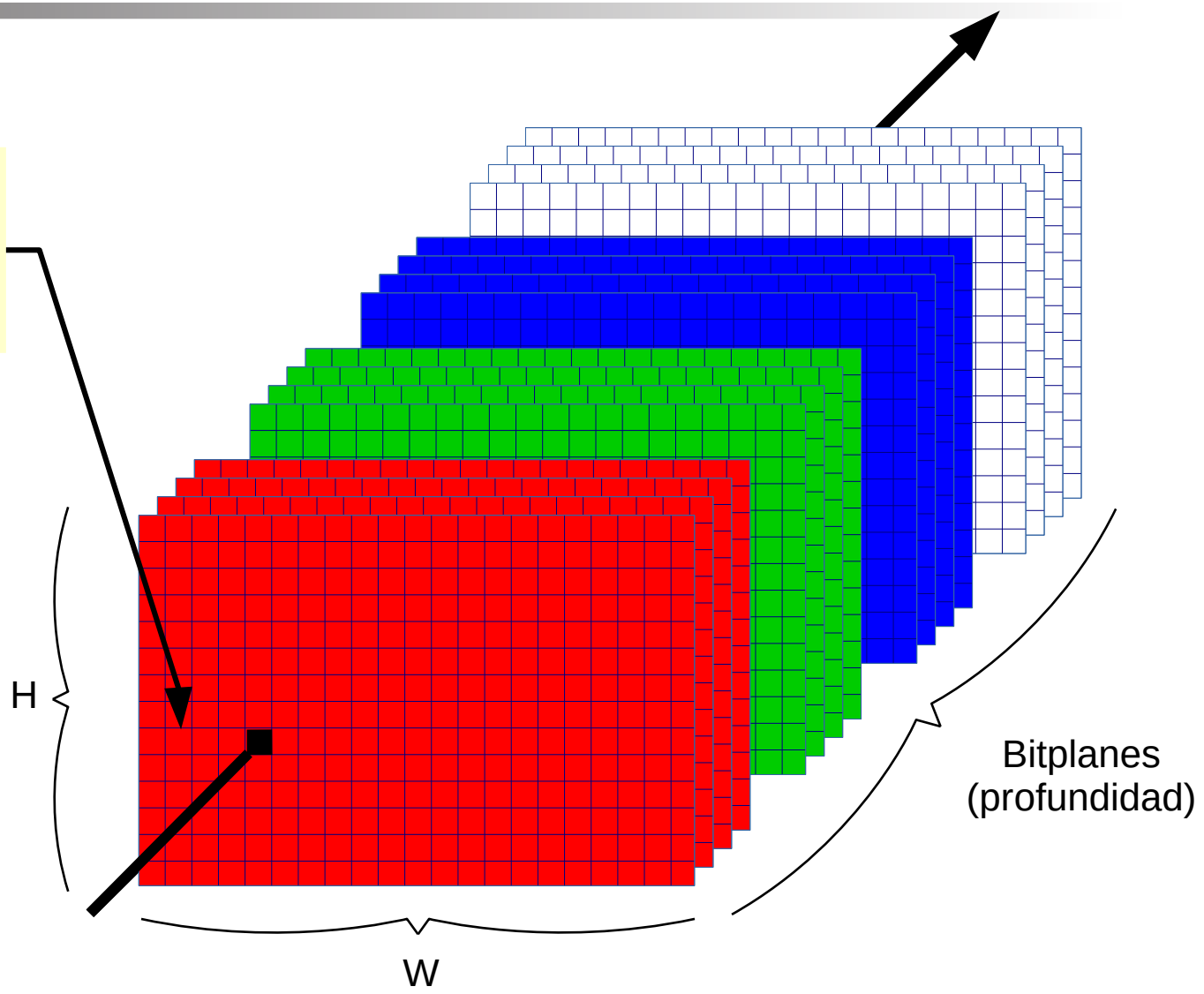
Unidad 7

Técnicas del Espacio de la Imagen

UNL FICH
Universidad Nacional del Litoral
Facultad de Ingeniería y Ciencias Hídricas

Buffer de Color

Pixel {x, y}
de color
{R, G, B, A}

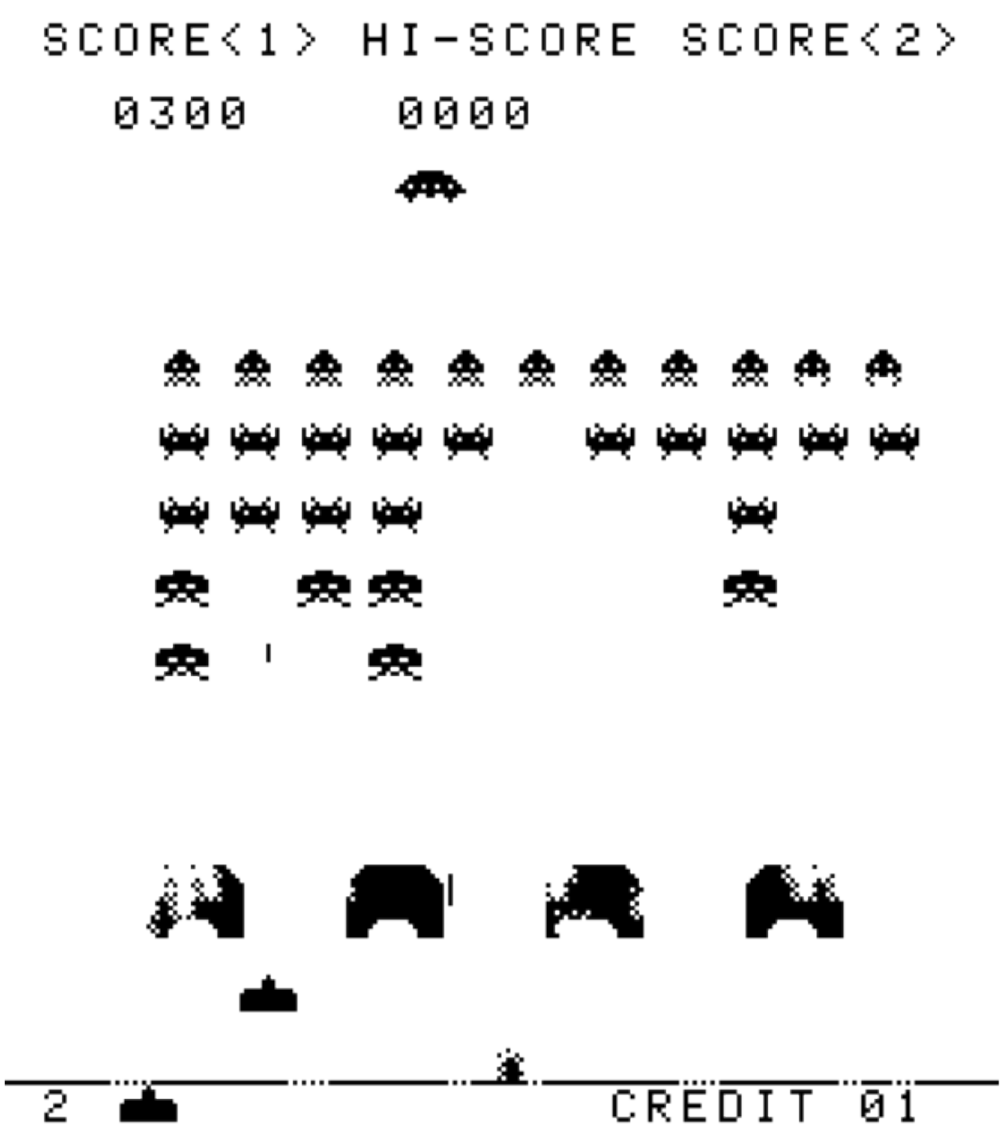


R0, G0, B0, A0, R1, G1, B1, A1, R2, G2, B2, A2, R3, G3, B3, A3...

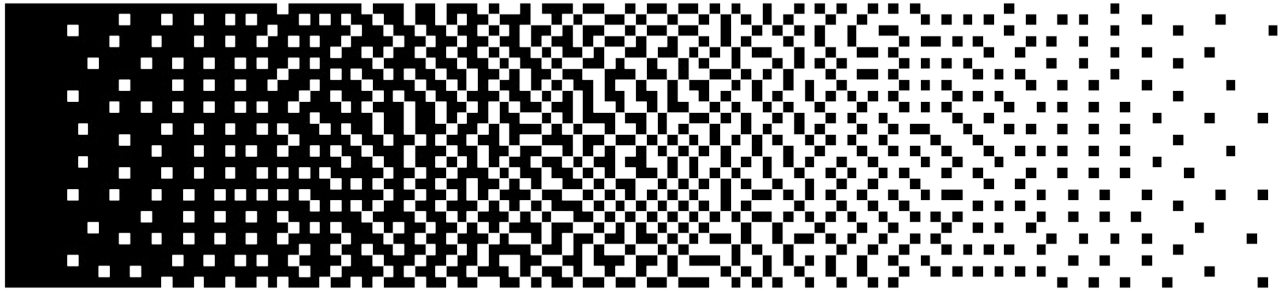
Profundidad de Color: Monocromo (1 bit)

1 bit (y poca resolución)

SPACE INVADERS

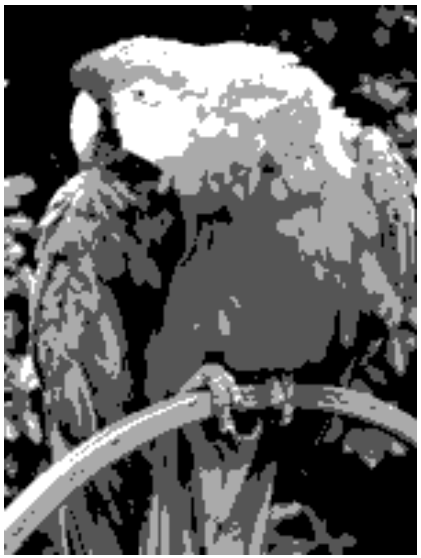


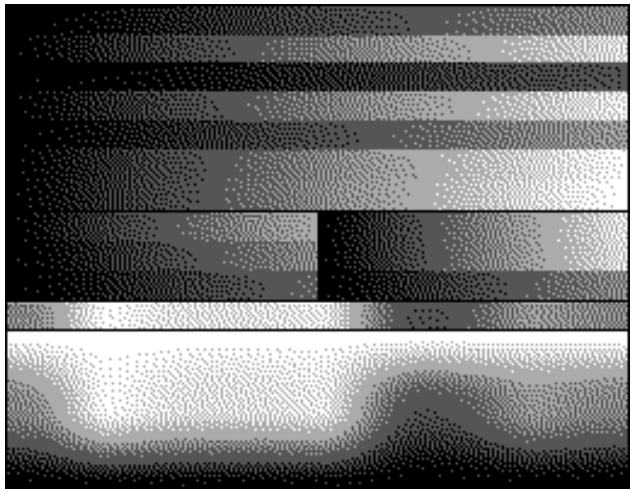
Profundidad de Color: Dithering



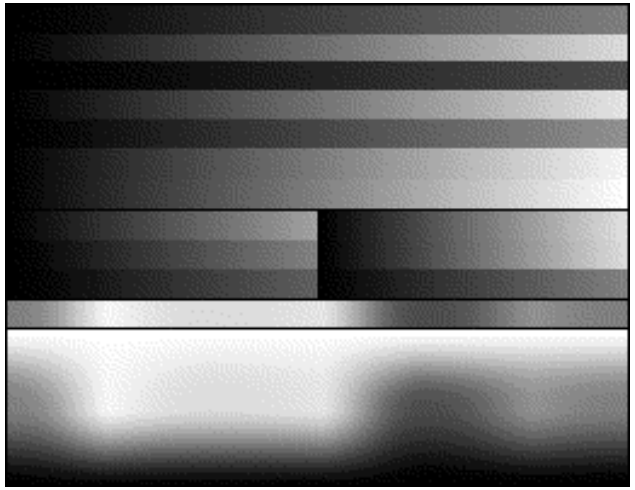
Profundidad de Color (2/4 bits)

2 bit





4 bit

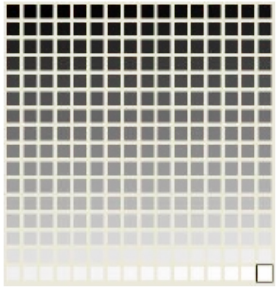


Profundidad de Color: 256 Tonos/Colores (8bits)

256 Colores

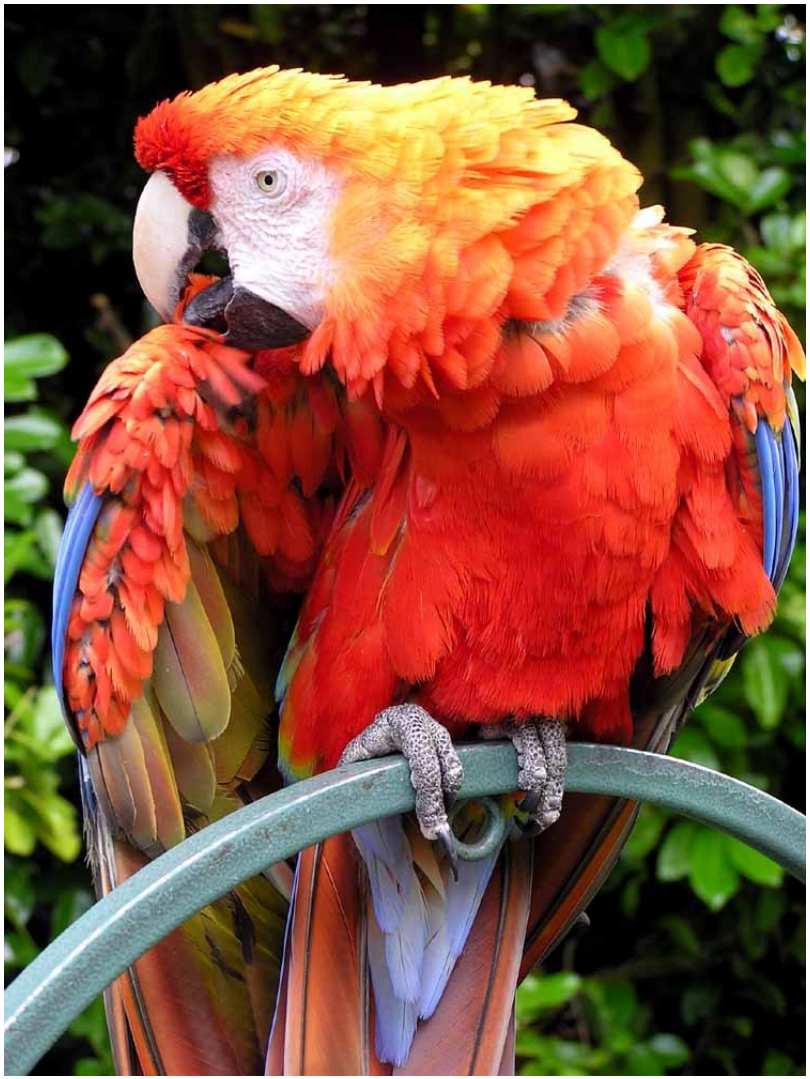
256 Tonos

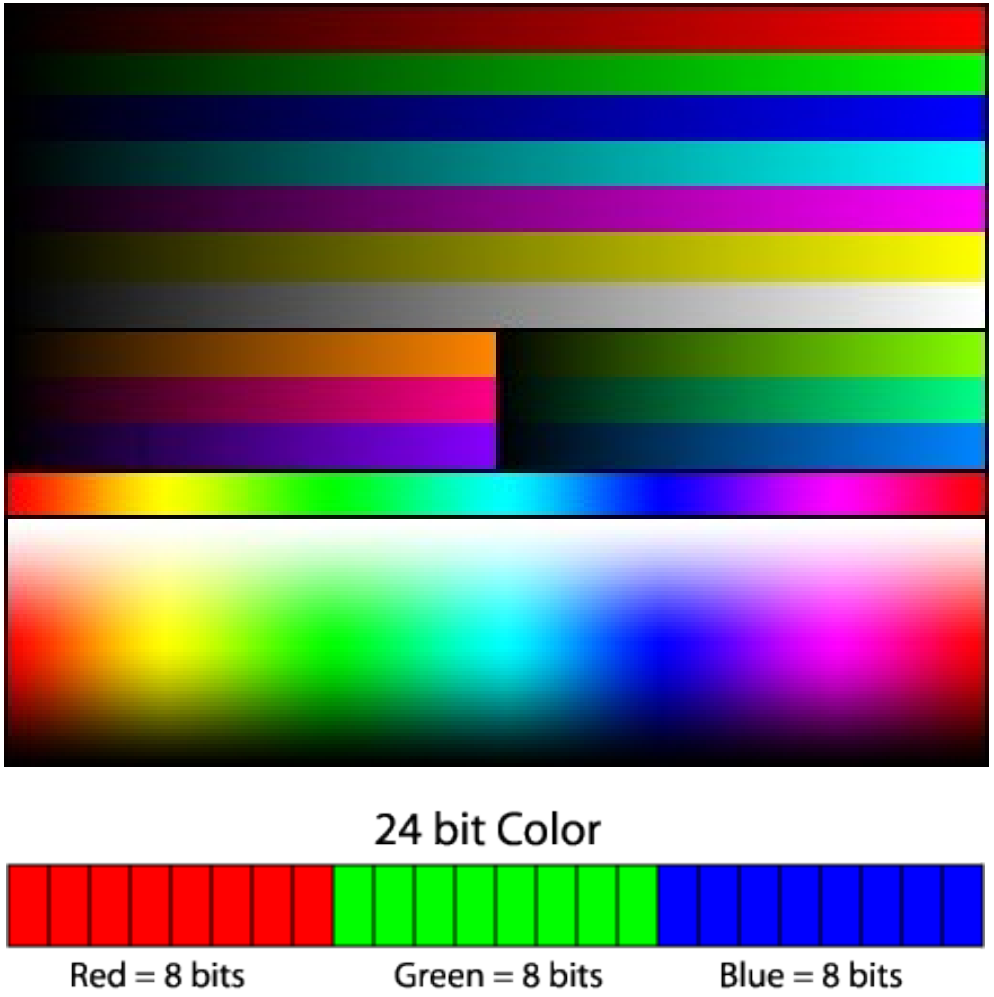
Colores indexados



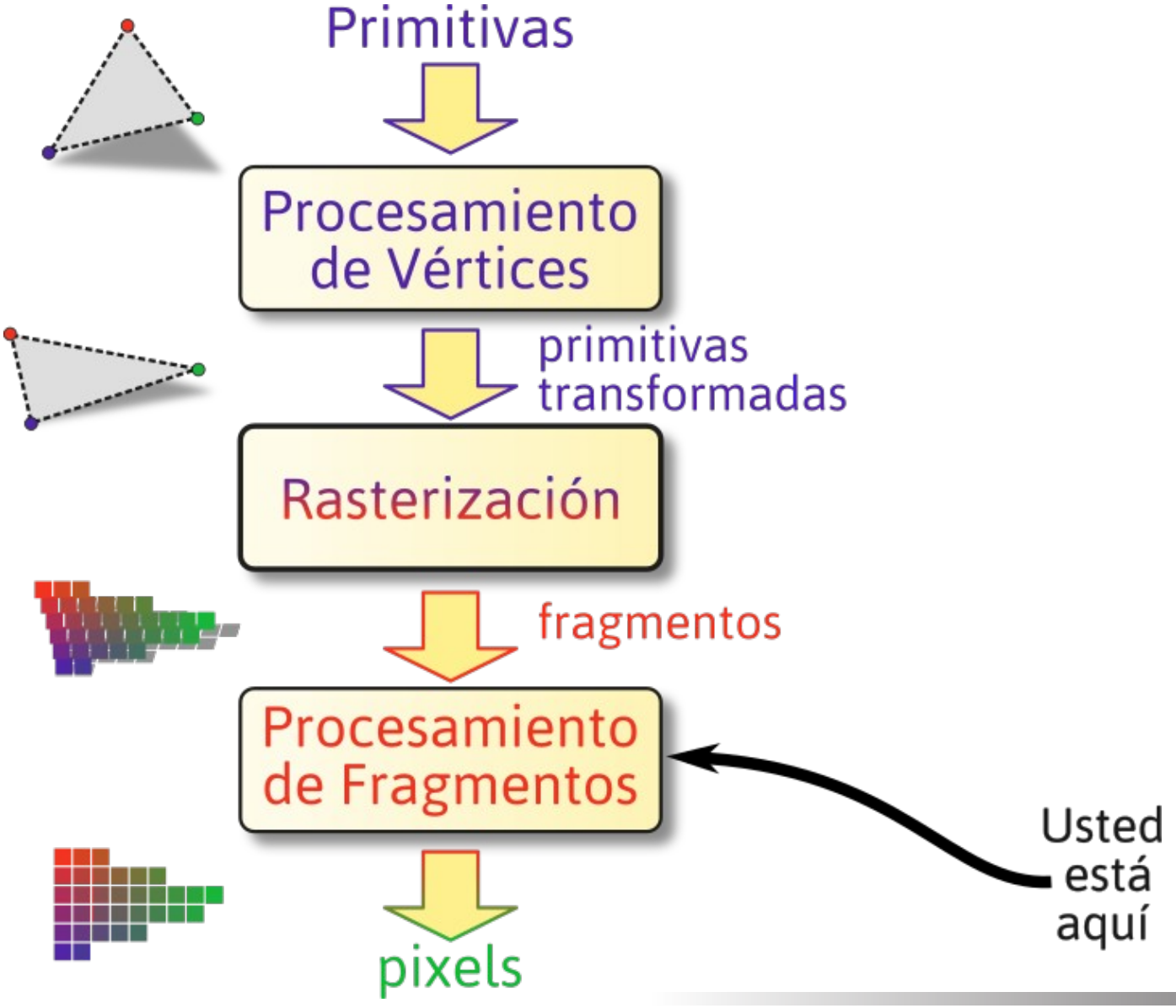
http://www.effectgames.com/demos/canvascycle/

Profundidad de Color: True Color (24bits)

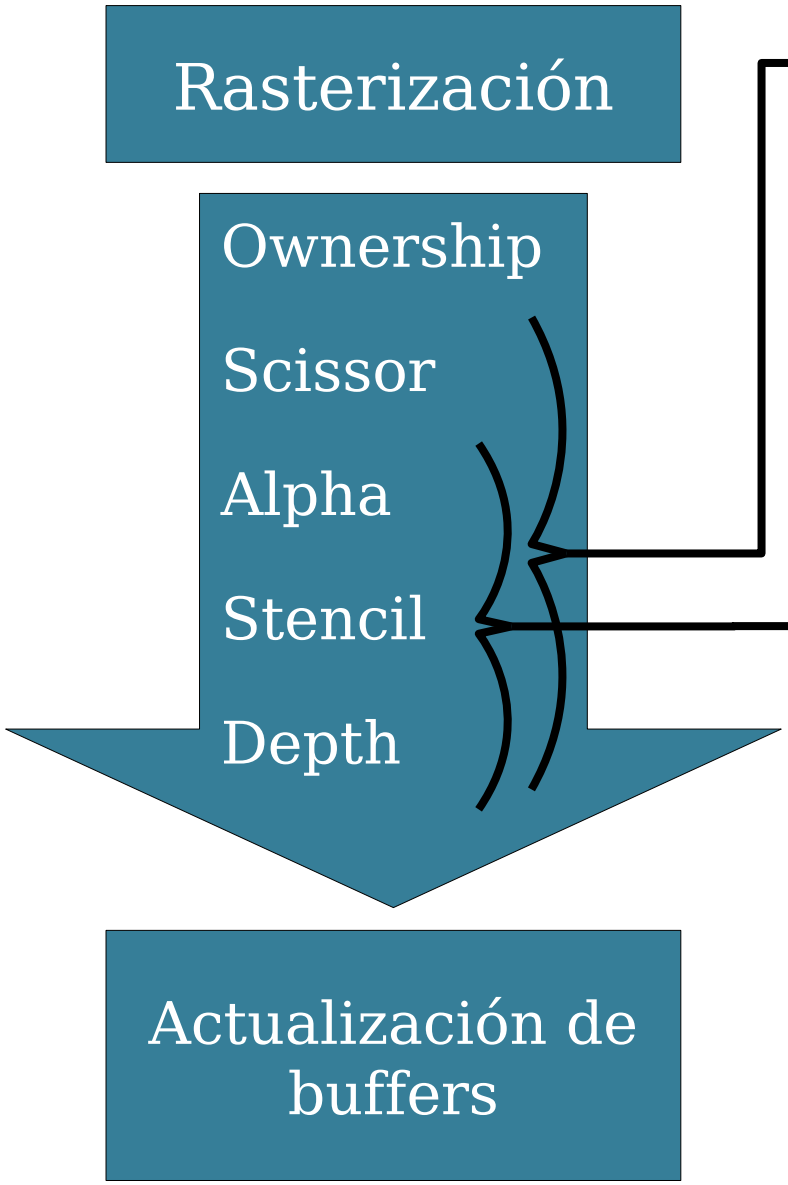




Usted está aquí



Tests de Fragmentos



glEnable(GL_xxxxxx_TEST)

glDisable(GL_xxxxxx_TEST)

glXxxxxFunc(*comp*, [*ref*], [*mask*])

comp: posibles comparadores

- GL_NEVER
- GL_ALWAYS
- GL_LESS
- GL_LEQUAL
- GL_EQUAL
- GL_GEQUAL
- GL_GREATER
- GL_NOTEQUAL

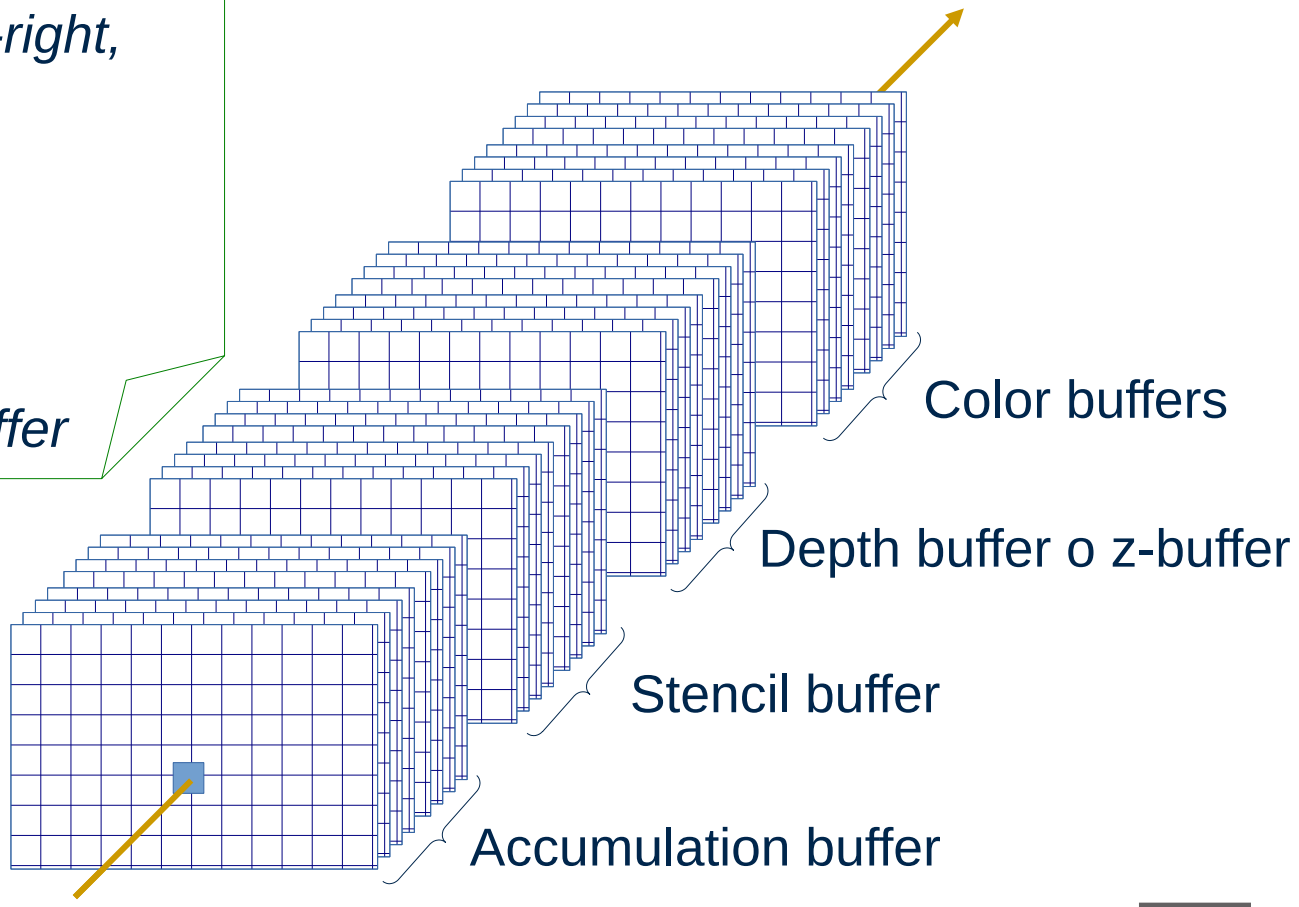
ref: valor con el que se compara (alpha y stencil)

mask: para enmascarar bits en la comparación (solo stencil)

Framebuffer

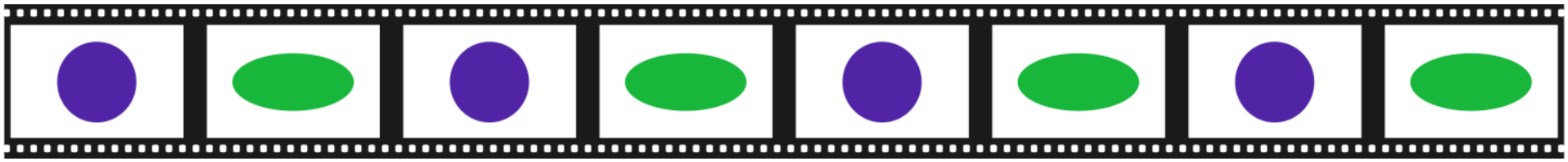
- *Color buffers:*
  - *front-left, front-right,*
  - *back-left, back-right,*
  - *Auxiliary[0-...]*
- *Depth buffer*
- *Stencil buffer*
- *Accumulation buffer*

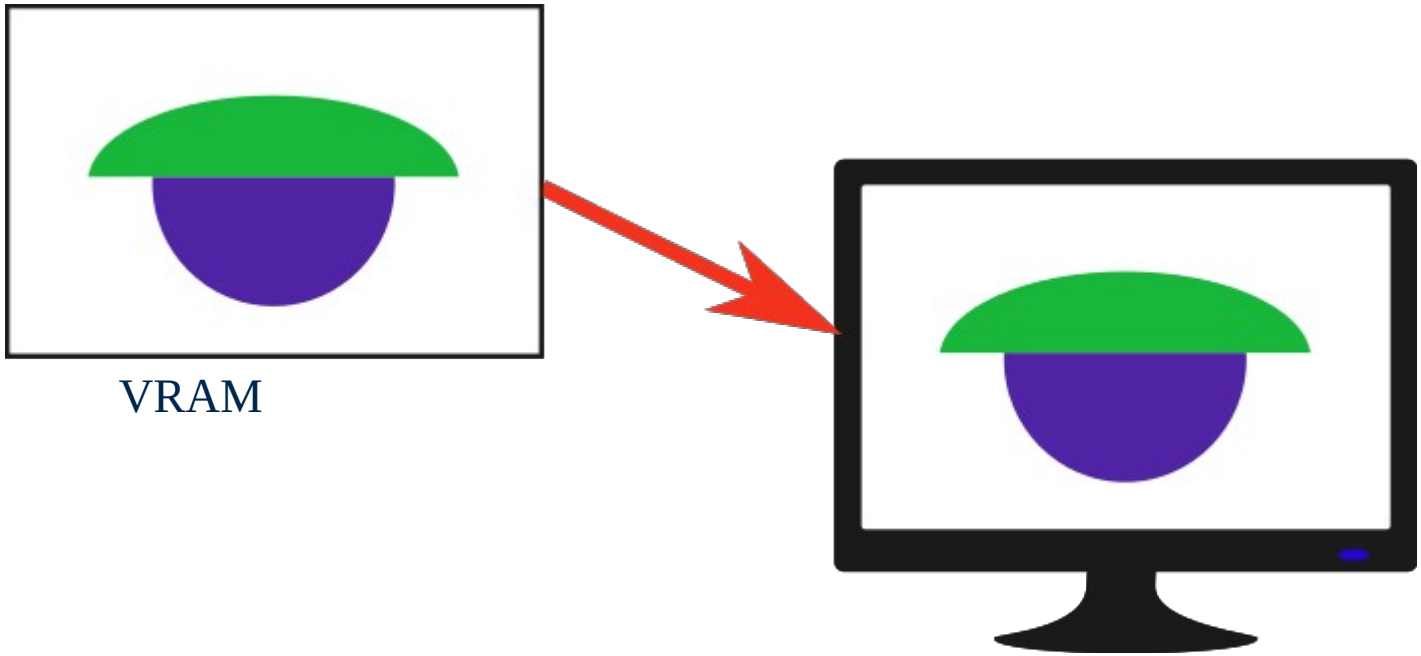
```
glutInitDisplayMode(buffers “or”eados)
glGetIntegerv(GL_DEPTH_BITS,&valor)
```



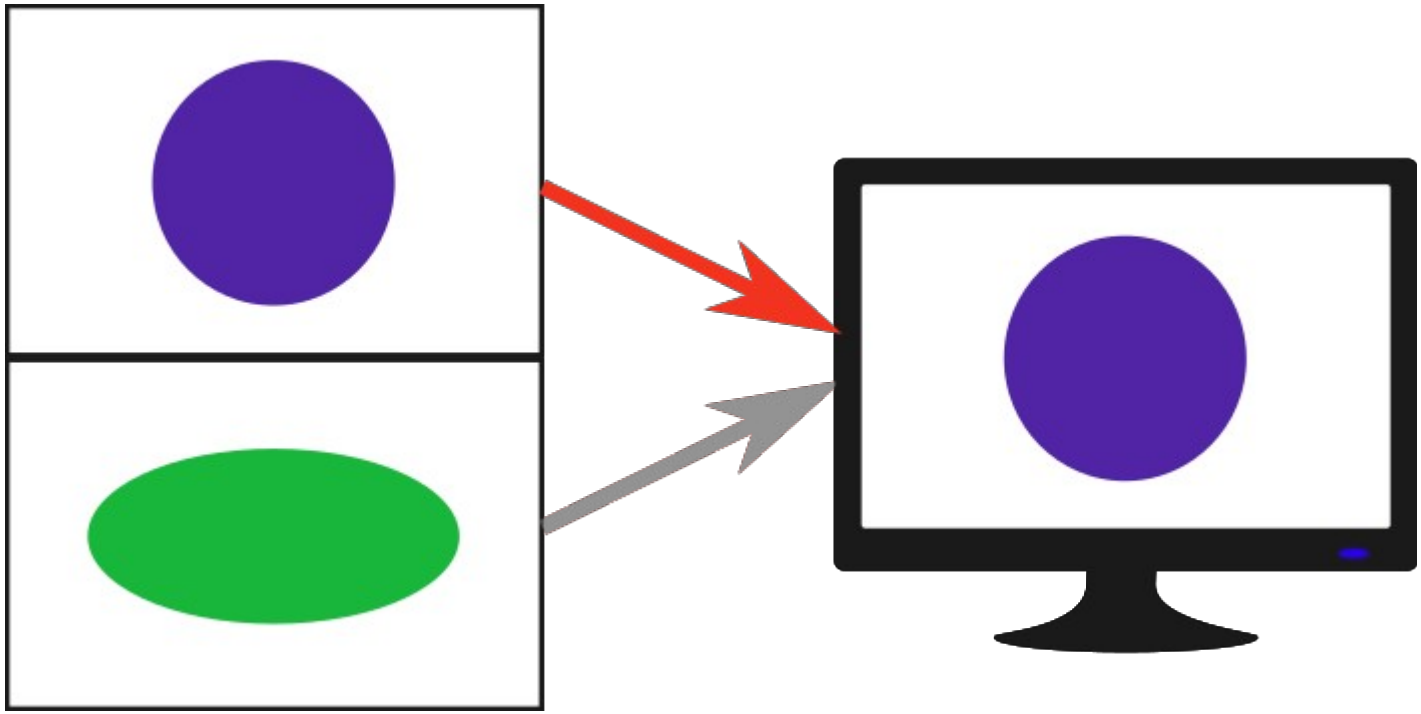
Framebuffer objects en OpenGL 3.0
Render Targets Model en Direct3D 11

Simple Buffer - Tearing



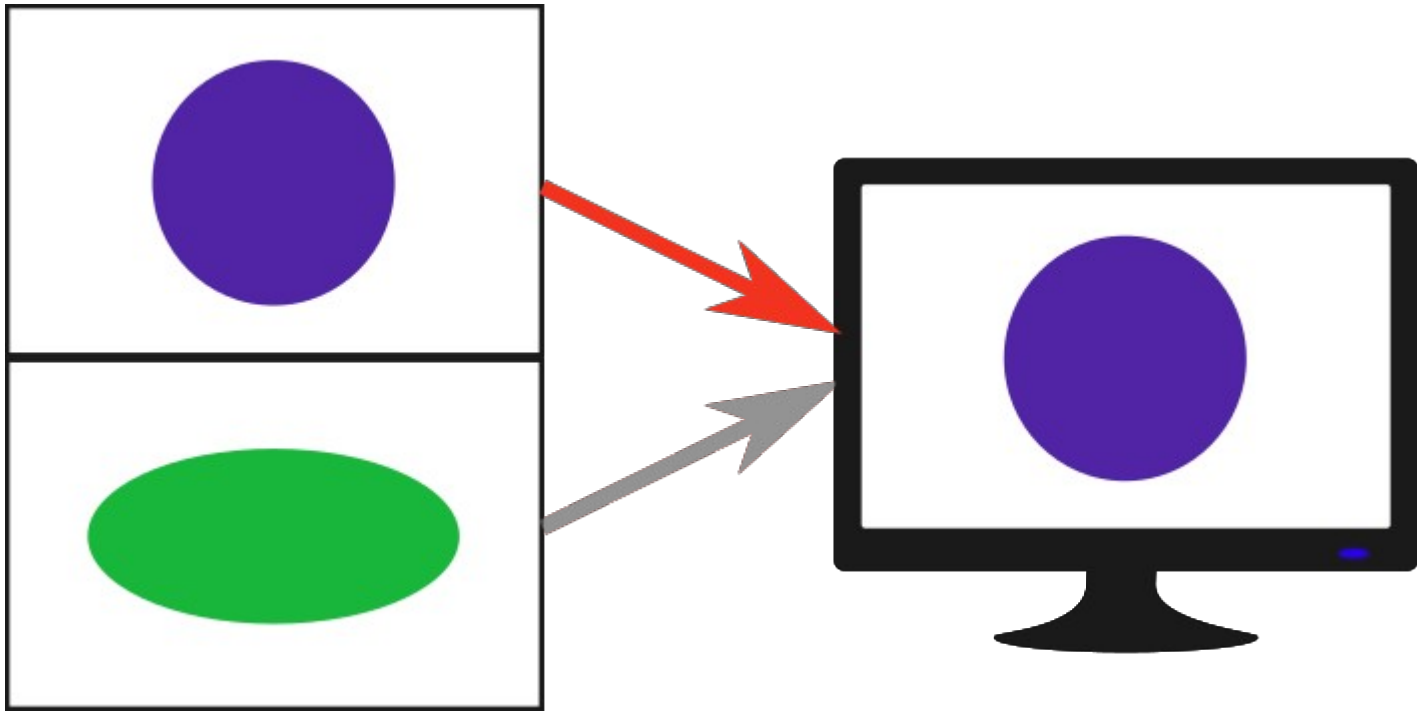


Double Buffer



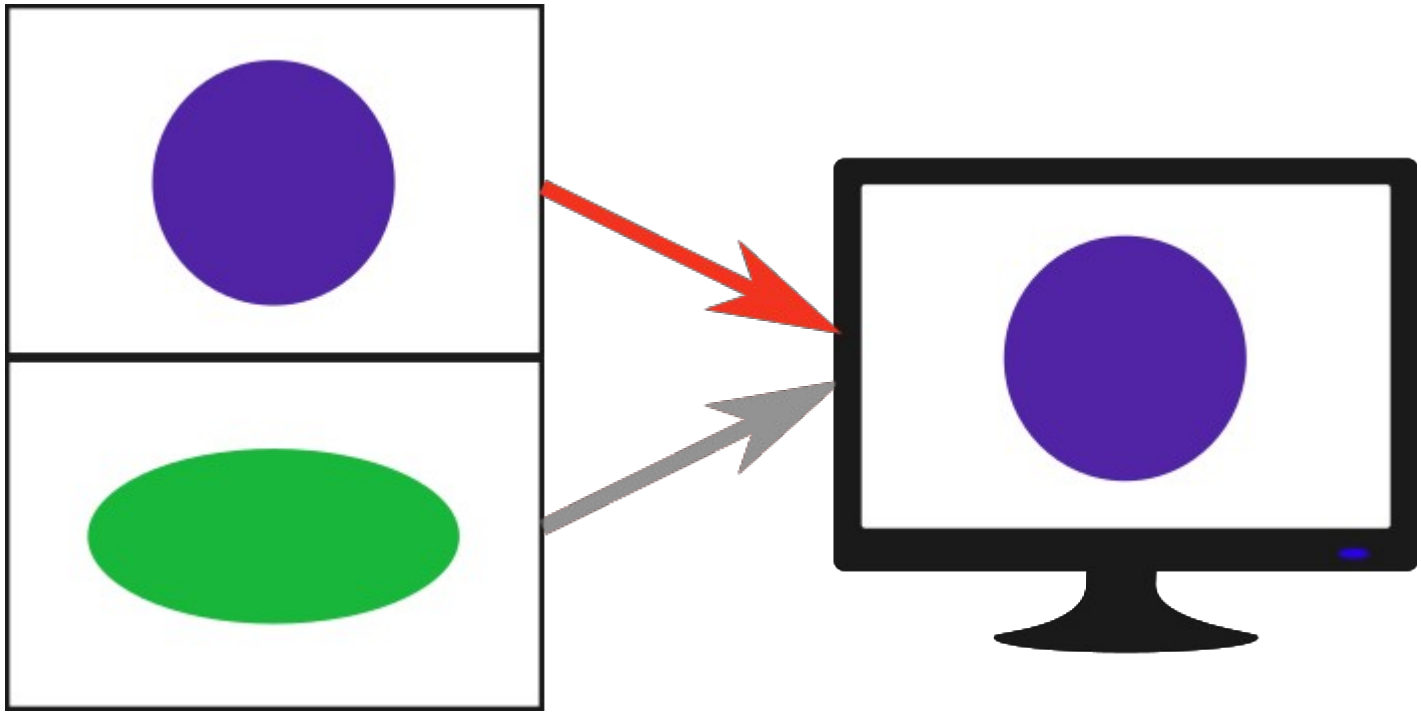
VRAM

Double Buffer



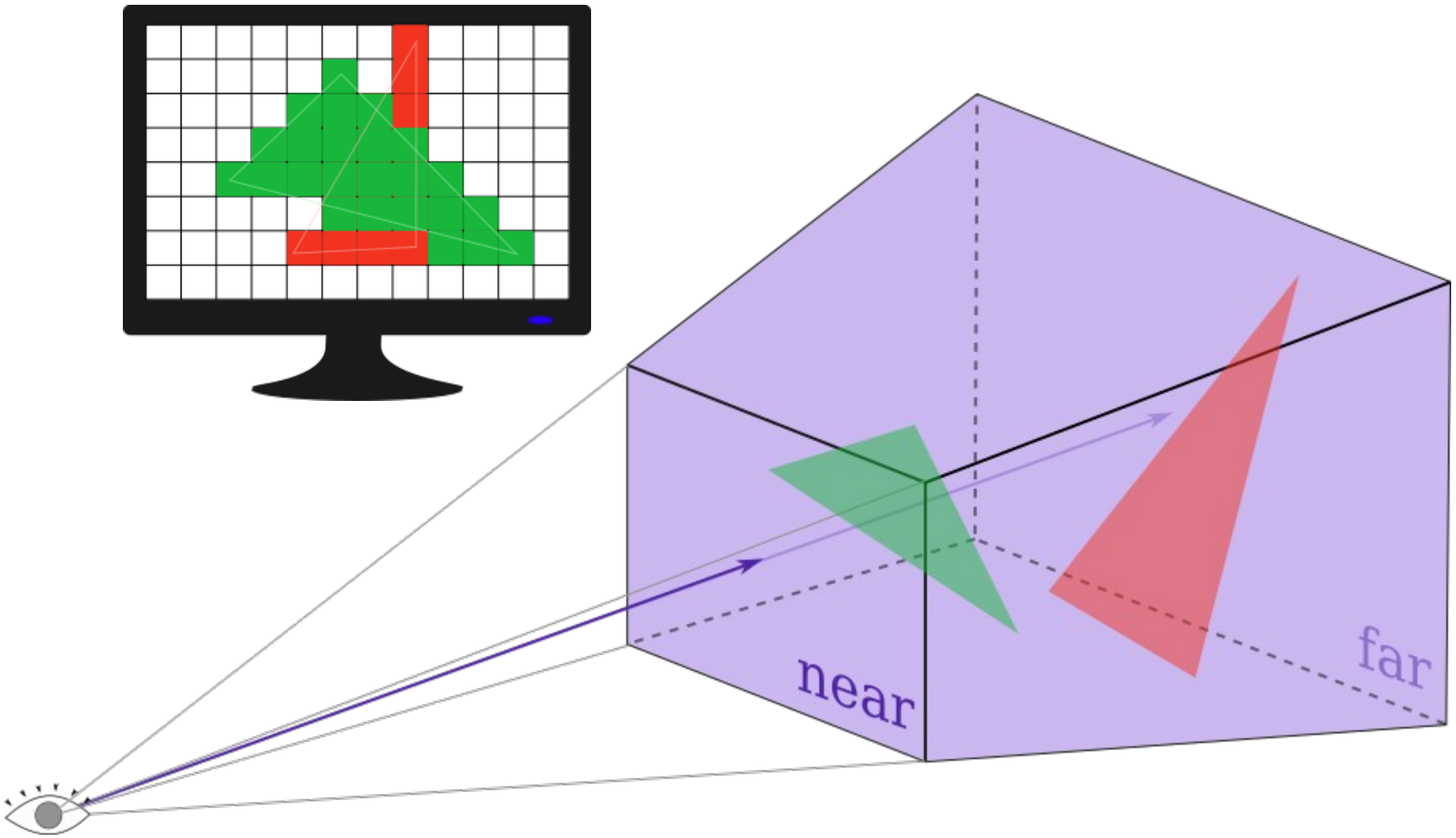
VRAM

Double Buffer

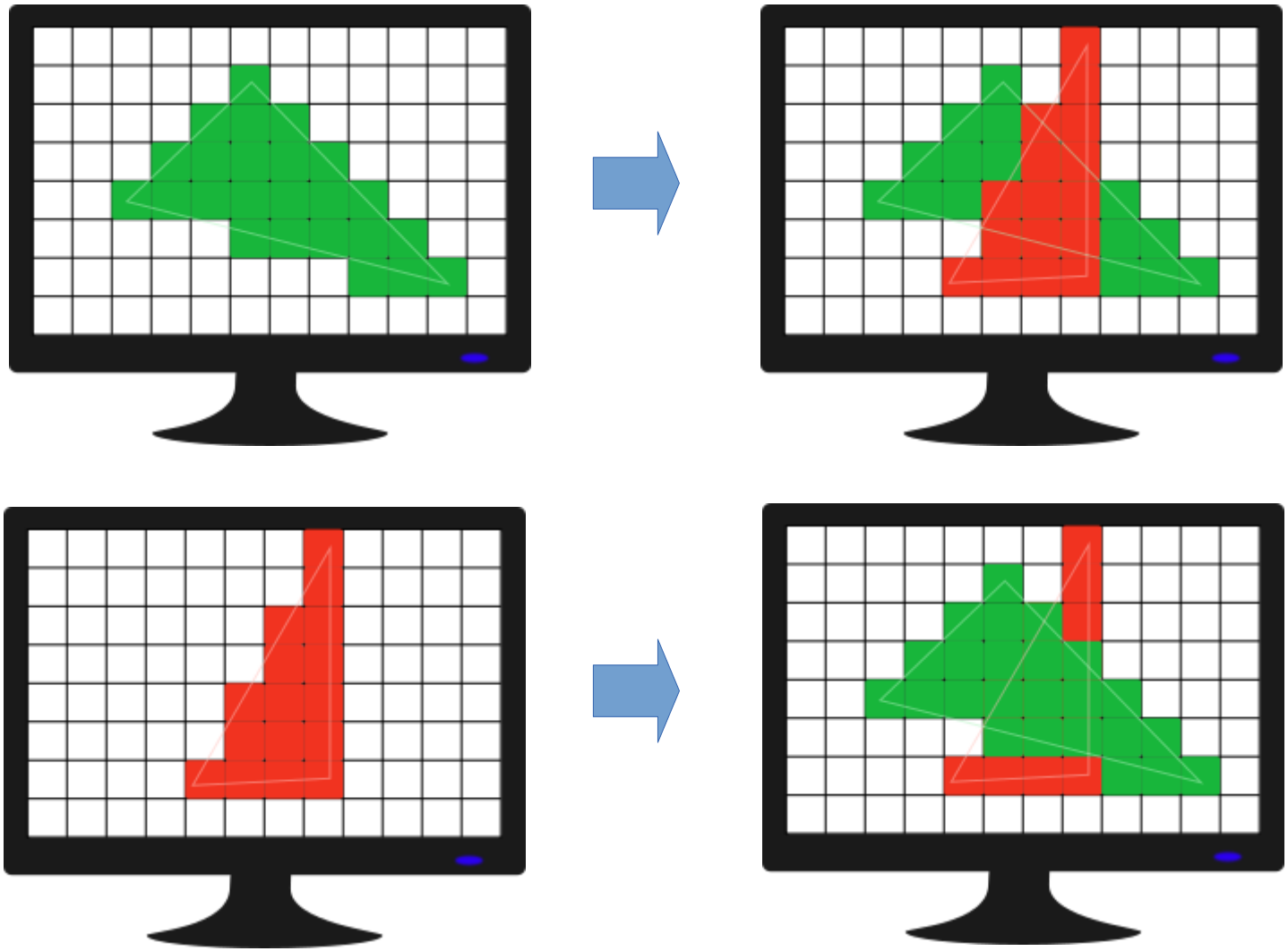


VRAM

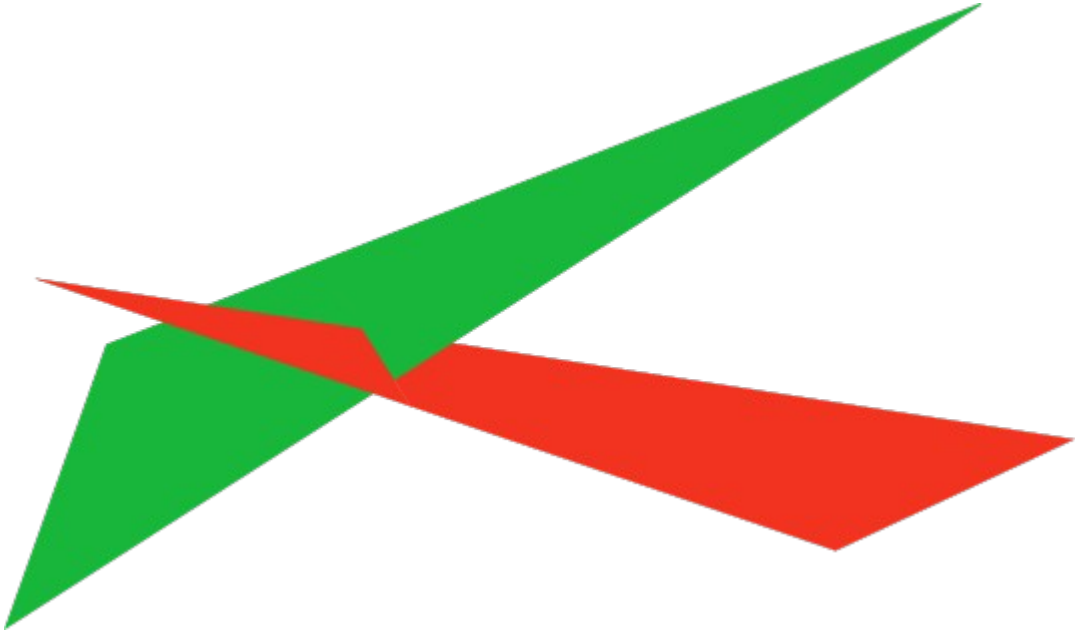
Visibilidad y Rasterización

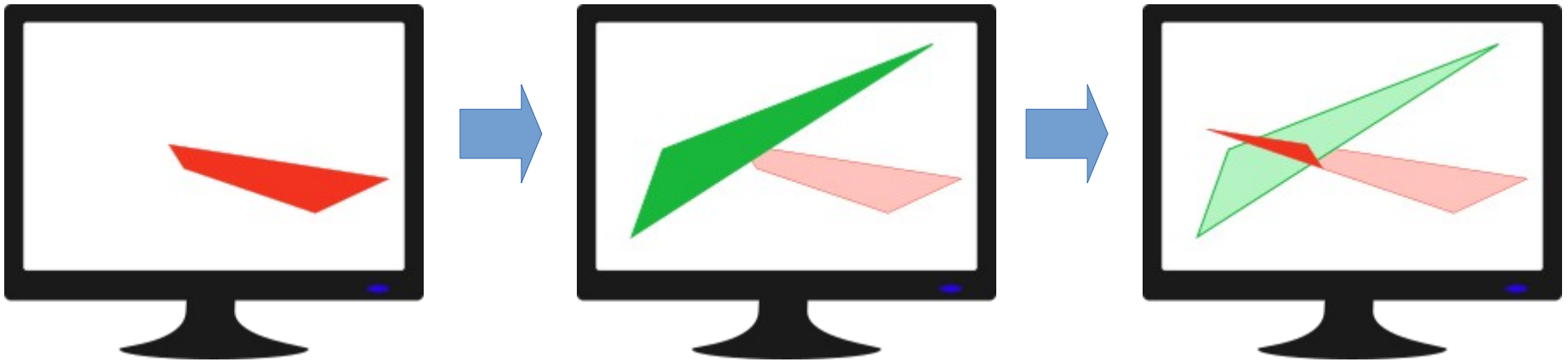


Visibilidad y Rasterización (sin z-buffer)

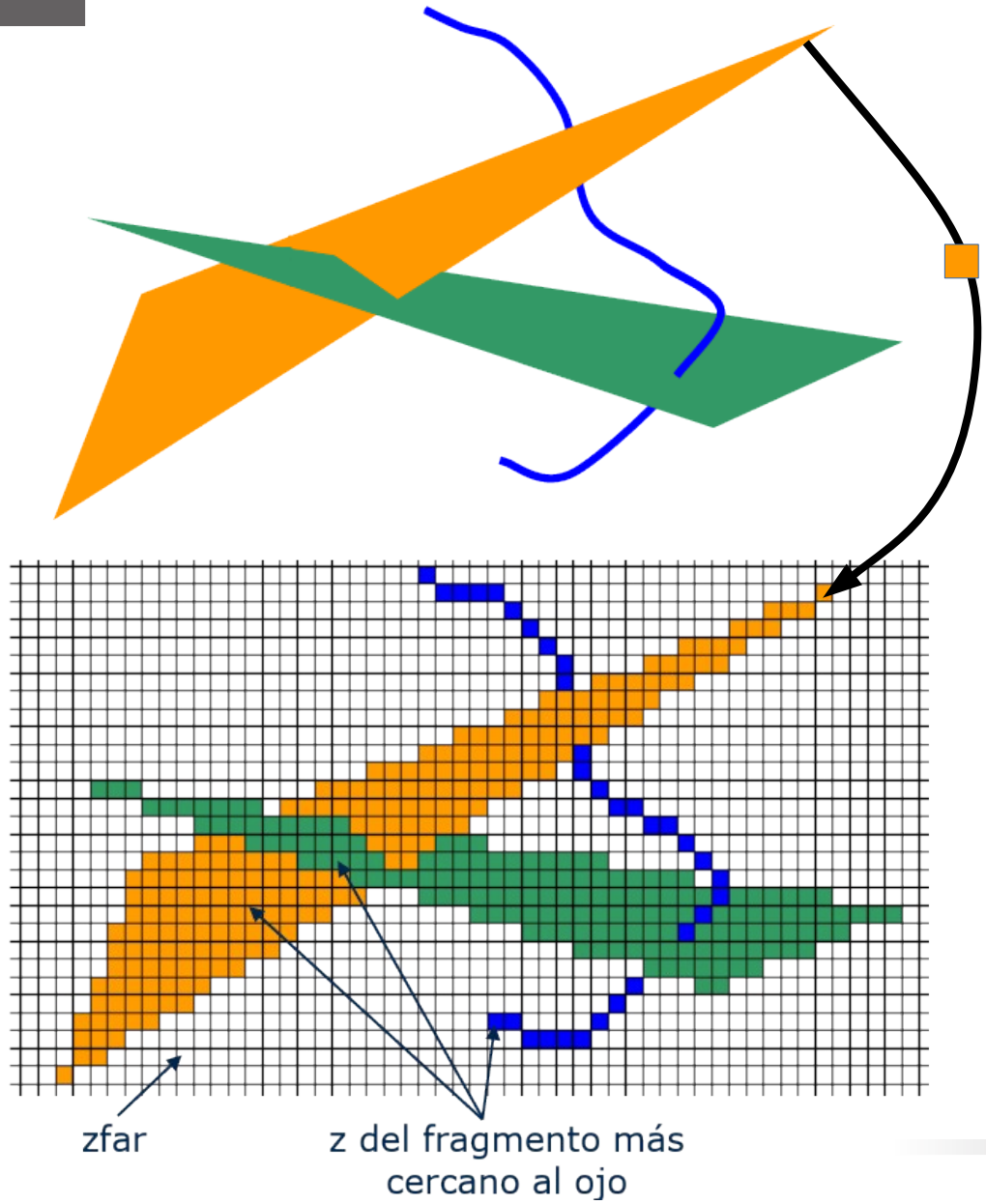


Visibilidad y Rasterización (sin z-buffer)



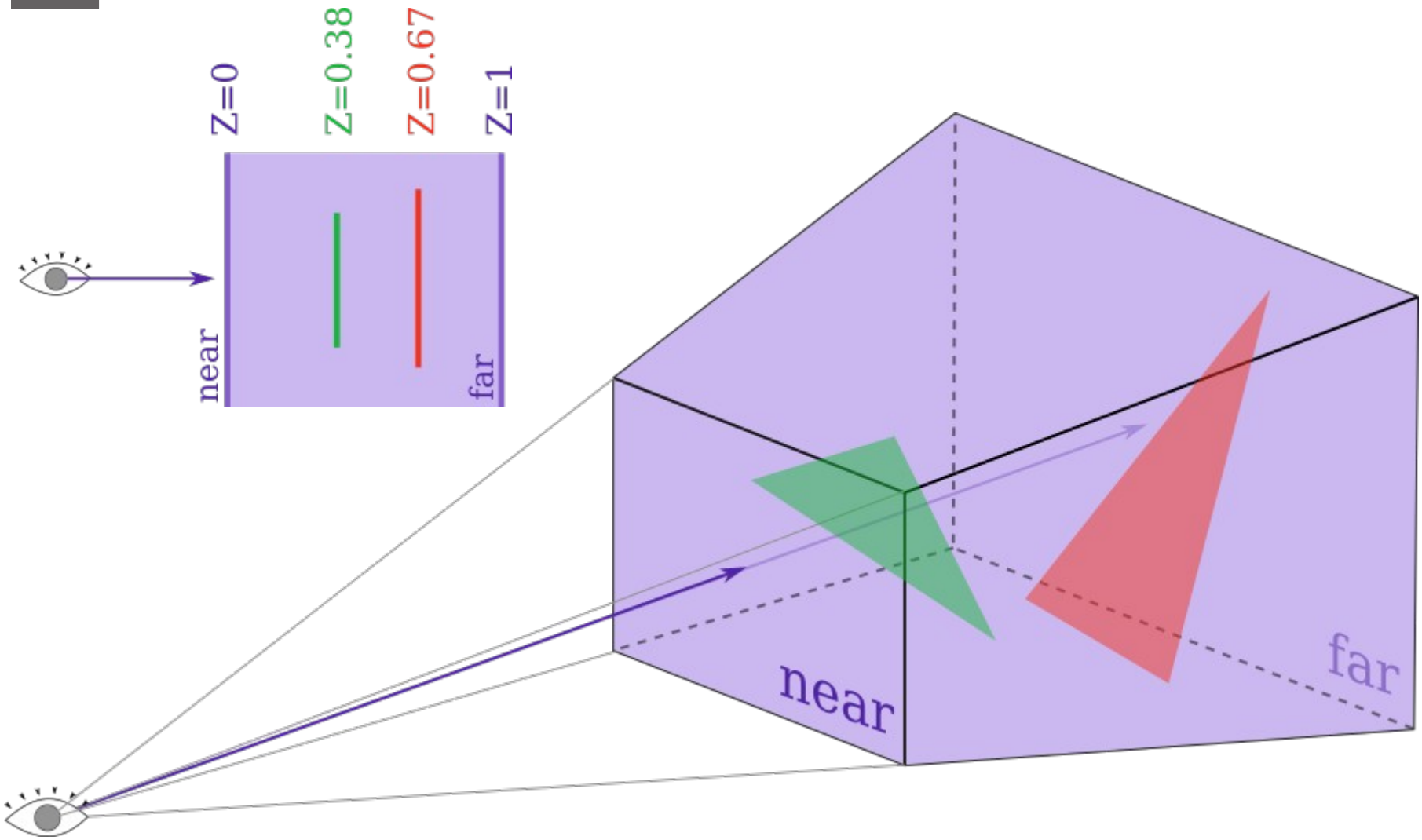


Depth Buffer



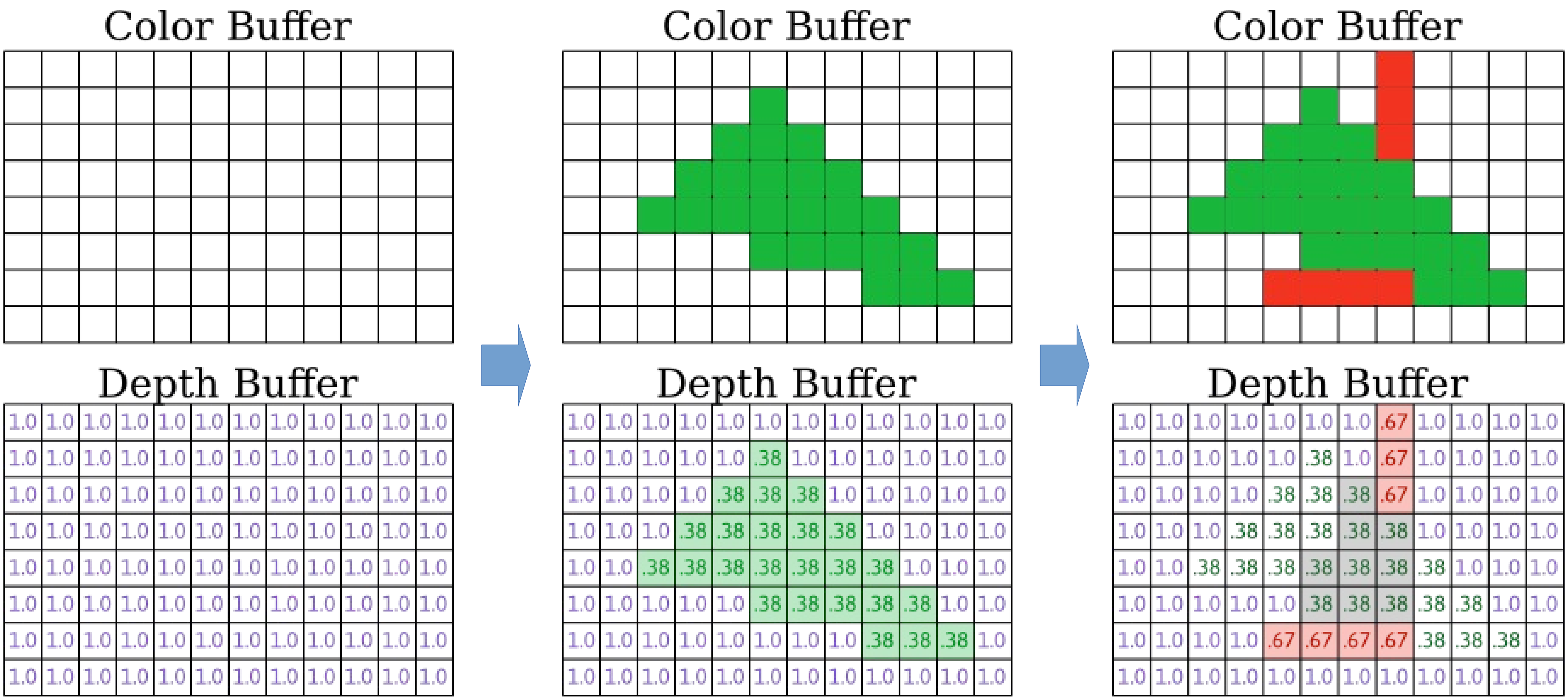

```
Para todo x, y
   depth(x,y) = zfar
Para cada Primitiva:
   Por cada fragmento:
      Si z < depth(x,y)
         color(x,y) = RGBA
         depth(x.y) = z
```

Algoritmo Z-Buffer



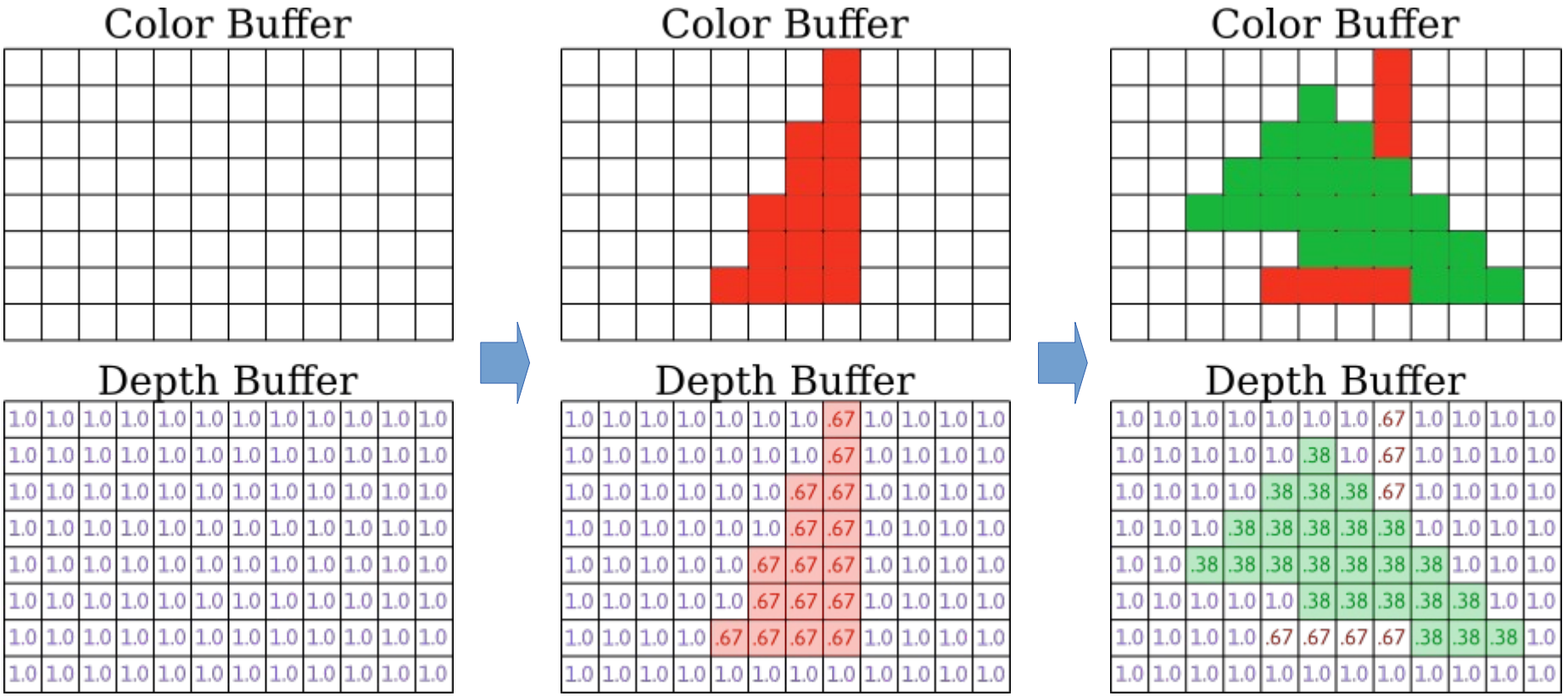
Algoritmo Z-Buffer

Rasterizando 1ro el Verde (z=.38), después el Rojo (z=.67)



Algoritmo Z-Buffer

Rasterizando 1ro el Rojo (z=.67), después el Verde (z=.38)

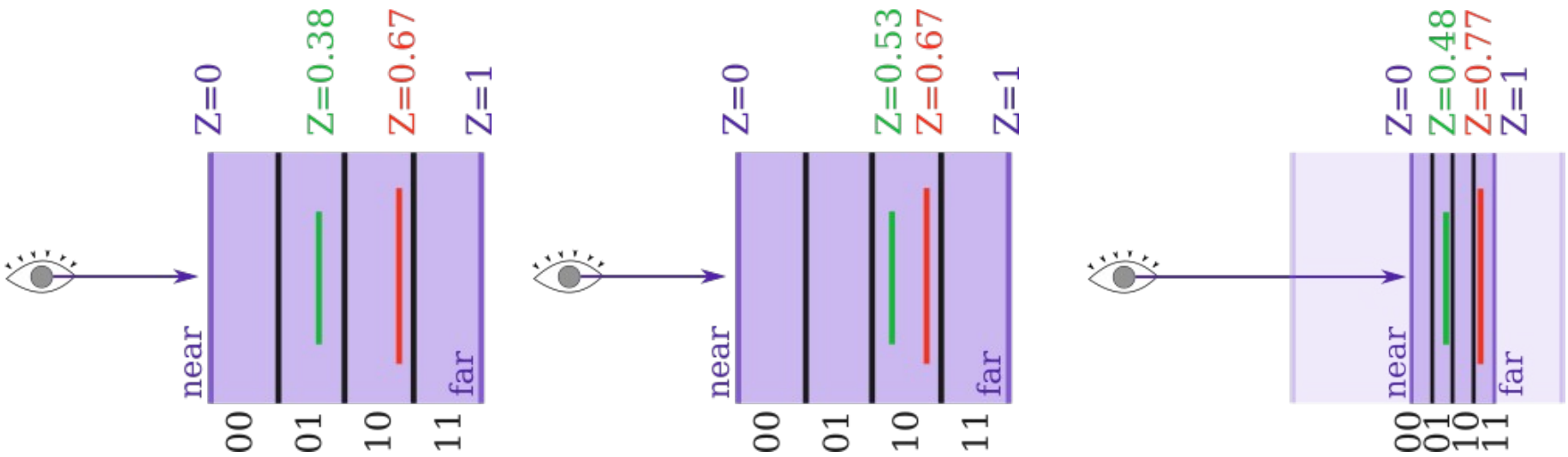


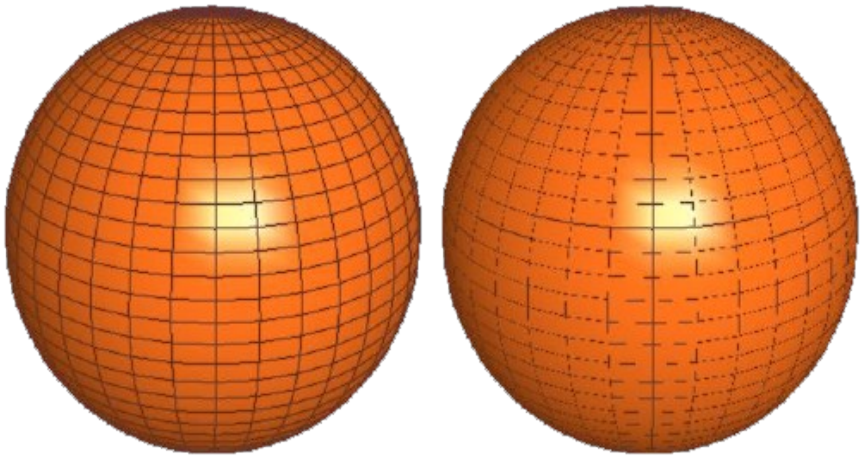
Depth Buffer

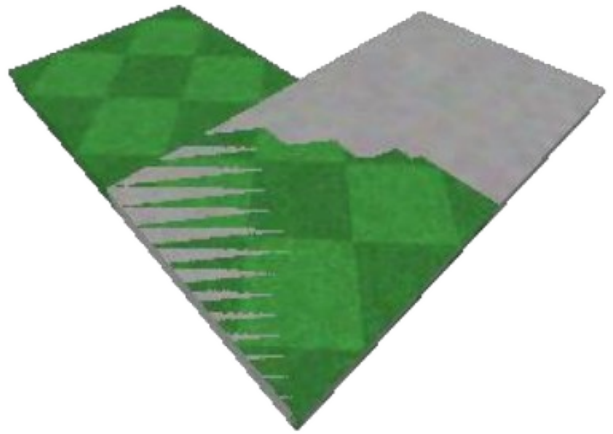
Color buffer

Depth buffer

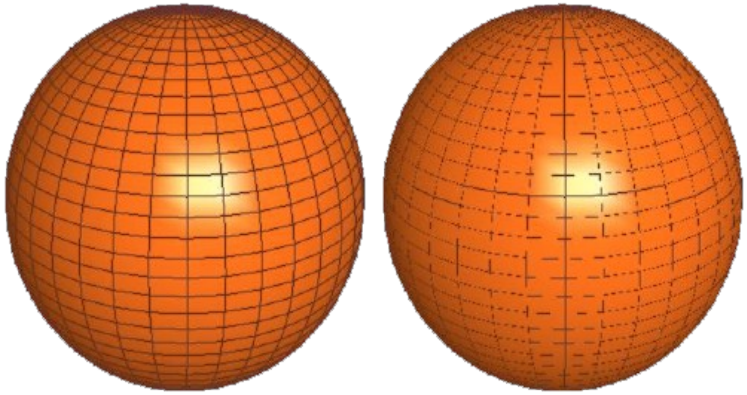
Depth Buffer: Z-Fighting







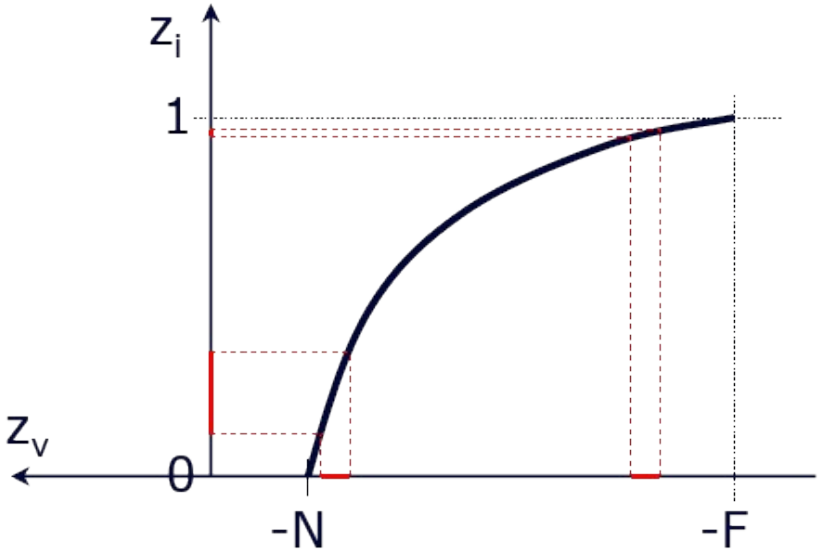
Depth Buffer: Z-Fighting



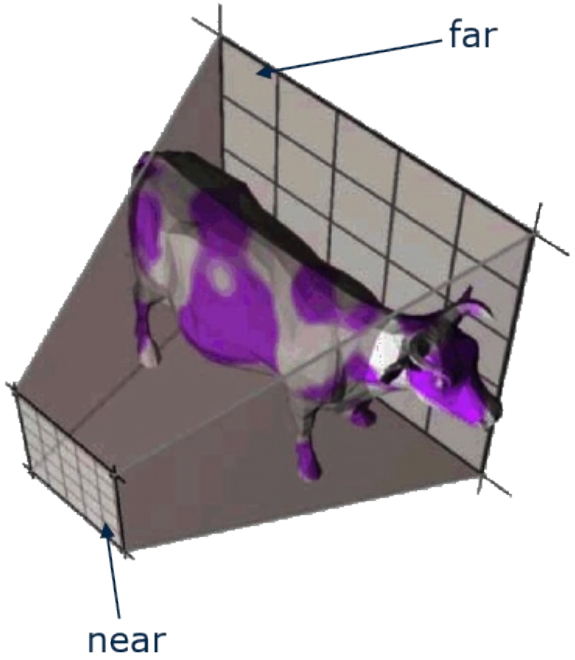
Modificar los valores de Z

```
void glPolygonOffset(GLfloat factor,
                     GLfloat units)
```

Ajustar los planos Near y Far



mayor precisión
cerca del plano near



Depth Buffer: Z Visual vs. Z de Imagen

Frustum:

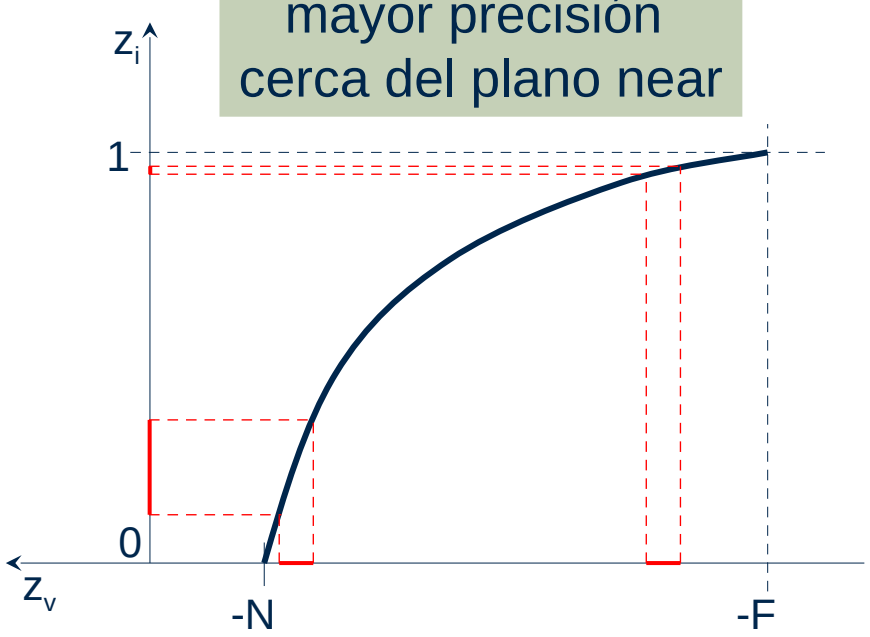
- **F**ar
- **N**ear
- **R**ight
- **L**eft
- **T**op
- **B**ottom

$$\begin{pmatrix} \frac{2N}{R-L} & 0 & \frac{R+L}{R-L} & 0 \\ 0 & \frac{2N}{T-B} & \frac{T+B}{T-B} & 0 \\ 0 & 0 & -\frac{F+N}{F-N} & \frac{-2FN}{F-N} \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} \dots \\ \dots \\ -\frac{F+N}{F-N}z - \frac{2FN}{F-N} \\ -z \end{pmatrix}$$

Matriz Proyectiva

$$z_{imagen} = \frac{F+N}{F-N} + \frac{1}{z}\frac{2FN}{F-N} = a + \frac{b}{z}$$

mayor precisión cerca del plano near

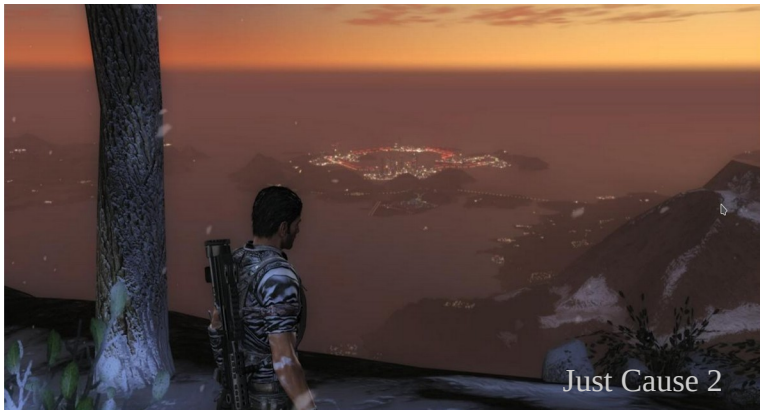


Depth Buffer: Z-Fighting

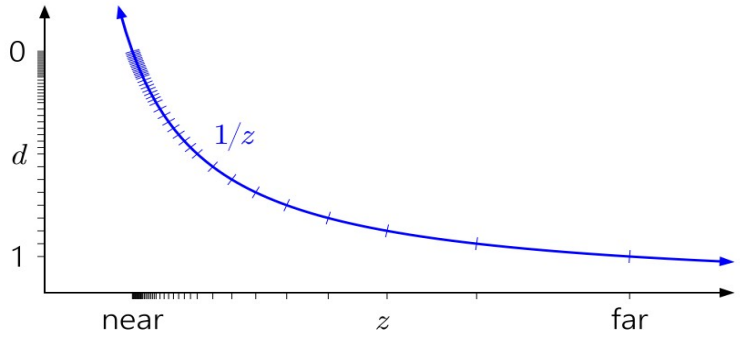
En la práctica hay muchos otros trucos y detalles que considerar...

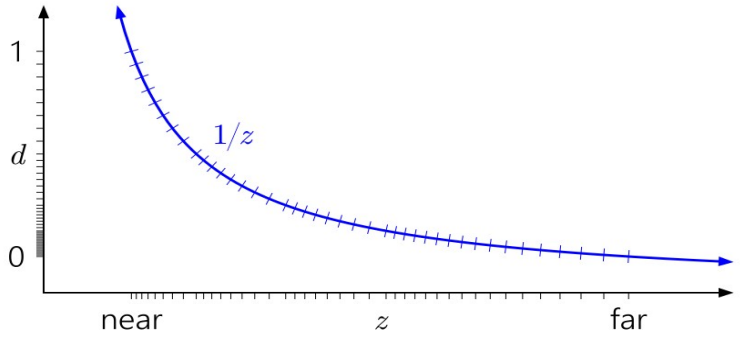
- Ej: Profundidad logarítmica:

$$z_{imagen} = \frac{2\log(z - N + 1)}{\log(F - N + 1)} - 1$$



- Ej: Z invertido



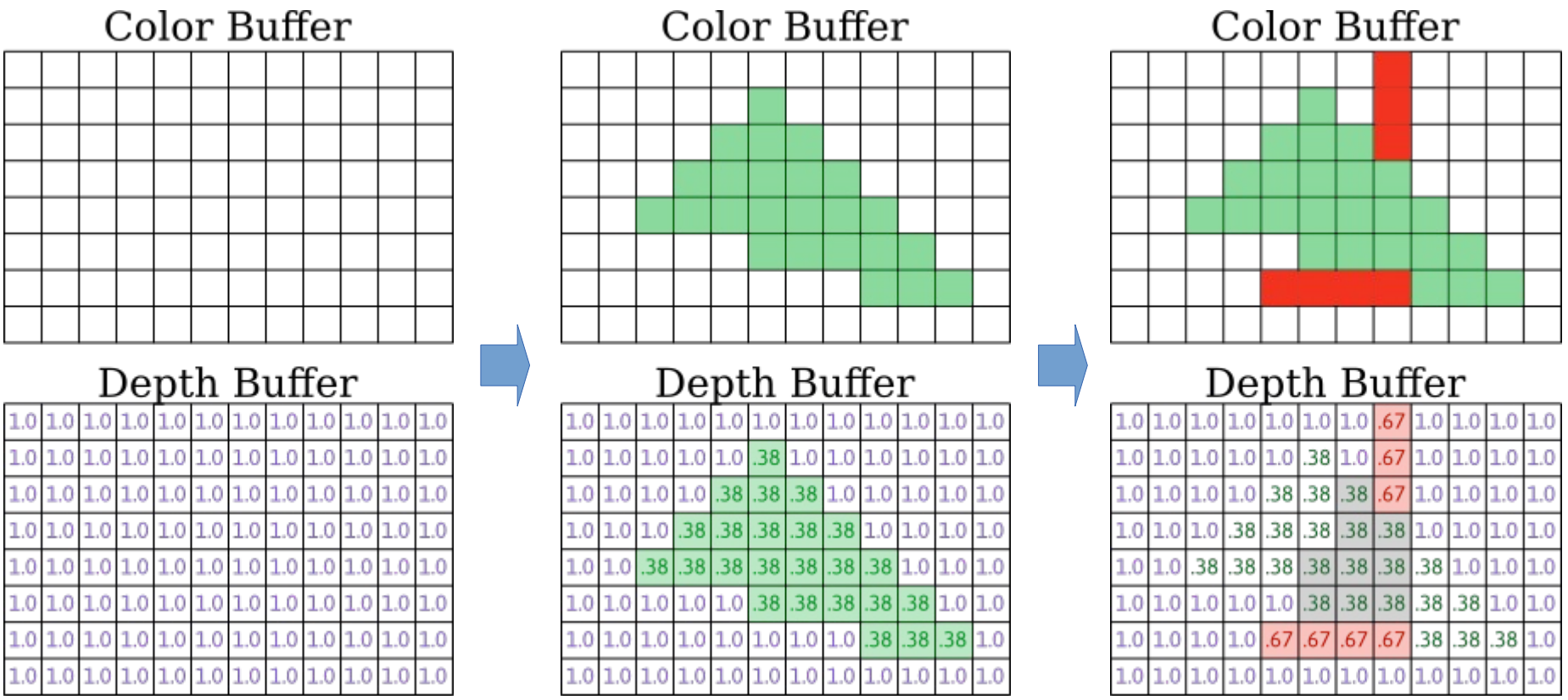


- Y más...

http://www.reedbeta.com/blog/depth-precision-visualized/

Algoritmo Z-Buffer y Transparencias

Rasterizando 1ro el Verde (z=.38), después el Rojo (z=.67)



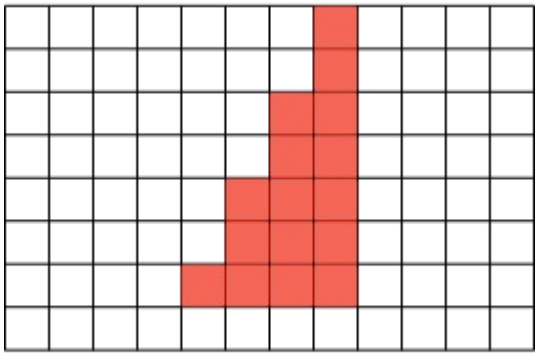
Depth Buffer: Transparencias Múltiples

Pocos objetos transparentes → Ordenamiento espacial

http://www.adriancourreges.com/blog/2015/11/02/gta-v-graphics-study/

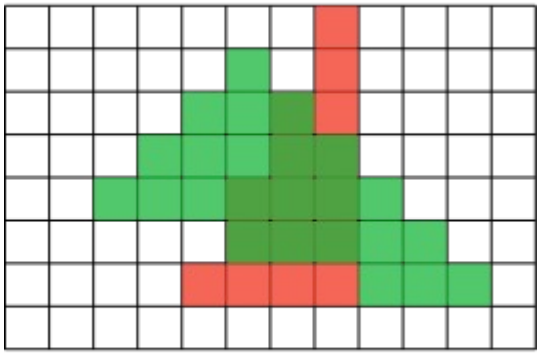
Transparencias Múltiples

Color Buffer



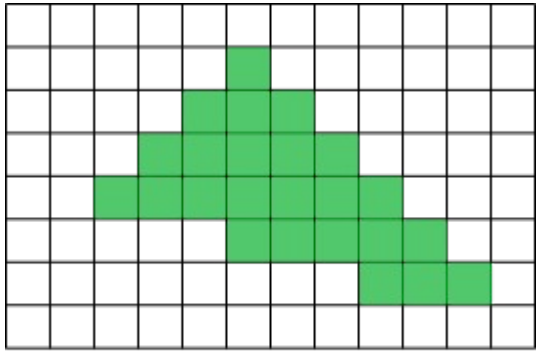
3/4 R + 1/4 B

Color Buffer

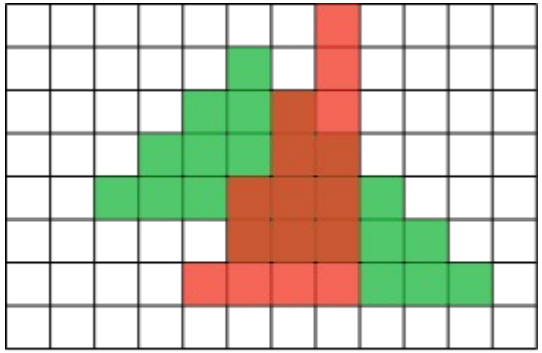


3/4 V + 1/4 (3/4 R + 1/4 B)

3/4 V + 3/16 R + 1/16 B



3/4 V+ 1/4 B



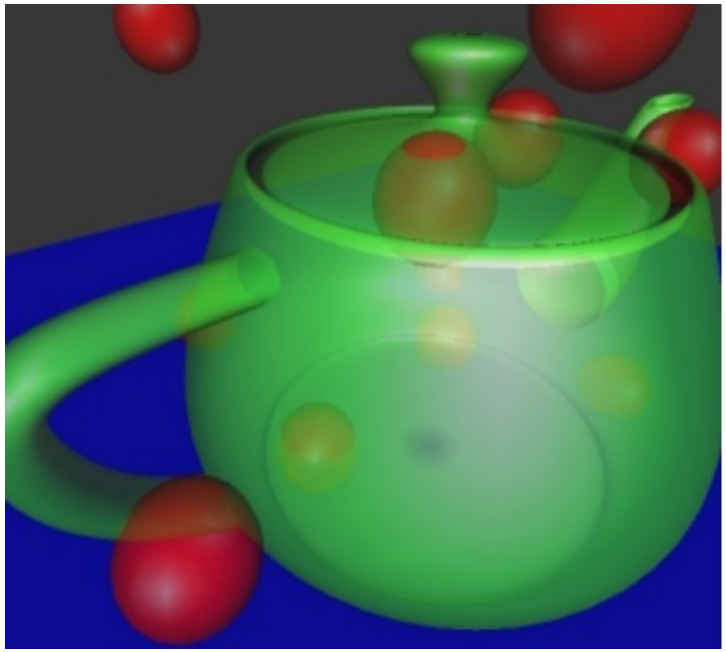
3/4 R + 1/4 (3/4 V + 1/4 B)

3/4 R + 1/16 V + 1/16 B

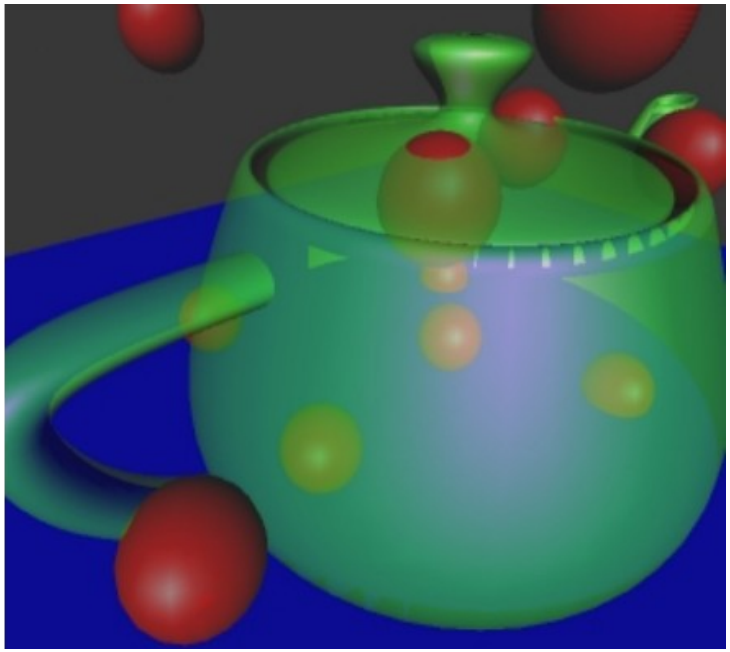
Depth Buffer: Transparencias Múltiples

Escenas más complejas → Técnicas especiales

Orden correcto



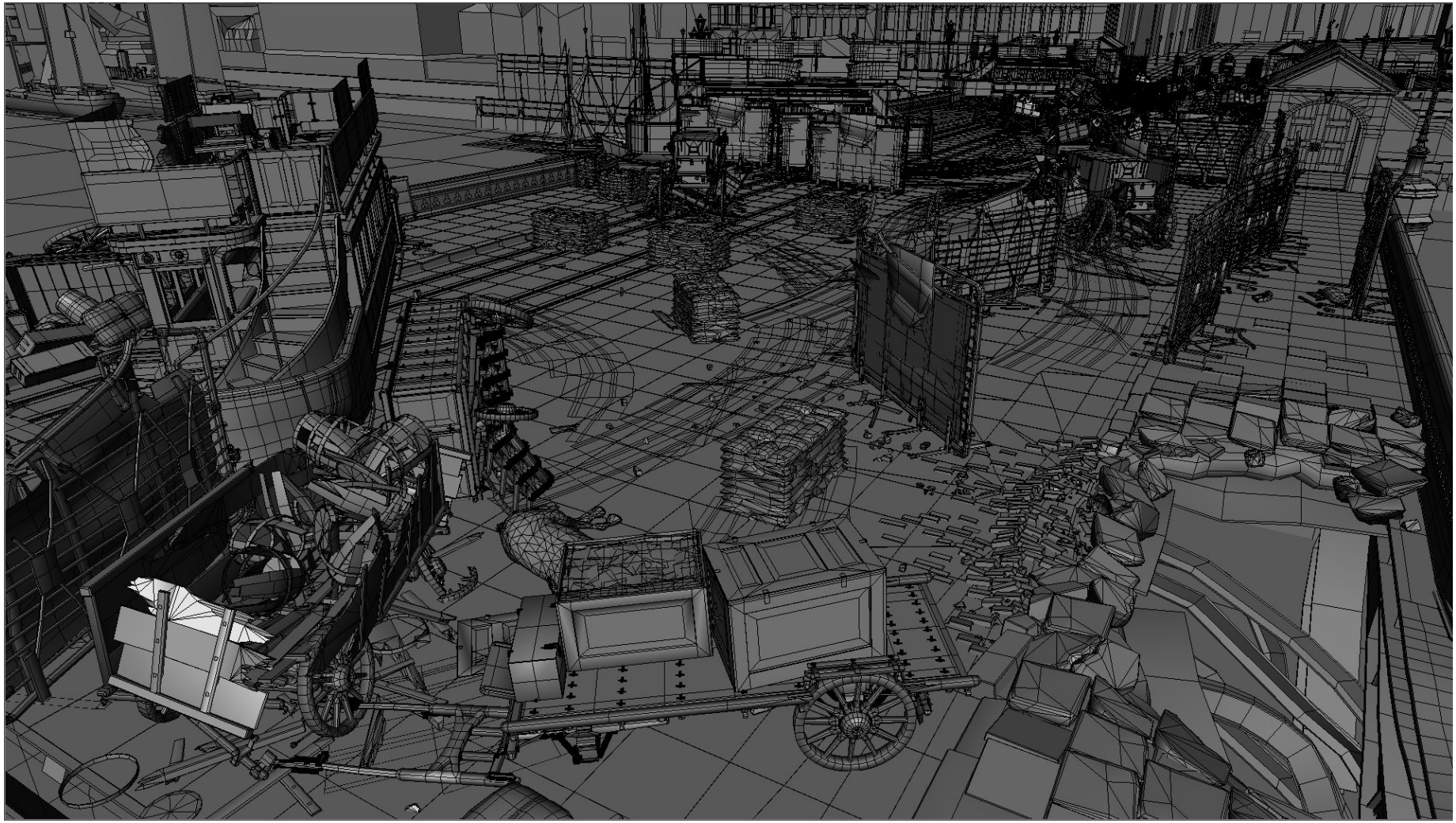
Orden incorrecto



Algunas:

- Depth peeling (doble z-buffer)
- Buffers con linked lists
- Order independent blending (más adelante)

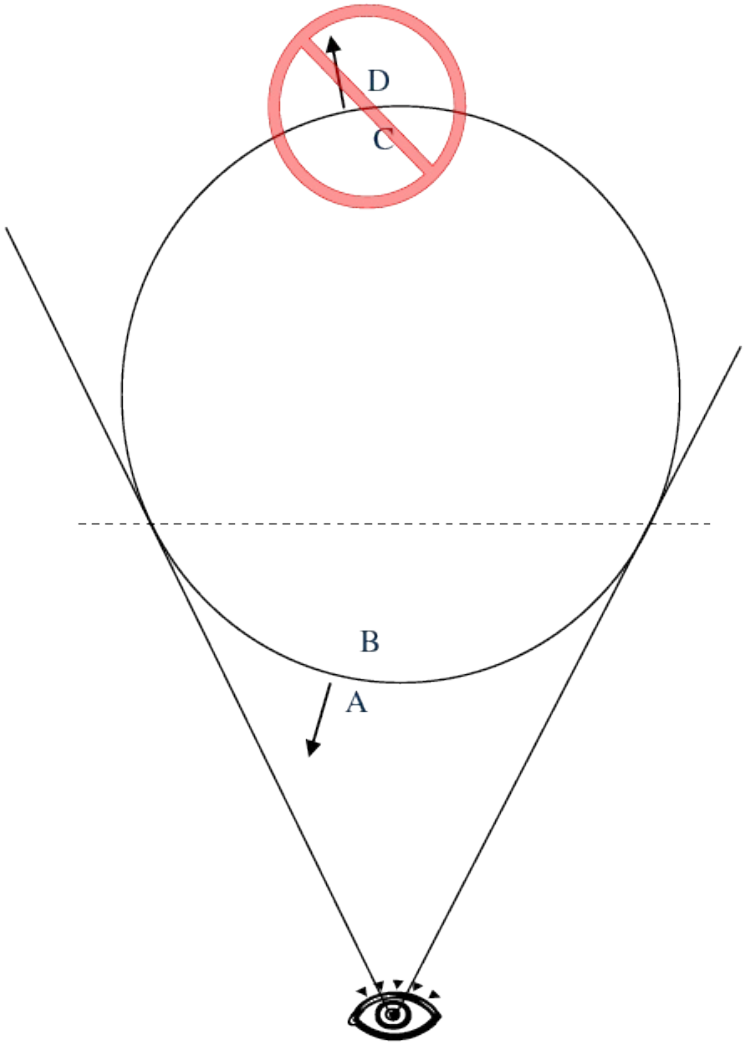
Depth Buffer: Escenas Complejas

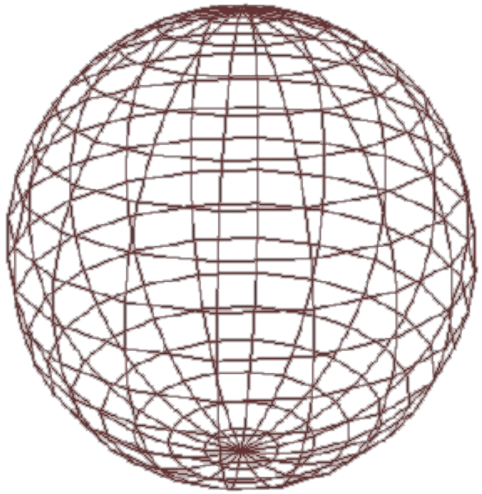


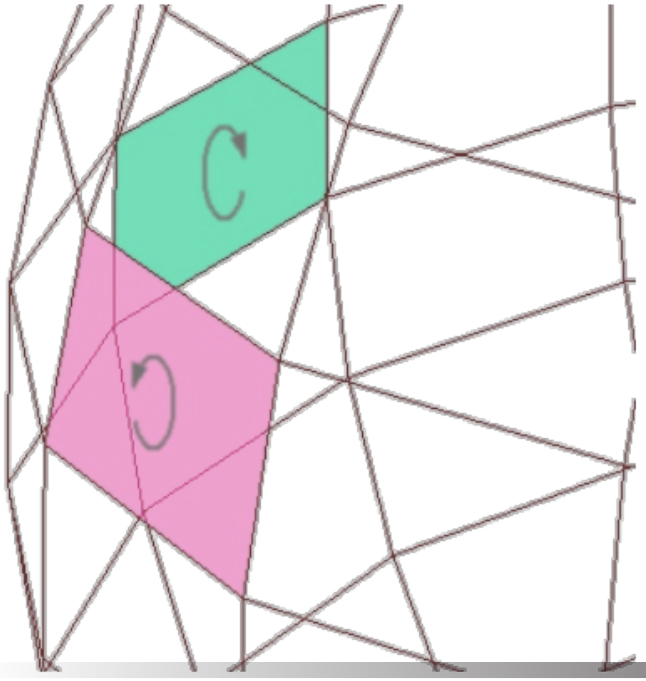
Las escenas complejas requieren rasterizar muchas primitivas

Backface Culling

glFrontFace(GL_CW or GL_CCW)
glCullFace(GL_FRONT or GL_BACK)

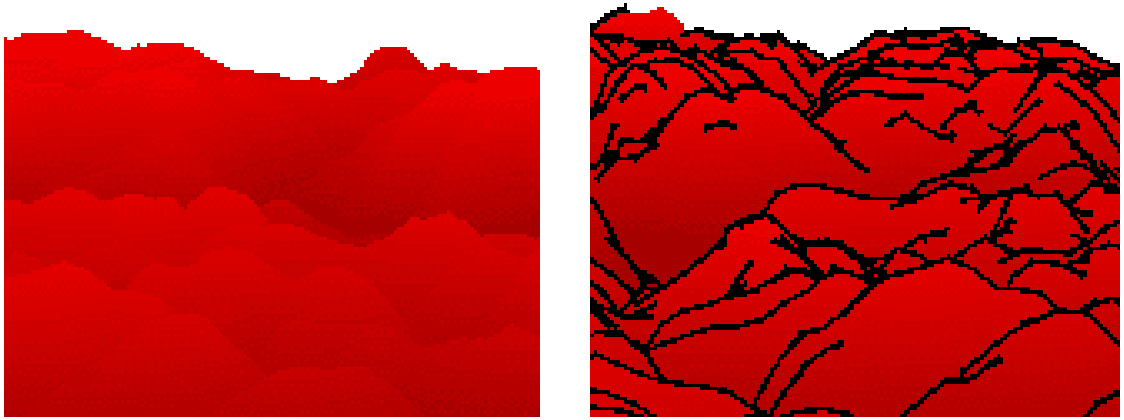


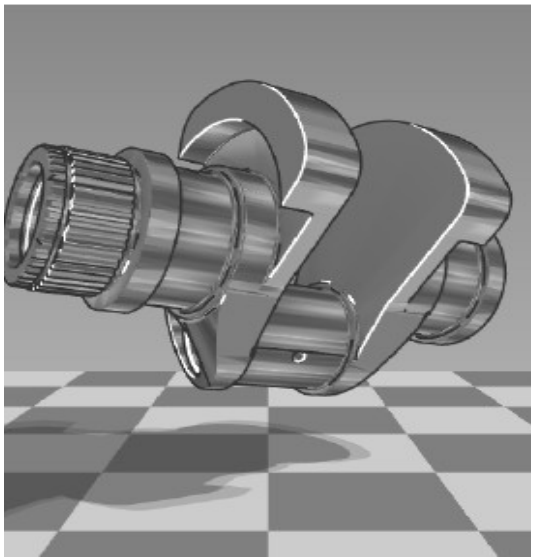




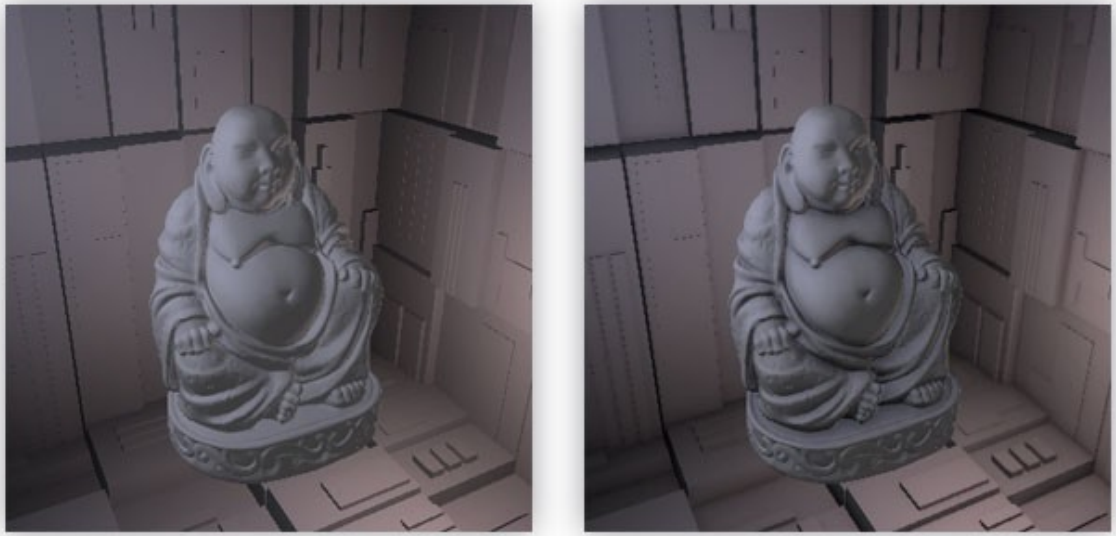
Depth Buffer: Otros Usos

Siluetas para mejorar la visualización





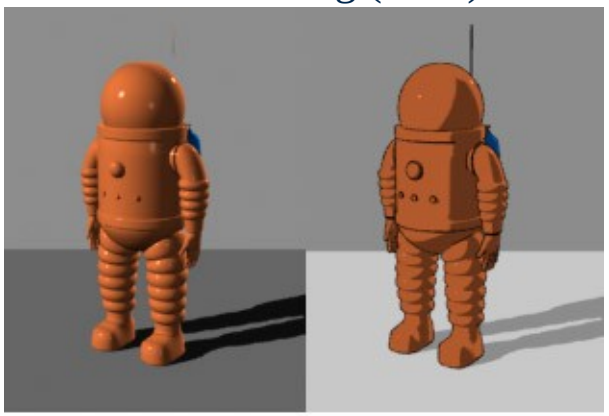
Screen Space Ambient Occlusion (SSAO)



without ambient occlusion

with ambient occlusion

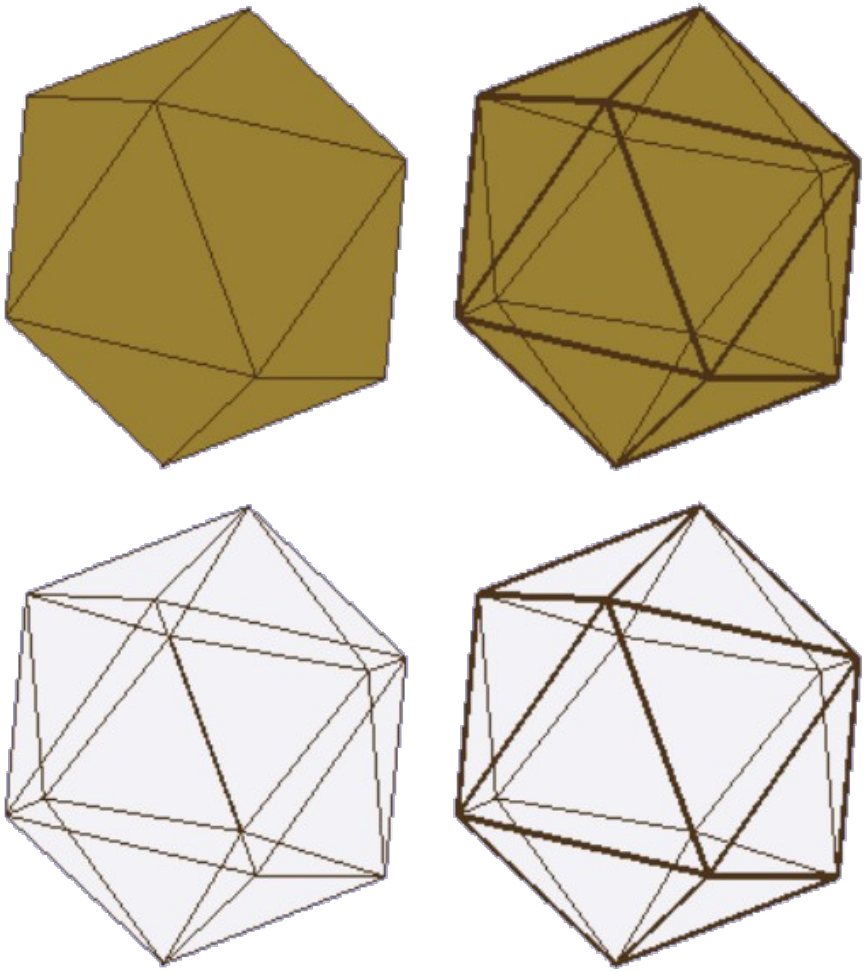
Toon-shading (NPR)



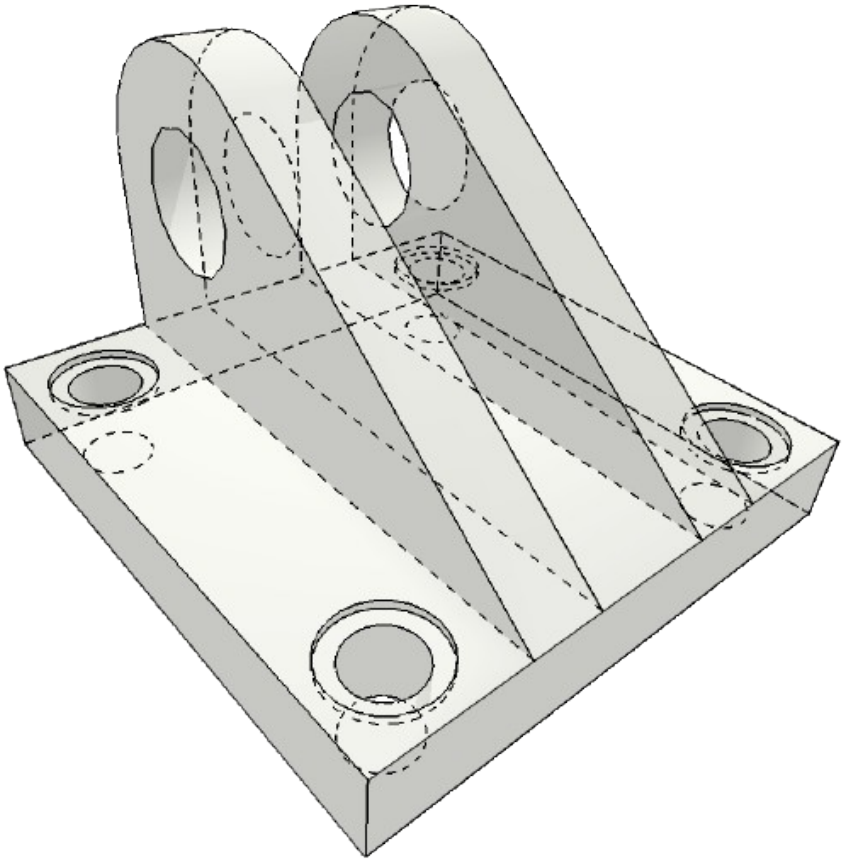
plastic shader toon shader

Depth Buffer: Otros Usos

Lineas ocultas

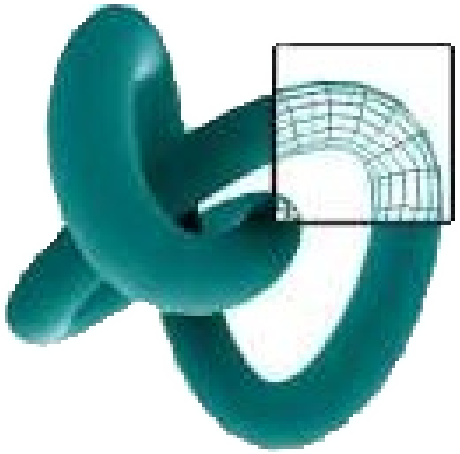


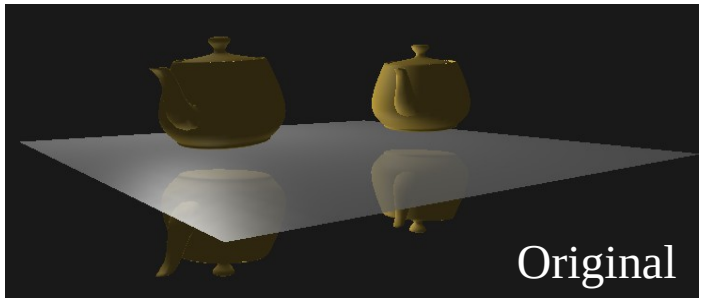
glColorMask(R,G,B,A)
glDepthFunc(comparador)

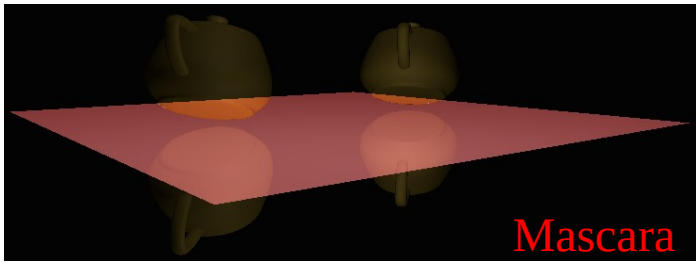


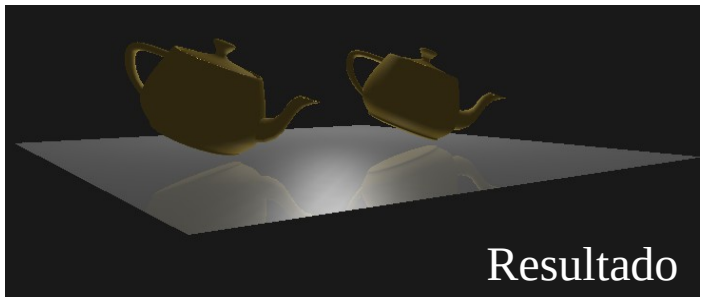
Stencil Buffer: Aplicaciones

- Reflejos y Espejos
- Sombras
- Outlining y Siluetas
- Visualización
- Refracción del agua
- Aplicar efectos a ciertos pixels
- Optimización
- Y más

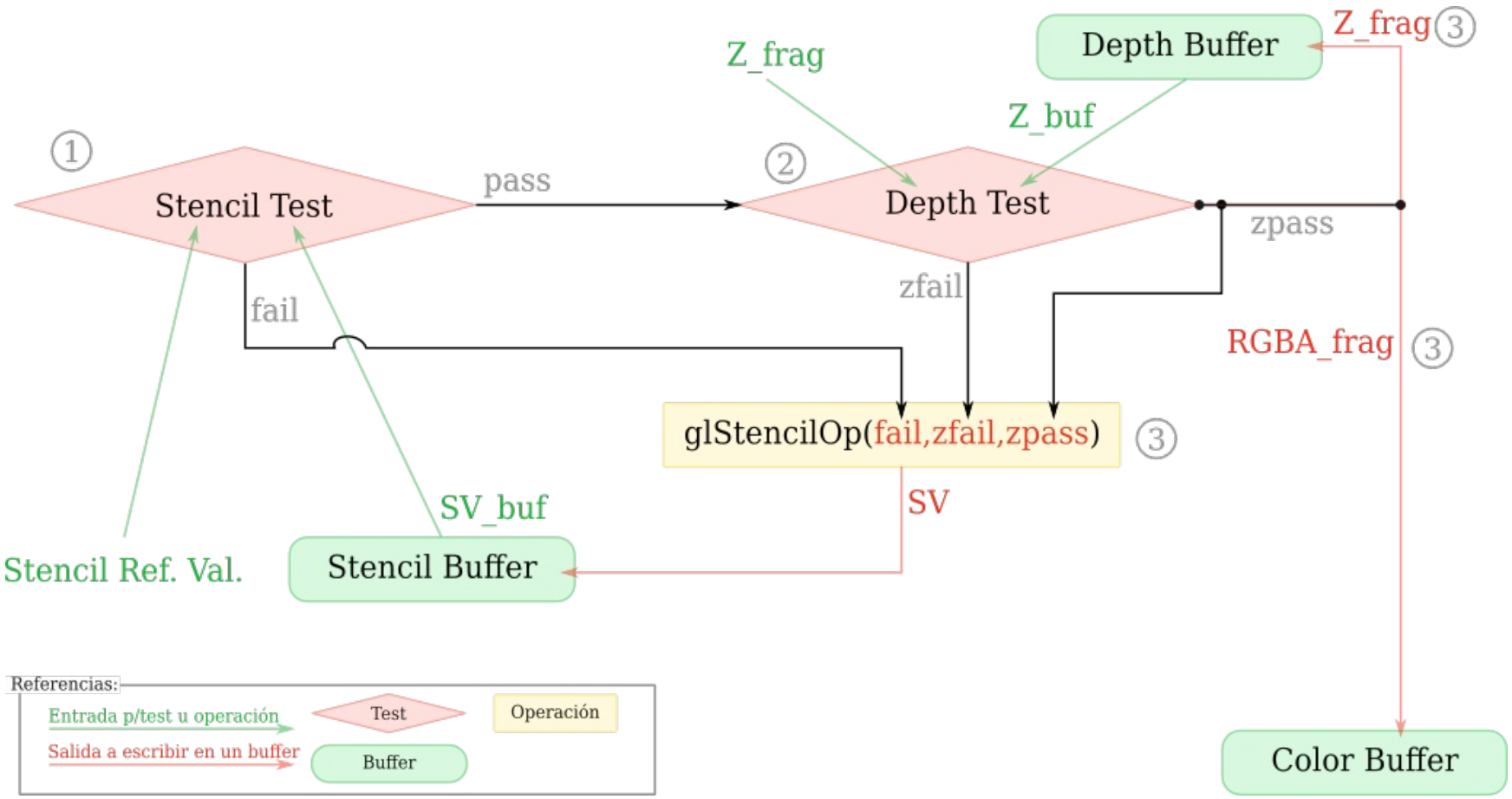




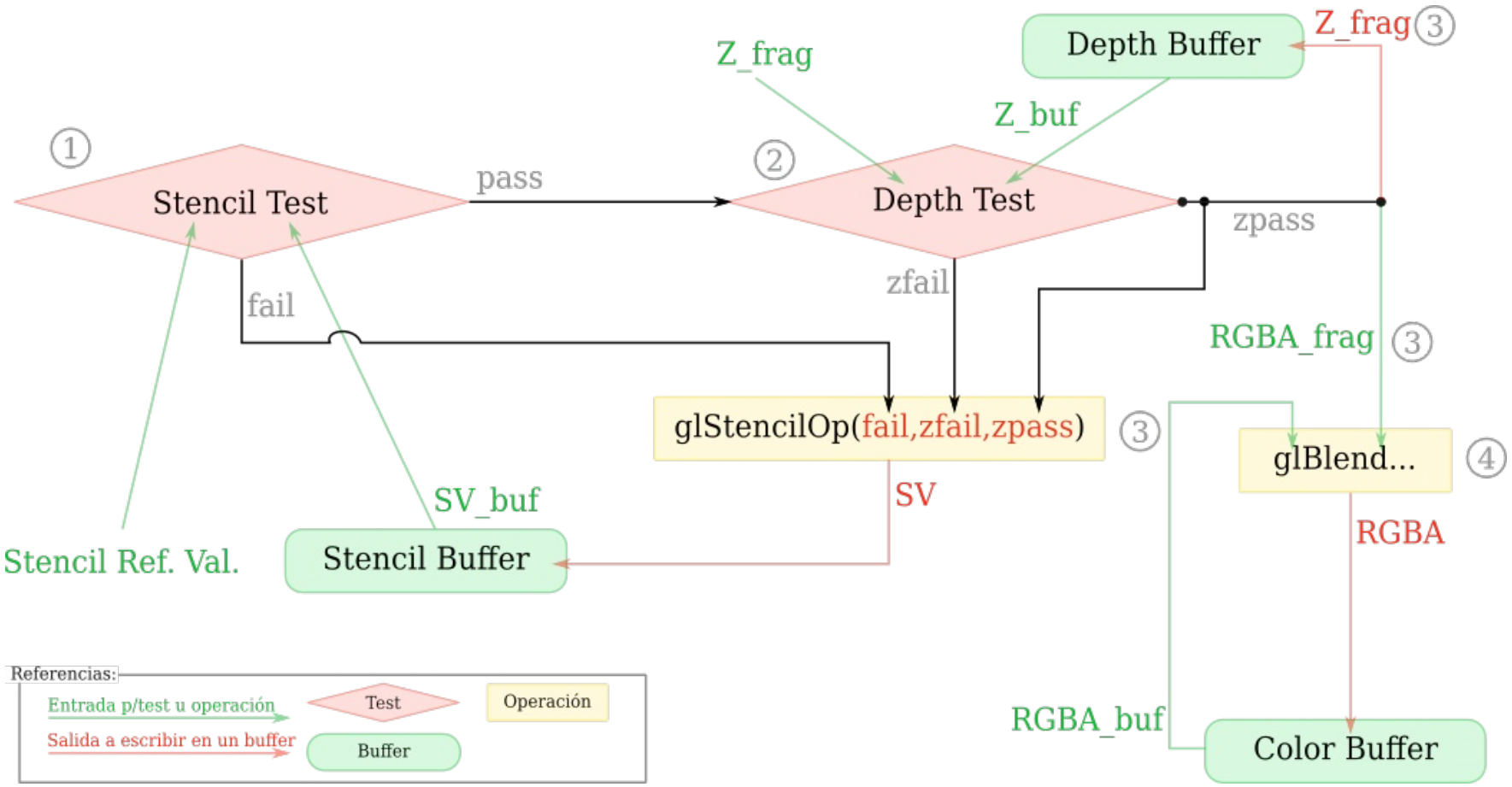




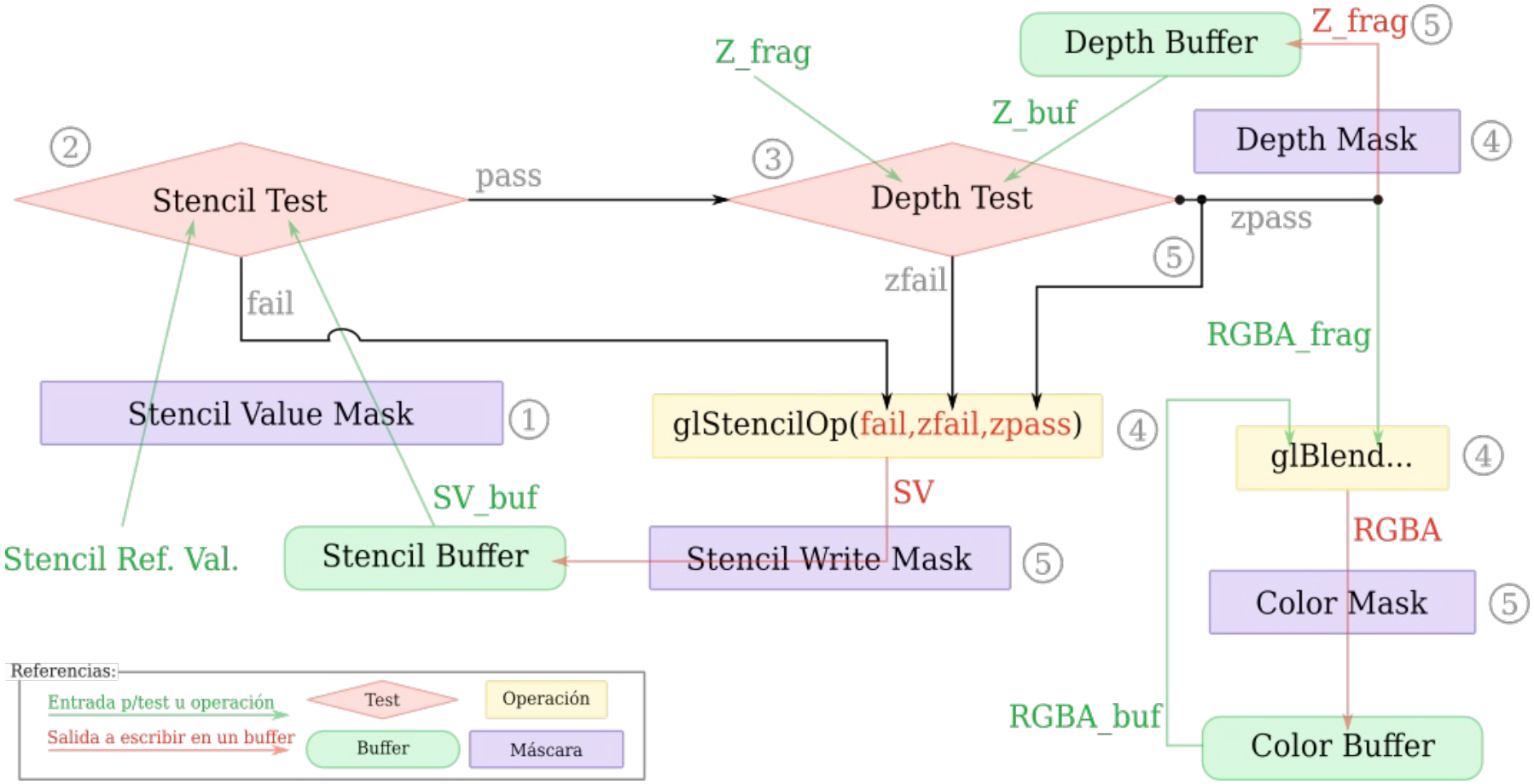
Stencil + Depth + Color



Stencil + Depth + Color

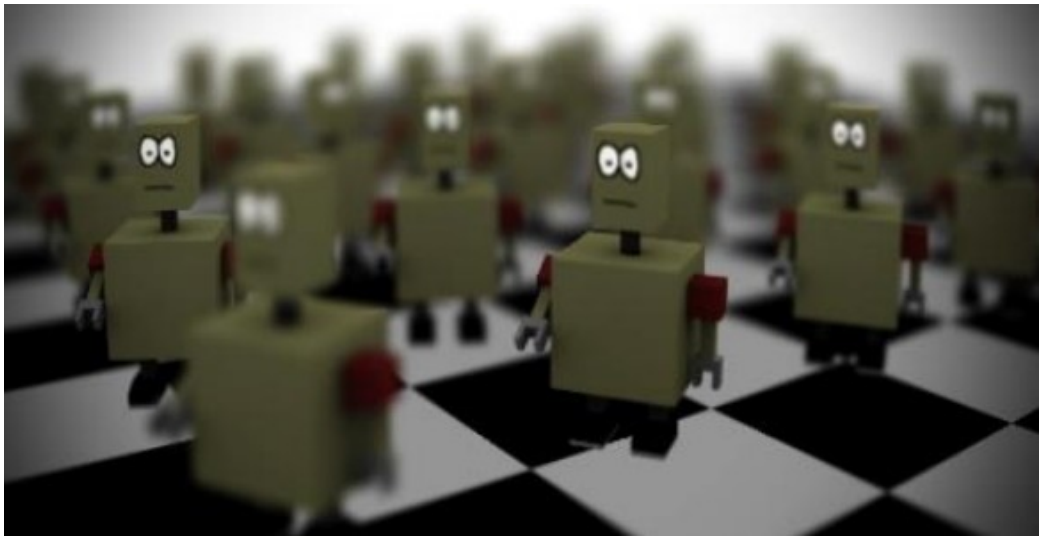


Stencil + Depth + Color

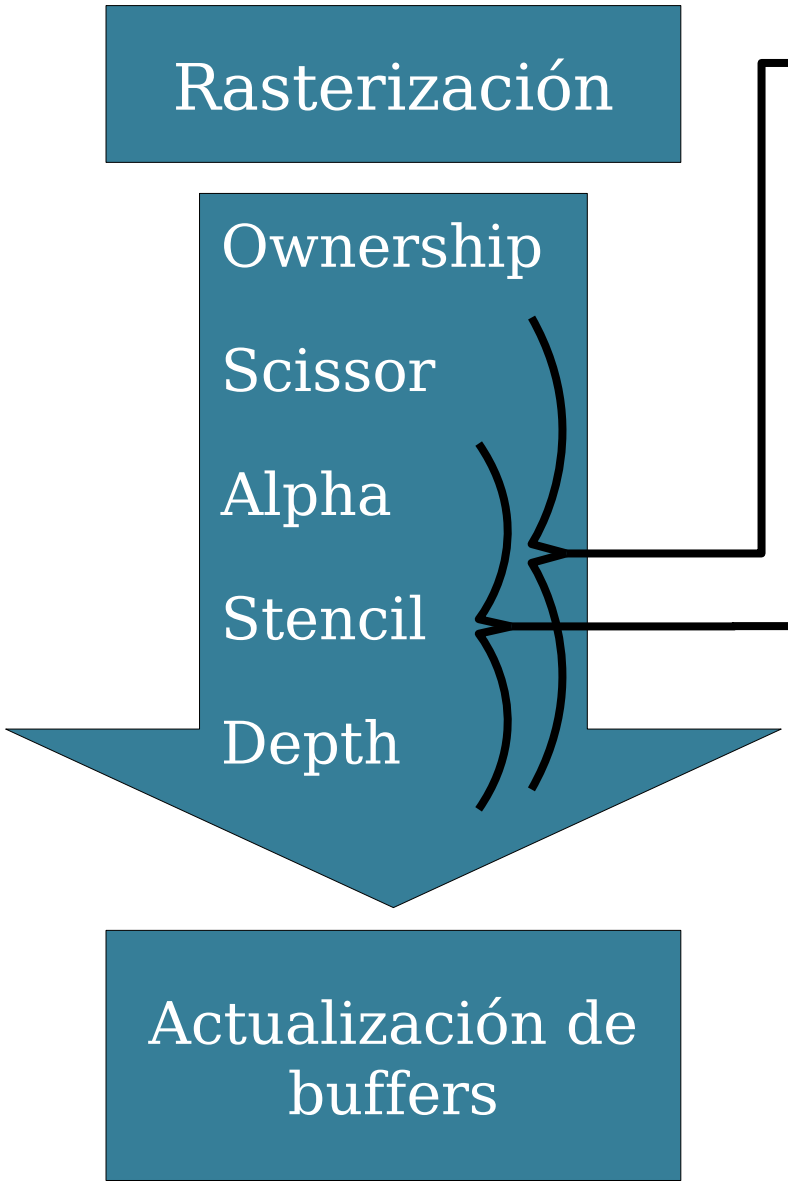


Accumulation Buffer

- Buffer de color “especial”
- 16 bits por color (48 bits)
- Reemplazado por FBOs
- Aplicaciones
  - Motion blur
  - Depth of field
  - Antialising
  - Soft shadows



Tests de Fragmentos



glEnable(GL_xxxxxx_TEST)

glDisable(GL_xxxxxx_TEST)

glXxxxxFunc(*comp*, [*ref*], [*mask*])

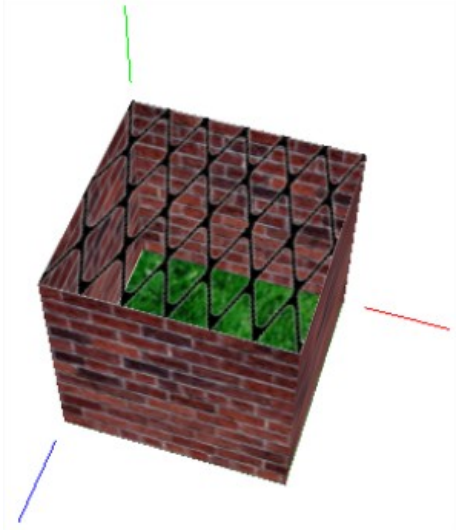
comp: posibles comparadores

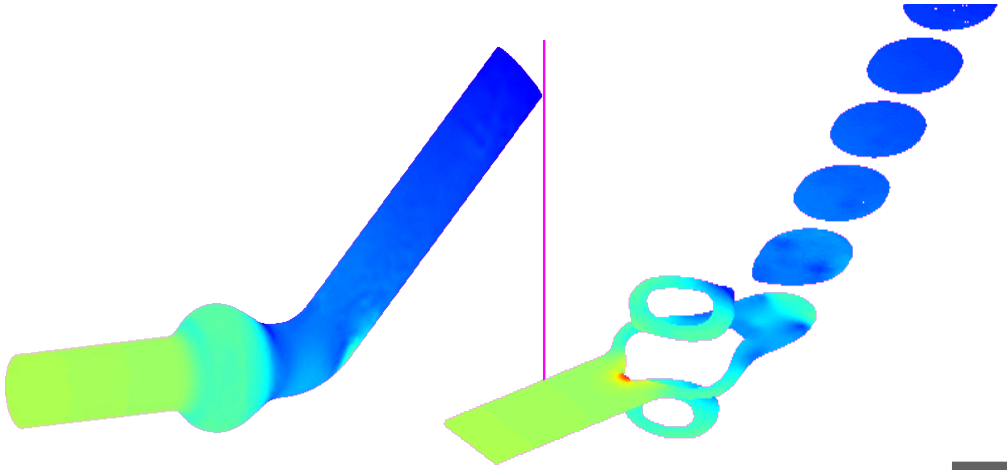
- GL_NEVER
- GL_ALWAYS
- GL_LESS
- GL_LEQUAL
- GL_EQUAL
- GL_GEQUAL
- GL_GREATER
- GL_NOTEQUAL

ref: valor con el que se compara (alpha y stencil)

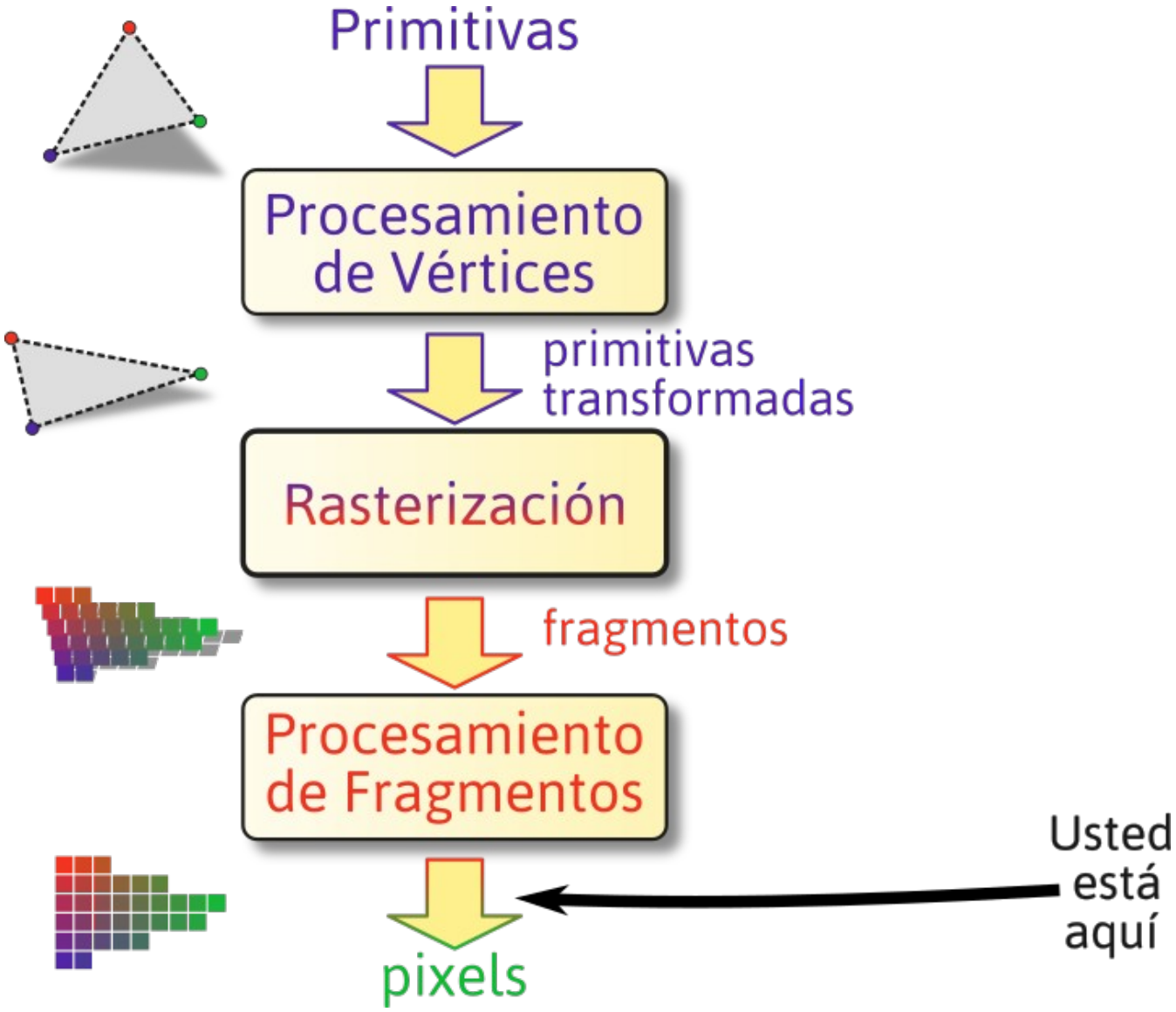
mask: para enmascarar bits en la comparación (solo stencil)

Tests de Fragmentos: Alpha-Test





Usted Está Aquí



Alteración de Fragmentos: Blending

- `glBlendEquation(op)`
- `glBlendFunc(`F_s`, `F_d`)`

$$D = F_s \; S \; op \; F_d \; D$$

D: color almacenado en el buffer

S: color del fragmento entrante

GL_FUNC_ADD	D = Fs S + Fd D
GL_FUNC_SUBTRACT	D = Fs S - Fd D
GL_FUNC_REVERSE_SUBTRACT	D = Fd D – Fs S
GL_MIN	D = min(Fs S, Fd D)
GL_MAX	D = max(Fs S, Fd D)

Alteración de Fragmentos: Blending

```
glBlendEquation(op)
glBlendFunc(Fs, Fd)
```

$D = F_s \; S \; op \; F_d \; D$

D: color almacenado en el buffer

S: color del fragmento entrante

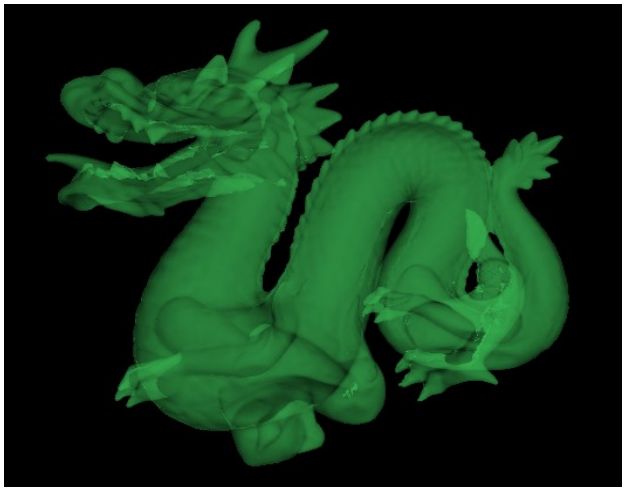
GL_ZERO	(0, 0, 0, 0)
GL_ONE	(1, 1, 1, 1)
GL_DST_COLOR	(R_d, G_d, B_d, A_d)
GL_SRC_COLOR	(R_s, G_s, B_s, A_s)
GL_ONE_MINUS_DST_COLOR	$(1, 1, 1, 1)-(R_d, G_d, B_d, A_d)$
GL_ONE_MINUS_SRC_COLOR	$(1, 1, 1, 1)-(R_s, G_s, B_s, A_s)$
GL_SRC_ALPHA	(A_s, A_s, A_s, A_s)
GL_ONE_MINUS_SRC_ALPHA	$(1, 1, 1, 1)-(A_s, A_s, A_s, A_s)$
GL_DST_ALPHA	(A_d, A_d, A_d, A_d)
GL_ONE_MINUS_DST_ALPHA	$(1, 1, 1, 1)-(A_d, A_d, A_d, A_d)$
GL_SRC_ALPHA_SATURATE	$(f, f, f, 1)$; $f=\min(A_s, 1-A_d)$

Alteración de Fragmentos: Blending

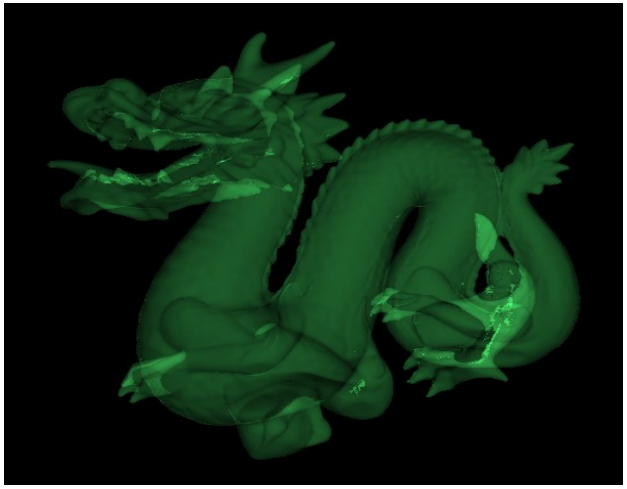
$$D = A_S S + (1 - A_S) D$$

```
glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA)

glBlendEquation(GL_FUNC_ADD)
```

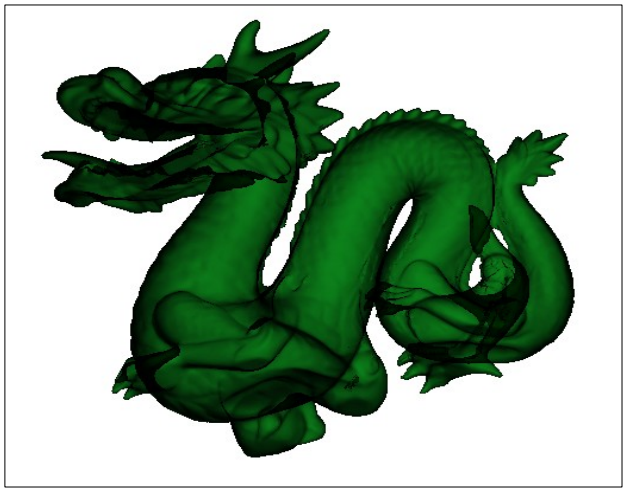


Independent Order Blending



$$D = S + D$$

```
GlBlendFunc(GL_ONE,GL_ONE)
glBlendEquation(GL_FUNC_ADD)
```



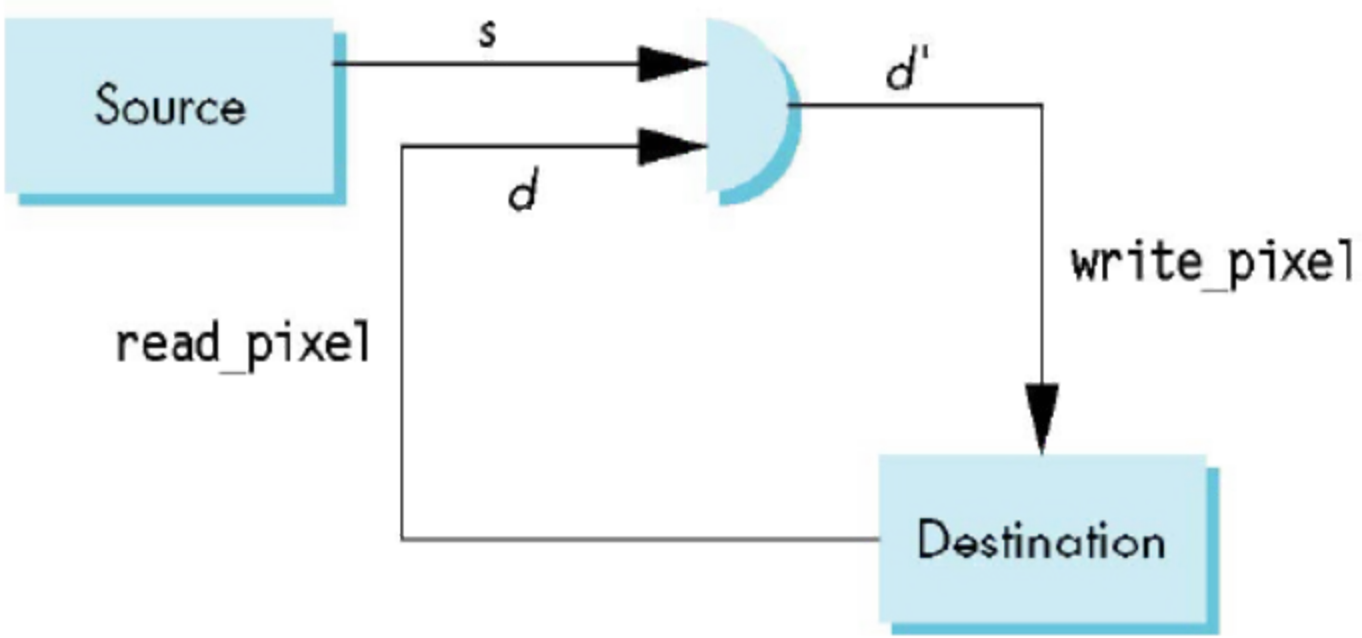
$$D = S \cdot D$$

```
GlBlendFunc(GL_ZERO,GL_SRC_COLOR)
```

Alteración de Fragmentos: Operaciones Lógicas

glLogicOp(op) ➡ D = S op D

D: color almacenado en el buffer
S: color del fragmento entrante



Alteración de Fragmentos: Operaciones Lógicas

glLogicOp(op) ➡ D = S op D

D: color almacenado en el buffer

S: color del fragmento entrante

Parameter	Operation	Parameter	Operation
GL_CLEAR	0	GL_AND	s&d
GL_COPY	**s**	GL_OR	s|d
GL_NOOP	d	GL_NAND	~(s&d)
GL_SET	1	GL_NOR	~(s|d)
GL_COPY_INVERTED	~s	**GL_XOR**	**s^d**
GL_INVERT	**~d**	GL_EQUIV	~(s^d)
GL_AND_REVERSE	s&~d	GL_AND_INVERTED	~s&d
GL_OR_REVERSE	s|~d	GL_OR_INVERTED	~s|d

Fin

Deben saber

Image precision vs. Model precision
Depth buffer: ventajas, problemas y usos
Stencil buffer: funcionamiento, usos
Tests: usos, técnicas multipaso
Blending y operaciones lógicas

Bibliografía

Red-Book:

- 5: A Hidden-Surface Removal Survival Kit
- 6: Blending, Polygon Offset
- 8: Lo que quieran saber sobre imágenes
- 10: Framebuffer (todo)
- 14: Usos alternativos

El material extra es para curiosear, ver otras fuentes o una punta para aprender más, no es necesario estudiarlo.