Documentatie
Airplane fighter

1. Introducere

Proiectul de față constă în realizarea unui joc 2D utilizând limbajul de programare **C** împreună cu biblioteca grafică **OpenGL**, integrată în mediu de dezvoltare **Eclipse IDE**. Jocul simulează o scenă simplă în care utilizatorul controlează un avion cu ajutorul tastelor săgeată stânga (←) și dreapta (→), scopul fiind evitarea unor obstacole care cad din partea superioară a ecranului, reprezentate sub forma unor **meteoriți**.

Mecanica jocului este una intuitivă: pe axa verticală (Y), în mod continuu, coboară meteoriți, iar jucătorul trebuie să manevreze avionul pe axa orizontală (X) pentru a evita coliziunile. Pentru fiecare meteorit evitat, se acumulează puncte, iar scorul este **salvat automat într-un fișier text**, împreună cu numele utilizatorului curent.

Înainte de a începe jocul propriu-zis, utilizatorul este nevoit să treacă printr-un **meniu de autentificare** afișat în consolă. Aici, are opțiunea de a alege între:

- l – pentru **login** (autentificare),
- s – pentru **signup** (creare cont nou).

Această funcționalitate permite fiecărui utilizator să aibă un profil unic, iar scorul său să fie salvat și diferențiat în fișierul corespunzător. Scopul principal al proiectului este de a integra noțiuni de grafică 2D în C folosind OpenGL, împreună cu concepte de bază din programarea procedurală, gestionarea fișierelor și interacțiunea cu utilizatorul prin tastatură.

2. Sistemul de autentificare

Pentru a asocia scorurile fiecărui utilizator, proiectul include un **mecanism de autentificare**, implementat sub forma unei interfețe textuale în consolă. Acesta este accesat **imediat după lansarea aplicației**, fiind prima funcție apelată din main() – și anume account_menu().

2.1. Meniul de autentificare – account_menu()

Funcția account_menu() are rolul de a interacționa cu utilizatorul pentru a-i permite să aleagă dacă dorește să se logheze (l) sau să-și creeze un cont nou (s). Această alegere determină fluxul următor:

- Opțiunea l declanșează procesul de **autentificare** prin login_menu().
- Opțiunea s declanșează procesul de **înregistrare** prin sign_up_menu().

Orice altă opțiune este considerată invalidă, iar utilizatorul este invitat să reîncerce.

Documentatie
Airplane fighter

2.2. Autentificarea utilizatorului – login_menu() și login_user()

Funcția login_menu() solicită introducerea unui **nume de utilizator** și a unei **parole**, apelând apoi login_user() pentru a verifica dacă datele introduse există în fișierul players_db.txt.

login_user() deschide fișierul în mod citire ("r") și caută o potrivire exactă între credențialele introduse și cele salvate. Dacă o potrivire este găsită:

- Se identifică poziția utilizatorului (player_id) în vectorul players[].
- Se incrementează automat numărul de jocuri jucate (nr_of_games).
- Se permite accesul în joc, returnând 1.

Dacă utilizatorul nu este găsit, funcția returnează 2, semnalând o eroare de autentificare.

2.3. Crearea unui cont nou – sign_up_menu() și sign_up_user()

Pentru utilizatorii noi, opțiunea s din meniu declanșează sign_up_menu(), care colectează un nume de utilizator și o parolă, verificând ulterior disponibilitatea acestora prin sign_up_user().

Dacă username-ul introdus nu este deja înregistrat:

- Se adaugă un nou element în vectorul players[].
- Se creează sau actualizează fișierul players_db.txt, scriind toate înregistrările existente plus cea nouă.
- Noul utilizator este considerat activ (player_id este setat), iar înregistrarea are succes (returnând 2).

Dacă username-ul este deja folosit, se returnează 1, iar utilizatorul este informat să încerce alt nume.

2.4. Structura fișierului de date – players_db.txt

Fișierul players_db.txt stochează informațiile despre utilizatori într-un format simplu:

username password nr_of_games score

Exemplu:

Documentatie
Airplane fighter

Ovidiu o_parola 3 150

Dumitru alta_parola 5 320

Fiecare linie este citită la pornirea jocului și rescrisă complet după fiecare sesiune pentru a asigura actualizarea datelor.

2.5. Avantaje și extensii posibile

Sistemul de autentificare aduce următoarele beneficii:

- Permite **salvarea scorului individual** pentru fiecare utilizator.
- Oferă **persistența datelor** între sesiunile de joc.
- Poate fi extins cu funcționalități precum criptarea parolelor sau autentificare grafică.

3. Inițializarea aplicației și ciclul principal de joc

3.1 Inițializarea graficii

După autentificare, aplicația pornește sistemul grafic OpenGL. Prin funcțiile glutInit, glutInitWindowSize și glutCreateWindow, se creează fereastra principală. Aceasta este centrată pe ecran, iar dimensiunea este adaptată înălțimii monitorului.

Funcția glutDisplayFunc(display) setează funcția principală de redare grafică a jocului. În această funcție (display()), logica de joc este împărțită în trei stări:

- **Start joc** – utilizatorul vede un buton de start;
- **Joc activ** – se desenează avionul, meteoriții și scorul;
- **Joc pierdut (restart)** – se afișează butonul de restart.

3.2 Gestionarea tastelor

Avionul este controlat folosind tastele săgeată stânga (←) și dreapta (→). Acestea sunt gestionate prin glutSpecialFunc() și glutSpecialUpFunc():

- specialKeyDown() setează steaguri booleene când o tastă este apăsată;
- specialKeyUp() le resetează când tasta este eliberată.

Documentatie
Airplane fighter

Mișcarea avionului se face în update_plane_position() prin modificarea coordonatei plane_x, ținând cont de timpul scurs (delta_time) și de marginile ecranului.

4. Mecanica jocului și scorul

4.1 Meteoriții și animația

Meteoriții sunt generați aleator la început prin funcția create_meteorites() și salvați într-un vector global meteorites[10]. Fiecare are o poziție (x, y), o rază (r) și o poziție de început (begin).

Meteoriții sunt redesenați constant în funcția animate_meteorite(), care se bazează pe timpul scurs pentru a determina viteza de coborâre. Dacă un meteorit iese din ecran, este resetat la o poziție nouă, generată aleatoriu.

Pentru fiecare meteorit evitat, se incrementează variabila current_score.

4.2 Coliziunea

Funcția colizion() verifică dacă avionul s-a intersectat cu un meteorit, folosind distanța dintre centrul avionului și cel al meteoritului. Dacă distanța este mai mică decât suma razelor, se consideră coliziune:

```
if (distance < meteorites->meteorite_r + 0.06) {
   my_plane->plane_red = 1.0;
   my_plane->plane_green = 0;
   restart_game = true;
}
```

Aceasta schimbă culoarea avionului (semnal vizual pentru pierdere) și oprește jocul.

4.3 Afișarea scorului

Scorul este afișat în colțul ferestrei folosind glRasterPos2f și glutBitmapCharacter(). Se afișează:

- Scorul curent,
- Scorul total (cumulat),
- Numărul de jocuri jucate.

Actualizarea scorului global (salvat în fișier) se face la fiecare secundă prin update_db(), pentru a asigura persistența datelor și corectitudinea statisticilor utilizatorului.

5. Bibliografie

1. **Documentația oficială OpenGL**. Disponibilă online la: https://www.opengl.org/documentation/
2. **Infopuc.ro** – Platformă educațională informatică. Disponibil online la: https://www.infopuc.ro
3. **Browser ChatBot** – Răspunsuri și explicații obținute la intrebarile mele in motorul de cautare;