Jinwoo Choi

- Keimyung University
  03/2017 – 08/2019
- Digipen Institute of Technology
  08/2021 – 04/2023

- Undergraduate Teaching Assistant
  09/2021 – 12/2021
- Tiktok, AR Effect Software Engineer Intern
  05/2022 – 08/2022
- Tiktok, Software Engineer(XR Engine and Runtime) Fulltime
  05/2023 –

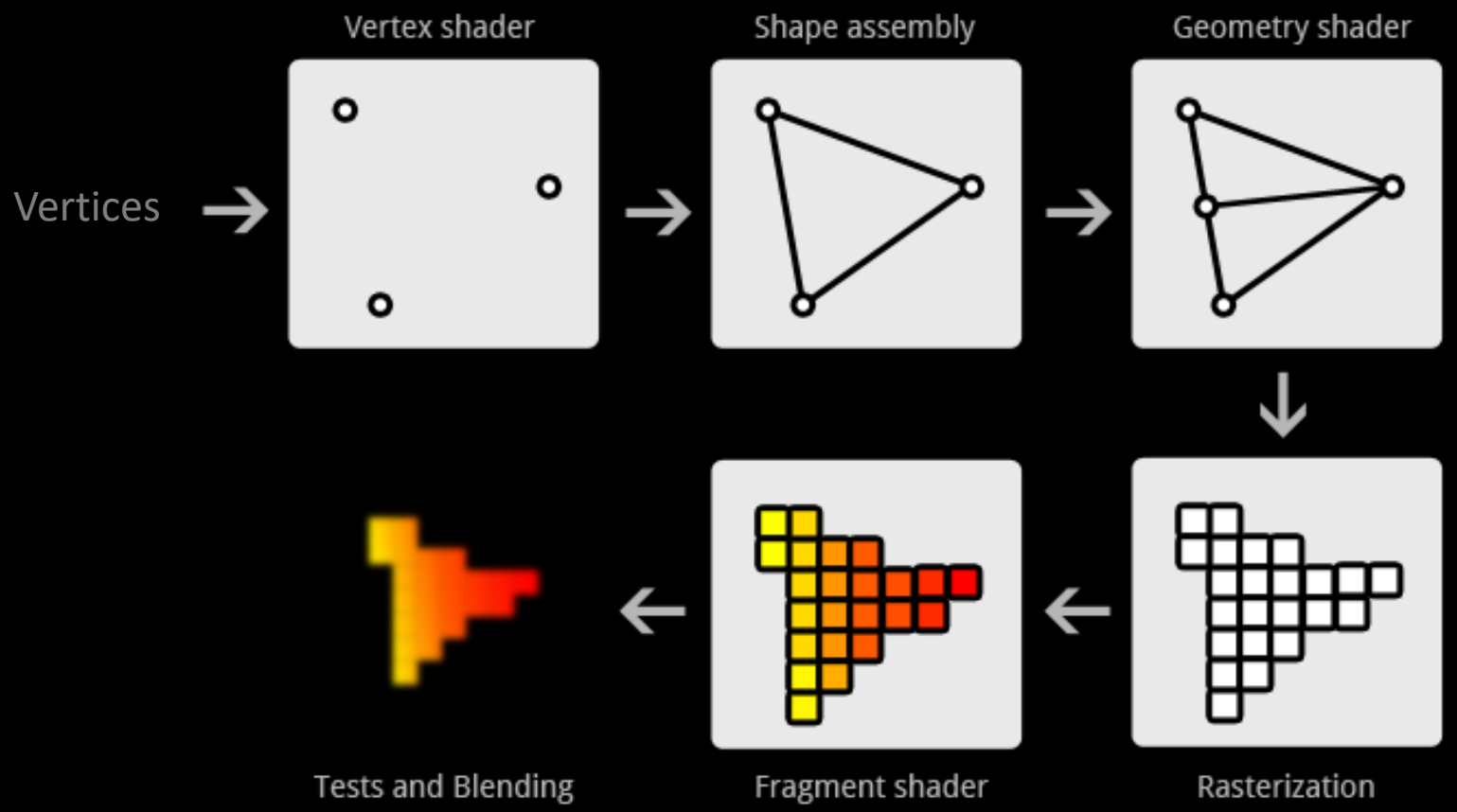
Email

imjinwoo98@gmail.com

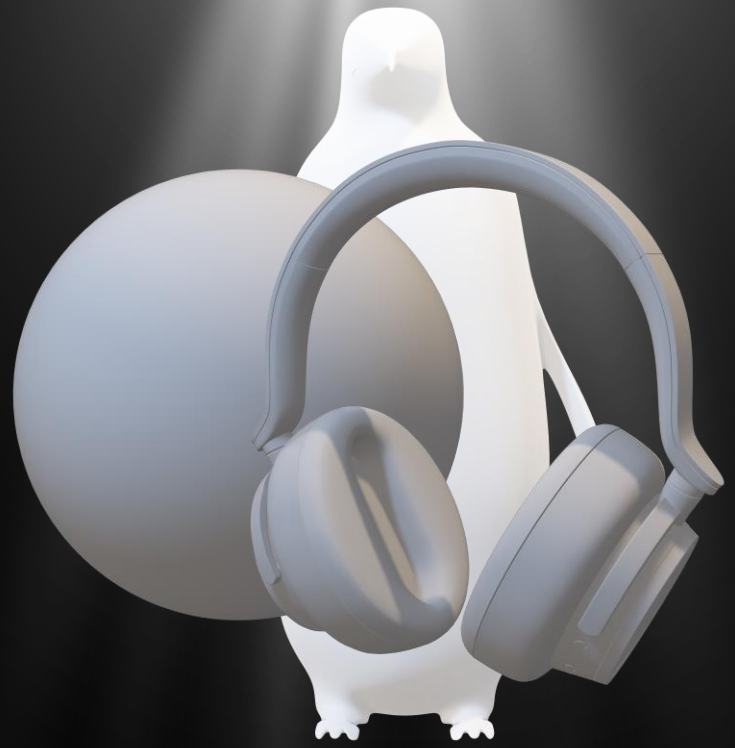
Deferred Shading

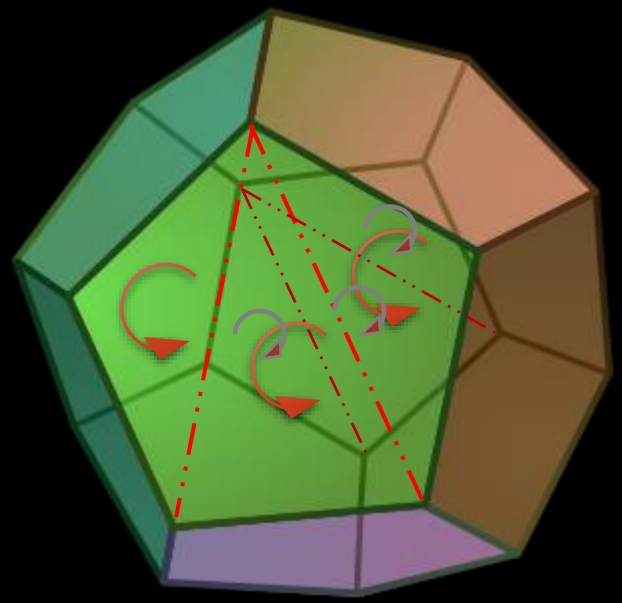
Jinwoo Choi

Rendering Pipeline



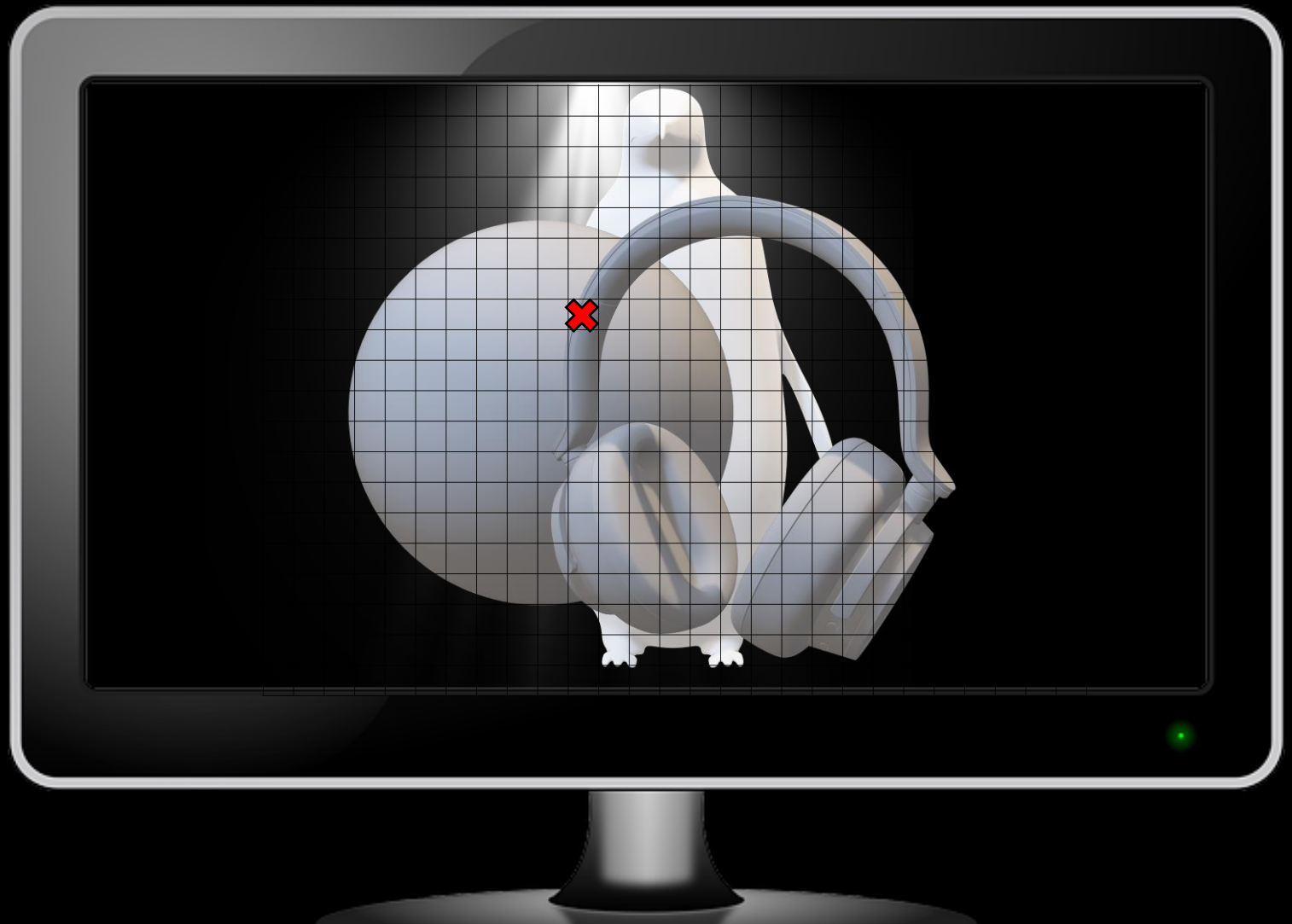
Lighting



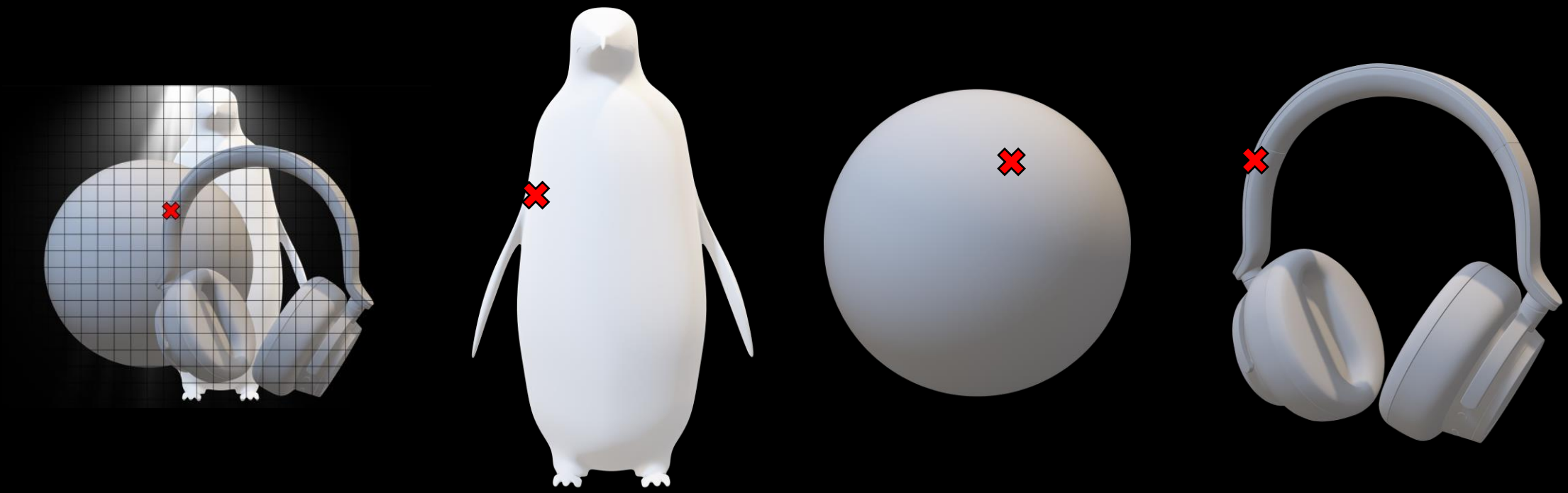


```
glEnable (GL_CULL_FACE);
glCullFace (GL_BACK);
```

Forward Rendering



Forward Rendering



Deferred Shading

Texture2D

Deferred Shading

Deferred Shading

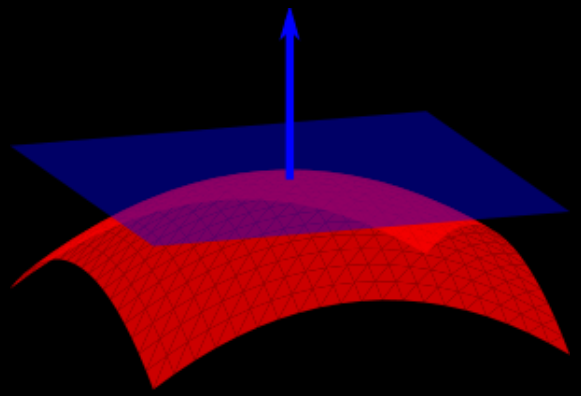
Deferred Shading

Deferred Shading

Deferred Shading

$$I_p = k_a i_a + \sum_{m}^{lights} (k_d i_{m,d} (N \cdot L) + k_s i_{m,s} (R \cdot V)^{\alpha})$$

Normal Vector



Light Vector

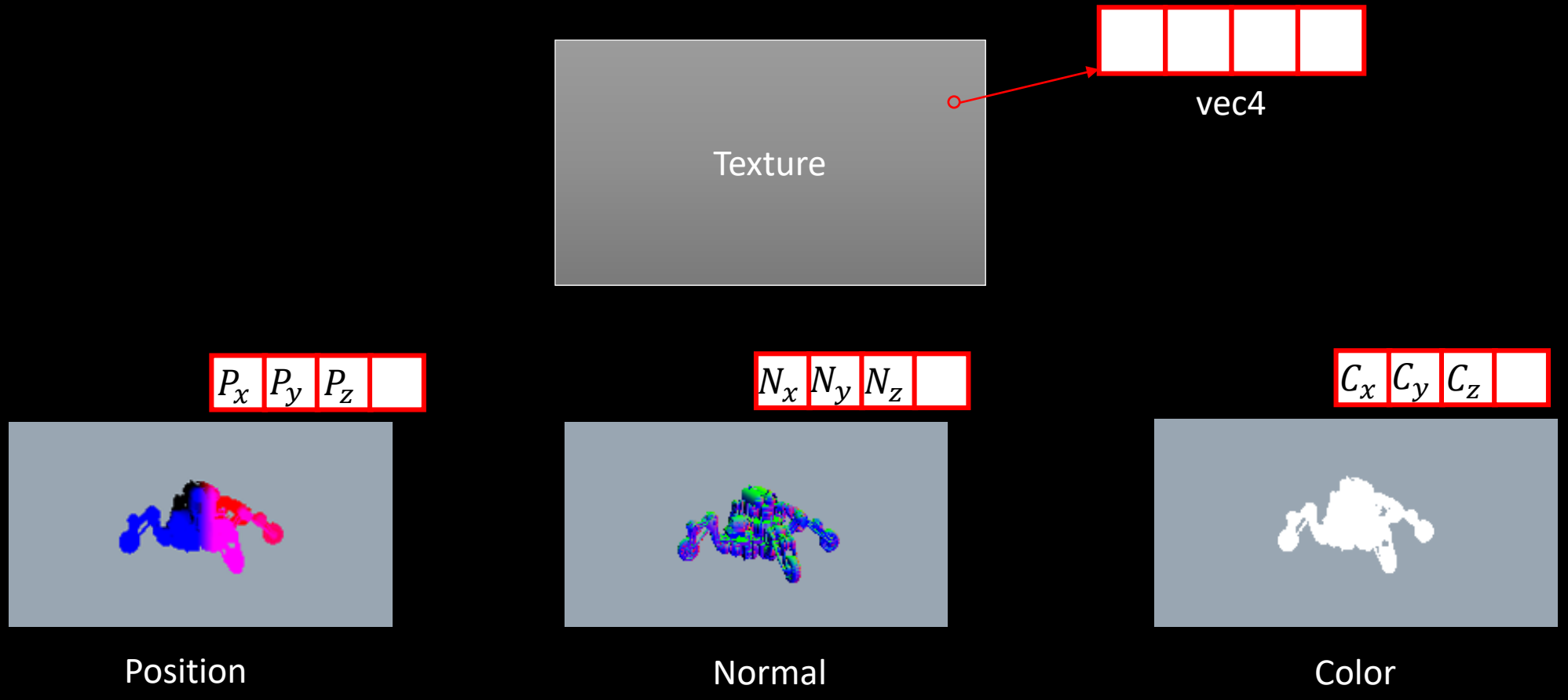
$\mathrm{L} = P_L - P$

P_L: Position of Light

P : **Position** of Vertex

Material

Texture



Texture

https://github.com/fendevel/Guide-to-Modern-OpenGL-Functions/blob/master/README.md

```
void  glGenTextures (GLsizei n, GLuint* textures);
```

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glGenTextures	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

```
glGenTextures (1, &texture);
glBindTexture (GL_TEXTURE_2D, texture);
```

```
void  glCreateTextures (GLenum target, GLsizei n, GLuint* textures);
```

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glCreateTextures	-	-	-	-	-	-	-	-	-	-	-	✓

```
glCreateTextures (GL_TEXTURE_2D, 1, &texture);
```

Texture

https://registry.khronos.org/OpenGL-Refpages/gl4/html/glTexParameter.xhtml

void glTexParameteri (GLenum target, GLenum pname, GLint param);

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glTexParameteri	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

```
glTexParameteri (GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_CLAMP_TO_EDGE);
glTexParameteri (GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_CLAMP_TO_EDGE);
glTexParameteri (GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
glTexParameteri (GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
```

void glTextureParameteri (GLuint texture, GLenum pname, GLint param);

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glTextureParameteri	-	-	-	-	-	-	-	-	-	-	-	✓

```
glTextureParameteri (texture, GL_TEXTURE_WRAP_S, GL_CLAMP_TO_EDGE);
glTextureParameteri (texture, GL_TEXTURE_WRAP_T, GL_CLAMP_TO_EDGE);
glTextureParameteri (texture, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
glTextureParameteri (texture, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
```

Texture

https://registry.khronos.org/OpenGL-Refpages/gl4/html/glTexImage2D.xhtml

void glTexImage2D (GLenum target, … GLsizei height, GLsizei width, … GLenum type, const void* data);

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glTexImage2D	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

```
glTexImage2D (GL_TEXTURE_2D, 0, GL_RGBA16F, width, height, 0, GL_RGBA, GL_FLOAT, nullptr);
```

void glTextureStorage2D (GLuint texture, GLsizei levels, … GLsizei width, GLsizei height);

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glTexStorage2D	-	-	-	-	-	-	-	-	✓	✓	✓	✓
glTextureStorage2D	-	-	-	-	-	-	-	-	-	-	-	✓

```
glTextureStorage2D (texture, 0, GL_ RGBA16F , width, height);
glTextureSubImage2D (texture, 0, 0, 0, width, height, GL_ RGBA, GL_FLOAT, nullptr);
```

Texture

void glActiveTexture (GLenum texture);

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glActiveTexture	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

```
glActiveTexture (GL_TEXTURE0 + 3);
glBindTexture (GL_TEXTURE_2D, texture);
```

void glBindTextureUnit (GLuint unit, GLuint texture);

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glBindTextureUnit	-	-	-	-	-	-	-	-	-	-	-	✓

```
glBindTextureUnit (3, texture);
```

Texture

OpenGL 2.0+

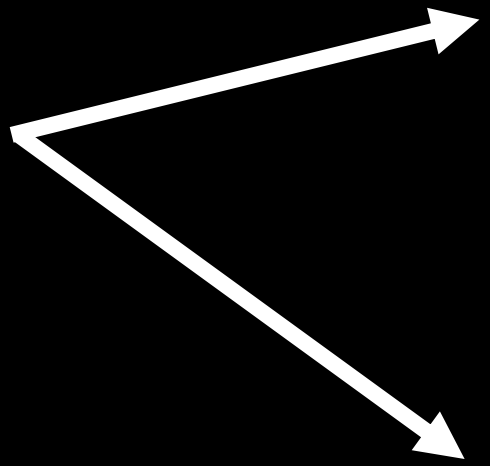
```
glGenTextures (1, &texture);                                                    n = 0, 1, 2 …
glBindTexture (GL_TEXTURE_2D, texture);
glTexParameteri (GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_CLAMP_TO_EDGE);
glTexParameteri (GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_CLAMP_TO_EDGE);
glTexParameteri (GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
glTexParameteri (GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
glTexImage2D (GL_TEXTURE_2D, 0, GL_RGBA16F, width, height, 0, GL_RGBA, GL_FLOAT, nullptr);
glActiveTexture (GL_TEXTURE0 + n);
glBindTexture (GL_TEXTURE_2D, texture);
```

OpenGL 4.5+

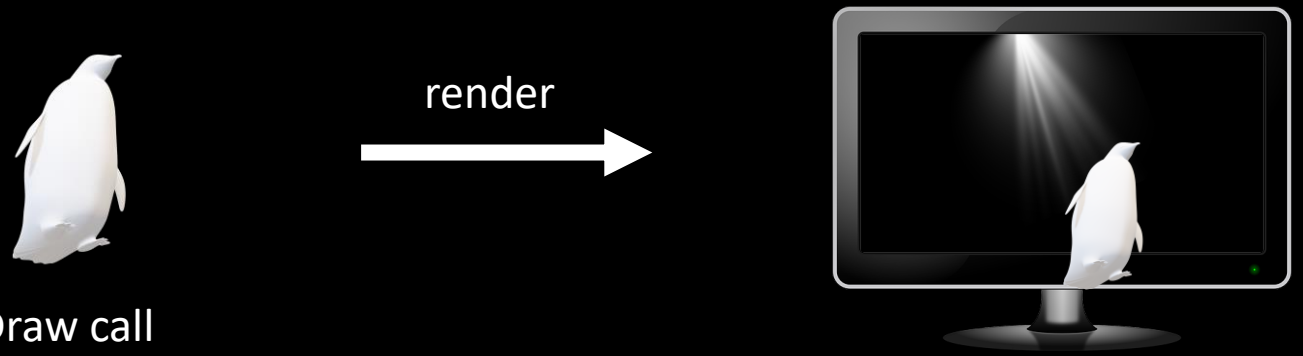
```
glCreateTextures (GL_TEXTURE_2D, 1, &texture);                                  n = 0, 1, 2 …
glTextureParameteri (texture, GL_TEXTURE_WRAP_S, GL_CLAMP_TO_EDGE);
glTextureParameteri (texture, GL_TEXTURE_WRAP_T, GL_CLAMP_TO_EDGE);
glTextureParameteri (texture, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
glTextureParameteri (texture, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
glTextureStorage2D (texture, 0, GL_ RGBA16F , width, height);
glTextureSubImage2D (texture, 0, 0, 0, width, height, GL_ RGBA, GL_FLOAT, nullptr);
glBindTextureUnit (n, texture);
```

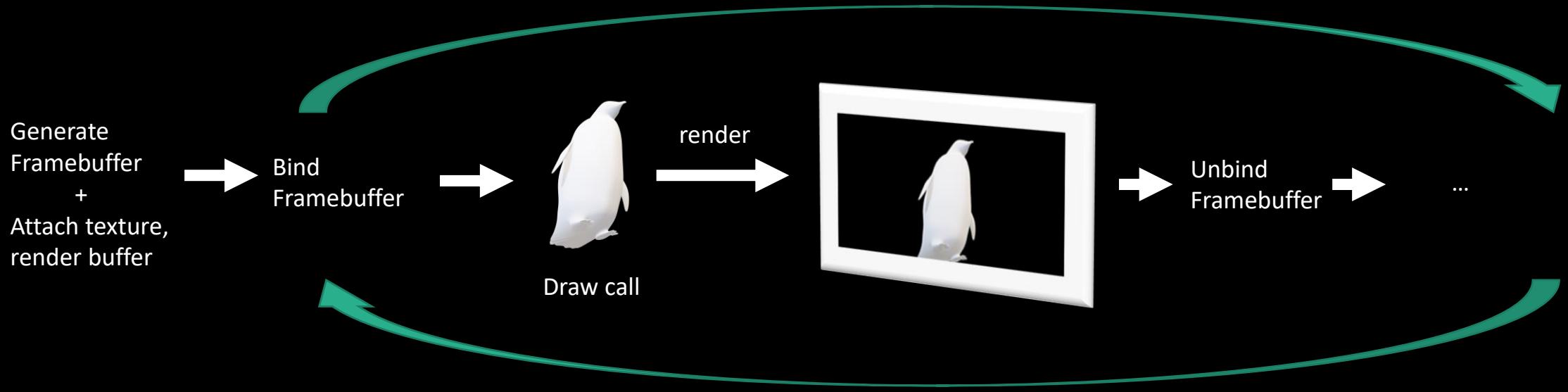
Frame Buffer(VRAM)

Memory that stores the color values for each pixel on the display



Frame Buffer





Frame Buffer

OpenGL 4.5+

```
class FBO // Frame Buffer Object
{
public:
	FBO();
	~FBO();
	void Initialize(unsigned width, unsigned height);
	void Bind();
	void Unbind();
private:
	unsigned GBuffer0, GBuffer1, GBuffer2; //texture, render target
	unsigned u_GBuffer0, u_GBuffer1, u_GBuffer2; // texture unit
	unsigned fbo_handle; //Frame Buffer Object Handle
	unsigned rbo_handle; //Render Buffer Object Handle
}
```

FBO-Create FBO

void glGenFramebuffers (GLsizei n, GLuint* ids);

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glGenFramebuffers	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

```
glGenFramebuffers (1, &fbo_handle);
glBindFramebuffer (GL_FRAMEBUFFER, fbo_handle);
```

void glCreateFramebuffers (GLsizei n, GLuint* framebuffers);

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glCreateFramebuffers	-	-	-	-	-	-	-	-	-	-	-	✓

```
glCreateFrameBuffers (1, &fbo_handle);
```

FBO-RBO

```
void  glGenRenderbuffers (GLsizei n, GLuint* ids);
```

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glGenRenderbuffers	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

```
glGenRenderbuffers (1, &rbo_handle);
glBindRenderbuffer (GL_RENDERBUFFER, rbo_handle); // replace rbo_handle with 0 means unbind
```

```
void  glCreateRenderbuffers (GLsizei n, GLuint* renderbuffers);
```

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glCreateRenderbuffers	-	-	-	-	-	-	-	-	-	-	-	✓

```
glCreateRenderbuffers (1, &rbo_handle);
```

FBO-Attach RBO

void glRenderbufferStorage (GLenum target, GLenum internalformat, GLsizei width, GLsizei height);
void glNamedRenderbufferStorage (GLenum renderbuffer, GLenum internalformat, GLsizei width, GLsizei height);

	OpenGL Version											
Function / Feature Name	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glNamedRenderbufferStorage	-	-	-	-	-	-	-	-	-	-	-	✓
glRenderbufferStorage	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

```
glRenderbufferStorage (GL_RENDERBUFFER, GL_DEPTH_COMPONENT, width, height);
glNamedRenderbufferStorage (rbo_handle, GL_DEPTH_COMPONENT, width, height);
```

void glFramebufferRenderbuffer (GLenum target, GLenum internalformat, GLenum renderbuffertarget, GLuint renderbuffer);
void glNamedFramebufferRenderbuffer (GLuint framebuffer, GLenum internalformat, GLenum renderbuffertarget, GLuint renderbuffer);

	OpenGL Version											
Function / Feature Name	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glFramebufferRenderbuffer	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
glNamedFramebufferRenderbuffer	-	-	-	-	-	-	-	-	-	-	-	✓

```
glFramebufferRenderbuffer (GL_FRAMEBUFFER, GL_DEPTH_ATTACHMENT, GL_RENDERBUFFER, rbo_handle);
glNamedFramebufferRenderbuffer (fbo_handle, GL_DEPTH_ ATTACHMENT, GL_RENDERBUFFER , rbo_handle);
```

FBO-Attach Texture

```
void  glFramebufferTexture2D (GLenum target, GLenum attachment, GLenum textarget, GLuint texture, GLint level);
void  glNamedFramebufferTexture (GLuint framebuffer, GLenum attachment, GLuint texture, GLint level);
```

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glFramebufferTexture	-	-	-	-	✓	✓	✓	✓	✓	✓	✓	✓
glFramebufferTexture1D	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
glFramebufferTexture2D	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
glFramebufferTexture3D	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
glNamedFramebufferTexture	-	-	-	-	-	-	-	-	-	-	-	✓

```
glFramebufferTexture2D (GL_FRAMEBUFFER , GL_COLOR_ATTACHMENT0, GL_TEXTURE_2D, GBuffer1, 0);
glNamedFramebufferTexture (fbo_handle, GL_COLOR_ATTACHMENT0, GBuffer1, 0);
```

FBO-Specify a list

```
void  glDrawBuffers (GLsizei n, const GLenum* bufs);
void  glNamedFramebufferDrawBuffers (GLuint framebuffer, GLsizei n, const GLenum* bufs);
```

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glDrawBuffers	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
glNamedFramebufferDrawBuffers	-	-	-	-	-	-	-	-	-	-	-	✓

```
glDrawBuffers (3, buffers);
glNamedFramebufferDrawBuffers (fbo_handle, 3, buffers);
```

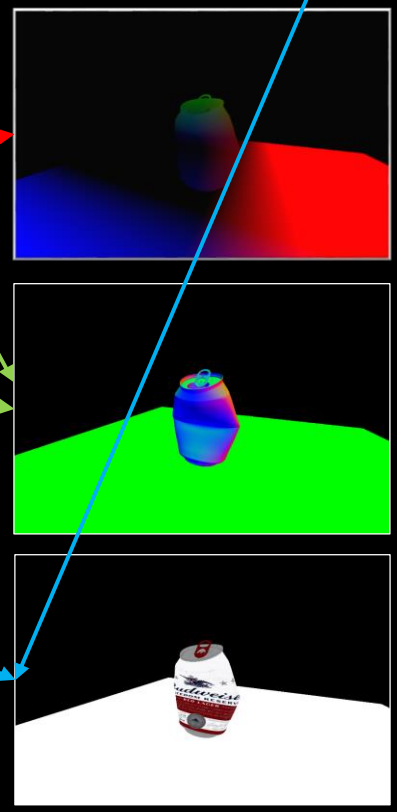
```
GLenum buffers[3] = {GL_COLOR_ATTACHMENT0, GL_COLOR_ATTACHMENT1, GL_COLOR_ATTACHMENT2};
```

FBO-Specify a list

```
GLenum buffers[3] = {GL_COLOR_ATTACHMENT0, GL_COLOR_ATTACHMENT1, GL_COLOR_ATTACHMENT2};
```

Fragment Shader

```
layout (location=0) in vec4 position;
layout (location=1) in vec4 normal;
layout (location=2) in vec4 color;
layout (location=0) out vec4 GBuffer0;
layout (location=1) out vec4 GBuffer1;
layout (location=2) out vec4 GBuffer2;
…
void main( )
{
        GBuffer0 = position ;
        GBuffer1 = normal;
        GBuffer2 = color;
}
```



FBO-Validate

```
GLenum  glCheckFramebufferStatus (GLenum target);
GLenum  glCheckNamedFramebufferStatus (GLuint framebuffer, GLenum target);
```

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glCheckFramebufferStatus	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
glCheckNamedFramebufferStatus	-	-	-	-	-	-	-	-	-	-	-	✓

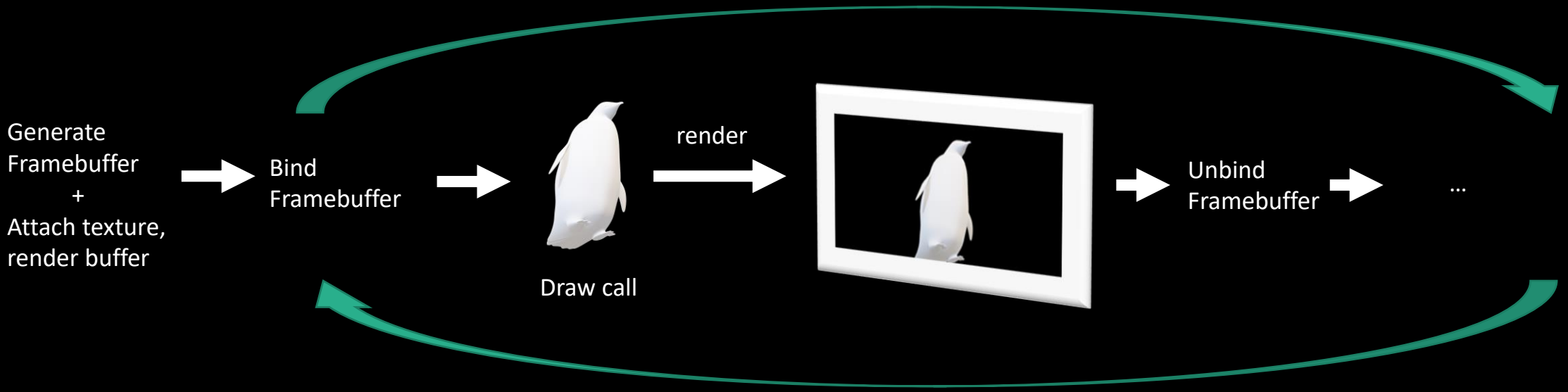
```
glCheckFramebufferStatus (GL_FRAMEBUFFER);
glCheckNamedFramebufferStatus (fbo_handle, GL_FRAMEBUFFER);
```

```
if ( glCheckNamedFramebufferStatus (fbo_handle, GL_FRAMEBUFFER) != GL_FRAMEBUFFER_COMPLETE)
{
        // Check error with return value
}
```

FBO-Validate

- GL_FRAMEBUFFER_UNDEFINED is returned if the specified framebuffer is the default read or draw framebuffer, but the default framebuffer does not exist.
- GL_FRAMEBUFFER_INCOMPLETE_ATTACHMENT is returned if any of the framebuffer attachment points are framebuffer incomplete.
- GL_FRAMEBUFFER_INCOMPLETE_MISSING_ATTACHMENT is returned if the framebuffer does not have at least one image attached to it.
- GL_FRAMEBUFFER_INCOMPLETE_DRAW_BUFFER is returned if the value of GL_FRAMEBUFFER_ATTACHMENT_OBJECT_TYPE is GL_NONE for any color attachment point(s) named by GL_DRAW_BUFFERi.
- GL_FRAMEBUFFER_INCOMPLETE_READ_BUFFER is returned if GL_READ_BUFFER is not GL_NONE and the value of GL_FRAMEBUFFER_ATTACHMENT_OBJECT_TYPE is GL_NONE for the color attachment point named by GL_READ_BUFFER.
- GL_FRAMEBUFFER_UNSUPPORTED is returned if the combination of internal formats of the attached images violates an implementation-dependent set of restrictions.
- GL_FRAMEBUFFER_INCOMPLETE_MULTISAMPLE is returned if the value of GL_RENDERBUFFER_SAMPLES is not the same for all attached renderbuffers; if the value of GL_TEXTURE_SAMPLES is the not same for all attached textures; or, if the attached images are a mix of renderbuffers and textures, the value of GL_RENDERBUFFER_SAMPLES does not match the value of GL_TEXTURE_SAMPLES.
- GL_FRAMEBUFFER_INCOMPLETE_MULTISAMPLE is also returned if the value of GL_TEXTURE_FIXED_SAMPLE_LOCATIONS is not the same for all attached textures; or, if the attached images are a mix of renderbuffers and textures, the value of GL_TEXTURE_FIXED_SAMPLE_LOCATIONS is not GL_TRUE for all attached textures.
- GL_FRAMEBUFFER_INCOMPLETE_LAYER_TARGETS is returned if any framebuffer attachment is layered, and any populated attachment is not layered, or if all populated color attachments are not from textures of the same target.

Geometry Pass



Geometry Pass-Bind

void glBindFramebuffer (GLenum target, GLuint framebuffer);

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glBindFramebuffer	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

```
glBindFramebuffer (GL_FRAMEBUFFER, fbo_handle);
glBindFramebuffer (GL_FRAMEBUFFER, 0); // default framebuffer
```

```
glBindFramebuffer(GL_READ_FRAMEBUFFER, m_fbo_deferred->GetHandle());
glBindFramebuffer(GL_DRAW_FRAMEBUFFER, m_fbo->GetHandle());
glBlitFramebuffer(
    0, 0, Input::s_m_windowSize.x, Input::s_m_windowSize.y, // source region
    0, 0, Input::s_m_windowSize.x, Input::s_m_windowSize.y, // destination region
    GL_DEPTH_BUFFER_BIT, // field to copy
    GL_NEAREST // filtering mechanism
);
```

Geometry Pass-Overview

OpenGL 3.0+

```
glGenFramebuffers (1, &fbo_handle);
glBindFramebuffer (GL_FRAMEBUFFER, fbo_handle);
glGenRenderbuffers (1, &rbo_handle);
glBindRenderbuffer (GL_RENDERBUFFER, rbo_handle);
glRenderbufferStorage (GL_RENDERBUFFER, GL_DEPTH_COMPONENT, width, height);
glFramebufferRenderbuffer (GL_FRAMEBUFFER, GL_DEPTH_ATTACHMENT, GL_RENDERBUFFER, rbo_handle);
glFramebufferTexture2D (GL_FRAMEBUFFER , GL_COLOR_ATTACHMENT0, GL_TEXTURE_2D, GBuffer0, 0);
glFramebufferTexture2D (GL_FRAMEBUFFER , GL_COLOR_ATTACHMENT1, GL_TEXTURE_2D, GBuffer1, 0);
glFramebufferTexture2D (GL_FRAMEBUFFER , GL_COLOR_ATTACHMENT2, GL_TEXTURE_2D, GBuffer2, 0);
glDrawBuffers (3, buffers);
if (glCheckFramebufferStatus (GL_FRAMEBUFFER) != GL_FRAMEBUFFER_COMPLETE)
{
        // Check error with return value
}
glBindRenderbuffer (GL_RENDERBUFFER, 0); //unbind
glBindFramebuffer (GL_FRAMEBUFFER, 0); //use default framebuffer
```

```
glBindFramebuffer (GL_FRAMEBUFFER, fbo_handle);
// Draw call here
glBindFramebuffer (GL_FRAMEBUFFER, 0);
```

Geometry Pass-Overview

OpenGL 4.5+

```
glCreateFrameBuffers (1, &fbo_handle);
glCreateRenderbuffers (1, &rbo_handle);
glNamedRenderbufferStorage (rbo_handle, GL_DEPTH_COMPONENT, width, height);
glNamedFramebufferRenderbuffer (fbo_handle, GL_DEPTH_ ATTACHMENT, GL_RENDERBUFFER , rbo_handle);
glNamedFramebufferTexture (fbo_handle, GL_COLOR_ATTACHMENT0, GBuffer0, 0);
glNamedFramebufferTexture (fbo_handle, GL_COLOR_ATTACHMENT1, GBuffer1, 0);
glNamedFramebufferTexture (fbo_handle, GL_COLOR_ATTACHMENT2, GBuffer2, 0);
glNamedFramebufferDrawBuffers (fbo_handle, 3, buffers);
if ( glCheckNamedFramebufferStatus (fbo_handle, GL_FRAMEBUFFER) != GL_FRAMEBUFFER_COMPLETE)
{
        // Check error with return value
}
glBindRenderbuffer (GL_RENDERBUFFER, 0); //unbind
glBindFramebuffer (GL_FRAMEBUFFER, 0); //back to default framebuffer
```

```
glBindFramebuffer (GL_FRAMEBUFFER, fbo_handle);
// Draw call here
glBindFramebuffer (GL_FRAMEBUFFER, 0);
```

Geometry Pass

OpenGL 4.5+

```
glBindFramebuffer (GL_FRAMEBUFFER, fbo_handle);
// Draw call here
glBindFramebuffer (GL_FRAMEBUFFER, 0);
```

```
glUseProgram (shader_handle);
…
glDrawArrays (…);  //or glDrawElements
glUseProgram (0);
```

Geometry Pass

```
glUseProgram (shader_handle);
…
glDrawArrays (…);  //or glDrawElements
glUseProgram (0);
```

Vertex Shader

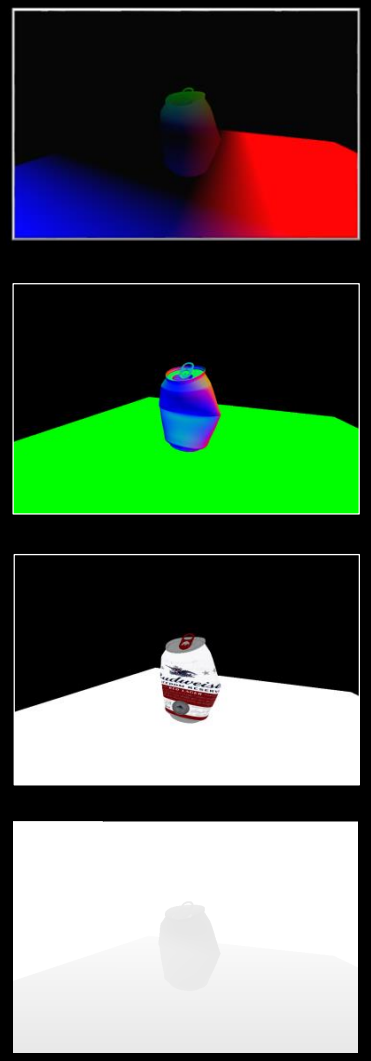
```
layout (location=0) in vec4 position_in;
layout (location=1) in vec4 normal_in;
layout (location=2) in vec2 texcoord_in;
layout (location=0) out vec4 position;
layout (location=1) out vec4 normal;
layout (location=2) out vec4 color;
uniform sampler2D color_texture;
void main( )
{
        position = … ; //world coordinate
        normal= normalize(…); //world coordinate
        color= texture2D(color_texture, texcoord_in) ;
        gl_Position = worldToNDC * position;
}
```

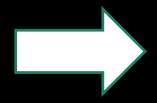
Fragment Shader

```
layout (location=0) in vec4 position;
layout (location=1) in vec4 normal;
layout (location=2) in vec4 color;
layout (location=0) out vec4 GBuffer0;
layout (location=1) out vec4 GBuffer1;
layout (location=2) out vec4 GBuffer2;
…
void main( )
{
        GBuffer0 = position ;
        GBuffer1 = normal;
        GBuffer2 = color;
}
```

Lighting Pass

Framebuffer

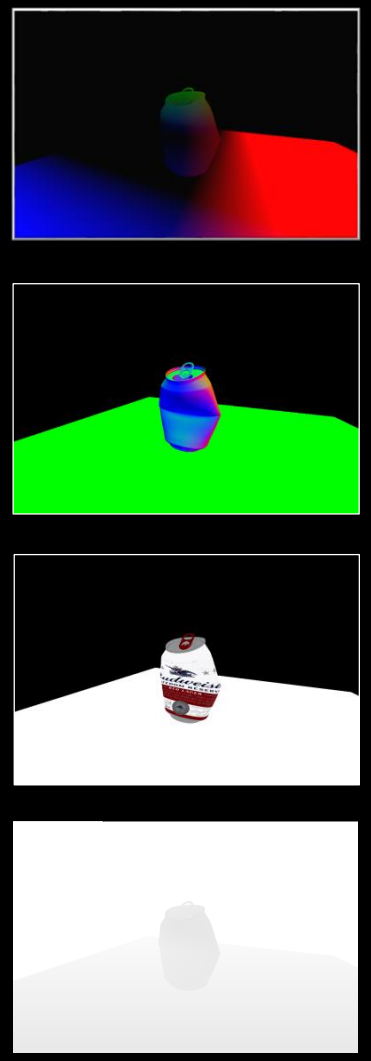




$$I_p = k_a i_a + \sum_{m}^{lights} (k_d i_{m,d}(N \cdot L) + k_s i_{m,s}(R \cdot V)^{\alpha})$$

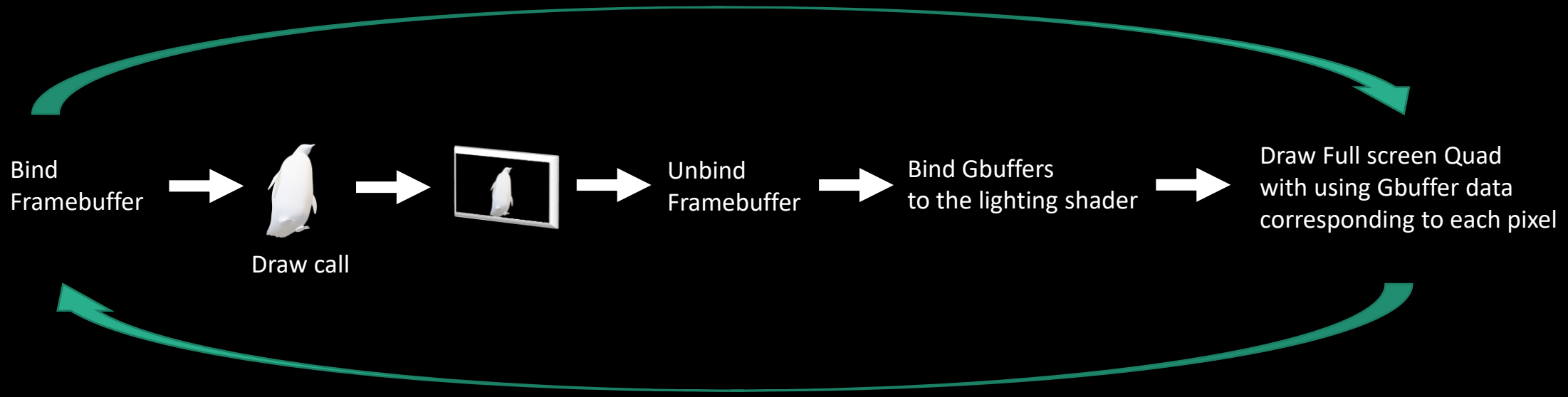
Lighting Pass

Framebuffer

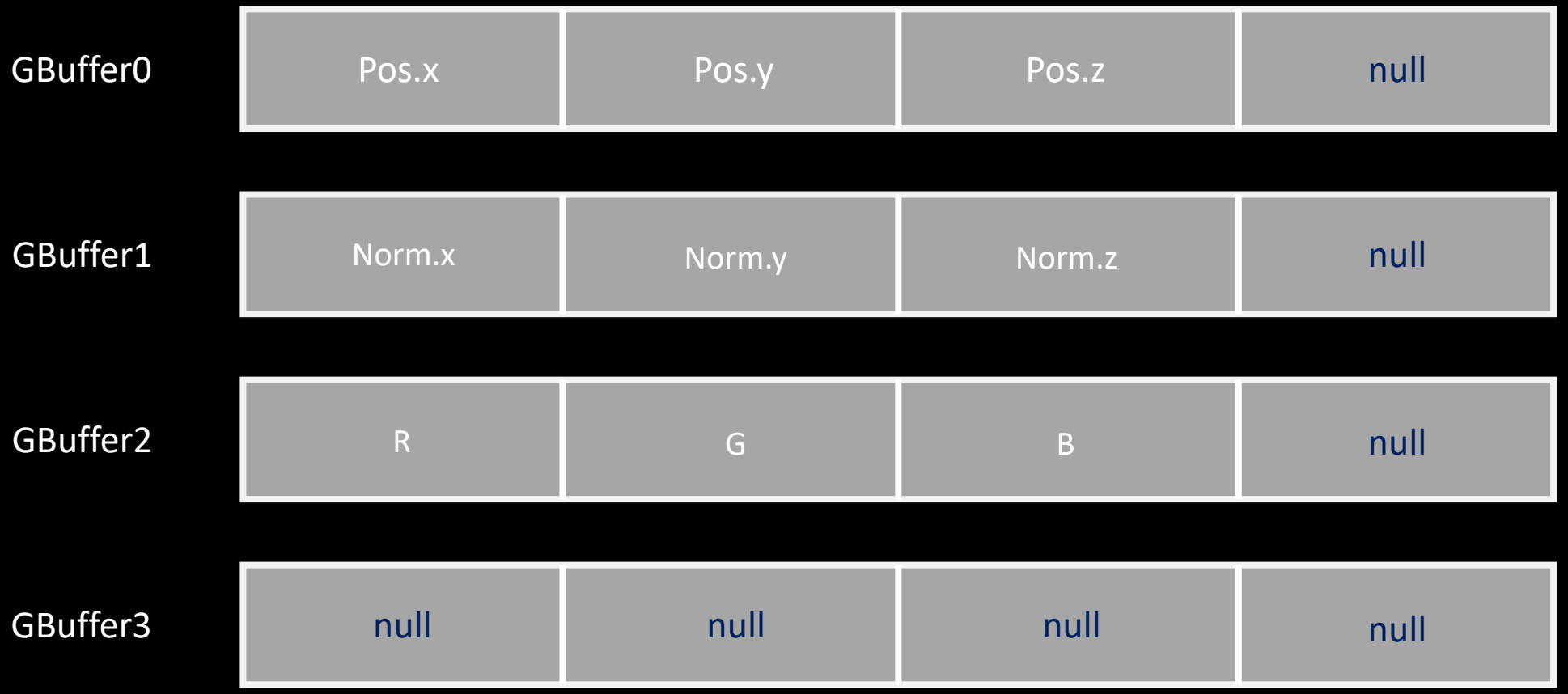


Review

Generate Framebuffer → Generate Gbuffers(textures) and Renderbuffer → Attach Gbuffers and renderbuffer to the framebuffer



Advanced

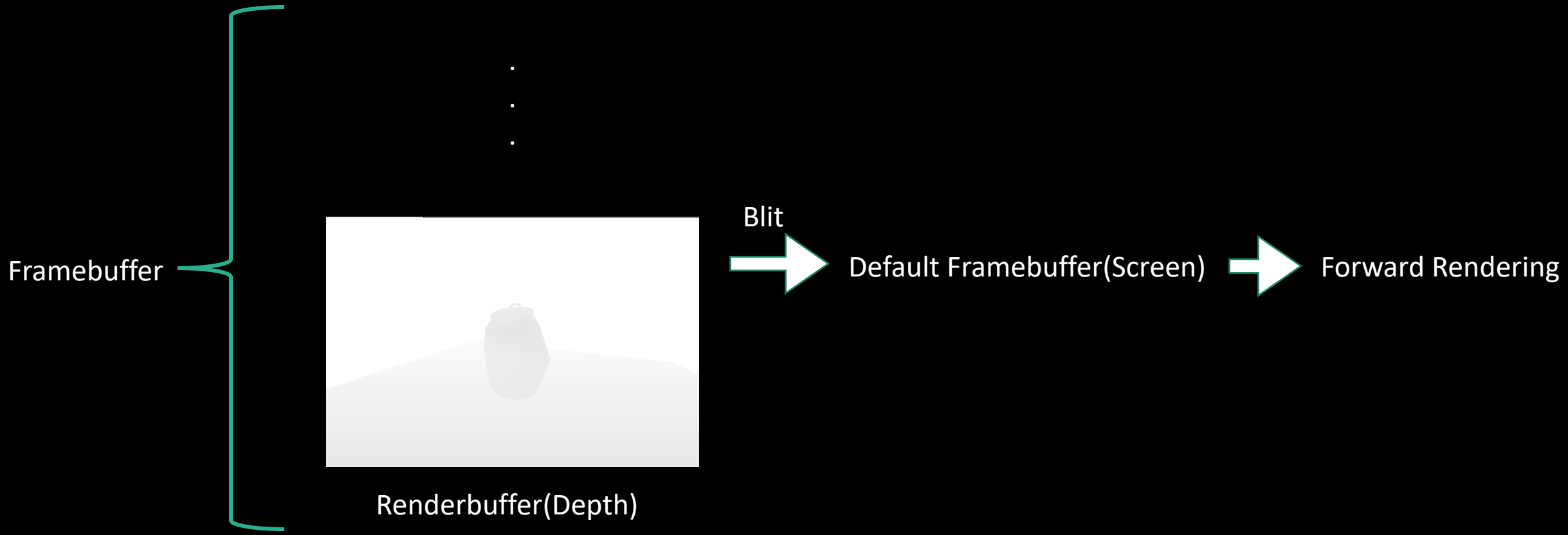

GBuffer0
Pos.x
Pos.y
Pos.z
null
GBuffer1
Norm.x
Norm.y
Norm.z
null
GBuffer2
R
G
B
null
GBuffer3
null
null
null
null

Pros & Cons

- Lighting is less dependent on geometry
- Worst case complexity depends on number of object and lights
- Easy to integrate with screen-space techniques such as shadow mapping, temporal anti-aliasing

- Hard to support a wide variety of materials
- Cannot handle translucent or transparent objects

Forward + Deferred



Forward + Deferred

void glBlitFramebuffer (GLint srcX0, GLint srcY0, GLint srcX1, GLint srcY1, GLint dstX0, GLint dstY0, GLint dstX1, GLint dstY1, GLbitfield mask, GLenum filter);

	OpenGL Version											
Function / Feature Name	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glBlitFramebuffer	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
glBlitNamedFramebuffer	-	-	-	-	-	-	-	-	-	-	-	✓

```
glBindFramebuffer (GL_READ_FRAMEBUFFER, fbo_handle);
glBindFramebuffer (GL_DRAW_FRAMEBUFFER, 0);

glBlitFramebuffer (
        0, 0, width, height,
        0, 0, width, height,
        GL_DEPTH_BUFFER_BIT,
        GL_NEAREST
);
```

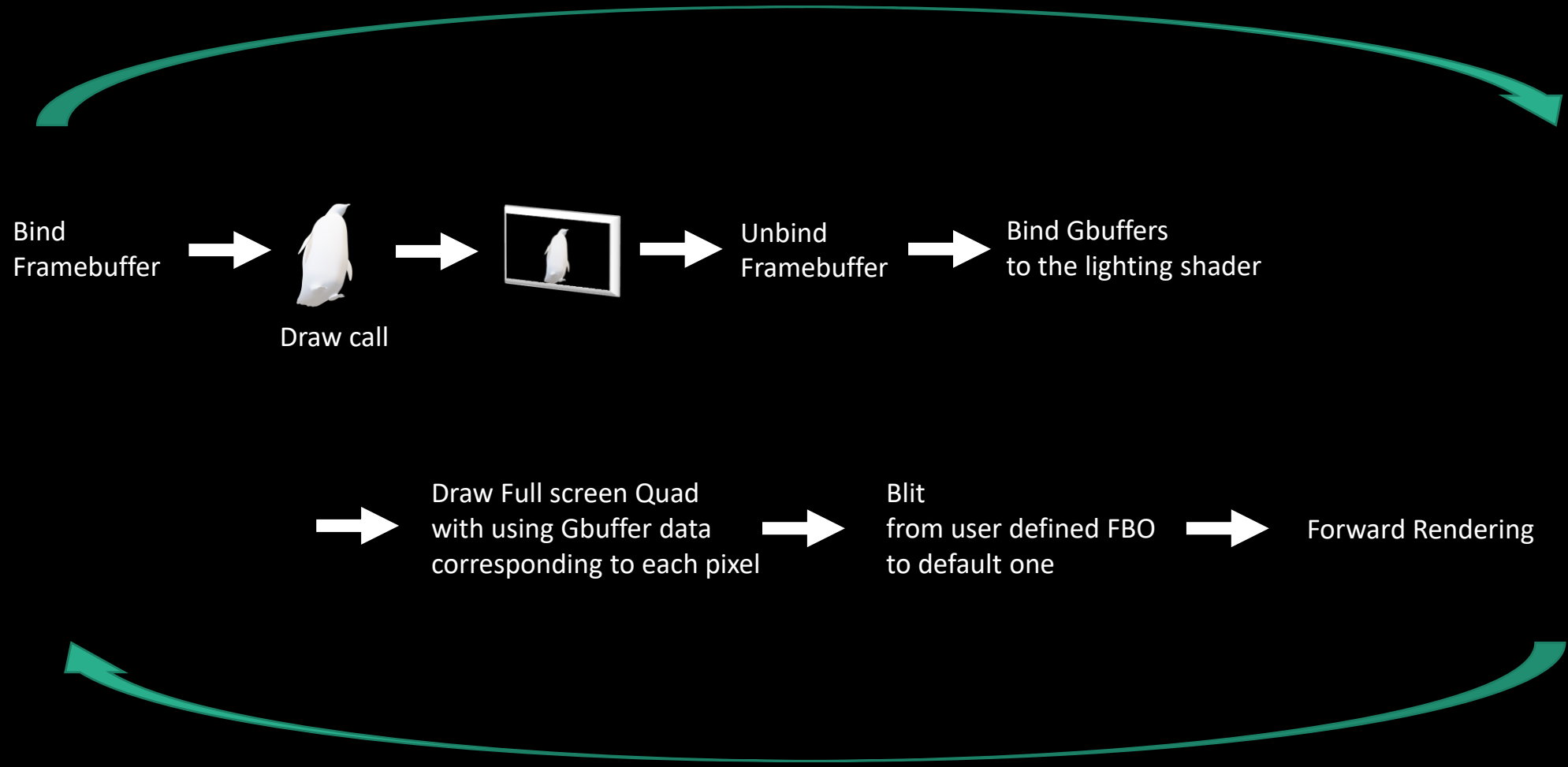
Forward + Deferred

void glBlitNamedFramebuffer (GLuint readFramebuffer, GLuint drawFramebuffer, GLint srcX0, GLint srcY0, GLint srcX1, GLint srcY1, GLint dstX0, GLint dstY0, GLint dstX1, GLint dstY1, GLbitfield mask, GLenum filter);

Function / Feature Name	OpenGL Version											
	2.0	2.1	3.0	3.1	3.2	3.3	4.0	4.1	4.2	4.3	4.4	4.5
glBlitFramebuffer	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
glBlitNamedFramebuffer	-	-	-	-	-	-	-	-	-	-	-	✓

```
glBlitFramebuffer (
        fbo_handle, 0,
        0, 0, width, height,
        0, 0, width, height,
        GL_DEPTH_BUFFER_BIT,
        GL_NEAREST
);
```

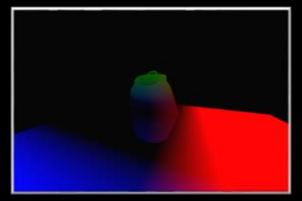
Forward + Deferred

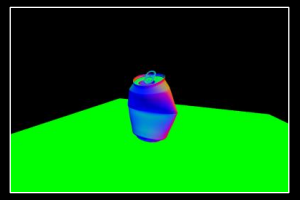


Local Lighting

Deferred Rendering →

Local Lighting





gl_FragCoord.xy/u_window_size.xy

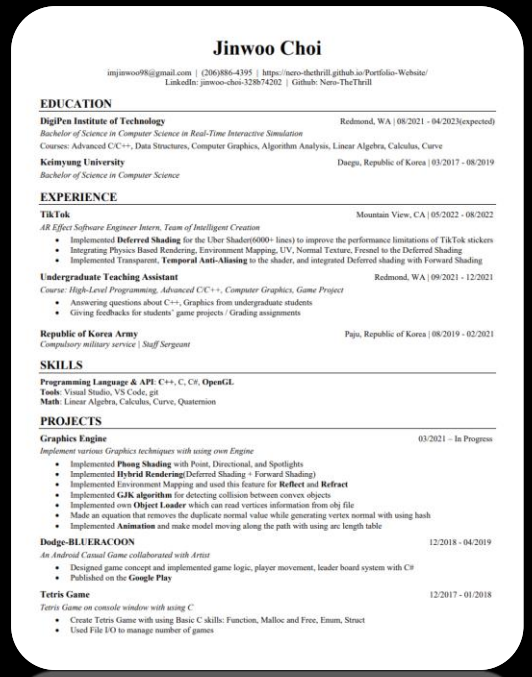
Additive Blending On

Depth Test Off

Face Culling On

Internship

Resume



Jinwoo Choi

imjinwoo98@gmail.com | (206)886-4395 | https://nero-thethrill.github.io/Portfolio-Website/
LinkedIn: jinwoo-choi-328b74202 | Github: Nero-TheThrill

EDUCATION

DigiPen Institute of Technology — Redmond, WA | 08/2021 - 04/2023(expected)
Bachelor of Science in Computer Science in Real-Time Interactive Simulation
Courses: Advanced C/C++, Data Structures, Computer Graphics, Algorithm Analysis, Linear Algebra, Calculus, Curve

Keimyung University — Daegu, Republic of Korea | 03/2017 - 08/2019
Bachelor of Science in Computer Science

EXPERIENCE

TikTok — Mountain View, CA | 05/2022 - 08/2022
AR Effect Software Engineer Intern, Team of Intelligent Creation
- Implemented **Deferred Shading** for the Uber Shader(6000+ lines) to improve the performance limitations of TikTok stickers
- Integrating Physics Based Rendering, Environment Mapping, UV, Normal Texture, Fresnel to the Deferred Shading
- Implemented Transparent, **Temporal Anti-Aliasing** to the shader, and integrated Deferred shading with Forward Shading

Undergraduate Teaching Assistant — Redmond, WA | 09/2021 - 12/2021
Course: High-Level Programming, Advanced C/C++, Computer Graphics, Game Project
- Answering questions about C++, Graphics from undergraduate students
- Giving feedbacks for students' game projects / Grading assignments

Republic of Korea Army — Paju, Republic of Korea | 08/2019 - 02/2021
Compulsory military service | Staff Sergeant

SKILLS

Programming Language & API: C++, C, C#, **OpenGL**
Tools: Visual Studio, VS Code, git
Math: Linear Algebra, Calculus, Curve, Quaternion

PROJECTS

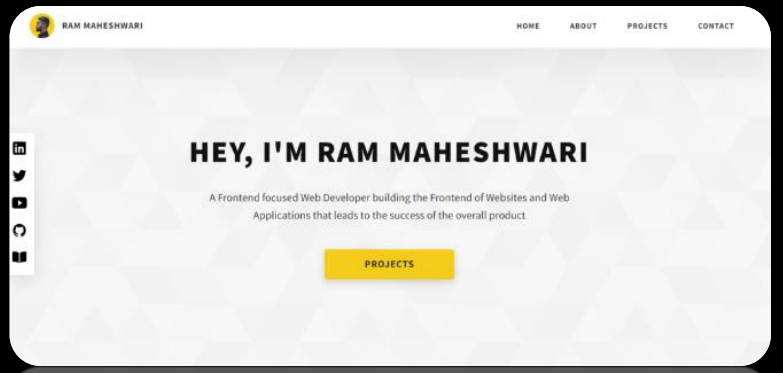
Graphics Engine — 03/2021 – In Progress
Implement various Graphics techniques with using own Engine
- Implemented **Phong Shading** with Point, Directional, and Spotlights
- Implemented **Hybrid Rendering**(Deferred Shading + Forward Shading)
- Implemented Environment Mapping and used this feature for **Reflect** and **Refract**
- Implemented **GJK algorithm** for detecting collision between convex objects
- Implemented own **Object Loader** which can read vertices information from obj file
- Made an equation that removes the duplicate normal value while generating vertex normal with using hash
- Implemented **Animation** and make model moving along the path with using arc length table

Dodge-BLUERACOON — 12/2018 - 04/2019
An Android Casual Game collaborated with Artist
- Designed game concept and implemented game logic, player movement, leader board system with C#
- Published on the **Google Play**

Tetris Game — 12/2017 - 01/2018
Tetris Game on console window with using C
- Create Tetris Game with using Basic C skills: Function, Malloc and Free, Enum, Struct
- Used File I/O to manage number of games

(Digipen Handshake resume guideline)
https://digipen.joinhandshake.com/stu/schools/1396/articles

Portfolio



Personal Projects

LinkedIn, Leetcode, etc.