LABORATORUL 1

ELEMENTE DE GRAFICĂ PE CALCULATOR

* TEHNOLOGIA 3D –
* MAȘINA CU STĂRI FINITE -

Realizator: Alexuc Răzvan

Prof. Coordonator: Gherman Ovidiu

In sistemele informatice, tehnologia 3D reprezintă o fotografie ce oferă informații vizuale. Acest cadru este supranumit ,,realitate augmentată”, atunci când imaginile 3D sunt realizate dinamic. Imaginea grafică este proiectată cu ajutorul a două nivele distincte. Primul este însuși suportul grafic, ce poate fi: display-ul (monitorul), o simplă foaie de hârtie cu anumite inscripții sau ecranul de proiecție. Cel de-al doilea este de natură cognitivă și reprezintă întreaga ,,scenă”.

În momentul în care ne referim la un spațiu 3D, primul gând pe care îl avem este cum un obiect poate fi proiectat și vizualizat. Afișajul grafic pe monitorul calculatorului este un port prin care mașina interacționează cu utilizatorul uman, prin urmare, pentru a utiliza o asemenea tehnologie, avem nevoie de cunoștințe din domeniul geometriei. Începând de la nivelul optim de cunoștințe, cum ar fi vectorul, ce reprezintă un segment de dreaptă orientat, până la sistemul de coordonate carteziene (René ,,Cartesius” Descartes – fondatorul geometriei analitice), utilizat pentru determinarea în mod unic a unui punct situat în plan, prin prisma abscisei și ordonatei, până la sistemele de coordonate polare, 2D și 3D, conștientizăm complexitatea acestei tehnologii.

Grafica 3D presupune utilizarea operatorilor matriceali tridimensionali, cu coordonatele omogene. Transformările grafice tridimensionale sunt:

* Translarea 3D (mișcarea unui obiect dintr-o poziție în alta);
* Scalarea 3D (modificarea sau distorsionarea dimensiunii unui obiect – poate fi locală sau globală);
* Rotația 3D (rotirea obiectului către un unghi – poate fi făcută în jurul unei axe oarecare);
* Forfecarea 3D (înclinarea unui obiect în spațiul 3D către direcția x, y sau z – schimbă forma obiectului);
* Oglindirea 3D (realizată față de un punct, față de o dreaptă sau față de un plan).

Acestea fiind spuse, tot ceea ce vizualizăm pe ecranul computer-ului nostru este o matrice de pixeli. Fiecare pixel de pe ecranul nostru are o culoare, iar practic, grafica oferă o cale pentru a determina ce culoare să afișeze pixelii respectivi. Grafica 3D este un sistem de producere a culorilor pentru pixeli pentru a ne convinge că scena pe care o vizualizăm este amplasată intr-un spațiu 3D, ci nu unul 2D. Procesul de conversie a lumii 3D într-o imagine 2D este numit *rendering.*

Sunt diverse metode pentru a reda lumea 3D. Procesul utilizat de hardware-ul grafic care este stocat în computer-ul nostru implică multă adaptare. Acest process este numit rasterizare, iar un sistem de redare ce utilizează rasterizarea este numită *rasterizer*.

Un obiect este realizat dintr-o serie de triunghiuri adiacente ce sunt definite pe suprafața exterioară a obiectului. Asemenea triungfhiuri sunt adesea denumite: *geometry*, *model* sau *mesh*.

Fiecare pixel dintr-o imagine 2D are o culoare particulară. Descrierea unei culori este o serie de numere. Fiecare număr corespunde la intensitatea unei culori particulare. Setul de referință al culorilor poartă denumirea de *colorspace*. Cel mai comun color space pentru ecrane este RGB. Pe 24-bit (numit și True Color), monitoarele pot afișa până la 16.777.215 de diferite combinații de culori. Pe 32-bit, suportă de asemenea până la 16.777.215 culori, dar au un canal alfa, deci colorspace-ul devine RGBA( Red – Green – Blue – Alpha). Ulterior, descoperim și *shader*-ul, ce este destinate pentru a fi rulat pe un renderer ca parte a operațiunii de randare.

OpenGL (Open Graphics Library) reprezintă un API (Application Programming Interface) graphic, lansat de Silicon Graphics în 1992 pentru a accesa rasterizer-ul ce este bazat pe hardware. Se conformează cu modelul pentru randarea 3D bazată pe rasterizare. Rasterizatorul recepționează o secvență de triunghiuri de la user, realizează operațiuni pe ele, scrie pixeli bazate pe datele triunghiului. API-ul utilizează ca limbaj principal C, definit de un număr de typedef-uri și funcții. Typedef-urile definesc GL-uri basic, precum Glint, Glfloat.

Pentru a randa un obiect cu succes, utilizând tehnologia OpenGL, trebuie să ținem cont de un pipeline, ce ne oferă o secvență de pași:

* Vertex Processing (fiecare vertex preluat din matrice este supus Vertex Shader-ului);
* Vertex Post-Processing (Transform Feedback-ul se întâmplă aici);
* Interpolarea parametrilor primitive, ce generează un număr de fragmente;
* Fragment Shader;
* Per-Sample Processing.

Cea mai nouă versiune de tip ,,stable” este 4.6, lansată în 2017, însă își propune să devină obsolete în câțiva ani, fiind înlocuită cu cea mai nouă tehnologie, numită Vulkan, lansată în 2016, ajungând până la versiunea 1.3. Există un tool numit OpenGL Extensions Viewer ce ne permite să vizualizăm versiunea actuală de OpenGL instalată și furnizează tool-uri pentru a testa sau actualiza driver-ul plăcii video.

Un competitor pentru OpenGL poate fi considerat Direct3D-ul (ca parte din DirectX). Fahrenheit a reprezentat o încercare de a crea un API high-level dezvoltat de către Microsoft și Silicon Graphics pentru a fuziona OpenGL și Direct3D în anii ’90, dar a fost anulat.

O mașină cu stări finite (FSM), denumit și automat cu stări finite poate fi definită ca fiind un model matematic de calcul. Permite descrierea cu precizie și într-un mod formal a comportamentului multor sisteme. Este popular în inginerie și știință.

Un FSM modelează o mașină ipotetică ce posedă un număr finit de stări. Ceea ce o caracterizează este că numai una dintre stări poate fi activă în oricare moment al timpului. Deci, pentru a respecta și executa acțiunile pentru care a fost realizată, este necesar ca starea ei activă să se schimbe, în funcție de anumite condiții prestabilite. Automatele finite pot recepționa intrări și pot furniza la ieșire diverse informații. Din punct de vedere al condițiilor pentru schimbarea de stare, avem în vedere două categorii:

1. Automate Mealy;
2. Automate Moore;

La automatele Mealy, remarcăm că ieșirea depinde de starea curentă și de input-ul curent.

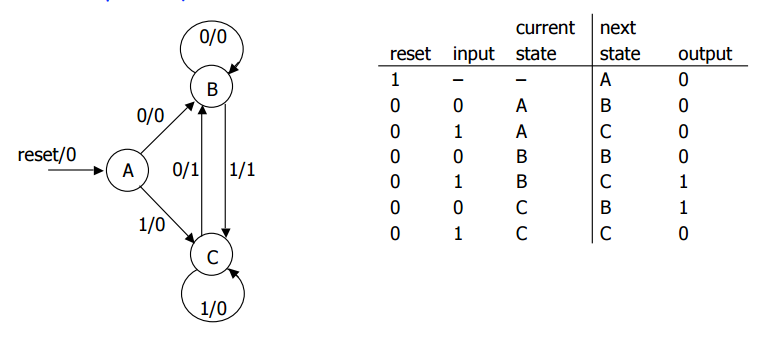


Figura 1. Diagrama pentru automatul Mealy

Pentru a trece din punctul A în punctul C, se citește astfel: din A, cu input-ul 1, se trece în C, cu output-ul 0.

La automatele Moore, output-ul va fi determinat strict de starea în care se află:

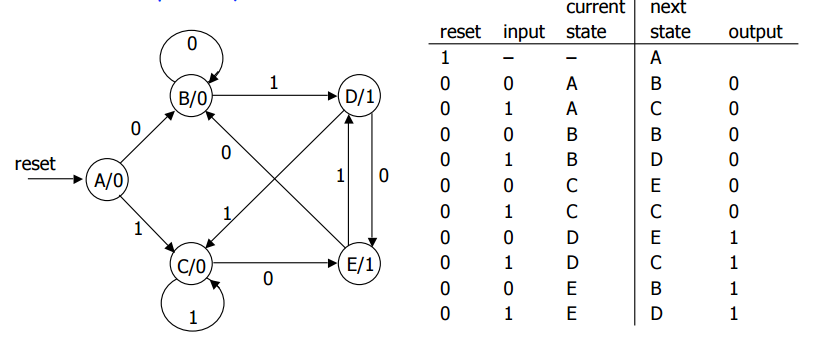


Figura 2. Diagrama pentru automatul Moore.

Cu toate acestea, există pentru fiecare mașină Mealy o mașină Moore echivalentă și invers.