Nama : Shopia Mudjahidah
NIM : 24060123120010
Lab : GKV E1

Materi 7 – Tekstur dengan Gambar

Tekstur adalah tampilan permukaan suatu benda yang bisa dikenali melalui penglihatan atau sentuhan. Dalam OpenGL, objek 3D memang terlihat lebih nyata dibanding objek 2D. Tapi, jika permukaannya polos saja, tampilannya akan terasa kaku dan kurang menarik. Untuk mengatasi hal ini, digunakanlah texture mapping, yaitu proses menempelkan gambar (yang disebut map) ke permukaan objek 3D. Dengan metode ini, kita bisa memberikan kesan realistis pada objek. Misalnya, sebuah kubus bisa terlihat seperti batu jika kita memberikan gambar tekstur batu pada permukaannya. Tekstur ini juga akan mengikuti transformasi objek, seperti rotasi atau pergeseran, sehingga hasil akhirnya terlihat lebih alami dan menyatu. Proses ini bisa pakai gambar BMP, dan gambar itu nanti disesuaikan agar mengikuti transformasi (kayak rotasi, geser, zoom) dari objek 3D-nya. Supaya dapat membaca gambar BMP, kita harus buat dua file tambahan yaitu Imageloader.h dan Imageloader.cpp. Kedua file ini fungsinya untuk membaca gambar berformat BMP dan mengubah datanya agar bisa digunakan sebagai tekstur di OpenGL. Setelah itu, kita juga perlu atur ulang tasks.json dan settings.json supaya ketika program dijalankan, file tambahan tadi ikut dikompilasi bersama main.cpp. Di dalam program utamanya, ada bagian initRendering() yang bertugas untuk mengatur pencahayaan dan mengaktifkan fitur tekstur. Lalu, pada drawScene(), kubus digambar lengkap dengan sisi-sisinya, diberi warna, dan juga ditambahkan tekstur gambar di sisi tertentu dengan perintah glTexCoord2f(). Hasil akhirnya, kita bisa melihat sebuah kubus berputar yang punya permukaan bergambar.

Materi 8 – Bayangan

Agar gambar 3D terlihat lebih nyata, kita perlu menambahkan efek cahaya seperti pantulan, pembiasan, dan bayangan. Pantulan digunakan untuk objek mengkilap seperti cermin, sedangkan pembiasan cocok untuk benda transparan seperti kaca. Yang paling penting adalah bayangan, karena membuat objek tampak benar-benar berada di suatu tempat. Bayangan harus sesuai dengan arah dan jumlah sumber cahaya agar hasilnya realistis. Objek yang digunakan adalah kerucut (seperti topi ulang tahun), dan kita juga buat lantai datar sebagai tempat bayangan jatuh. Kita tentukan dulu posisi cahaya, bidang tempat bayangan jatuh, dan posisi objeknya. Dari situ, digunakan fungsi glShadowProjection() untuk menghitung bagaimana bayangan dari objek bisa diproyeksikan ke permukaan lantai. Jadi, bukan bayangan otomatis seperti di game engine modern, tapi kita hitung manual proyeksinya pakai matriks. Di bagian render(), objek utama (kerucut) digambar seperti biasa, lalu setelah itu bayangannya digambar ulang dengan matriks proyeksi tadi dan diberi warna abu-abu gelap. Supaya interaktif, kita bisa menggerakkan sumber cahaya dengan tombol-tombol di keyboard. Efek bayangan ini penting banget karena bikin visualisasi objek terlihat lebih nyata, seolah-olah benar-benar ada dalam ruang 3D, apalagi saat bayangannya bergerak sesuai arah cahaya.

Materi 9 – Interaksi Antar Objek

Kode dibuat untuk mensimulasikan interaksi antar objek 3D dengan mengandalkan posisi dan status variabel, misalnya menendang bola atau mengangkat kotak. Karena OpenGL tidak punya sistem deteksi tabrakan atau sentuhan, maka kita pakai logika koordinat. Misalnya dalam simulasi menendang bola, kita punya objek kaki dan bola. Saat kaki digerakkan ke posisi tertentu dan cukup dekat dengan bola, maka bola akan otomatis bergerak menjauh, seolah-olah ditendang. Pergerakan ini dikontrol lewat variabel status seperti kick, roll, dan touch yang diatur dalam kondisi if. Sedangkan untuk simulasi memegang, kita membuat karakter yang bisa menggerakkan tangan. Saat tangan menyentuh kotak (dengan memeriksa posisi koordinatnya), dan tombol yang sesuai ditekan, maka status grab aktif dan kotak akan ikut bergerak saat tubuh bergerak. Ketika tangan diturunkan, kotaknya juga ikut turun. Kita tidak benar-benar mendeteksi tabrakan secara otomatis, tapi kita buat logika seolah-olah itu terjadi berdasarkan posisi.