Exercici lliurable 2 de laboratori d'IDI 2023–2024 Q2

Instruccions

1. Aquests exercicis són individuals, així que només pots lliurar **codi que hagis generat tu**. No pots fer servir codi que altres estudiants hagin compartit amb tu ni que tu hagis compartit amb d'altres estudiants. Altrament es considerarà còpia.
2. Partiràs del codi que tens a `Exercici-2.tgz` adjunt a aquesta pràctica. Has de desplegar aquest arxiu en un directori teu. La solució que lliuris ha de compilar i executar correctament al laboratori. Els exercicis que es demanen només requereixen canvis a la classe `MyGLWidget`, **No has de modificar cap altre fitxer, No pots modificar la classe `LL2GLWidget`!**
3. Per fer el lliurament has de generar un arxiu `tar` que inclogui tot el codi del teu exercici i que es digui `<nom-usuari>-Ex2.tgz`, on substituiràs `<nom-usuari>` pel teu nom d'usuari. Fes que el directori de treball sigui aquell en el què has desenvolupat el codi de l'exercici i, si per exemple el teu nom és `Pompeu Fabra`, has d'executar

```
make distclean
tar zcvf pompeu.fabra-Ex2.tgz *
```

4. Un cop fet això, al teu directori tindràs l'arxiu `<nom-usuari>-Ex2.tgz` que és el que has de lliurar a la pràctica corresponent del Racó de la FIB **abans del dilluns dia 15 d'abril a les 23:59**.

Enunciat

L'objectiu de l'exercici és intentar fer una versió del clàssic joc "Snake" que es va popularitzar tant en 1998 perquè estava preinstal·lat als mòbils Nokia de l'època. Tot i mantenir una jugabilitat 2D voldrem visualitzar l'escena amb elements 3D: una serp que es mou sobre un terra i "creix" (s'allarga) al menjar uns ítems (bales en el nostre cas). A més a més, ficarem unes canonades (del Super Mario) per fer de frontera de l'espai de joc.

Primer haurem de pintar el terra amb les canonades fent de frontera o vorera, i després construirem i pintarem la serp considerant que té 3 parts: el cap, el cos i la cua (vegeu imatge del fitxer `escenaFinal.png`). Després haurem de fer que la serp es pugui moure, i que la part del cos sigui la que anirem replicant entre el cap i la cua, per fer la serp més llarga cada cop que es mengi una bala (vegeu imatge del fitxer `escenaFinal2.png`). Finalment, voldrem poder veure l'escena amb una càmera ortogonal en planta com si fos el joc 2D original (vegeu imatge del fitxer `escenaFinal3.png`).

Et proporcionem un codi bàsic que crea i visualitza una escena formada per **un terra de 10x10** unitats ubicat sobre el pla XZ i centrat a l'origen, **una bala** escalada per a que tingui alçada 1.5 amb el centre de la seva base al punt (10, 0, 0) (`marblePos`), **una cua de serp** escalada per a que tingui alçada 1 amb el centre de la seva base al punt (4, 0, 0) (`tailPos`), **un cos de serp** escalat per a que tingui alçada 1 amb el centre de la seva base al punt (-1, 0, 0) (`bodyPos`), **un cap de serp** escalat per a que tingui alçada 1 amb el centre de la seva base al punt (-5, 0, 0) (`headPos`), i **una canonada** de decoració escalada per a que tingui alçada 1 amb el centre de la seva base al punt (-8, 0, 0) (vegeu imatge del fitxer `escenaInicial.png`). **Analitzeu el codi donat abans d'implementar funcionalitats**.

A partir d'aquest codi, resol els següents exercicis:

1. Modifica l'escena per a que:
   (a) El terra faci **30x30** i segueixi centrat al punt **(0,0,0)**.
   (b) Hi hagi **canonades** al voltant de tot el terra però a dins, no a fora, que facin **3** unitats d'alçada i es pintin amb una separació de 1 unitat. *Nota: Fixa't que t'hi cabran 30 canonades en cada costat del terra.*
   (c) Es munti una serp sencera, d'alçada 1, mirant cap a les **X positives**, fent servir **el cap, la peça de cos i la cua** que ja tens, adientment col·locats i orientats (1 unitat de distància entre ells) amb les posicions dels centres de les seves respectives bases als punts **(0,0,0), (-1,0,0) i (-2,0,0)**. Fixa't que tens les variables `headAngle` i `tailAngle` que cal que les utilitzis per fer l'orientació inicial.
   (d) Hi hagi **1 bala** amb alçada **0.75** i amb el centre de la seva base al punt (10,0,0).
2. Calcula els paràmetres d'una càmera perspectiva per tal de veure l'escena sencera, centrada i sense retallar. Per posicionar la càmera, has de fer servir els dos angles d'Euler (psi, theta) per tal de mostrar l'escena amb una inclinació **vertical inicial de 45 graus**.

   Afegeix també el codi d'interacció per al ratolí necessari per tal que es puguin modificar els angles d'Euler. Fixa't bé en el que ja tens implementat a la classe `LL2GLWidget`. Fes servir les variables donades `factorAngleX` i `factorAngleY` per traduir el desplaçament en píxels del ratolí a l'angle de rotació corresponent.

El moviment d'inspecció que es demana ha de fer que en moure l'usuari el ratolí cap a la dreta del viewport la càmera es mou justament en sentit contrari, cap a l'esquerra, i el mateix anant cap amunt. Fixa't que amb aquest moviment d'inspecció sembla que l'usuari arrossegui l'escena en moure el ratolí.

Pots veure la imatge de la solució als exercicis 1 i 2 en el fitxer `escenaFinal.png`.

3. Volem que al prémer les tecles `Key_Left`, `Key_Right`, `Key_Down` i `Key_Up` tota la serp es mogui 1 unitat en les respectives direccions -X, +X, +Z, -Z. L'orientació del cap ha de ser sempre mirant en la direcció de moviment, i la direcció de la cua ha de ser sempre mirant cap a la peça de cos a la cual ha d'anar enganxada.

   Per resoldre aquest exercici, el codi proporcionat ja us dóna implementat un mètode `updateSnakeGame()` que actualitza la posició de la serp tenint en compte la direcció `direction` que estigui definida. Aquest mètode també s'encarrega de que quan el cap de la serp arriba a la posició de la bala se la menja, i llavors fa créixer la serp afegint una nova instància de cos de serp entre el cap i la cua. Un cop la serp ha menjat la bala, també tenim un mètode `computeRandomMarblePosition()` que fa aparèixer una nova bala a una posició aleatòria sobre el terra (en una posició vàlida que no està ocupada per la serp ni per les canonades).

   El moviment de la serp també ha de ser restringit. Cal que **completis la implementació del mètode** `checkPosition (glm::vec3 pos)` que s'ha d'encarregar de comprovar si es pot fer el moviment següent o no. Per fer això cal controlar que el cap de la serp no pot anar a un espai que estigui ja ocupat pel seu propi cos (o cua) ni on hi hagi canonades (veure imatge del fitxer `escenaFinal2.png`).

4. Afegeix al codi una segona **càmera ortogonal en planta** que faci l'efecte de visualitzar el joc com si fos l'original en 2D (veure imatge del fitxer `escenaFinal3.png`). Fixa't que ha de coincidir el moviment de la serp provocat per les tecles amb l'orientació del tauler (Up - amunt, Down - avall, Left - esquerra i Right - dreta). Aquesta càmera s'activa/desactiva amb la tecla `C`.

   En aquesta càmera no ha de funcionar la rotació dels angles d'Euler però sí que cal evitar retallats en fer un resize.

5. Afegeix el tractament de la tecla `R` de manera que permeti reinicialitzar l'escena i la càmera al resultat dels exercicis 1 i 2 (és a dir que es vegi tot com a la imatge del fitxer `escenaFinal.png`).

6. **Opcional**: Afegeix una animació a l'aplicació de manera que quan l'usuari prem la tecla `T` s'activa un *timer* que fa que el moviment de la serp es fa de manera automàtica, fent que cada 200 mil·lisegons tota la serp avanci 1 unitat en la direcció en la que està mirant el cap. En prémer la tecla `T` un altre cop es desactiva el *timer*. Nota: Fixa't que si el *timer* està activat les tecles no han de fer avançar la serp (perquè ja ho fa el *timer*) però sí que han de direccionar-la.

A ∼`/assig/idi/LabEx/Exercici-2` tens un executable de la solució.