Ministerul Educației și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Departamentul Ingineria Software și Automatica

Raport

Lucrarea de laborator nr. 4

Grafică pe Calculator

Varianta 3

A efectuat:	Student grupa TI-231 FR	Apareci Aurica
A verificat:	asistent universitar	Ursu Adriana

Chișinău

2024

Cuprins

1. Cadrul teoretic

Tema: Scene statice 3D

Scopul lucrării: Obținerea cunoștințelor practice in sinteza scenelor 3D statice, utilizând funcțiile standard de reprezentare a modelelor 3D din biblioteca p5.js

Sarcina (conform variantei): Elaborați un program care creează o scenă 3D statică conform variantei indicate: Automobil blindat

Laborator nr. 4 Varianta 3

2. Rezumat succint la temă

Formatul de fișier OBJ este unul dintre cele mai importante formate de fișiere în aplicațiile de imprimare 3D și grafică 3D. Este formatul preferat pentru imprimarea 3D multicolor și este utilizat pe scară largă ca format de schimb neutru pentru modelele 3D non-animate în aplicațiile grafice.

Un fișier OBJ este un format standard de imagine 3D care poate fi exportat și deschis de diverse programe de editare 3D. Acesta conține un obiect tridimensional, care include coordonatele 3D, hărțile de texturi, fețele poligonale și alte informații despre obiect.

Pe scurt, formatul de fișier OBJ stochează informații despre modelele 3D. Poate codifica geometria suprafeței unui model 3D și poate stoca, de asemenea, informații despre culoare și textură. Acest format nu stochează nicio informație despre scenă cum ar fi poziția luminii sau animații.

Un fișier OBJ este de obicei generat de un software CAD (Computer Aided Design) ca produs final al procesului de modelare 3D. Extensia de fișier corespunzătoare formatului de fișier OBJ este pur și simplu „.OBJ”.

În forma sa cea mai simplă, formatul de fișier OBJ permite utilizatorului să țeseleze suprafața modelului 3D cu forme geometrice simple precum triunghiuri, patrulatere sau poligoane mai complexe. Vârfurile poligoanelor și normala fiecărui poligon sunt apoi stocate într-un fișier care conține geometria suprafeței modelului.

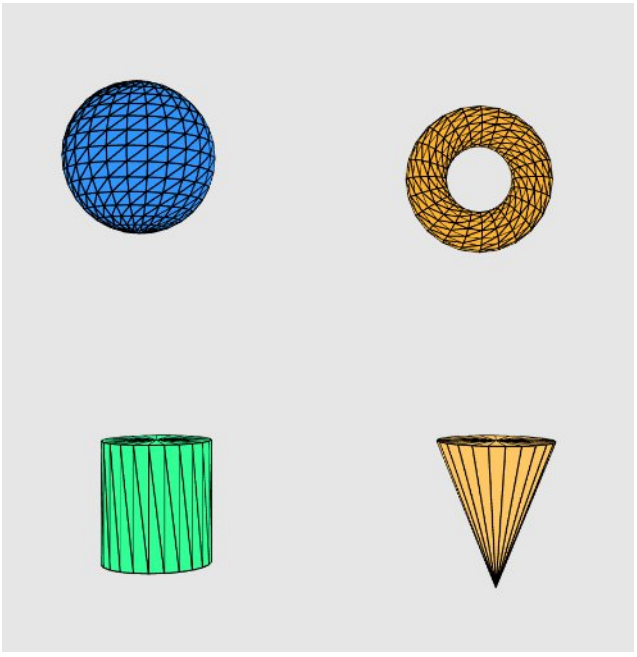
Țesălările cu fețe poligonale au o serie de avantaje și dezavantaje. Poligoanele sunt forme geometrice simple și această metodă este de fapt cel mai simplu mod de a descrie geometria suprafeței. Cu toate acestea, aproximarea unei suprafețe curbate cu poligoane duce la modificarea grosimii modelului.

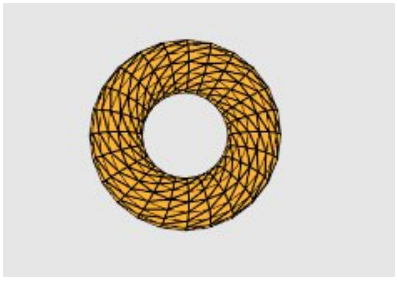
În cazul tipăririi 3D, imprimanta 3D va imprima obiectul cu aceeași grosime specificată de fișier. Desigur, făcând triunghiurile din ce în ce mai mici, aproximarea poate fi făcută din ce în ce mai exact, rezultând imprimări de o calitate mare. Cu toate acestea, pe măsură ce se micșorează dimensiunea triunghiului, crește și numărul de triunghiuri necesare pentru a descrie suprafața. Acest fapt cauzează creșterea semnificativă a dimensiunii fișierului, problemă care încearcă să o soluționeze slicer-ele de imprimare 3D. De asemenea, devine dificilă distribuția sau gestionarea unor astfel de fișiere uriașe.

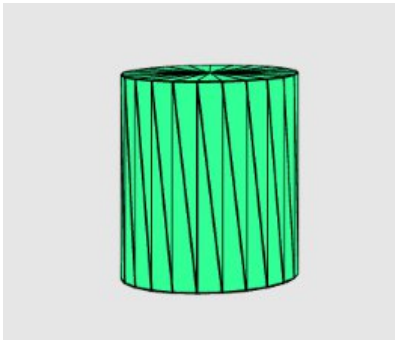
Prin urmare, este foarte important să se găsească echilibrul corect între dimensiunea fișierului și calitatea imprimării. Nu are sens să se reducă dimensiunile triunghiurilor la infinit, deoarece la un moment dat nu se va mai putea distinge cu ochiul liber diferența între calitățile tipăririi.

3. Listingul programului

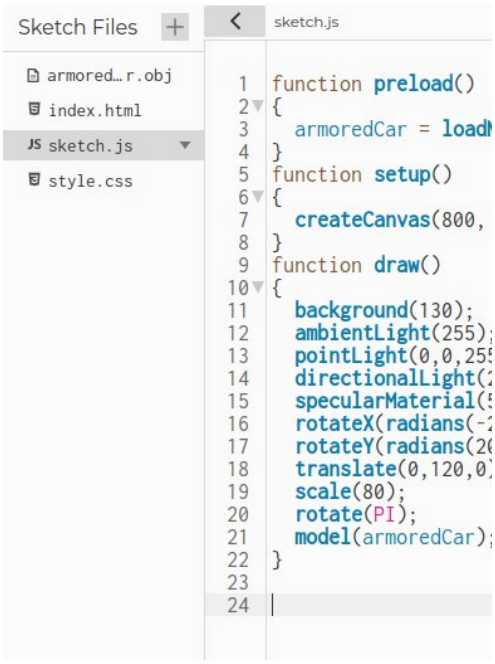
```
function setup() {
  createCanvas(400, 400, WEBGL);
}

function draw() {
  background(230);

  // Sfera
  push();
  translate(-120, -120, 50);
  fill(50, 150, 255);
  sphere(50);
  pop();

  // Tor
  push();
  translate(120, -120, -50);
  fill(255, 180, 50);
  torus(40, 15);
  pop();

  // Cilindru
  push();
  translate(-120, 120, -30);
  fill(50, 255, 150);
  cylinder(40, 90);
  pop();

  // Con
  push();
  translate(120, 120, 30);
  fill(255, 200, 100);
  cone(40, 100);
  pop();
}
```

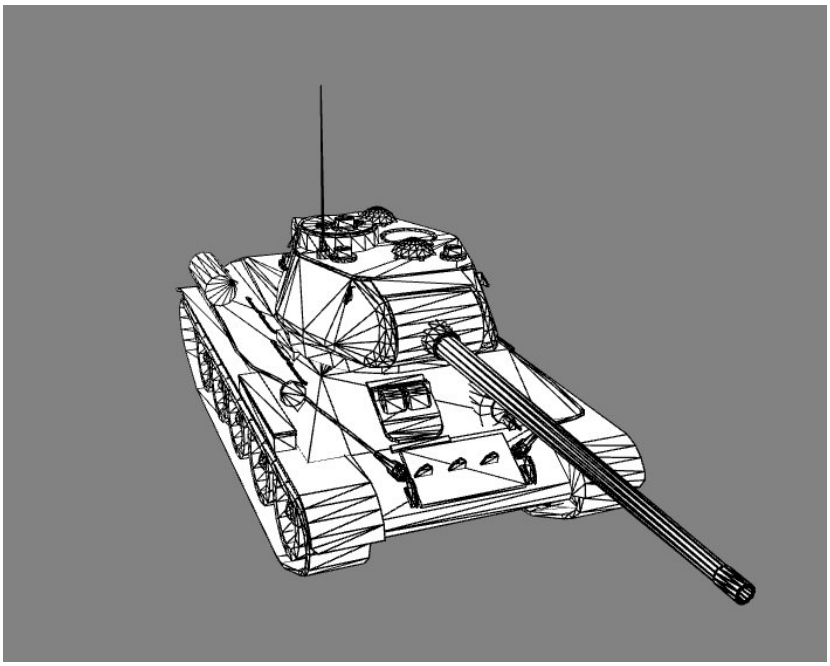






```
function preload()
{
  armoredCar = loadModel('armoredCar.obj');
}
function setup()
{
  createCanvas(800, 650, WEBGL);
}
function draw()
{
  background(130);
  ambientLight(255);
  pointLight(0,0,255,0,-200,0);
  directionalLight(255,255,255,0,0,1);
  specularMaterial(50,150,150);
  rotateX(radians(-20));
  rotateY(radians(20));
  translate(0,120,0);
  scale(80);
  rotate(PI);
  model(armoredCar);
}
```





4. Concluzii

Pentru efectuarea acestei lucrări de laborator am preluat un model 3D conform variantei primite. Din motivul ca modelul 3D era efectuat in Blender (software pentru modelare 3D si creare de scene), am redactat modelul pentru a îndeplini pe deplin sarcina din laborator, după care l-am exportat in format .obj pentru a putea fi importat cu ajutorul WEBGL si p5.js, în browser.

Un dezavantaj întâlnit este ca p5.js nu permite texturarea modelelor 3D cu ajutorul fișierelor .mtl, iar pentru a da culoare modelului suntem nevoiți sa modificam proprietățile materialelor, cât sa ajustam si luminarea scenei prin diferite procedee oferite de p5.js.