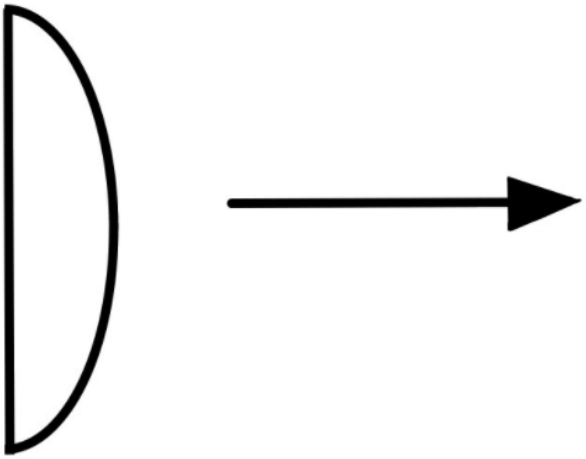
Tema 1 - Bow and Arrow

In cadrul temei 1 se va implementa o versiune simplificata a jocului Bow and Arrow. Tot ce nu este specificat ramane la creativitatea voastra.

Jucatorul

Jucatorul va fi reprezentat doar prin arc si sageata. Sageata va fi creata prin combinarea a minim 2 primitive geometrice(ex: triunghi, dreptunghi sau linie). Arcul va fi alcatuit dintr-o linie verticala si un arc de cerc.

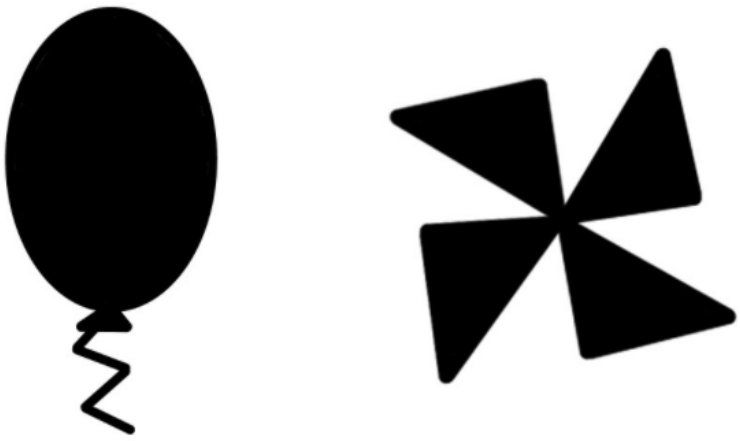


Tinte

Obiectele in care jucatorul trebuie sa traga cu sageata vor fi de 2 feluri: baloane si shuriken. Baloanele vor fi create prin combinarea a minim 2 primitive(ex: un cerc scalat diferit pe axele x si y si o polilinie). Acestea se vor deplasa ascendent pe axa OY(vor aparea in partea de jos a scenei si vor urca). Baloanele vor fi de doua culori diferite: baloanele rosii- vor mari scorul, baloanele galbene- vor micsora scorul la coliziunea cu sageata.

Stelutele shuriken vor fi create prin combinarea a 4 triunghiuri. Ele se vor deplasa pe axa OX(vor aparea din partea dreapta a scenei si se vor indrepta catre jucator). Stelutele vor fi animate - se vor roti in jurul centrului pe tot parcursul deplasarii.

Obiectele trebuie definite ca in laboratorul 2. Nu se accepta incarcarea de obiecte 3D!



Gameplay

Arcul jucatorului se afla in partea stanga a scenei. Arcul se poate deplasa doar pe axa OY, prin apasarea tastelor W, S. Directia de tragere a arcului se poate ajusta prin miscarea mouse-ului. Arcul va fi mereu rotit astfel incat sageata se va deplasa catre pozitia curenta a mouse-ului. Jucatorul va putea trage o sageata la cateva secunde. Viteza de deplasare a sagetii este determinata de timpul de apasare al

butonului mouse-ului. Cu cat butonul este apasat mai mult timp, cu atat viteza va fi mai mare, pana la o limita setata.

Jucatorul va avea 3 vieti. La coliziunea dintre arc si un shuriken jucatorul va pierde o viata. La pierderea tuturor vietilor, jocul se va termina.

Scorul va creste la coliziunea sagetii cu baloanele rosii sau shuriken. Scorul se va afisa periodic in consola sau se va ilustra pe ecran sub forma unui dreptunghi scalat (scorebar).

Detalii de implementare

- Sagetile pot fi lansate doar catre partea dreapta a scenei.In momentul coliziunii sagetii cu un obiect se va declansa o animatie de distrugere a obiectului(ex: balonul va fi scalat cu factor subunitar pana dispare de tot).
- Afisarea puterii de tragere se va realiza astfel: un dreptunghi amplasat sub jucator, care va fi scalat pe axa OX cat timp butonul mouse-ului este tinut apasat.
- Coliziunea dintre sageata si obiectele tinta se va implementa prin testarea daca varful sagetii este inclus in cercul incadrator al fiecarui obiect tinta.
- Coliziunea dintre shuriken si arcul jucatorului se va implementa prin testarea intersectiei cercurilor incadratoare ale obiectelor.
- Crearea mediului inconjurator trebuie sa se realizeze in asa fel incat consumul de memorie si timpul de redare sa fie optime. Nu creati incontinuu obiecte care sa reprezinte obiectele tinta si sagetile care apar si dispar din spatiul de desenare!!!! O implementare eleganta este sa creati un singur obiect in functia init(), iar in functia Update() sa dati comanda de desenare pentru acel obiect de mai multe ori, de fiecare data la alta pozitie si cu alt factor de scalare.
- Toate animatiile trebuie sa fie independente de timpul de procesare al unui cadru.

Mai multe informatii despre ce reprezinta coliziunea si cum se poate implementa:

- https://developer.mozilla.org/en-US/docs/Games/Techniques/2D_collision_detection [https://developer.mozilla.org/en-US/docs/Games/Techniques/2D_collision_detection]
- https://learnopengl.com/In-Practice/2D-Game/Collisions/Collision-detection [https://learnopengl.com/In-Practice/2D-Game/Collisions/Collision-detection]
- https://www.youtube.com/watch?v=aTbw71EpamY [https://www.youtube.com/watch?v=aTbw71EpamY]
- https://developer.mozilla.org/en-US/docs/Games/Techniques/3D_collision_detection [https://developer.mozilla.org/en-US/docs/Games/Techniques/3D_collision_detection]

Mentiune: in cazul acestei teme, coliziunile sunt mult mai simple (punct-cerc si cerc-cerc).

Bonus

- Scena cat mai realista (adaugarea unui personaj in scena, etc)
- Cresterea dificultatii jocului (viteza de deplasare a obiectelor tinta, numarul obiectelor tinta, etc)
- Animatii (ex: baloanele sa se deplaseze pe un traseu serpuitor; la coliziunea dintre sageata si balon, balonul sa fie deformat realist si sa cada conform gravitatiei)
- Sageata sa tina cont de gravitatie (aruncarea sub un unghi inclinat)
- Diverse alte obiecte tinta (exemple in video)

Exemplu (doar pentru orientare): https://youtu.be/dcv-XKmUPjo [https://youtu.be/dcv-XKmUPjo]

Functionalitati obligatorii

Barem orientativ pentru realizarea functionalitatilor (din 150 puncte):

- Desen scena - 40 p (10p sageata, 10p arcul, 10p baloane, 10p shuriken)
- Animatie obiecte tinta - 10p

- Stabilirea pozitiei arcului cu tastele - 10p
- Stabilirea directiei de tragere si deplasare a sagetii - 25p (15p directie, 10p deplasare)
- Coliziunea sageata-obiecte tinta - 15p
- Coliziunea shuriken-personaj - 15p
- Sistemul de tragere(viteza sageata + power bar) - 15p
- Animatie tinte doborate - 10p
- Actualizare scor - 10p

Intrebari si raspunsuri

Pentru intrebari vom folosi forumurile de pe moodle.

Notare

Baremul este orientativ. Fiecare asistent are o anumita libertate in evaluarea temelor (de exemplu, sa dea punctaj partial pentru implementarea incompleta a unei functionalitati sau sa scada pentru hard coding). Acelasi lucru este valabil atat pentru functionalitatile obligatorii, cat si pentru bonusuri.

Tema trebuie incarcata pe moodle. Pentru a fi punctata, tema trebuie prezentata la laborator. Vor exista laboratoare speciale de prezentare a temelor (care vor fi anuntate).

Indicatii suplimentare

Tema va fi implementata in OpenGL si C++. Este indicat sa folositi framework-ul si Visual Studio.

Pentru implementarea temei, in folderul Source/Laboratoare/ puteti crea un nou folder, de exemplu Tema1, cu fisierele Tema1.cpp si Tema1.h (pentru implementare POO, este indicat sa aveti si alte fisiere). Pentru a vedea fisierele nou create in Visual Studio in Solution Explorer, apasati click dreapta pe filtrul Laboratoare si selectati Add→New Filter. Dupa ce creati un nou filtru, de exemplu Tema1, dati click dreapta si selectati Add→Existing Item. Astfel adaugati toate fisierele din folderul nou creat. In fisierul LabList.h trebuie adaugata si calea catre header-ul temei. De exemplu: #include <Laboratoare/Tema1/Tema1.h>

Arhivarea proiectului

- in mod normal arhiva trebuie sa contina toate resursele necesare compilarii si rularii
- inainte de a face arhiva asigurati-va ca ati dat clean la proiect
  - click dreapta pe proiect in **Solution Explorer** → **Clean Solution**, sau
  - stergeti folderul **/Visual Studio/obj**
- stergeti fisierul **/Visual Studio/Framework_EGC.sdf** (in caz ca exista)
- stergeti fisierul **/Visual Studio/Framework_EGC.VC.db** (in caz ca exista)
- stergeti folderul **/Visual Studio/.vs** (daca nu il vedeti, **este posibil sa fie ascuns**)
- stergeti folderul **/x64** sau **/x86** (in caz ca exista)
  - executabilul final este generat in folderul **/x86** sau **/x64** la finalul link-editarii in functie de arhitectura aleasa la compilare (32/64 biti)
- in cazul in care arhiva tot depaseste limita de 20MB (nu ar trebui), puteti sa stergeti si folderul **/libs** sau **/Resources** intrucat se pot adauga la testare. Nu este recomandat sa faceti acest lucru intrucat ingreuneaza mult testarea in cazul in care versiunea curenta a librariilor/resurselor difera de versiunea utilizata la momentul scrierii temei.

Deadline tema

17 noiembrie ora 23:59

Responsabili tema

- Stefania Cristea

- Andrei Lapusteanu
- Anca Morar

egc/teme/2020/01.txt · Last modified: 2020/11/17 19:31 by florin_eugen.iancu