Baker. Tercera Edición, Editorial Prentice Hall, 2003. (biblioteca UMNG).*
- ***Graficos por Computador con Openg GL.*** *Donal Hearn, M. Pauline Baker. Tercera Edición, Editorial Prentice Hall, 2003.*
- ***Computer Graphics: C Version.*** *Donal Hearn, M. Pauline Baker. Segunda Edición, Editorial Prentice Hall, 1997. (biblioteca UMNG).*
- ***Computer Graphics: Principles and Practice****. James D. Foley, Andries van Dam, Steven K. Feiner, John F. Hughes. Editorial Addison Wesley, 2014. (biblioteca UMNG).*
- ***Real time Rendering. 2nd Edition.*** *Tomas Akenine-Möller y Eric Haines. Editorial Ak Peters. 2002*
- ***OpenGL Graphics Through Applications.*** *Robert Whitrow, Springer, 2008.*
- ***Introduction to Computer Graphics***. Eck, David. Hobart and William Smith Colleges. 2016. Disponible en: http://math.hws.edu/eck/cs424/downloads/graphicsbook-linked.pdf


Computación Gráfica en la Biblioteca UMNG Campus Cajicá (link)

# Ahora si... Comencemos!

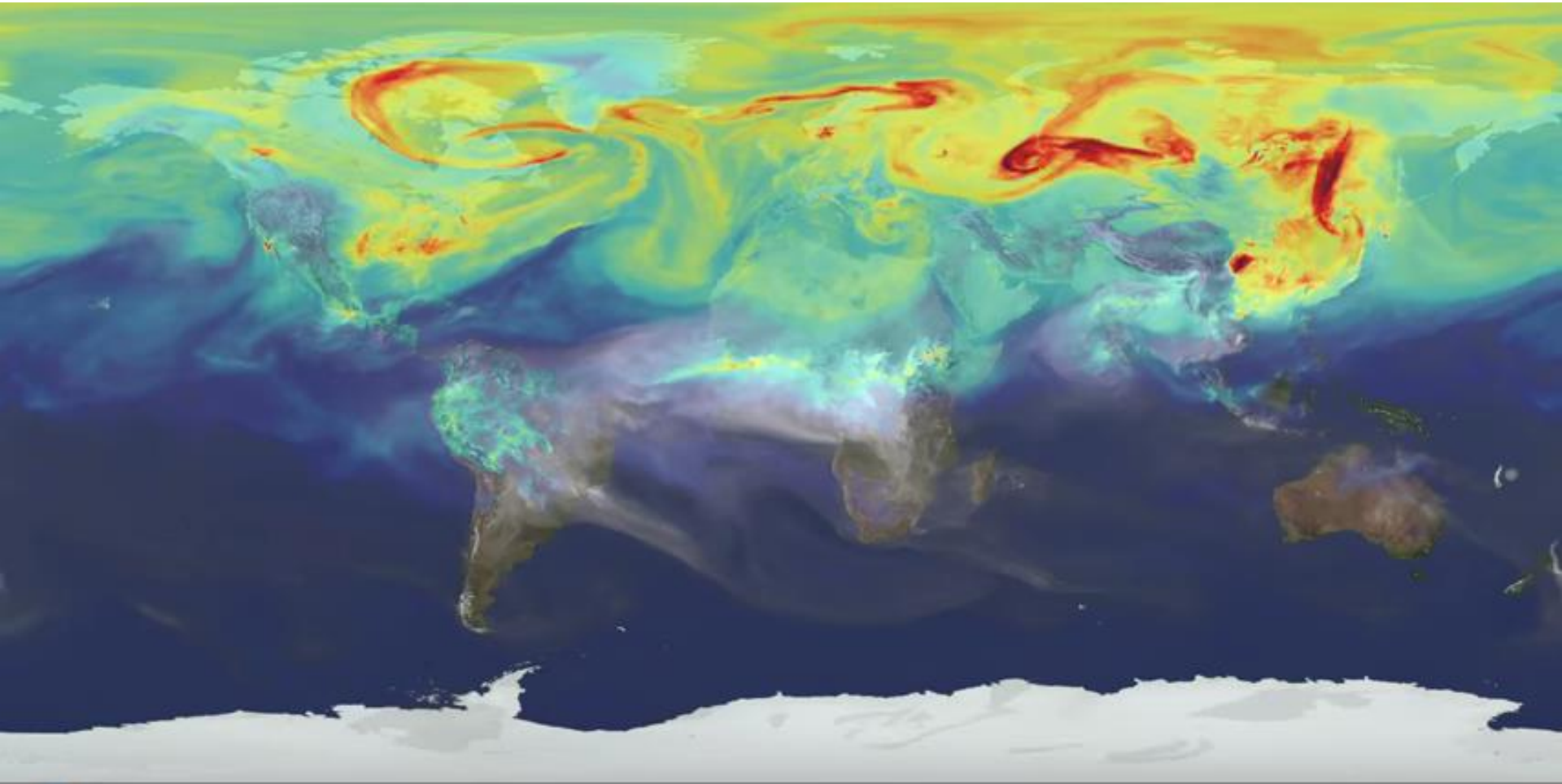


Tomado de: https://medium.freecodecamp.org/make-your-hobby-harder-programming-is-difficult-thats-why-you-should-learn-it-e4627aee41a1

# ¿Para qué la computación gráfica?

# ¿Para qué la computación gráfica?

***Visualización científica***

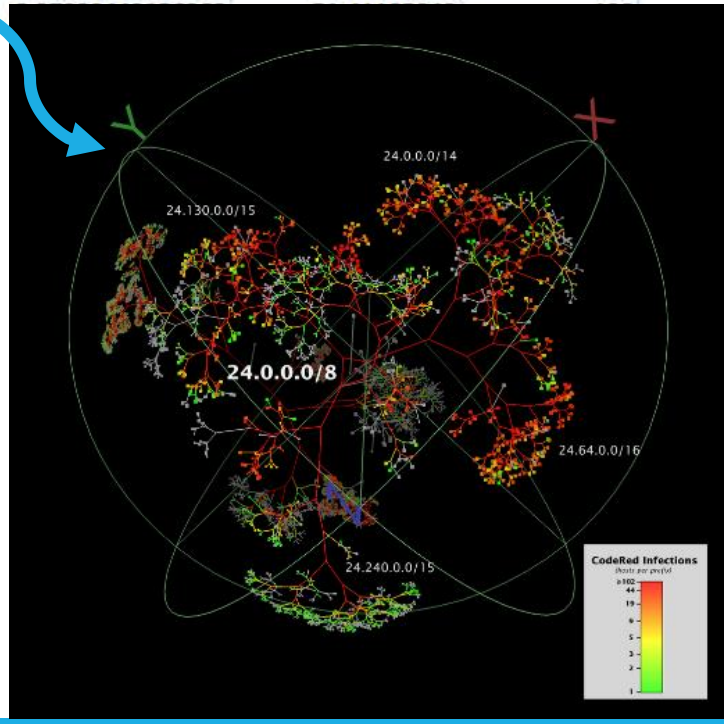


# ¿Para qué la computación gráfica?

***Visualización científica***

| | A | B | C | D |
|---|---|---|---|---|
| 1 | DateTime | Longitude | Latitude | Altitude |
| 281 | 4.5.15 7:18:37 | 6,97196211665869 | 51,41349929 | 134 |
| 282 | 4.5.15 7:18:41 | 6,97143933735788 | 51,41299747 | 136 |
| 283 | 4.5.15 7:18:45 | 6,97095804847777 | 51,4125539 | 137 |
| 284 | 4.5.15 7:18:49 | 6,97055228054523 | 51,41216489 | 137 |
| 285 | 4.5.15 7:18:53 | | | |

- Permite visualizar grandes cantidades de información.
- Utilizada para estudiar comportamientos complejos.
- Presentar tendencias y patrones de los datos, de manera comprensible.
- Facilita la toma de decisiones.




Tomado de: https://www.webdesignerdepot.com/2009/06/50-great-examples-of-data-visualization/

# ¿Para qué la computación gráfica?

***Simulación***

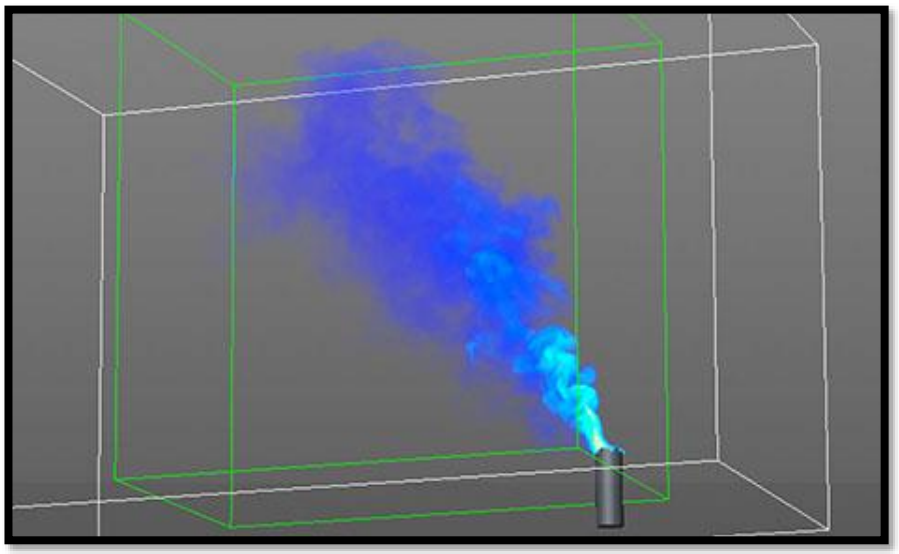
# ¿Para qué la computación gráfica?

***Simulación***

- Para realizar predicciones y análisis de comportamientos.
- Permite estudiar fenómenos naturales, partiendo de su comportamiento matemático.

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} + \nu \nabla^2 \mathbf{u} + \mathbf{f}$$

$$\frac{\partial \rho}{\partial t} = -(\mathbf{u} \cdot \nabla)\rho + \kappa \nabla^2 \rho + S$$



Tomado de: https://www.webdesignerdepot.com/2009/06/50-great-examples-of-data-visualization/

# ¿Para qué la computación gráfica?

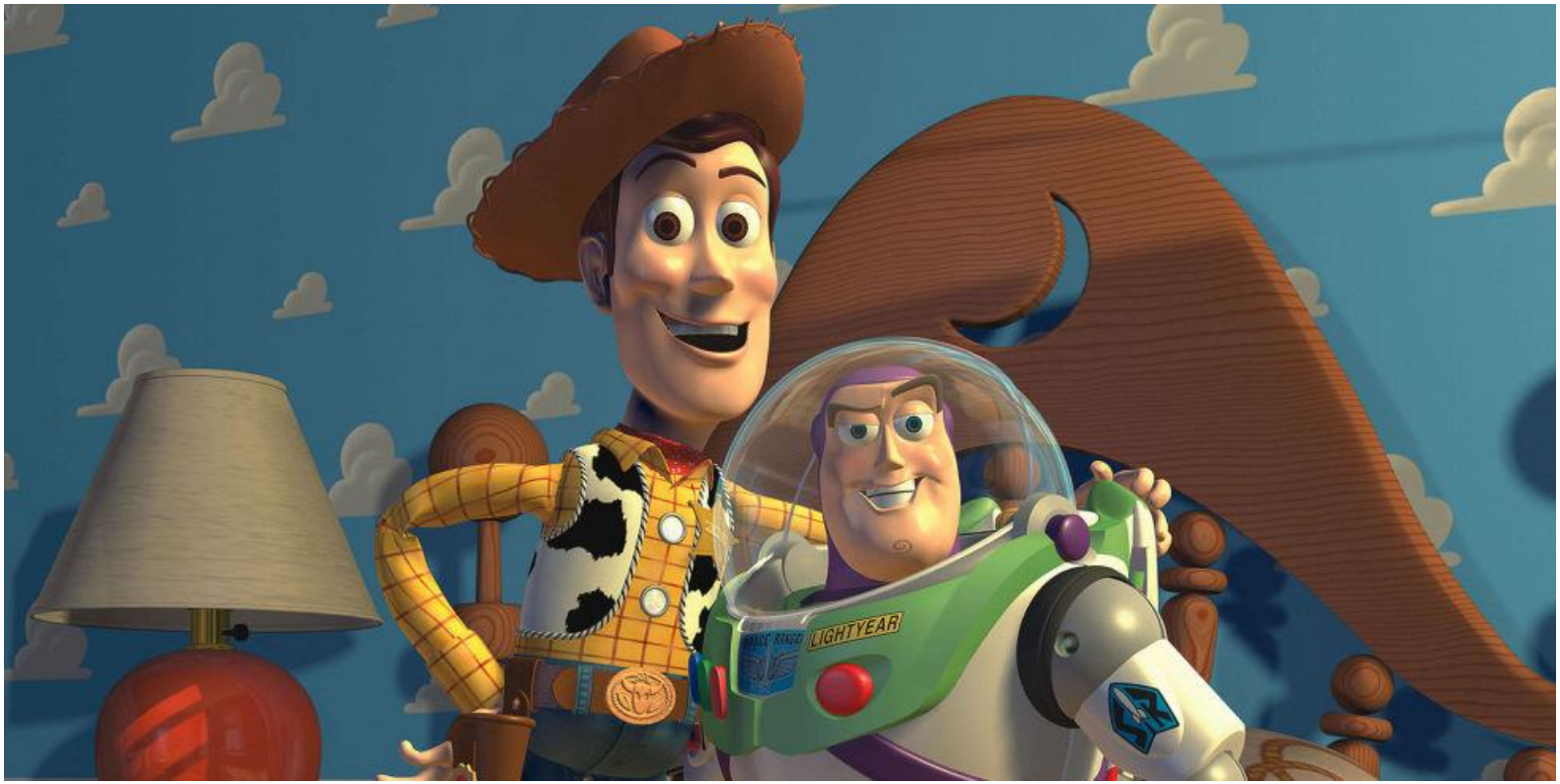
***Videojuegos***


- En el diseño de motores de videojuegos o la creación de nuevos tipos de interacción.
- Para aplicar en el funcionamiento de efectos especiales, interfaces, mejora de materiales, iluminación, etc.
- Permite generar todo tipo de gráficos, con diferente nivel de realismo.

Tomado de: http://whatculture.com/gaming/17-best-video-game-covers-time

# ¿Para qué la computación gráfica?

***VFX CG***



# ¿Para qué la computación gráfica?

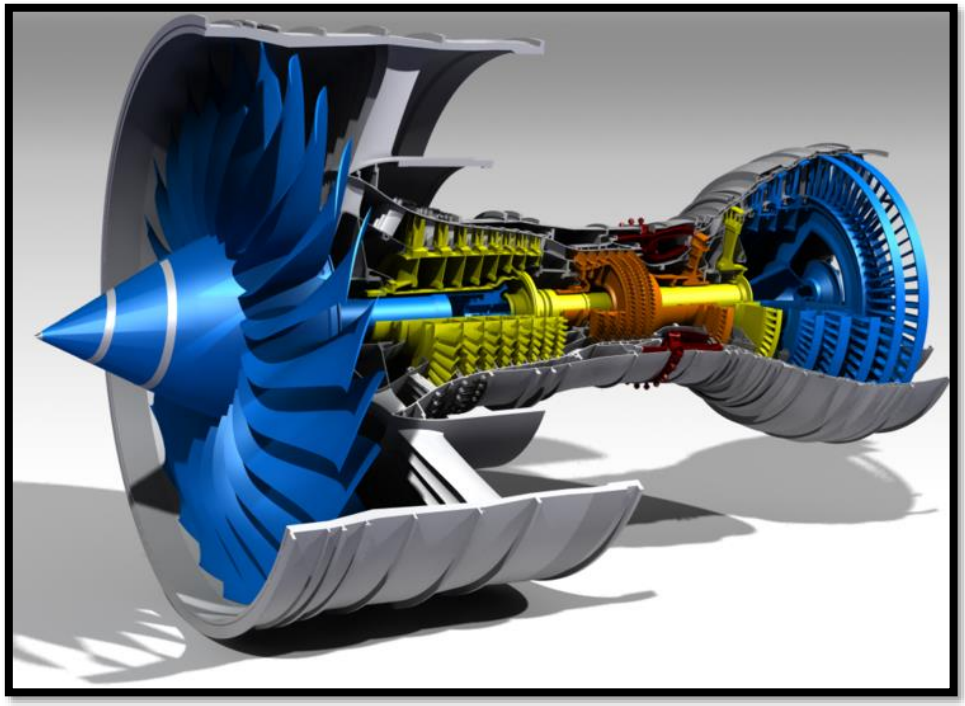
***Diseño gráfico***

- Permite producir imágenes para diversas áreas: arquitectura, mercadeo, etc.
- Diseño de escenas en 2D y 3D.

Tomado de: https://simplydavidb.wordpress.com/2016/04/20/illustrator-example/#jp-carousel-233

# ¿Para qué la computación gráfica?

***Diseño Asistido por Computador***



- Visualización y diseño de componentes mecánicos, eléctricos y de diferentes áreas.
- Facilita la estimación de materiales.
- Permite interactuar con los componentes de un sistema complejo.

# ¿Para qué la computación gráfica?

***Realidad virtual y realidad aumentada***

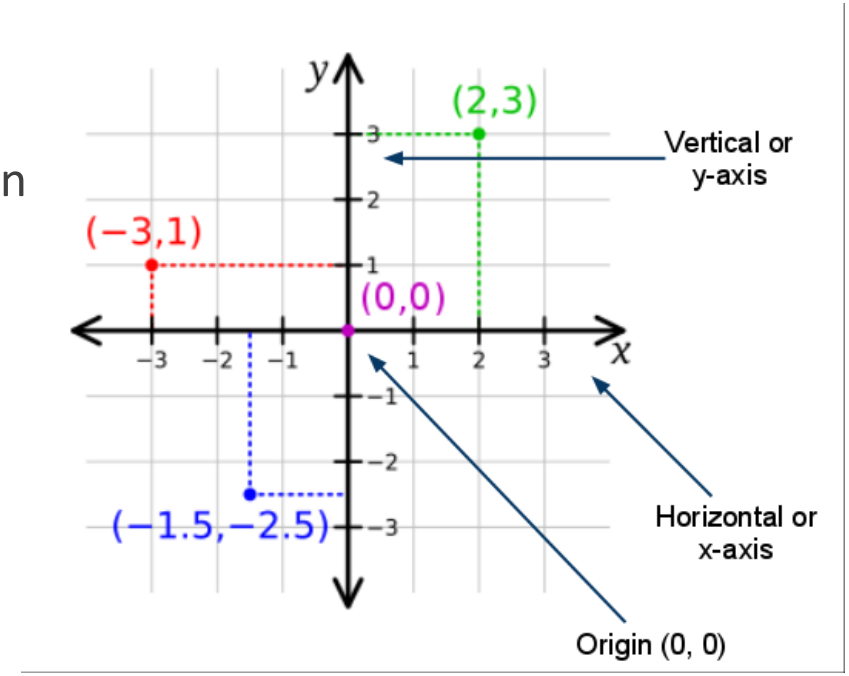
- Inmersión en entornos simulados.
- Dispositivos especializados para la interacción con objetos, bien sea en mundos virtuales o en el mundo real.

# Sistemas de coordenadas

# Recordando… Coordenadas cartesianas 2D

Elementos importantes:

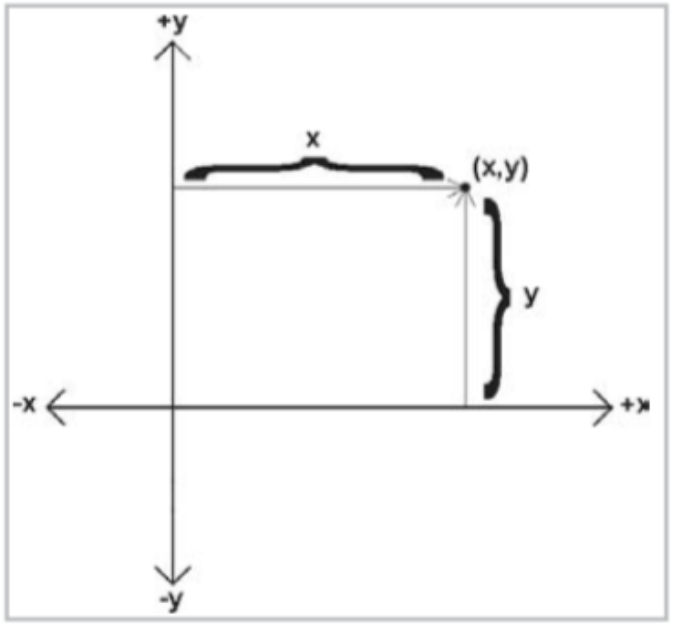
- ***Origen***.
- Dos ***ejes*** que pasan por el origen y son perpendiculares entre sí.



# Recordando… Coordenadas cartesianas 2D

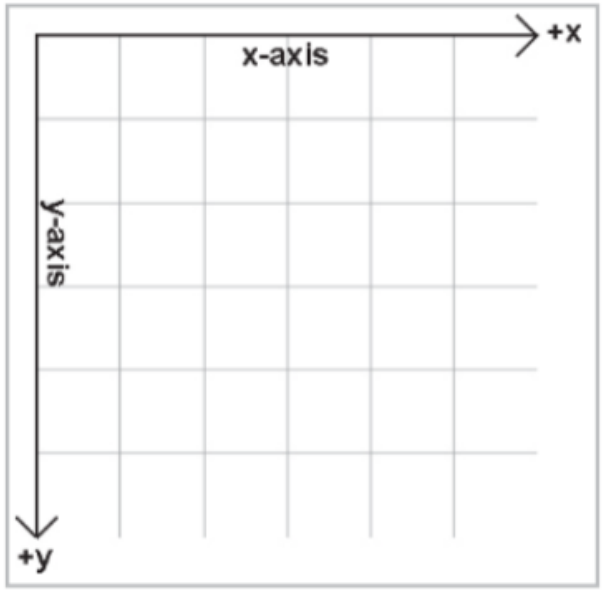
Elementos importantes:

- 2 números (x, y) permiten obtener la ubicación de un punto en particular.
- Cada número representa la distancia a uno de los ejes, medida de forma paralela al otro eje.
- El signo determina la ubicación, positiva o negativa en el plano.



# Coordenadas 2D en pantalla

Orientación en pantalla:



# Coordenadas polares (2D)

Sistema bidimensional de coordenadas, en el que un punto puede ser representado mediante una distancia, tomada desde el origen, y un ángulo.

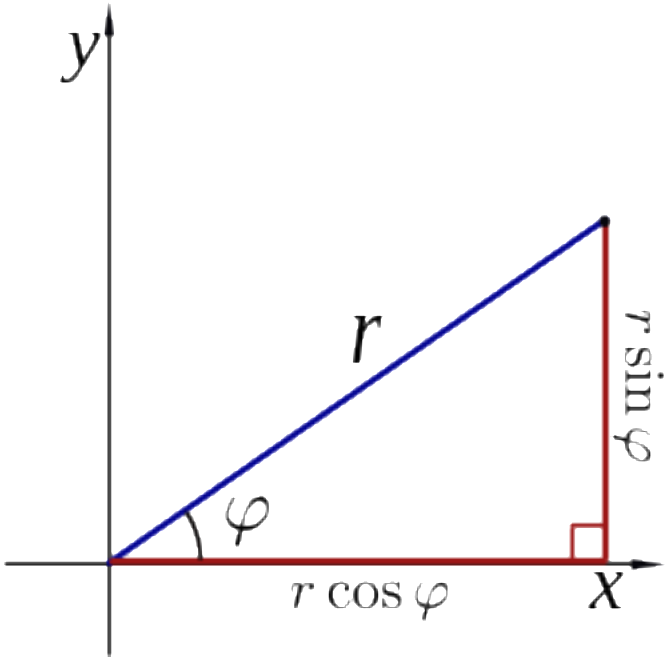
La relación entre los sistemas de coordenadas está dada por:

**Cartesiano a polar:**

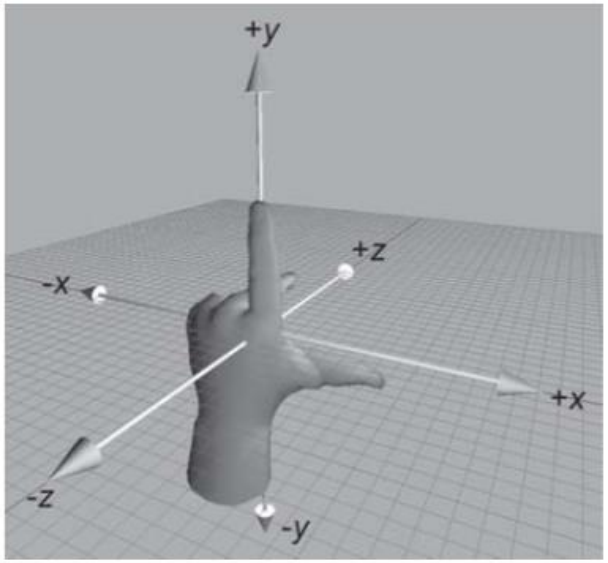
$$\boldsymbol{r = \sqrt{x^2 + y^2}} \quad , \quad \boldsymbol{\theta = \tan^{-1}\left(\frac{y}{x}\right)}$$

**Polar a cartesiano:**

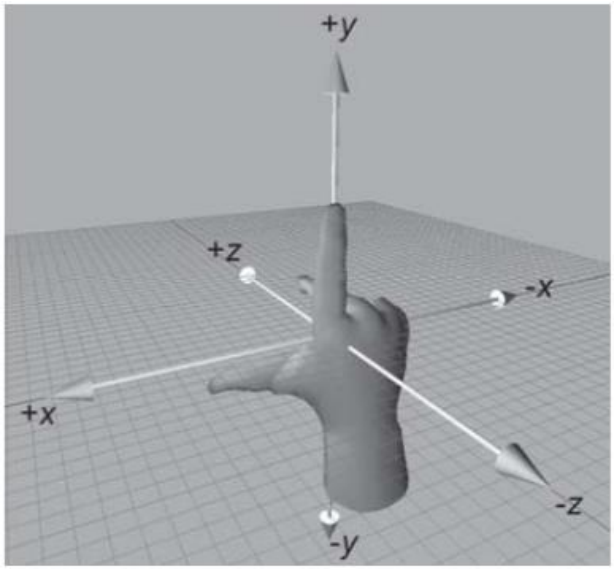
$$\boldsymbol{x = r\cos\theta} \quad , \quad \boldsymbol{y = r\sin\theta}$$



# Espacios de coordenadas 3D




Mano izquierda

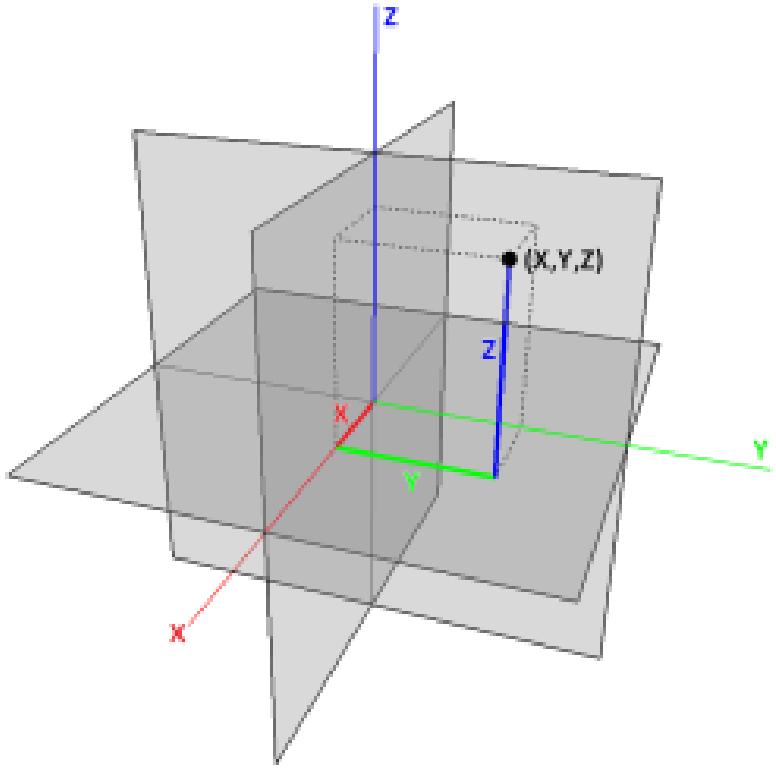



Mano derecha

# Sistema de coordenadas cartesianas 3D

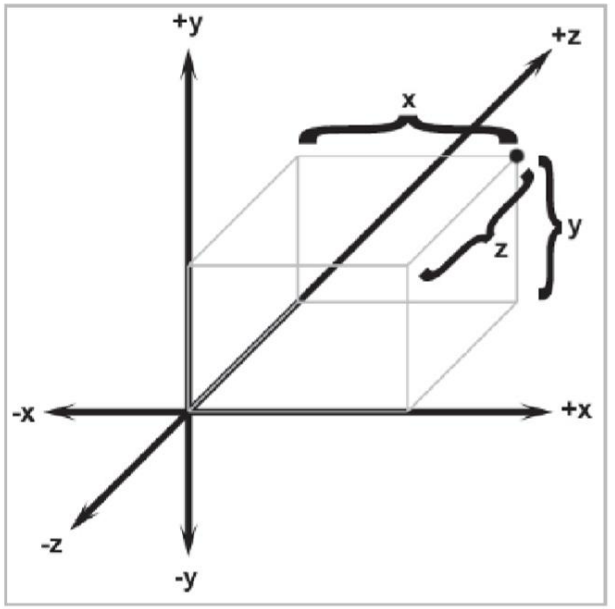
Se requieren 3 ejes para representar un espacio 3D.

- Se agrega el eje z.
- Usualmente los ejes son mutuamente perpendiculares.



# Sistema de coordenadas cartesianas 3D

Un punto se representa entonces como (x, y, z).



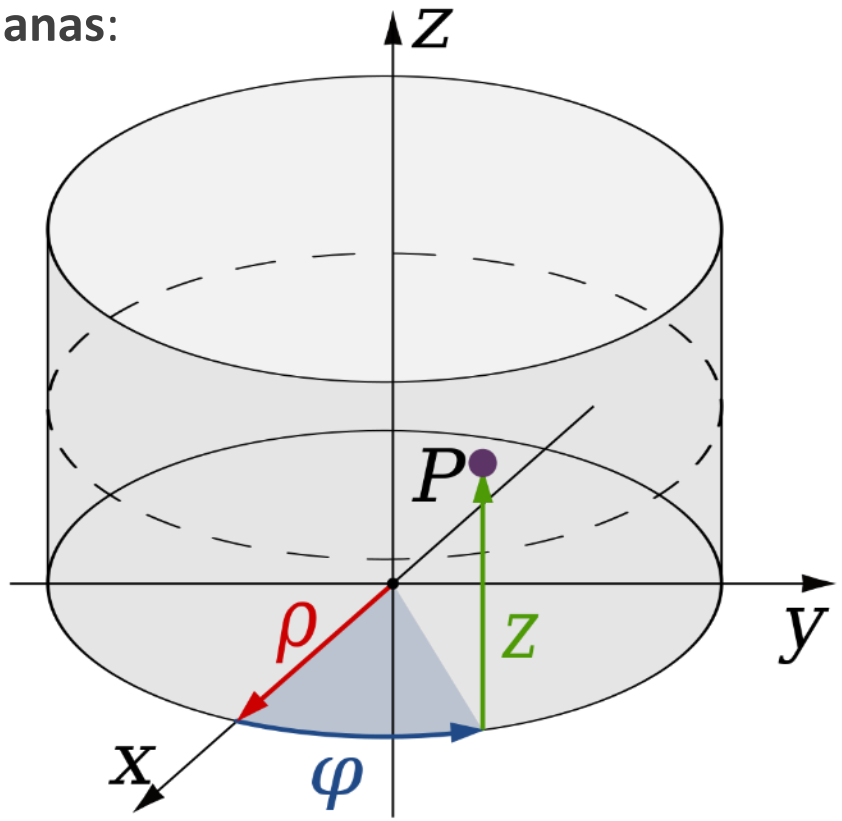
# Coordenadas cilíndricas

**Coordenadas cilíndricas a cartesianas**:

$$x = \rho \cos \varphi$$

$$y = \rho \sin \varphi$$

$$z = z$$



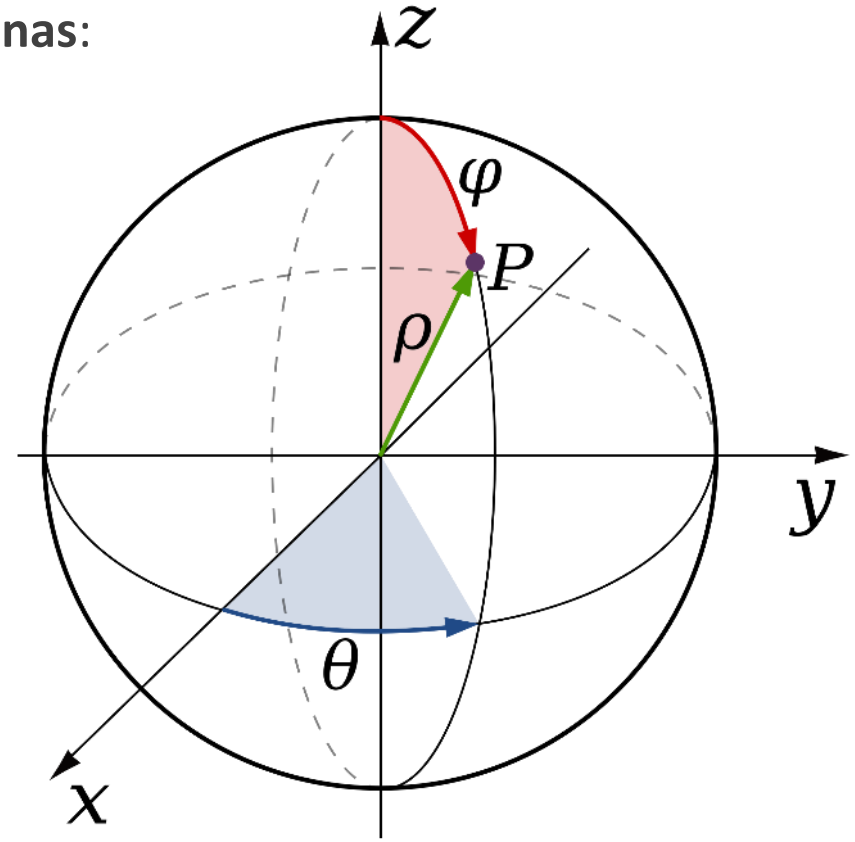
# Coordenadas esféricas

**Coordenadas esféricas a cartesianas**:

$$x = \rho \cos\theta \sin\varphi$$

$$y = \rho \sin\theta \sin\varphi$$

$$z = \rho \cos\varphi$$



# Resumen:
# Sistemas de coordenadas en 3D

**Coordenadas cartesianas**

Desplazamiento en los ejes de coordenadas (x, y, z).

**Coordenadas cilíndricas**

Coordenadas polares sobre un plano 2D, más un desplazamiento en una tercera dimensión, de manera perpendicular.

**Coordenadas esféricas**

Coordenadas polares sobre un plano 2D, más un ángulo de azimut.

# Three.js

DE LA TEORÍA A LA PRÁCTICA

# Three.js

Librería de Javascript para renderizado de gráficos interactivos en 2D y 3D, basada en WebGL.

Los elementos están dentro del <canvas> de una página HTML.

Soportado en casi todos los navegadores.

Incluye escenas, cámaras, geometría, carga de modelos 3D, luces, materiales, shaders, partículas, animación, matemáticas...

Editor 3D: https://threejs.org/editor/

# Cómo iniciar

1. Descargar la librería Three.js
2. Crear un archivo llamado “index.html”
3. En la misma carpeta, crear una carpeta llamada “js” y poner dentro una copia del archivo three.min.js o three.js

# Index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Plantilla Base</title>
    <style>
      html, body { margin: 0; padding: 0; overflow: hidden; }
    </style>
  </head>
  <body>
    <script src="js/three.min.js"></script>
    <script>
        //Acá va el código de la aplicación
    </script>
  </body>
</html>
```

```
<!DOCTYPE html>
<html>
  <head>
    <title>Plantilla Base</title>
    <style>
      html, body { margin: 0; padding: 0; overflow: hidden; }
    </style>
  </head>
  <body>
    <script src="js/three.min.js"></script>
    <script>
        var scene = new THREE.Scene();
        var aspect = window.innerWidth / window.innerHeight;
        var camera = new THREE.PerspectiveCamera( 75, aspect, 0.1, 1000);
        var renderer = new THREE.WebGLRenderer();
        renderer.setSize( window.innerWidth, window.innerHeight );
        document.body.appendChild( renderer.domElement );
    </script>
  </body>
</html>
```

# Ejercicio 1

Revisar el código en el aula virtual Tema01.zip

Buscar en la documentación de Three.js los elementos ***ArrowHelper***, ***AxesHelper*** y ***PlaneHelper***.

Hacer una aplicación que muestre:

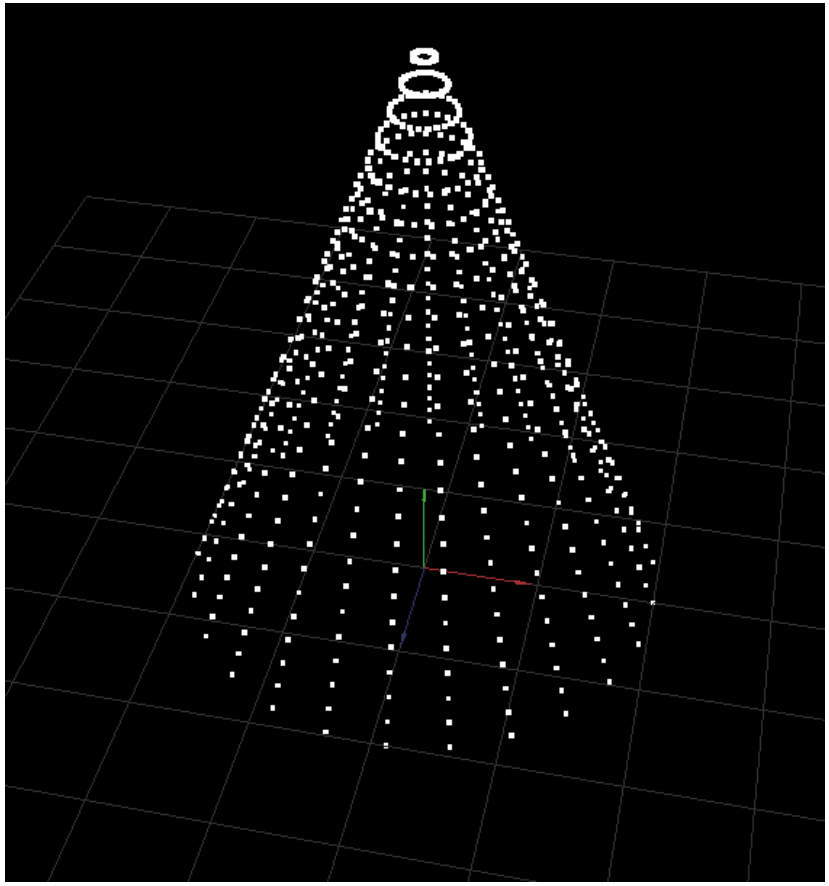
1. El plano XZ.
2. Las flechas con los vectores X, Y y Z en el origen.

# Debugging en Three.js

***Debugging*** *is the process of finding and resolving defects or problems within a computer program that prevent correct operation of computer software or a system.*

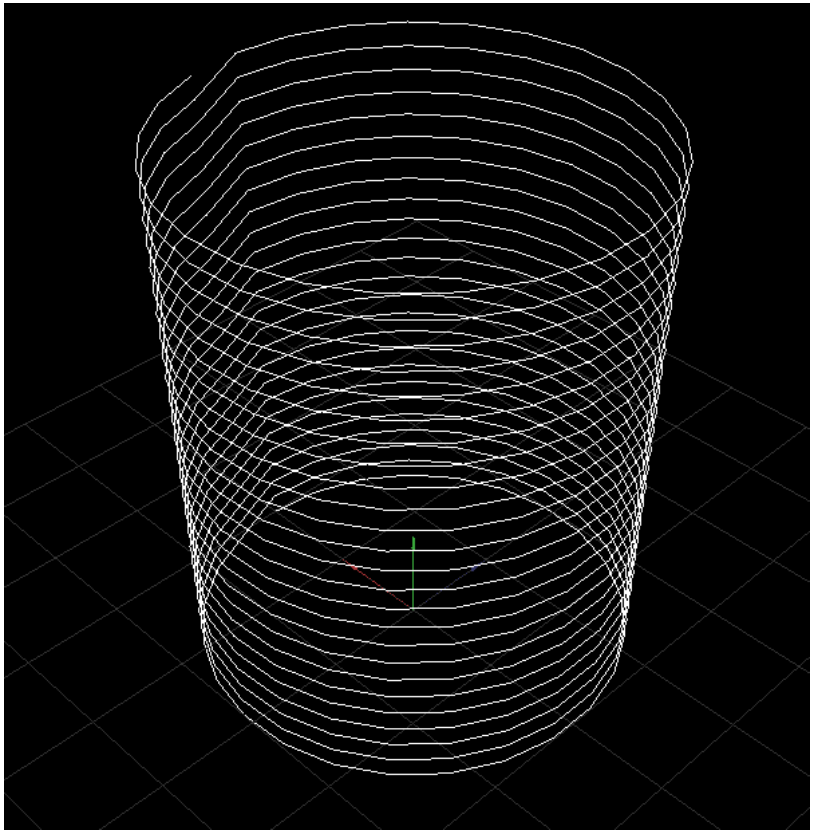
Es muy probable que al tratar de realizar los ejercicios, el navegador no muestre ningún resultado. Esto ocurre principalmente por errores en la sintaxis o de tipografía en el código.

Para hacer depuración (*debugging*) es necesario activar las opciones de desarrollador del navegador. Se mostrará en la consola web el origen de los errores. También se puede utilizar en el código la opción *console.log("texto")* para verificar.



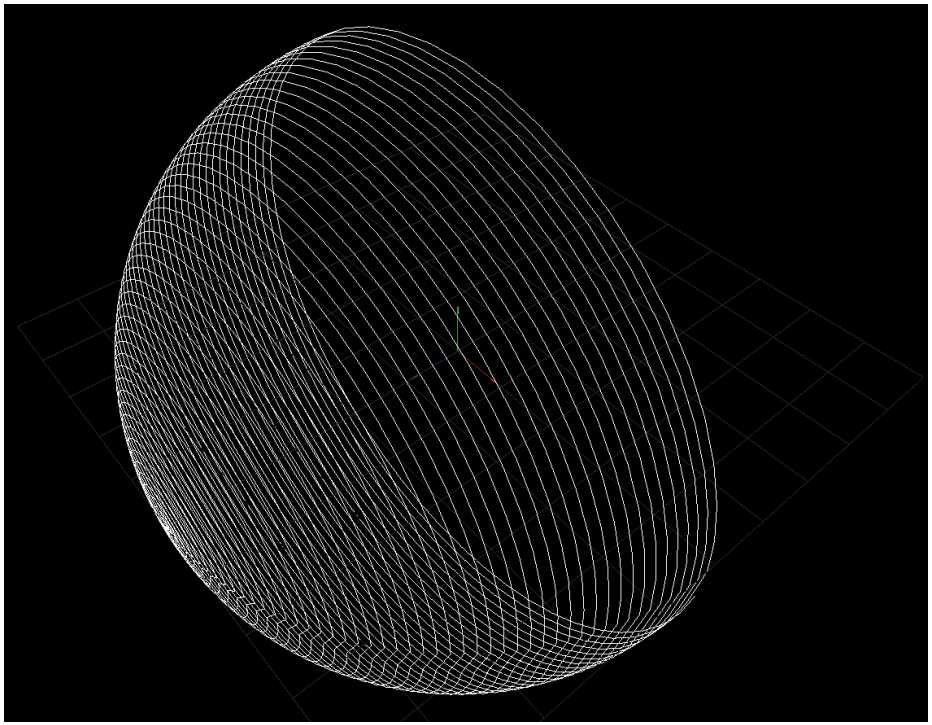
# Practica: Ejemplo 1

***02 - Coord3D04-EjA-2019-2***



# Practica: Ejemplo 2

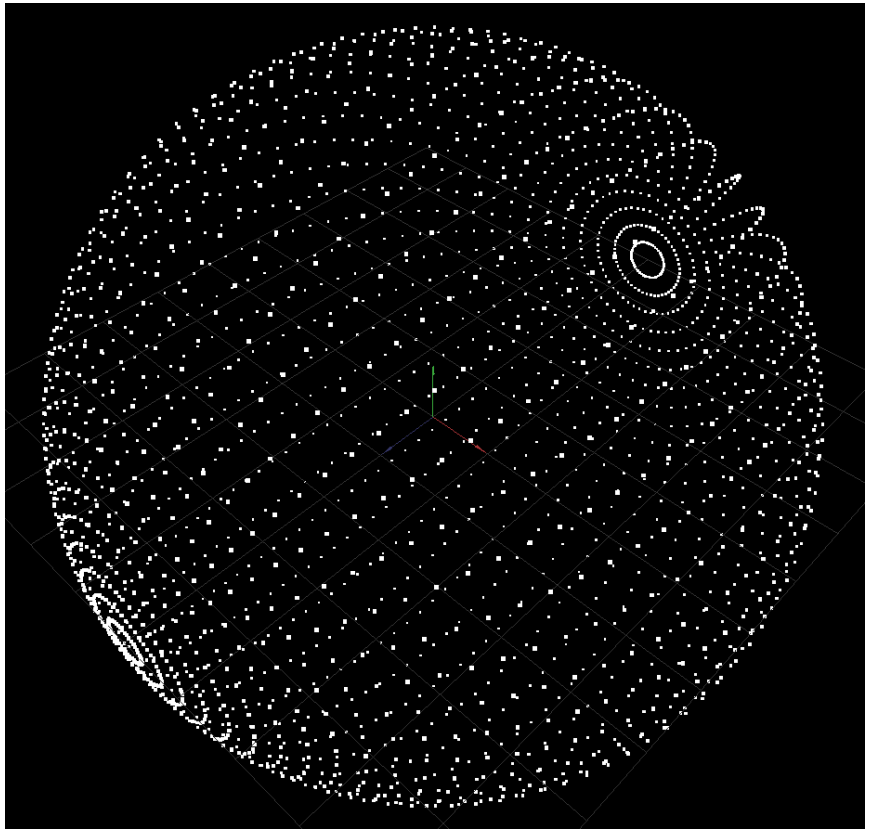
***02 - Coord3D04-EjB-2019-2***



# Practica: Ejercicio 1

Dados los ejemplos, genere media esfera, con radio 5 ubicada en el origen (usando líneas interconectadas).

*Tip 1: Tenga en cuenta lo visto sobre las coordenadas esféricas !*



# Practica: Ejercicio 2

Dados los ejemplos, genere una esfera, con radio 5 ubicada en el origen (usando solo puntos).

*Tip 1: Tenga en cuenta lo visto sobre las coordenadas esféricas !*

# Tarea

Dados los ejemplos, de forma individual:

1. Crear una cuenta en Github, para manejar repositorio. Si ya tiene un repositorio de Introducción a la Computación Gráfica, organice los archivos en otra carpeta.
2. Subir el enlace del repositorio en el aula virtual.
3. Hacer un modelo libre en 3D, usando puntos y coordenadas cartesianas, esféricas y cilíndricas.

# Bibliografía recomendada


- ***Computer Graphics: Principles and Practice***. *James D. Foley, Andries van Dam, Steven K. Feiner, John F. Hughes. Editorial Addison Wesley, 2014. (biblioteca UMNG).*
- ***Introduction to Computer Graphics***. Eck, David. Hobart and William Smith Colleges. 2016. Disponible en: http://math.hws.edu/eck/cs424/downloads/graphicsbook-linked.pdf
- Three.js. Documentation. (2018) Tomado de: https://threejs.org/docs/index.html#manual/introduction/Creating-a-scene
- Lyons, David. Introduction to Three.js. Tomado de: http://davidscottlyons.com/threejs-intro