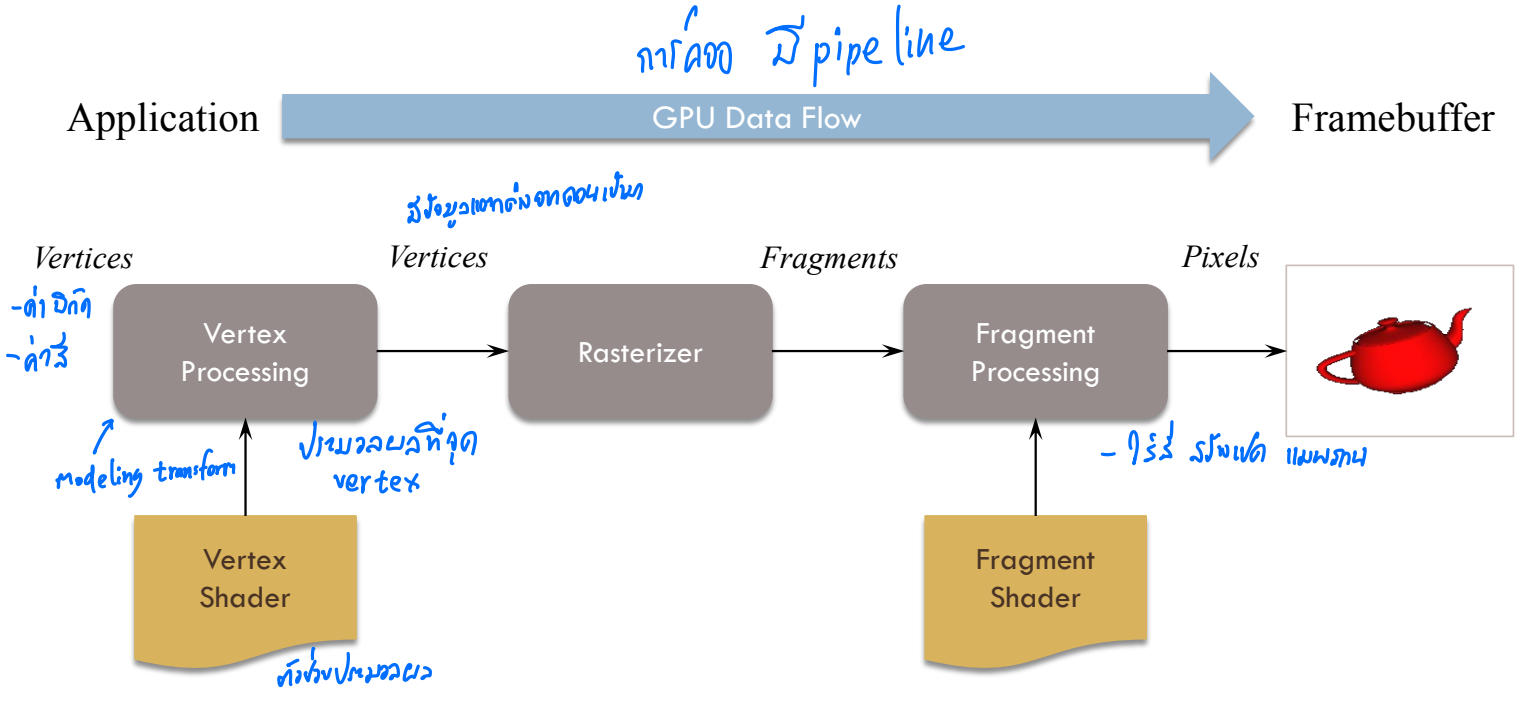
Week 4:

- Client-Side Vertex Arrays
  - แตกต่างจากการเก็บแบบ display list
  - ใช้ array ที่จัดเก็บชุด vertex จะส่งผ่านระหว่าง ฝั่ง client กับ ฝั่ง server → GPU

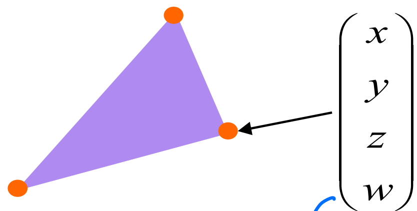
Chakrit Watcharopas

A Simplified Pipeline Model



Representing Geometric Objects

- Geometric objects are represented using *vertices* รูปเรขาคณิต มีจุด vertices เป็นองค์ประกอบ
- A vertex is a collection of generic attributes
  - positional coordinates ตำแหน่งจุด
  - colors สีตรงจุดนั้น
  - texture coordinates พิกัดที่เอารูปภาพของมา map บนตัวโมเดล
  - any other data associated with that point in space

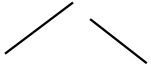


- Position stored in 4 dimensional homogeneous coordinates

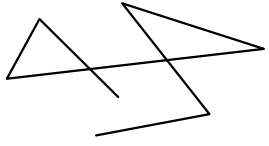
OpenGL's Geometric Primitives

- All primitives are specified by vertices

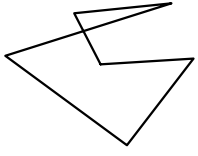
GL_POINTS



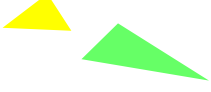
GL_LINES



GL_LINE_STRIP



GL_LINE_LOOP



GL_TRIANGLES

GL_TRIANGLE_STRIP

GL_TRIANGLE_FAN

Cube Data

- Vertices of a unit cube centered at origin
  - sides aligned with axes

ลูกบาศก์ที่เป็นด้านเท่า

```
position_data = [
    ( -0.5, -0.5,  0.5, 1.0 ),
    ( -0.5,  0.5,  0.5, 1.0 ),
    (  0.5,  0.5,  0.5, 1.0 ),
    (  0.5, -0.5,  0.5, 1.0 ),
    ( -0.5, -0.5, -0.5, 1.0 ),
    ( -0.5,  0.5, -0.5, 1.0 ),
    (  0.5,  0.5, -0.5, 1.0 ),
    (  0.5, -0.5, -0.5, 1.0 )
]
```

Cube Data (cont'd)

- We'll also set up an array of RGBA colors

```
color_data = [
    ( 0.0, 0.0, 0.0, 1.0 ),  # black
    ( 1.0, 0.0, 0.0, 1.0 ),  # red
    ( 1.0, 1.0, 0.0, 1.0 ),  # yellow
    ( 0.0, 1.0, 0.0, 1.0 ),  # green
    ( 0.0, 0.0, 1.0, 1.0 ),  # blue
    ( 1.0, 0.0, 1.0, 1.0 ),  # magenta
    ( 1.0, 1.0, 1.0, 1.0 ),  # white
    ( 0.0, 1.0, 1.0, 1.0 )   # cyan
]
```

Generating the Cube from Faces

- Generate 6 quads for the cube
  - 24 vertices with 24 colors

```
face_data = [
    ( 1, 0, 3, 2 ),
    ( 2, 3, 7, 6 ),
    ( 3, 0, 4, 7 ),
    ( 6, 5, 1, 2 ),
    ( 4, 5, 6, 7 ),
    ( 5, 4, 0, 1 )
]
```

Drawing the Cube


```
def drawCube():
    glBegin(GL_QUADS)
    for i in range(6):
        for j in range(4):
            vid = face_data[i][j]
            glColor4fv(color_data[vid])
            glVertex4fv(position_data[vid])
    glEnd()
```

มีการเรียก function 4 ครั้ง

Client-Side Vertex Arrays

- We can reduce the number of function calls for supplying vertex attributes to GPU
- With Vertex Arrays, we can transfer multiple vertex attributes in memory using a single call
- We convert vertex data into *numpy* array

Client-Side Vertex Arrays

- Put vertex data in numpy array

```
import numpy as np
position_np = np.array(position_data,
                       dtype=np.float32)
color_np = np.array(color_data,
                    dtype=np.float32)
face_np = np.array(face_data, dtype=np.int32)
positions = position_np[face_np.flatten()]
colors = color_np[face_np.flatten()]
```

np array 2D

list ของ turple

Array 2 ตัว แต่ละตัวมี 4 float

Client-Side Vertex Arrays เรียกใช้ครั้งเดียว

- Prepare passing vertex data to GPU

```
glEnableClientState(GL_VERTEX_ARRAY)
glEnableClientState(GL_COLOR_ARRAY)
glVertexPointer(4, GL_FLOAT, 0, positions)
glColorPointer(4, GL_FLOAT, 0, colors)
```

- Now pass vertex data to GPU and draw quads with glDrawArrays()

```
glDrawArrays(GL_QUADS, 0, 24)
```

References

- Angel, E., & Shreiner, D. (2013, July). An introduction to OpenGL programming. In *SIGGRAPH Courses* (pp. 3-1).
- Khronos.org. (2018). *Client-Side Vertex Arrays - OpenGL Wiki*. Available at: www.khronos.org/opengl/wiki/Client-Side_Vertex_Arrays.