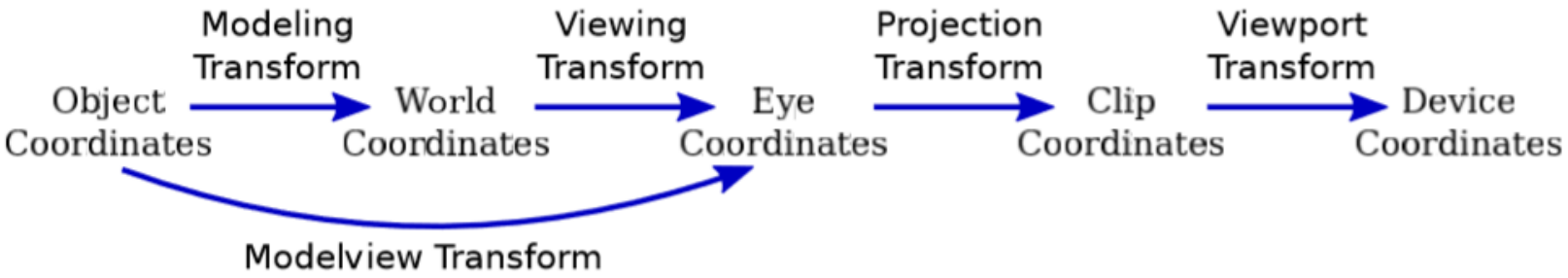
Computación Gráfica

Ing. Gabriel Ávila, MSc.

La tubería de visualización

GRAPHICS PIPELINE OR RENDERING PIPELINE

De coordenadas globales a coordenadas del dispositivo



1. Vértices ubicados en el espacio (*object coordinates*).
2. Se transforman a coordenadas de la cámara (*eye coordinates*).
3. Se genera el volumen (cubo) de visualización (*projection transform*) y se eliminan aquellos objetos que están fuera de este volumen.
4. Se obtienen la coordenadas en pantalla de los objetos de la escena (*device coordinates*).

The viewport transform

Transforma las coordenadas (x, y) de los objetos dentro del volumen recortado, a coordenadas en el dispositivo de visualización. Define el rectángulo en donde se verá la escena.

En three.js esto depende del estilo del *canvas:*

```
renderer.setSize( window.innerWidth, window.innerHeight );
```

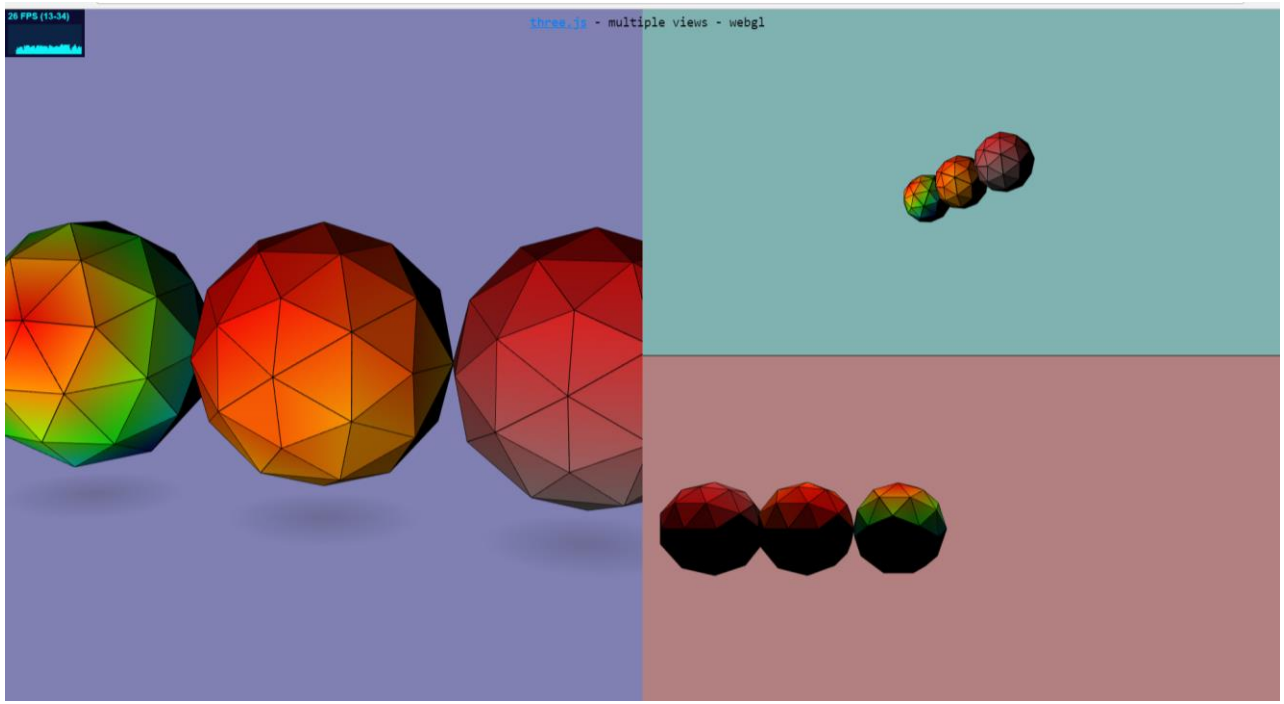
Como de las características del *viewport particular:*

```
renderer.setViewport( x, y, width, height );
```

Que define el viewport desde un punto (x, y) hasta $(x+width, y+height)$.

Viewport

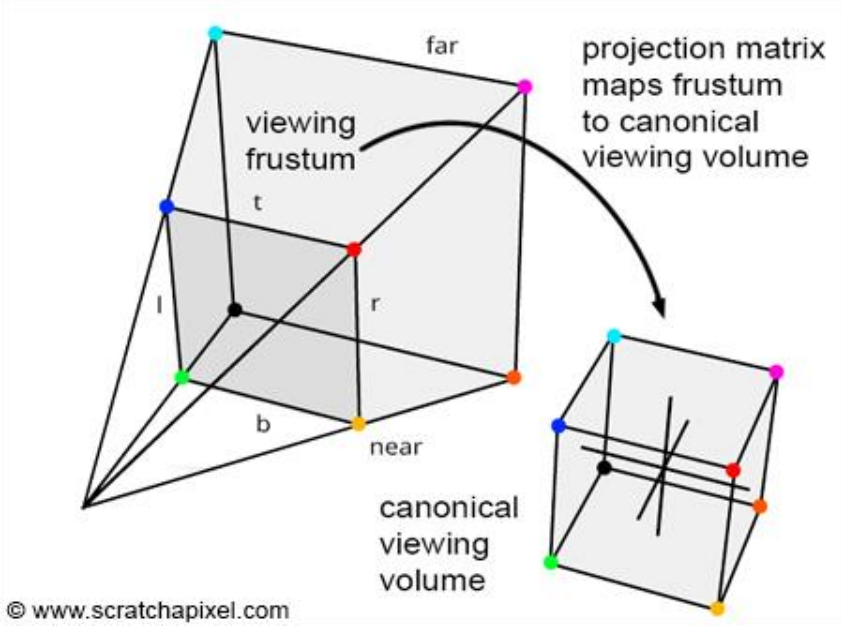
Es posible tener mas de una cámara y mas de un viewport para visualización de la escena.



Volumen de visualización

El volumen de visualización es la parte del mundo visible en la imagen.

Este volumen se determina por la combinación de la Transformación de Visualización (***Viewing Transformation***) y de la Transformación de Proyección (***Projection Transformation***).



The viewing transform and the projection transform

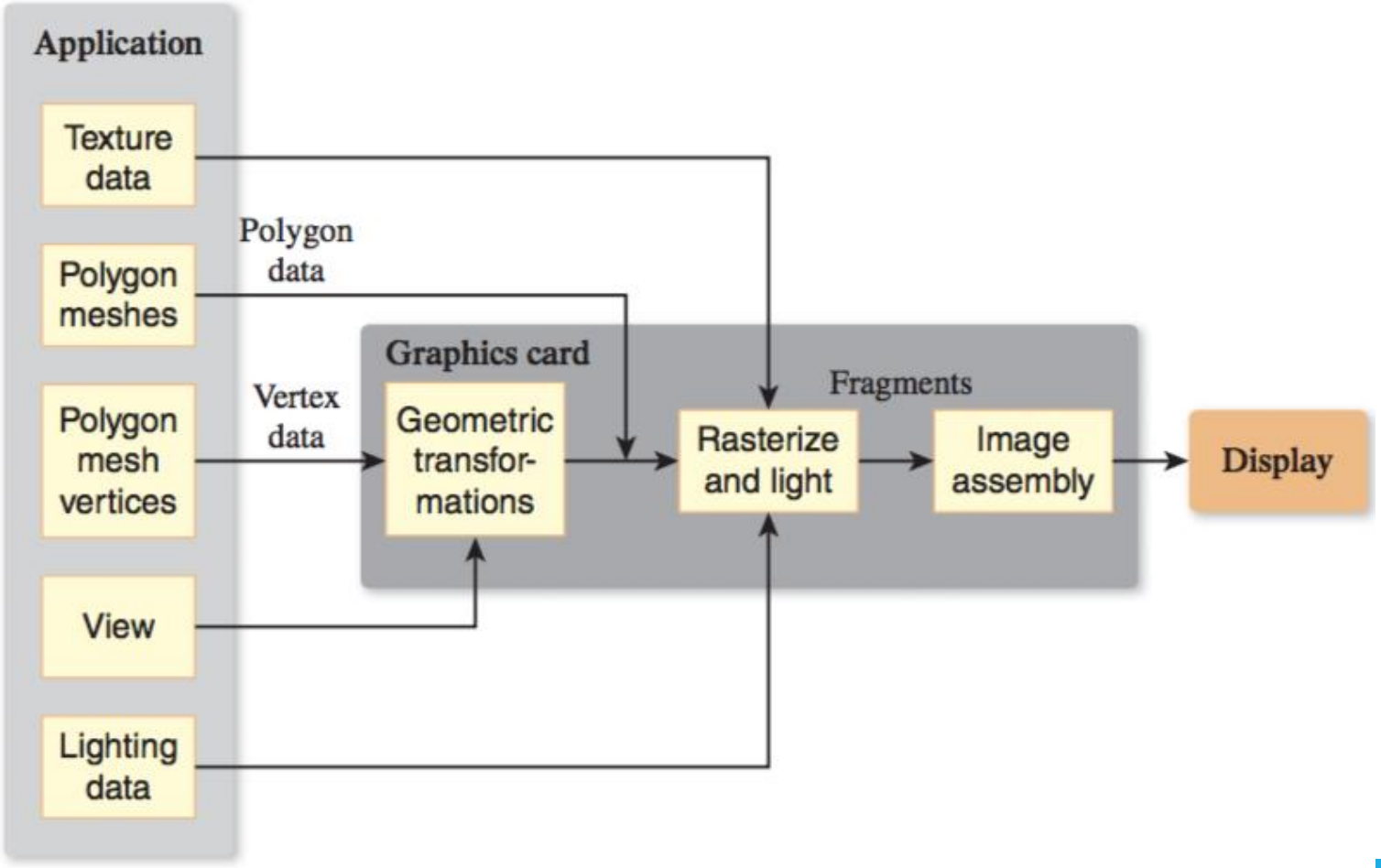
Viewing Transform: Determina dónde esta localizado el usuario (o la cámara) y hacia dónde está mirando. No indica qué tanto del mundo puede ver el usuario.

Projection Transform: Especifica la forma y extensión de la región que está en la vista, es decir, lo que va dentro del volumen de visualización.

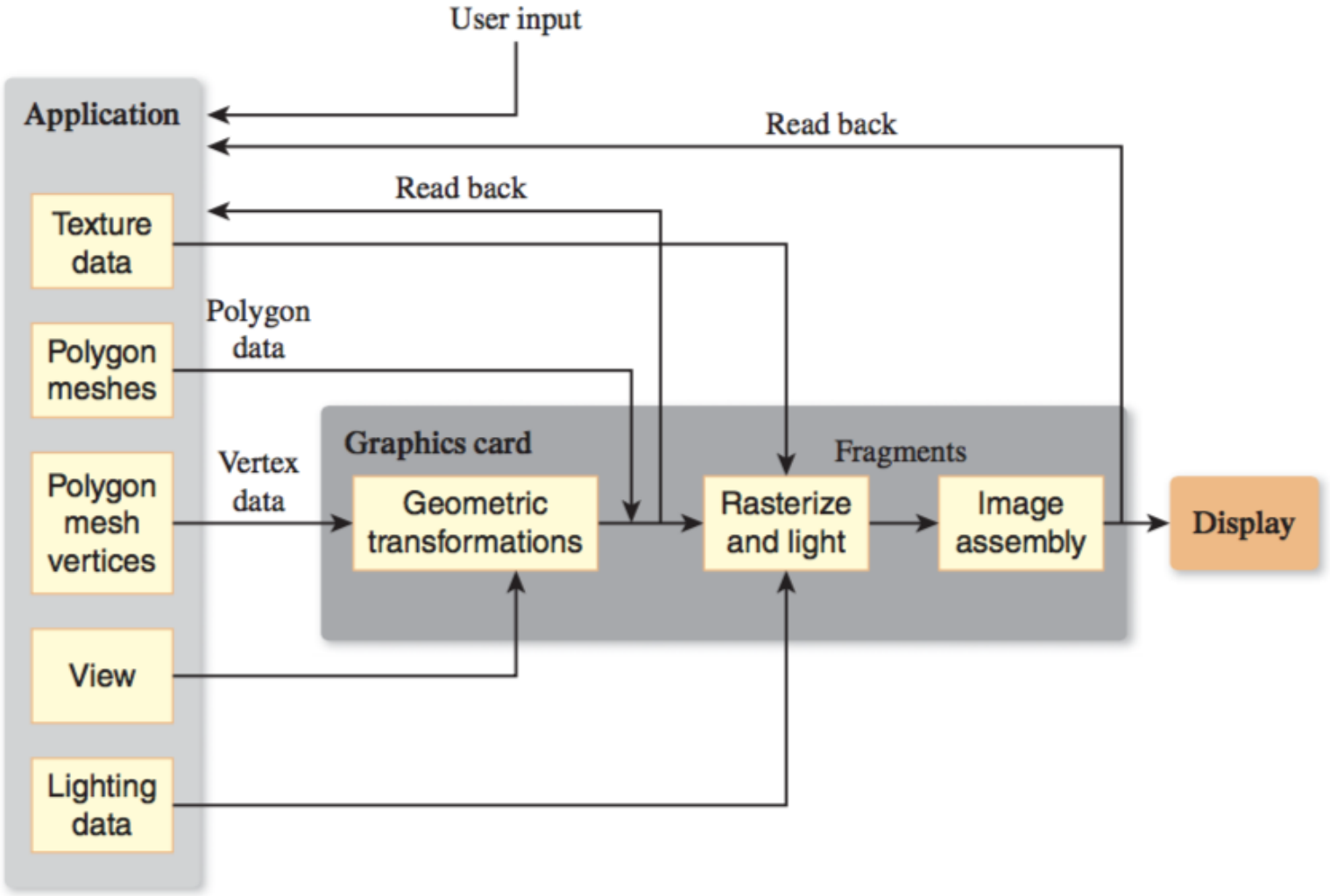
The modeling transform

Convierte las coordenadas de los objetos, de coordenadas locales a coordenadas globales.

Graphics pipeline V1

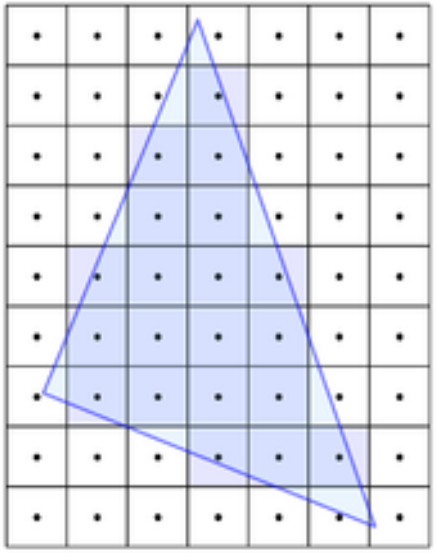


Graphics pipeline
More detailed

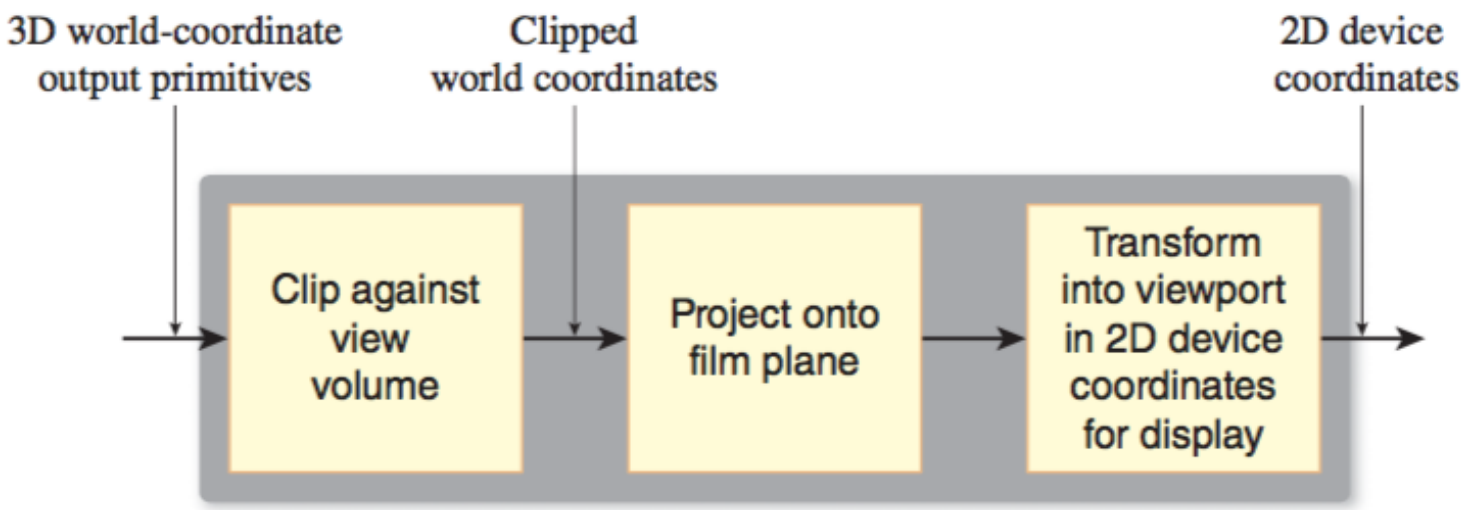


Rasterization

Proceso en el que las formas que aparecen en el viewport se convierten a pixeles.

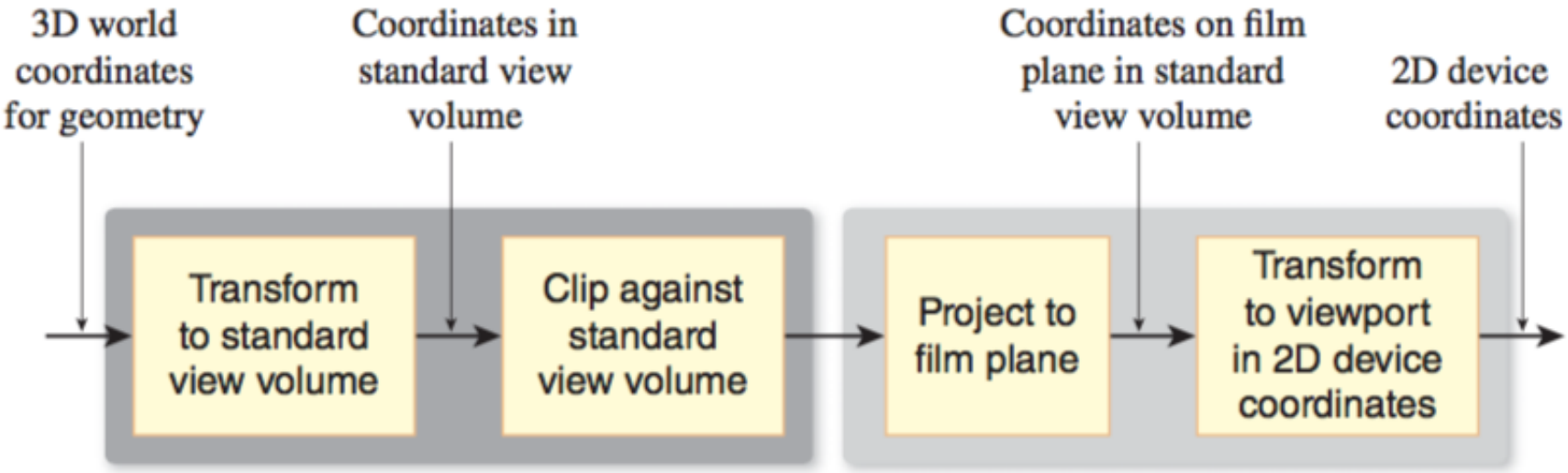


https://www.scratchapixel.com/lessons/3d-basic-rendering/rasterization-practical-implementation/rasterization-stage

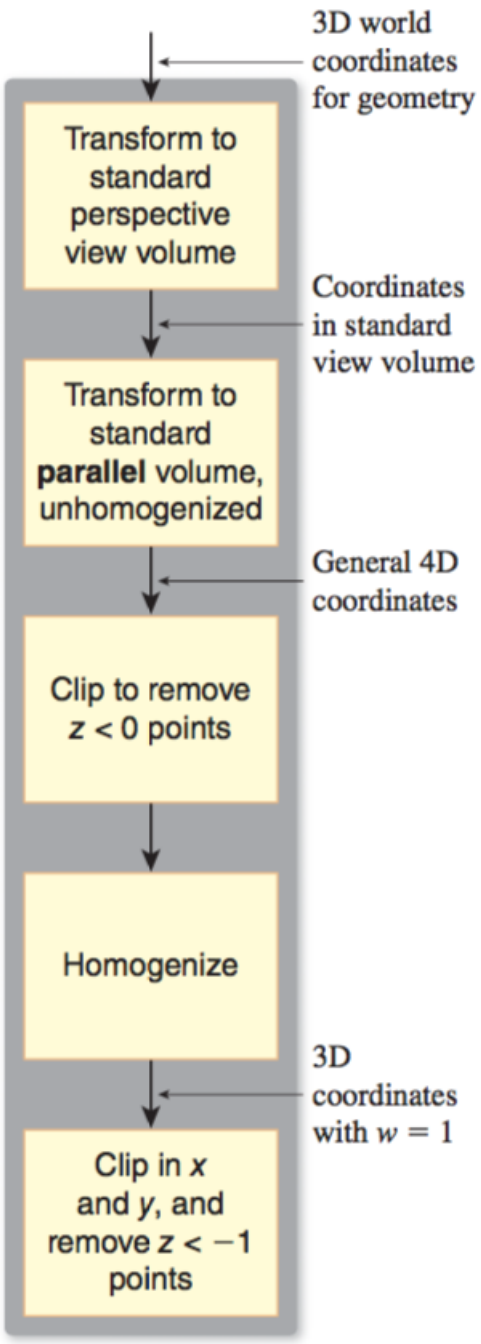


Rasterizing pipeline

PROCESAMIENTO DE GEOMETRÍAS PARA CREAR IMÁGENES



Standard view transform

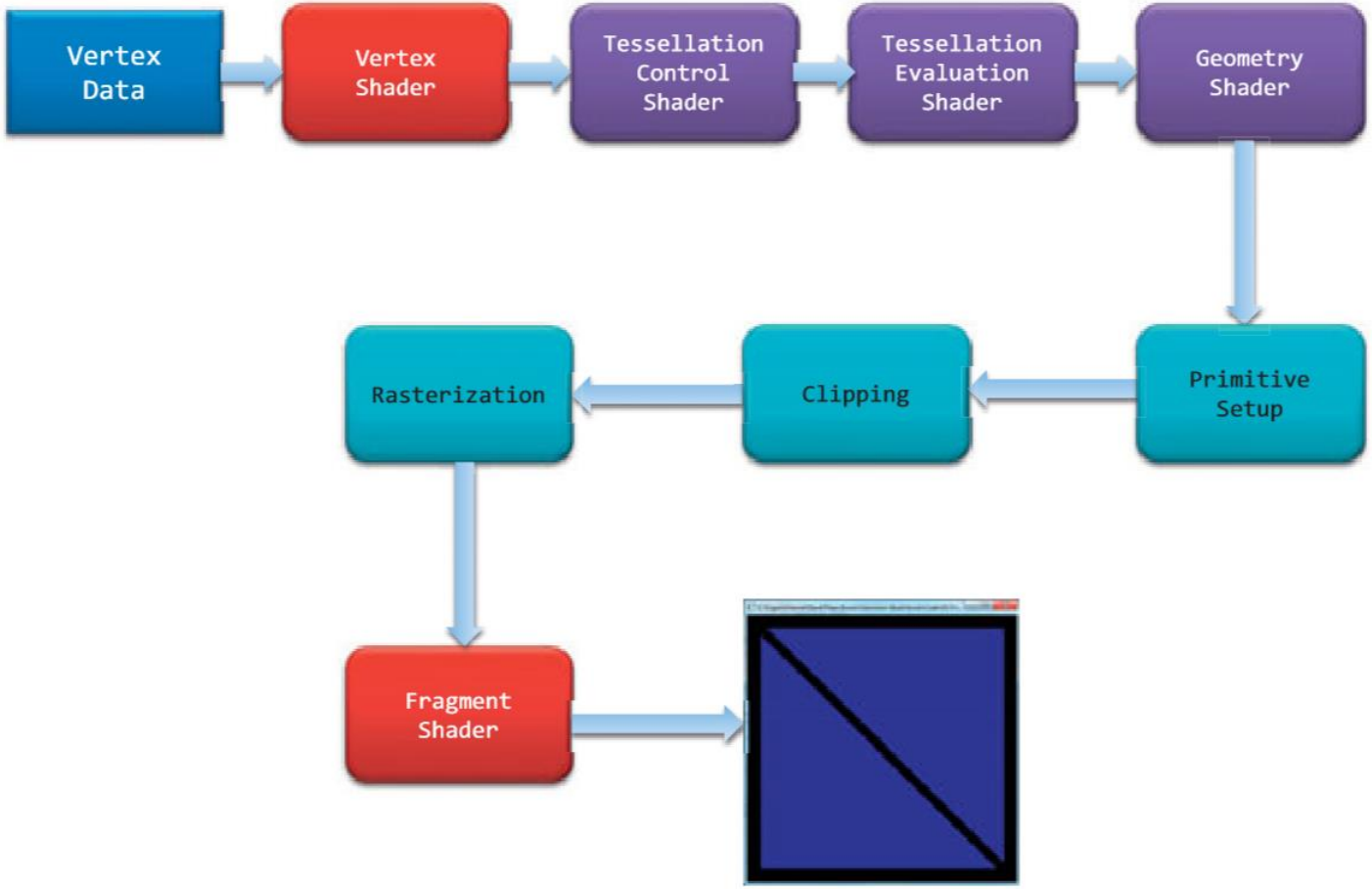


Clipping con perspectiva

Modern pipeline

1. *Setting up the scene.*
2. *Visibility determination.*
3. *Setting object-level rendering states.*
4. *Geometry generation/delivery.*
5. *Transformation and lighting.*
6. *Backface culling and cliping.*
7. *Projection to screen space.*
8. *Rasterization.*
9. *Pixel shading.*

OpenGL Pipeline



Trabajo en clase (en grupo)

Utilizando el ejemplo "webgl_multiple_views", visualizar el brazo robótico desde 3 cámaras ortogonales, que apunten a los planos XY, XZ y YZ. Estas cámaras deberán usar el 50% superior del canvas.

Ubicar otra cámara en perspectiva en la mano (grip) del brazo robótico, para visualizar en el 50% inferior lo que ve el brazo a medida que se mueve por el ambiente.

Agregar otros elementos a la escena que permitan al usuario visualizar la dirección de movimiento del brazo robótico.


XY
XZ
YZ
VISTA DEL GRIP

Bibliografía

Dunn, F. y Parberry, I. (2002). Chapter 15: 3D Math for Graphics en: ***3D Math Primer for Graphics and Game Development***. Wordware Publishing, Inc.

Hughes, J et al. (2014). Chapter 13: Camera Specifications and Transformations en: ***Computer Graphics: Principles and Practice***. 3rd Ed. Addison-Wesley.

Joy, Ken. (2009). ***Lecture 05: The pinhole camera model and associated transform matrix***. UC Davis

Tutorial 3: Matrices. http://www.opengl-tutorial.org/beginners-tutorials/tutorial-3-matrices/