Laborator 01

Cuprins

1. Obiective 2
2. Protecția muncii 2
3. Regulamentul EGC 2
4. Introducere în OpenGL 2
4.1 Elemente generale 2
4.2 Materiale multimedia introductive 3
4.3 Pipeline-ul de execuție 4
4.4 Materiale multimedia introductive 6
4.5 Tutori OpenGL 6
5. Probleme de rezolvat 8

1. Obiective

- Asimilarea elementelor specifice de protecția muncii.
- Luarea la cunoștință a regulamentului.
- Introducere în OpenGL. Elemente generale.

2. Protecția muncii

Citiți și asimilați normele specifice de securitatea și protecția muncii care se regăsesc atașate acestui laborator.

3. Regulamentul EGC

Citiți și asimilați regulamentul conform căruia se vor desfășura laboratoarele la disciplina EGC.

4. Introducere în OpenGL

4.1 Elemente generale

OpenGL[1] nu este un limbaj de programare, ci un API de redare grafică, ce prezinta programatorului un set de funcții specifice care permite utilizarea componentelor hardware ale unui PC (în speță placa grafică) pentru a genera scene 2D/3D complexe și a le desena pe ecran.

Un program care utilizează acest API grafic va fi scris într-un limbaj de programare precum C/C++, C#, Java, Python, etc. care va invoca funcții specifice din biblioteca OpenGL, fie direct fie prin intermediul unui *wrapper* care translatează apeluri de funcții *high-level* la apeluri directe *low-level*. OpenGL are un sistem propriu (nativ) de creare a ferestrelor.

OpenGL este standard liber, *open-source*, *cross-platform*, administrat de Grupul Khronos[2]. Implementări OpenGL:

- OpenGL standard, ajuns la versiunea 4.6.
- OpenGL ES, ajuns la versiunea 3.2, o implementare pentru dispozitive mobile (telefoane, tablete, dispozitive *embedded*, etc.). Extinde OpenGL 3.x.
- WebGL, ajuns la versiunea 2.0, o implementare *cross-platform* pentru web (API 3D *low-level* pentru browser-e). Extinde OpenGL ES 2.0.

[1] https://www.opengl.org/
[2] https://www.khronos.org/

- Vulkan, ajuns la versiunea 1.1, o implementare simplificată, de înaltă performanță, dedicat plăcilor grafice (GPU) de generații moderne. Este următorul pas în dezvoltarea OpenGL (deși acesta este încă în uz și activ dezvoltat).

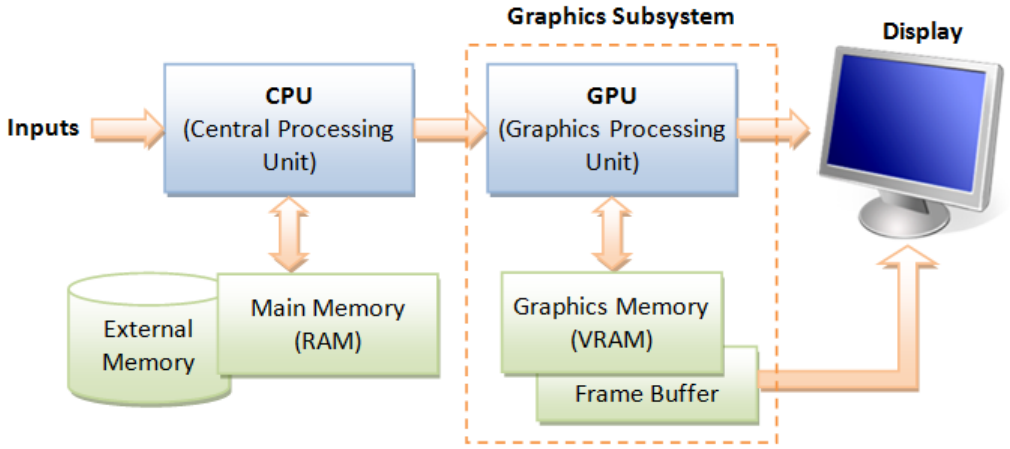

Fig. 1. Utilizarea OpenGL – schemă de principiu.

Există și o implementare non-standard, incompatibilă cu versiunea de bază OpenGL, dedicată dispozitivelor de calcul de la Apple (OSX/macOS/iOS) numită Metal, optimizată pentru platformele specifice Apple.

Microsoft a furnizat o alternativă la OpenGL numită DirectX (ajuns la versiunea 12). Structura celor 2 API concurente este similară, la fel ca și capabilitățile oferite – excepția fiind că DirectX este oferit doar pe sisteme PC cu Windows (Xbox inclusiv), pe când OpenGL este un produs *cross-platform*. De asemenea, flexibilitatea și versatilitatea OpenGL-ului a dus la dezvoltarea a numeroase *binding*-uri care permit utilizarea diverselor limbaje de programare (în afară de C – limbajul primar folosit în dezvoltarea OpenGL) pentru scrierea de programe grafice.

Producătorii de plăci grafice (nVidia și AMD în principal) furnizează seturi de extensii (*vendor extensions*) care implementează anumite caracteristici specifice respectivelor plăci grafice, permițând programatorilor care se folosesc de acele extensii să aibă acces la noile caracteristice ale echipamentelor grafice de generație nouă.

Pe scurt: OpenGL randează triunghiuri. 

4.2 Materiale multimedia introductive

Urmăriți/studiați materialele multimedia informative de mai jos:

- Introducere OpenGL (elemente fundamentale):
  - ✓ detalii în documentul atașat laboratorului (***Joey de Vries - Learn OpenGL. An Offline Transcript of learnopengl.com.pdf***), pag. 18-21.
  - ✓ detalii în documentul atașat laboratorului (***Jason L. McKesson - Learning Modern 3D Graphics Programming.pdf***), pag. 11-20.
  - ✓ versiunea scurtă: https://www.youtube.com/watch?v=IXxc9yNBpuo

- ✓ versiunea lungă: https://www.youtube.com/watch?v=6-9XFm7XAT8

- Introducere în OpenGL-ul modern:
  - ✓ https://glumpy.github.io/modern-gl.html
- Ce este Vulkan:
  - ✓ detalii în documentul atașat laboratorului (***Khronos Group - Ecosistemul OpenGL, SIGGRAPH 2014.pdf***);
  - ✓ https://www.youtube.com/watch?v=wWYRFwIHdJc

4.3 Pipeline-ul de execuție

Execuția unui set de comenzi adresate subsistemului grafic se face printr-o serie de etape predefinite care împreună formează așa-numita *pipeline* de execuție. Aceasta asigură generarea scenei 2D/3D pornind de la setul de comenzi dat de programator până generarea efectivă pe ecran a scenei.

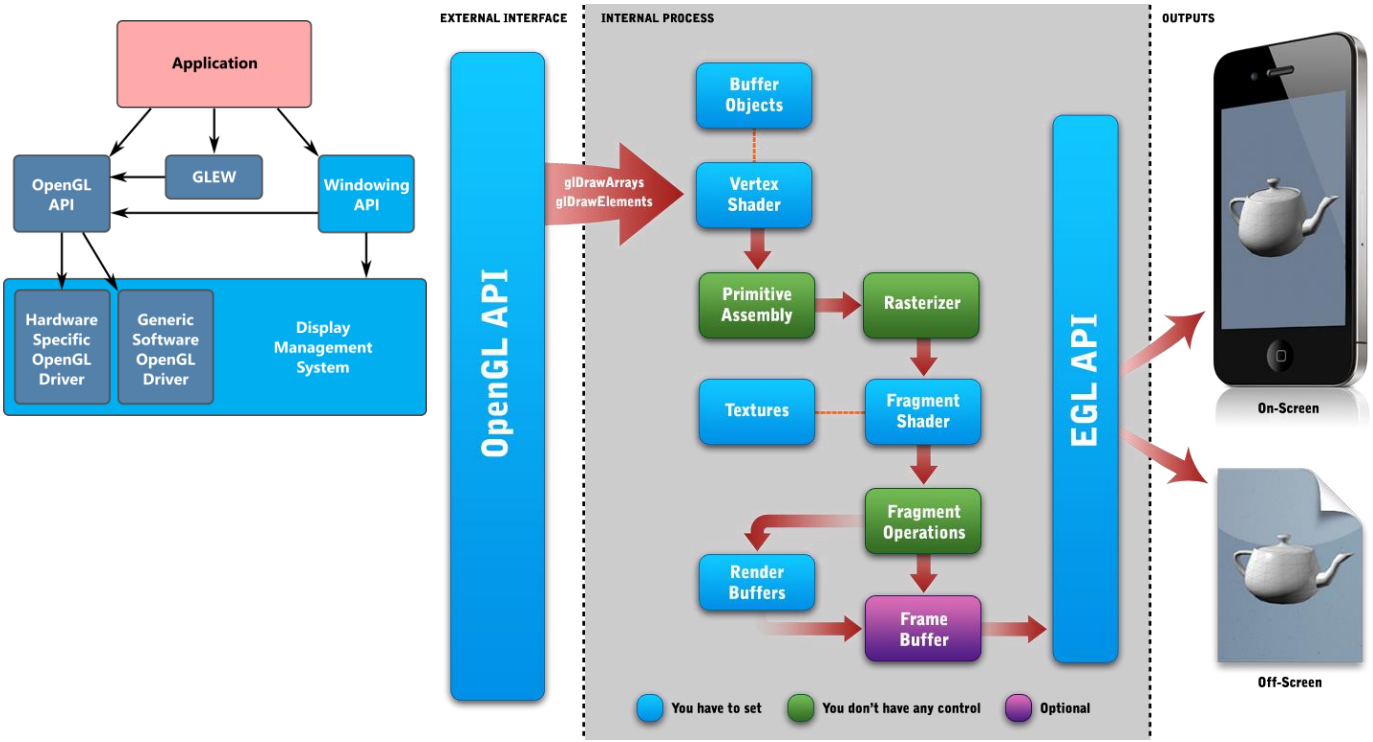

Fig. 2. Modul de lucru intern a API-ului OpenGL (organizare software și schemă-bloc).

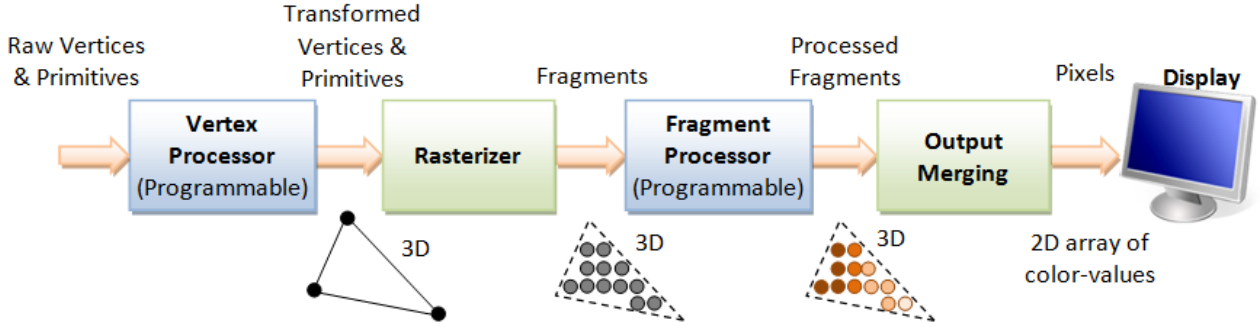

Fig. 3. Modul de lucru abstract al API-ului OpenGL.

Până la versiunea OpenGL 3.1, acest *pipeline* era fix, programatorul având grijă să descrie fiecare element al scenei grafice prin intermediul funcțiilor predefinite, lucru limitator din punct de vedere a ceea ce poate fi implementat cu ajutorul acestui API (funcțiile posibile fiind predefinite). Începând cu versiunea 3.1, *pipeline*-ul de execuție implementat este unul programabil, mai eficient, care utilizează *rastere* și *shadere* programabile de către dezvoltator.

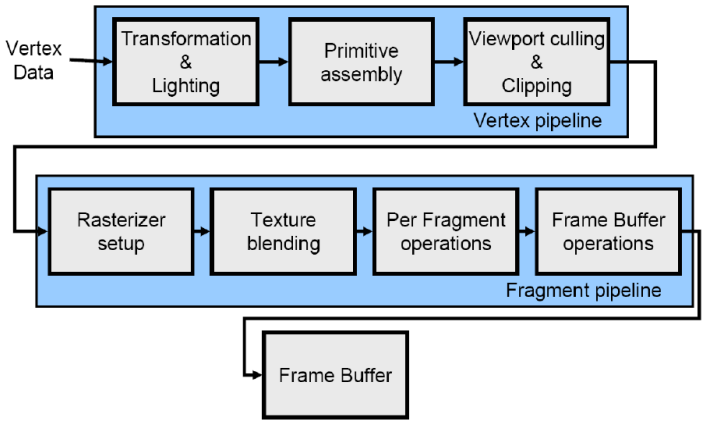

Fig. 4. OpenGL – Pipeline de execuție fix.

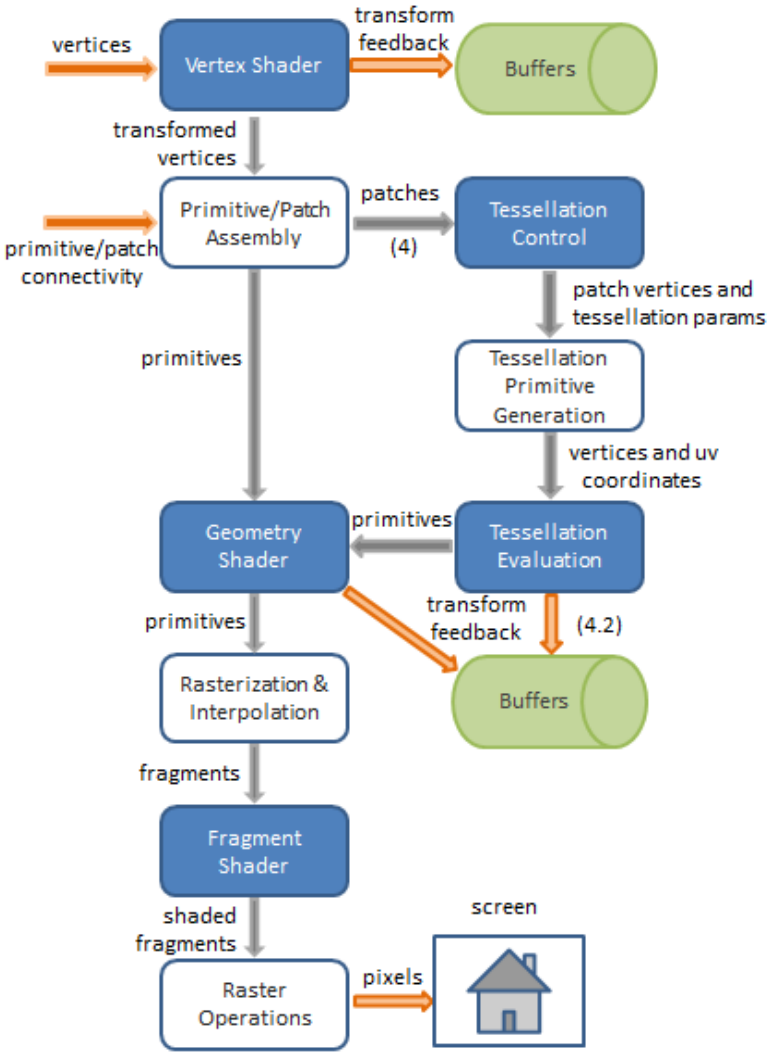

Fig. 5. OpenGL – Pipeline de execuție programabil.

4.4 Materiale multimedia introductive

- *Pipeline* de execuție:
  - ✓ introducere (1): https://www.youtube.com/watch?v=bgckX62f4EA&list=PLEETnX-uPtBXT9T-hD0Bj31DSnwio-ywh&index=6
  - ✓ introducere (2): https://www.youtube.com/watch?v=FeUr-S1f7Qk
  - ✓ introducere (3): https://www.youtube.com/watch?v=cvcAjgMUPUA
  - ✓ elemente avansate (1): https://www.youtube.com/watch?v=0PTBOX1HHIo
  - ✓ elemente avansate (2): https://www.youtube.com/watch?v=qHpKfrkpt4c
  - ✓ elemente avansate (3): https://www.youtube.com/watch?v=98SSgxDe-5A
  - ✓ Vulkan: https://www.youtube.com/watch?v=Apjnf4KhgFI
- Transformări OpenGL (translări, rotații, scalări/deformări):
  - ✓ (ignorați partea de cod): https://www.youtube.com/watch?v=u-LoUvilIIQ

4.5 Tutori OpenGL

OpenGL poate fi considerat ca fiind o mașină cu stări finite. Acest lucru înseamnă că putem modela OpenGL ca o structură compusă din stări și acțiuni discrete, iar schimbările de stare de la un moment dat *t* reflectă schimbările produse de intrări de la inițializarea sistemului si până la momentul *t*, în baza stării inițiale (*default*) a sistemului. Orice acțiune efectuată de dezvoltator se realizează în pași discreți, scena 3D rezultată descriind corespunzător setul de instrucțiuni.

Funcțiile OpenGL sunt de 2 tipuri: funcții generatoare de primitive geometrice (generează output vizual, starea scenei fiind controlată de starea mașinii) și funcții de schimbare de stare (transformări, modificarea atributelor).

Pentru o mai bună înțelegere a conceptelor specifice OpenGL, urmăriți următorii tutori OpenGL:

- Nate Robin's Tutors: https://user.xmission.com/~nate/tutors.html . Veți găsi atașat laboratorului codul-sursă și binarele pentru execuția tutorilor.

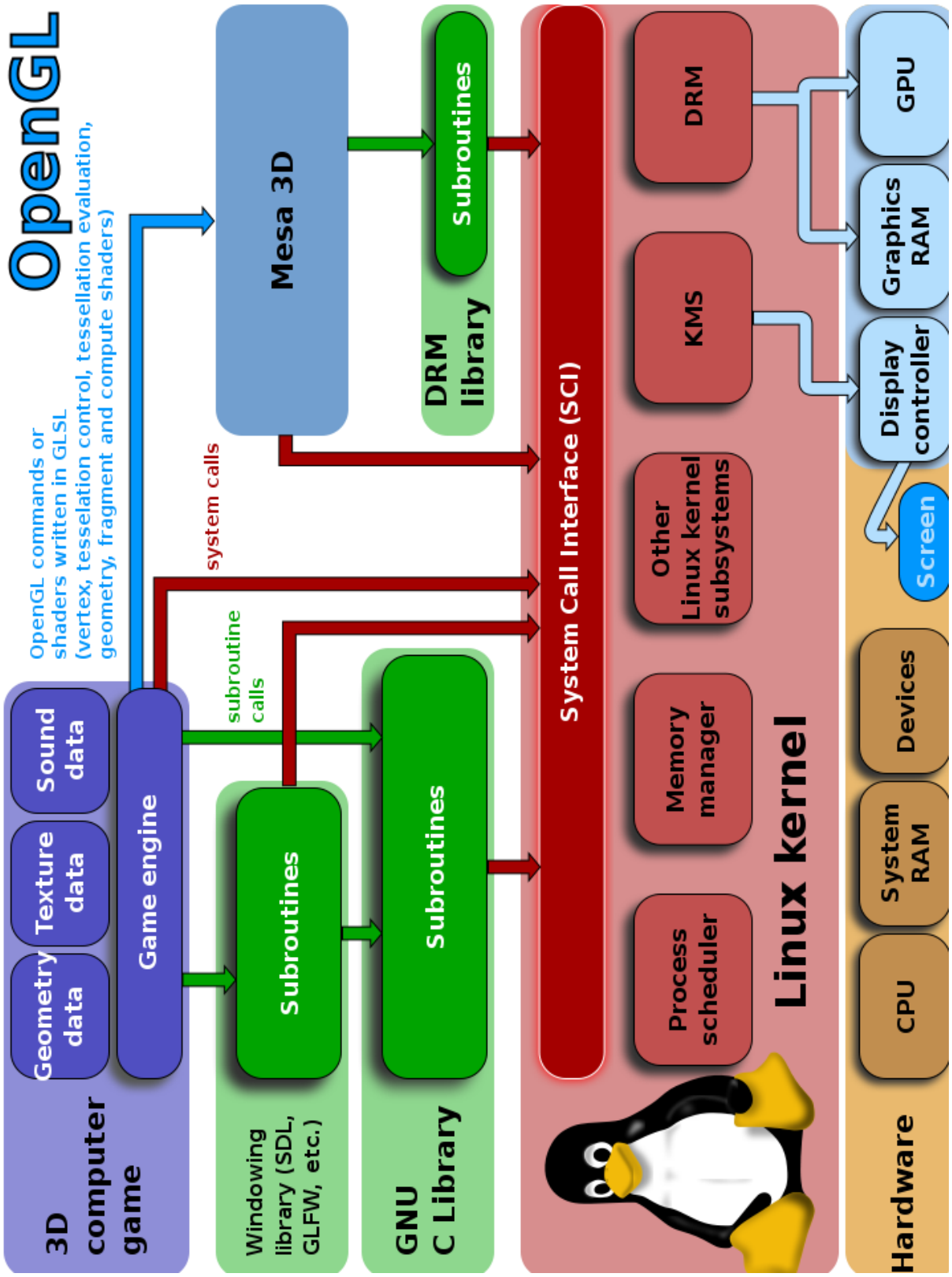

Fig. 6. Aplicație grafică ce utilizează API-ul OpenGL pentru a comunica cu subsistemul grafic (GPU).

5. Probleme de rezolvat

1. Creați un referat cu privire la tehnologia OpenGL (și cele derivate din aceasta) în care să prezentați opiniile proprii cu privire la acestea, precum și punctele slabe și punctele tari ale acestor tehnologii. Cum explicați modelul de automat cu stări finite al OpenGL și cum afectează acest lucru procesul de randare al scenei 3D de către biblioteca grafică/API?
2. Încărcați referatul pe formularul online a cărui link se regăsește în pagina materialului de laborator. Respectați data-limită pentru încărcarea temei.