Gede Prasidha Bhawarnawa

IF 2020 - 13520004

BENTUK 3D

Apa saja yang harus dilakukan saat ingin merender suatu bentuk kubus?

Jelaskan bagaimana webgl memproses bangun 3D yang telah Anda spesifikasikan dalam kode sehingga dapat ditampilkan di layar Anda!

Jawabannya digabung ya kak biar tidak repetisi :D

Terdapat lima langkah dasar untuk webgl memproses bangun 3D (kubus) di dalam source code ini.

1. Menyiapkan kanvas dan memanggil context webgl untuk digambarkan objek 3D yang diinginkan. Ini dapat dilakukan dengan cara memanggil ID dari objeck kanvas pada html dan menghubungkannya dengan webgl di script JavaScript.
2. Langkah kedua adalah memasukkan informasi mengenai kubus tersebut ke dalam buffer object. Informasi ini dapat berupa titik sudut, warna, urutan penggambaran atau pengindeksan (indices), dll. Terdapat dua jenis buffer yang digunakan dalam proses binding data ke dalam buffer di source code ini, yaitu ARRAY_BUFFER dan ELEMENT_ARRAY_BUFFER. ARRAY_BUFFER digunakan untuk menyimpan buffer dari kubus sedangkan ELEMENT_ARRAY_BUFFER digunakan untuk menyimpan indeks dari kubus. Perbedaan ini akan lebih jelas terlihat ketika dilakukan binding di langkah ketiga dan penggambaran objek di langkah kelima.
3. Langkah ketiga adalah pembuatan dan kompilasi program shader program. Shader program adalah program-program yang berjalan di GPU dan ditulis dengan bahasa OpenGL ES Shader Language (ESSL) atau disebut juga sebagai OpenGL Shading Language (GLSL). Bahasa ini mirip dengan bahasa C/C++ dan memiliki tipe data, I/O, dan penggunaan variabel dan konstanta sendiri. Terdapat dua jenis shader yang dipakai di source code ini: vertex shader dan fragment shader. Vertex shader adalah program shader yang dipanggil untuk setiap titik sudut dari objek yang didefinisikan di langkah dua sebelumnya. Sesuai dengan namanya, vertex shader mengendalikan dan memproses data setiap titik sudut seperti posisi, transformasi, warna, dan koordinat tesktur. Lalu, terdapat fragment shader. Di dalam konteks webGL, fragment adalah permukaan dari setiap segitiga yang terbentuk oleh titik-titik sudut yang telah didefinisikan. Fragment shader bertanggung jawab untuk mengolah dan memberikan warna ke setiap pixel di fragment. Setelah fragment shader dan vertex shader dibuat dan dilink ke source code GLSL, maka shader terus akan dikompilasi sehingga dapat dijalankan oleh GPU. Lalu, program yang sudah dikompilasi akan diattach ke program shader utama untuk digunakan saat proses penggambaran.
4. Di dalam program GLSL yang sudah dikompolasi pada langkah ketiga, terdapat atribut-atribut yang sebenarnya menunjuk langsung ke elemen di webGL. Maka dari itu, pada langkah keempat ini, perlu dilakukan asosiasi variabel di dalam shader program ke objek-objek webgl yang ingin digambar. Terdapat tiga tahapan dalam melakukan ini: menentukan lokasi dari objek tersebut, mengasosiasikan lokasi dari objek tersebut ke variabel di dalam shader program, dan mengaktifkan variabel atau atribut tersebut. Pada source code program ini, dilakukan tiga kali binding. Binding pertama dilakukan terhadap posisi dari vertex. Binding kedua dilakukan terhadap warna dari vertex dan binding ketiga dilakukan

terhadap posisi setiap vertex dalam tiga dimensi dan rotasi. Perlu diperhatikan bahwa digunakan vertexAttribPointer untuk atribut-atribut yang disimpan dalam ARRAY_BUFFER sementara ELEMENT_ARRAY_BUFFER hanya akan digunakan nanti saat proses penggambaran. Lalu, terdapat proses unbinding dimana ARRAY_BUFFER yang sudah digunakan akan didealokasi memori bindingnya. Ini dilakukan agar jika program dikompilasi ulang, maka tidak akan terjadi stacking antara memori buffer yang lama yang masih terdapat di memori dengan memori buffer yang baru. Karena pada source code ini juga terdapat rotasi, maka projeksi untuk rotasi terhadap waktu untuk setiap sisi juga diperlukan terhadap matriks perpindahan dan penampilan (movement matrix dan view matrix).

5. Langkah terakhir adalah menggambar bangunnya itu sendiri. Terdapat beberapa persiapan yang perlu dilakukan, khususnya untuk menggambar bentuk 3D seperti melakukan depth testing, inisiasi matriks 3D, dan pembersihan (reset) COLOR_BUFFER_BIT dan DEPTH_BUFFER_BIT. Setelah itu, terdapat dua opsi yang dapat digunakan untuk menggambar bentuknya, yaitu drawArray dan drawElements. drawArray digunakan apabila tidak ada informasi indeks sehingga webGL akan menggambar berdasarkan urutan yang ada di array saat pendefinisian vertex di langkah kedua. drawElements digunakan ketika terdapat urutan indeks yang dapat dimanfaatkan oleh webGL menggunakan buffer ELEMENT_ARRAY_BUFFER. Karena terdapat vertex yang digunakan berulang kali pada kubus, maka metode yang paling tepat digunakan adalah drawElements.

Jika ingin merubah bentuk kubus menjadi bangun 3D lain yang telah Anda buat, perubahan apa saja yang perlu dilakukan?

Perubahan yang perlu dilakukan adalah perubahan informasi tentang vertex yang didefinisikan dan dibind ke ARRAY_BUFFER pada langkah kedua dari penjelasan sebelumnya. Selain itu, ada kemungkinan terdapat modifikasi yang harus dilakukan terhadap shader program yang dikompilasi karena ada perubahan (dapat berupa penambahan atau pengurangan) atribut dan qualifier di source codenya. Selain itu, terdapat juga perubahan pada indices atau pengindeksan yang digunakan karena besar kemungkinan ada perbedaan metode penggambaran segitiga untuk bentuk 3D lainnya.