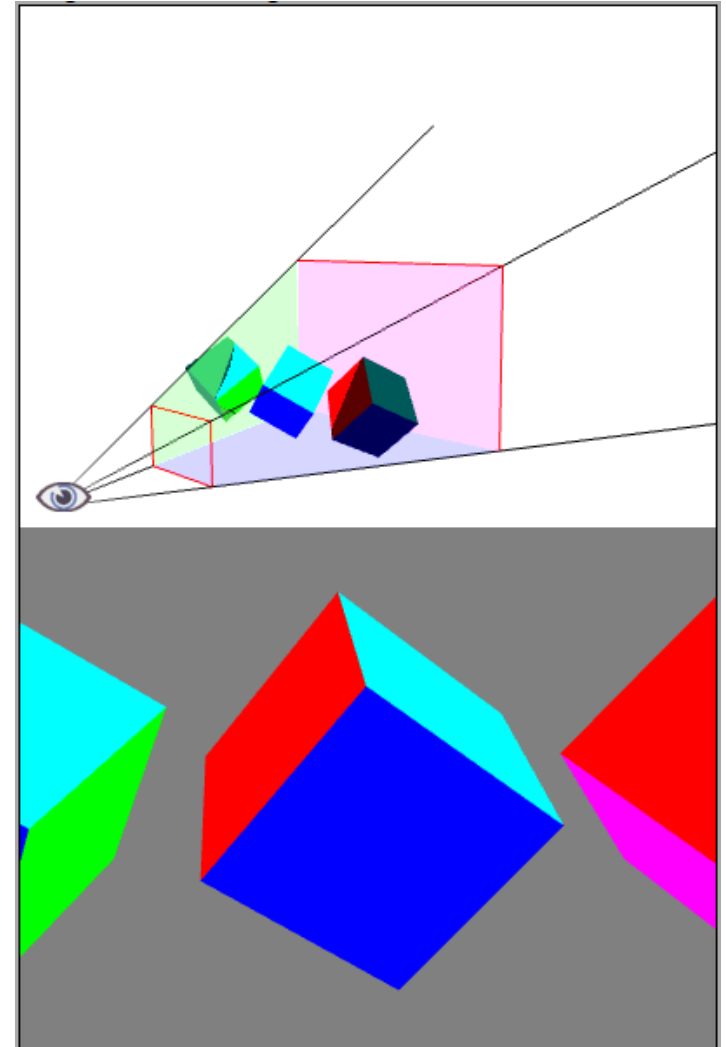


3D EN LA WEB

ESCENA 3D: PUNTOS DE VISTA

Cuando observamos una escena tridimensional de un videojuego en la computadora, por ejemplo, vemos una versión **bidimensional** de la misma.

Nosotros vemos lo que ve una **cámara** situada en un punto determinado del espacio R^3 que posee cierta rotación en sus ejes de tal forma que enfoque a alguna parte de la escena.



ESCENA 3D: PUNTOS DE VISTA

Las librerías que dan soporte a este tipo de escenas en los navegadores, como por ejemplo **WebGL**, realizan este proceso de transformación de la información contenida en el entorno tridimensional mediante **proyecciones** y el resultado es llevado a un ambiente bidimensional, que es el que luego el usuario verá en la pantalla.



PROYECCIONES

Tipos de proyecciones posibles:

- Proyección Ortográfica
 - Misma escala, intercambia y por z.
- Proyección Perspectiva Débil
 - Escala según distancia (y).
- Proyección Perspectiva
 - Escala según distancia y posición de la cámara que enfoca al/los punto/s.

PROYECCION ORTOGRAFICA

Para proyectar:

$$a = \langle x, y, z \rangle$$

Con x, y, z en R^3

en

$$b = \langle x, y \rangle$$

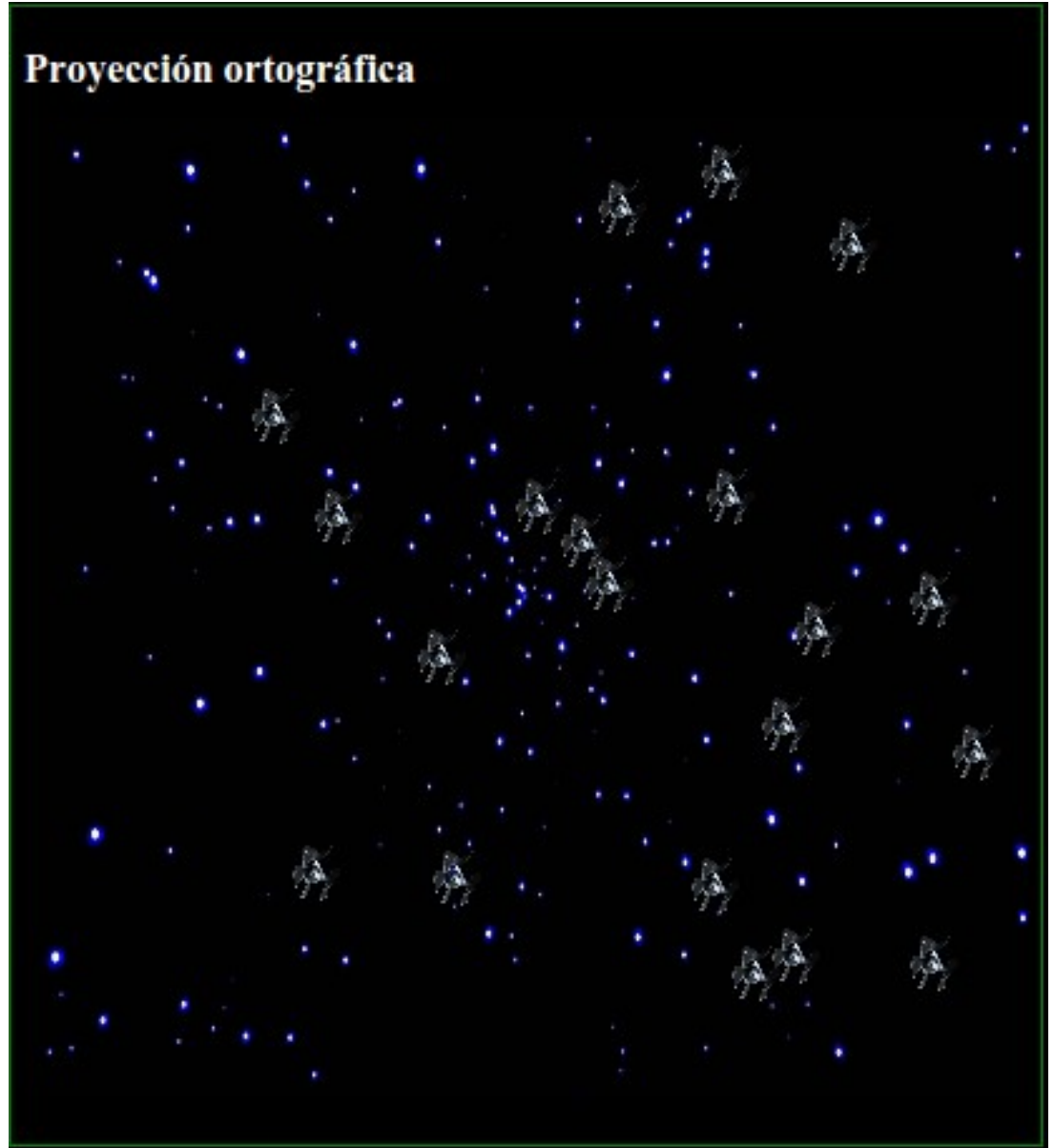
Con x, y en R^2

Entonces:

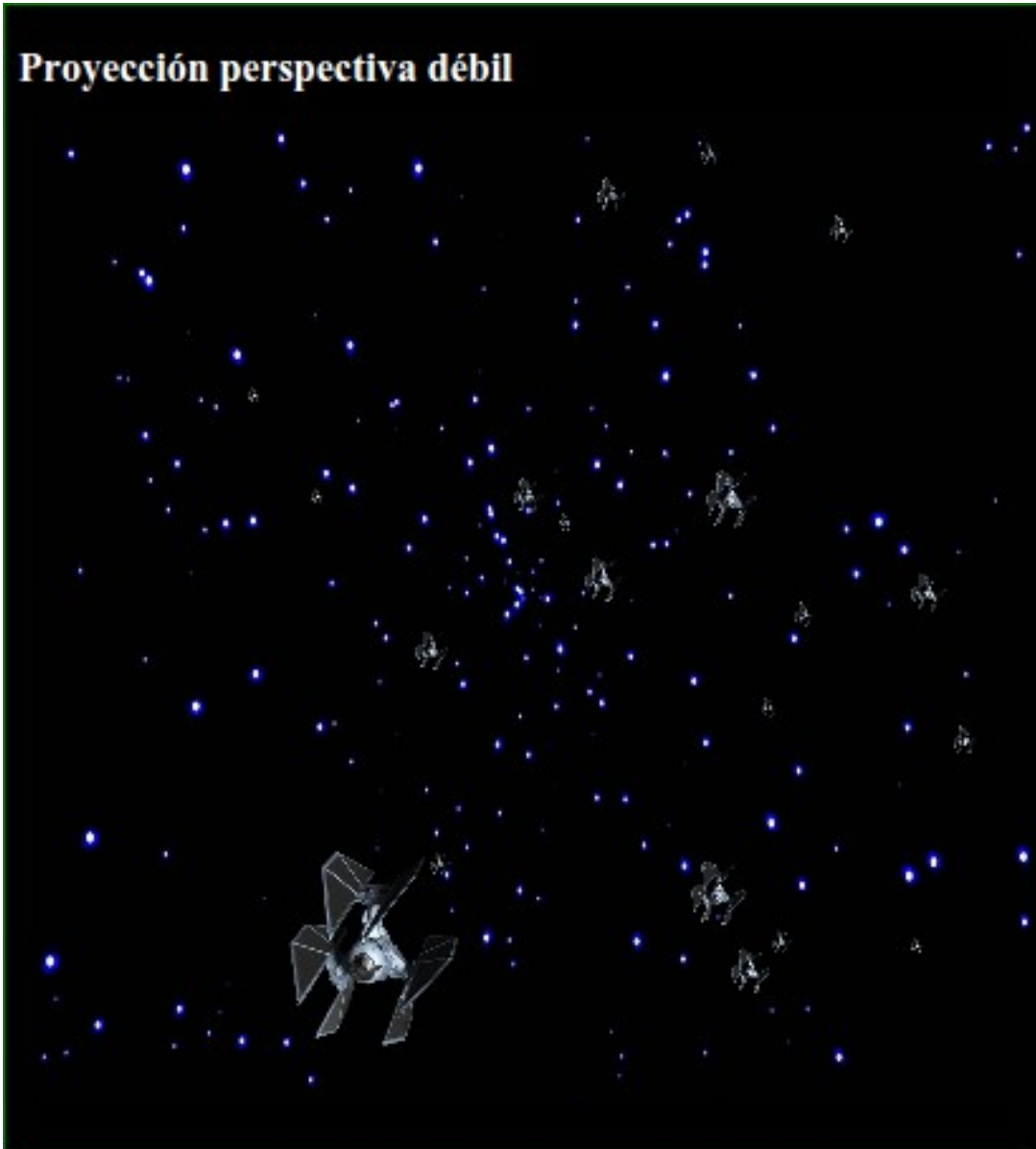
$$b_x = a_x$$

$$b_y = a_y$$

Proyección ortográfica



PROYECCION PERSPECTIVA DEBIL



Para proyectar:

Se hace lo mismo que en la anterior.

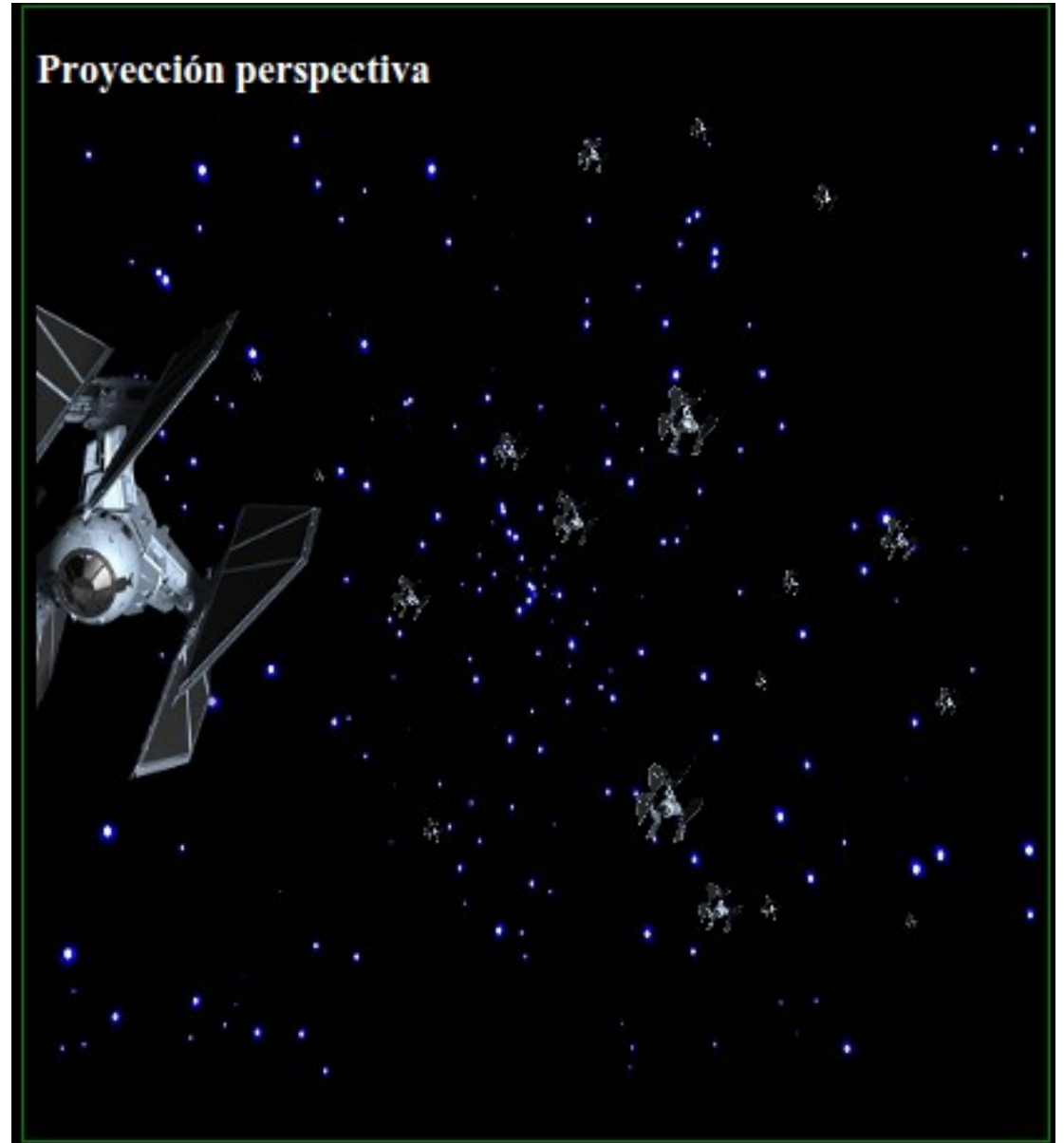
Escala:

Aumentamos la escala cuanto más cerca de $y=0$ está el punto.

PROYECCION PERSPECTIVA

Para proyectar:

Se tiene en cuenta el angulo y la posición de la cámara, la posición del objeto y la distancia entre la pantalla y el espectador.



PROYECCION PERSPECTIVA

Resta de matrices C y A. O:

$$\mathbf{d}_x = c_y(s_z \mathbf{y} + c_z \mathbf{x}) - s_y \mathbf{z}$$

$$\mathbf{d}_y = s_x(c_y \mathbf{z} + s_y(s_z \mathbf{y} + c_z \mathbf{x})) + c_x(c_z \mathbf{y} - s_z \mathbf{x})$$

$$\mathbf{d}_z = c_x(c_y \mathbf{z} + s_y(s_z \mathbf{y} + c_z \mathbf{x})) - s_x(c_z \mathbf{y} - s_z \mathbf{x})$$

C_{y,x,z} coseno de camara.posicion.y, x, z

S_{y,x,z} seno de camara.posicion.y,x,z

X, Y, Z objeto.x, y, z

$$\mathbf{b}_x = (\mathbf{d}_x \mathbf{s}_x) / (\mathbf{d}_z \mathbf{r}_x) \mathbf{r}_z$$

$$\mathbf{b}_y = (\mathbf{d}_y \mathbf{s}_y) / (\mathbf{d}_z \mathbf{r}_y) \mathbf{r}_z$$

b: nuevo punto en entorno 2d

s: tamaño de la pantalla

rx,y: tamaño de la superficie que se mostrará.

rz: distancia de la superficie de que se mostrará hacia el centro de la camara

dz: distancia entre el punto y el centro de la camara.

PUNTOS EN UN ESPACIO R3

Tal como sucede en las matemáticas, en la programación web podemos representar la posición de ciertos puntos en un espacio tridimensional mediante una estructura llamada **vector**. En **javascript**, por conveniencia para la posterior programación de los algoritmos de proyección, usamos **objetos**.

Como vector:

```
Var punto = [3,2,1]
```

Como objeto:

```
Var punto = {  
  X: 3,  
  Y: 2,  
  Z: 1  
}
```



PROYECCION ORTOGRAFICA

```
function proyeccionOrtografica(){  
  //los muestro  
  puntos.forEach(function(object){  
    var bx = object.x;  
    var by = object.z;  
    dibujarPunto(bx,by,1,"canvasOrtografica");  
  });  
}
```

A yellow square containing the letters 'JS' in a bold, dark grey, sans-serif font.

PROYECCION PERSPECTIVA DEBIL

```
function proyeccionPerspectivaDebil(){  
  //los muestro  
  puntos.forEach(function(object){  
    var coeficiente = 0.0001;  
    var escala = object.y*coeficiente;  
    var bx = object.x;  
    var by = object.z;  
    dibujarPunto(bx,by,escala,"canvasPerspectivaDebil");  
  });  
}
```


A yellow square containing the letters 'JS' in a bold, dark grey, sans-serif font, representing JavaScript.

PROYECCION PERSPECTIVA

```
function proyeccionPerspectiva(){  
  //los muestro  
  puntos.forEach(function(object){  
    var coeficienteEscala = 0.0001;  
    var escala = (object.y-camara.posicion.y)*coeficienteEscala;  
    //veamos si no lo pasamos con la cámara..  
    escala = (escala<0)? 0 : escala;  
    var bx, by;  
    var coeficiente = 100;  
    bx = object.x - (camara.posicion.x/(camara.posicion.y-object.y)*coeficiente);  
    by = object.z + (camara.posicion.z/(object.y-camara.posicion.y)*coeficiente);  
    dibujarPunto(bx,by,escala,"canvasPerspectiva");  
  });  
}
```

A yellow square containing the letters 'JS' in a bold, dark blue font, representing JavaScript.

EJEMPLO EN NAVEGADOR

A yellow square containing the letters 'JS' in a bold, dark grey font, representing JavaScript.

JS

BIBLIOGRAFIA

- 3D Projection, [wikipedia.org](https://en.wikipedia.org)
- WebGL, [wikipedia.org](https://en.wikipedia.org)
- WebGL 3d Ortographic, webglfundamentals.org
- WebGL 3d Perspective, webglfundamentals.org
- Perspective Projection, ogldev.org

A yellow square containing the letters 'JS' in a bold, dark blue, sans-serif font, representing JavaScript.