CCWC 2023

Aspects of VR
WebXR

Phillip G. Bradford

University of Connecticut

Outline

Motivation

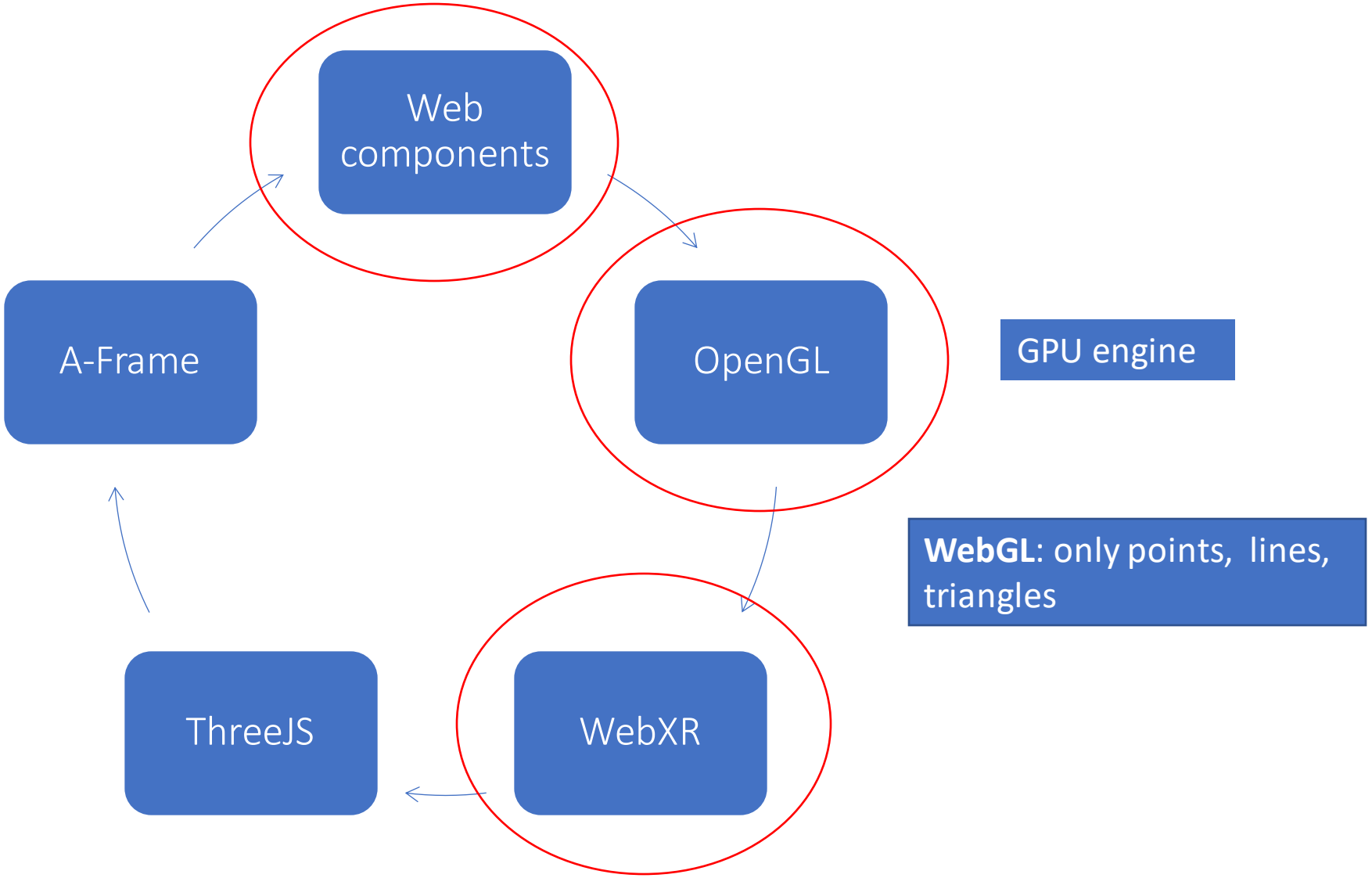
High level view and learning path

What is WebXR?
    WebGL - OpenGL

WebXR pipeline

WebXR API

High level view and learning path



Setup

Ensure Python3 is installed – download and install

https://www.python.org/

Windows> mkdir workshop

Windows> cd workshop

Windows> python3 -m http.server 9000

What is WebXR?

WebXR is a Specification for WebVR/WebAR and WebMR

https://www.w3.org/TR/webxr/

The WebXR specification is by the *Immersive Web Working Group*

https://www.w3.org/immersive-web/

The WebXR API is an implementation of the WebXR spec

It allows webpages to be presentation and rendering engines

WebGL

WebGL is a rasterizer used by WebXR

- The WebGL API is an implementation of the WebGL spec
- The WebGL spec (Kronos group) is based on the OpenGL spec
- Draws points, lines, and triangles

WebGL is a Javascript API to OpenGL

OpenGL is an open graphics library in C/C++ for rendering graphics using GPU/Graphics cards

WebXR fundamentals

https://developer.mozilla.org/en-US/docs/Web/API/WebXR_Device_API/Fundamentals

https://addons.mozilla.org/en-US/firefox/addon/webxr-api-emulator/

WebGL Rendering

WebGLRenderingContext - Web APIs | MDN (mozilla.org)

https://github.com/mdn/dom-examples/tree/main/webgl-examples/tutorial

https://github.com/mdn/dom-examples

Start the server

Windows/MacOS> python3 –m http.server 9000

Avoids CTRL-O open file:/// which is not secure – see CORS discussion

WebGL

Rasterization engine

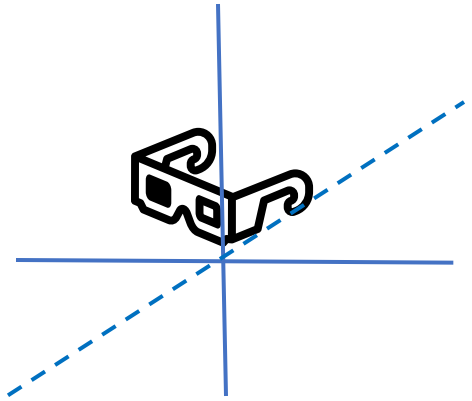
- Converts vector images into pixel images

Leverages the GPU

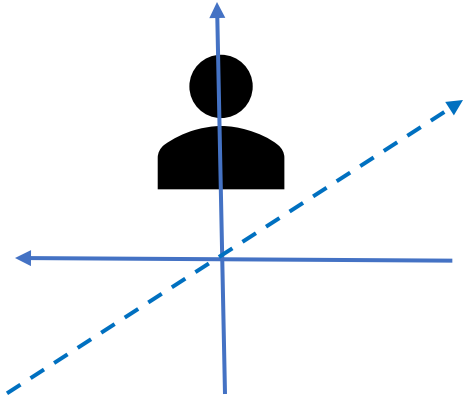
- Vector shader – computes vertex positions
- Fragment shader – computes colors of vertices

Uses OpenGL

3DoF or 6DoF



Rotation in 3D



Translation in 3D

<canvas></canvas> Tags: Rakesh Baruah

Drawing 2D and 3D graphics – render state and behavior

```
<body>
   <h3>WebGL</h3>
   <canvas id='myCanvas'></canvas>
</body>
<script type="text/javascript">
   const g1 = myCanvas.getContext('webgl');
   if (!g1) {
      alert('WebGL not available');
      console.log('WebGL not available');
   } else {
      alert('WebGL good');
      console.log('WebGL good');
   }
   </script>
```

canvas-gettransform-settransform

```
<canvas></canvas>
 <canvas></canvas>

 <script>
  const canvases = document.querySelectorAll('canvas');
  const ctx1 = canvases[0].getContext('2d');
  const ctx2 = canvases[1].getContext('2d');

  ctx1.setTransform(1, .2, .8, 1, 0, 0); // h-scale, v-skew, h-skew, v-scale, h-trans, v-trans
  ctx1.fillRect(25, 25, 50, 50);  // fillRect(x, y, width, height)

  let storedTransform = ctx1.getTransform();
  console.log(storedTransform);

  ctx2.setTransform(storedTransform);
  ctx2.beginPath();
  ctx2.arc(50, 50, 50, 0, 2 * Math.PI);   // arc(x, y, radius, startAngle, endAngle)
  ctx2.fill();
 </script>
```

OpenGL - http://www.dgp.toronto.edu/~ah/csc418/fall_1999/tut/square

```
#include <stdio.h>
#include <GL/glut.h>
void display(void)
{
glClear( GL_COLOR_BUFFER_BIT);
glColor3f(0.0, 0.0, 0.0);
glBegin(GL_POLYGON);
  glVertex3f(2.0, 4.0, 0.0);
  glVertex3f(8.0, 4.0, 0.0);
  glVertex3f(8.0, 6.0, 0.0);
  glVertex3f(2.0, 6.0, 0.0);
glEnd();
glFlush();
}
```

OpenGL -

http://www.dgp.toronto.edu/~ah/csc418/fall_1999/tut/square

```
int main(int argc, char **argv) {
printf("hello world\n");
glutInit(&argc, argv);
glutInitDisplayMode ( GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);

glutInitWindowPosition(100,100);
glutInitWindowSize(300,300);
glutCreateWindow ("square");

glClearColor(1.0, 1.0, 1.0, 0.0);        // black background
glMatrixMode(GL_PROJECTION);             // setup viewing projection
glLoadIdentity();                        // start with identity matrix
glOrtho(0.0, 10.0, 0.0, 10.0, -1.0, 1.0);   // setup a 10x10x2 viewing world

glutDisplayFunc(display);
glutMainLoop();

return 0;
}
```

Matrix transformations

Magnitude

x.M	0	0	
0	y.M	0	
0	0	z.M	
			1

product

x
y
z
1

=

x*x.M	y*y.M	z*z.M	1

Matrix translation

Translation

1	0	0	x.T
0	1	0	y.T
0	0	1	z.T
			1

product

x
y
z
1

=

x+x.T	y+y.T	z+z.T	1

Matrix transformations

Skew

1	h.sk	0	
v.sk	1	0	
0	0	1	
		1	

x
y
z
1

Matrices

https://glmatrix.net/