RG47 - Bilijar

Dokumentacija

Aleksa Kojadinović
130/2017

shconsts.h

Skraćeno od **sh**ared**const**ant**s**. Sadrži dosta konstantni koje se inicijalizuju na samom početku igre. Među najvažnijima su dimenzije bilijarskog stola: *tableLength, tableWidth*, kao i koordinate ivica stola: *tableEdgeUp, tableEdgeDown, tableEdgeLeft, tableEdgeRight*, poluprečnici kugli i rupa: *ballRadius i pocketRadius*, dekartove koordinate kamere: *lookingAt(XZY), lookingFrom(XZY),* sferne koordinate tačke iz koje se gleda *camRho, camTheta, camR*, itd. Takođe sadrži makroe sa vrednostima nekih često korišćenih uglova u radijanima, logičke vrednosti:

- ***bool*** *inShotMode* - označava da li je igrač u režimu udaranja kugle ili u režimu gledanja oko stola
- ***bool*** *fineTune - označava da li je igrač u režimu preciznijeg ciljanja*
- ***bool*** *fullScreen - označava da li je veličina prozora fullscreen*
- ***bool*** *controlLock - označava da li su kontrole zaključanje*
- ***bool*** *textures - označava da li su teksture uključene*

Sadrži i inicijalizacije nekih vrednosti osvetljenja.

putils.h, putils.cpp

Neke kratke utility funkcije.

bool *withinBounds* - proverava da li je vrednost zadata prvim parametrom unutar njenih ograničenja koja su zadata drugim parametrom kao niz *double* vrednosti dužine dva. Razlikuje se od funkcije withinBoundsSimple po tome što dozvoljava prekoračenje ali ga odmah vraća. Daje efekat treperanja da bi korisnik znao šta se dešava.

- **bool** *withinBoundsSimple* - isto kao withinBounds samo bez efekta treperenja
- **void** *drawCircle* - crtanje kruga trigonometrijskom parametrizacijom (polarnim koordinatama)
- **bool** *circleDrop* - proverava da li se krug sa centrom (smallX, smallY) nalazi unutar kruga sa centrom (bigX, bigY), tačnije da li je centar manjeg kruga unutar centra većeg kruga. Koristi se za proveru da li je kugla upala u rupu
- **void** *set_normal_and_vertex,* **void** *draw_cylinder,* **void** *draw_cylinder 2* - koriste cilindrične koordinate za iscrtavanje cilindra preko *triangle strip*-ova

image.h, image.cpp

Kod preuzet sa jednog od časova. Služi za učitavanje bmp slike za teksturu.

Vec2.h, Vec2.cpp

Klasa koja predstavlja vektor u dve dimenzije.

Obezbeđuje osnovne operacije sa vektorima (sabiranje, oduzimanje, množenje skalarom, skalarni proizvod).

Kod operacija čiji je rezultat vektor date su dve verzije, verzija koja menja trenutni vektor i verzija koja vraća novi.
Npr *.add(***Vec2** *v)* dodaje v na pozivajući vektor *dok .r_add(***Vec2** *v)* vraća novi vektor koji predstavlja zbir trenutnog vektora i vektora *v*.
Implementirana vrlo direktno, prateći formule. Metoda *squaredMag* vraća kvadriranu normu vektora. Korisno za poređenje dve norme, kao i proveru da li je norma vektora jednaka nuli.

Ball.h, Ball.cpp

Klasa koja predstavlja bilijarsku kuglu.
Polja:

- **Vec2** *position - vektor pozicije*
- ***Vec2*** *velocity* - vektor brzine
- **double** *radius* - poluprečnik
- **bool** *onTable* - označava da li je kugla trenutno na stolu
- **GLfloat** *r, g, b* - boja kugle
- **unsigned int** *bmaskTurnOn, bmaskTurnOff - bmaskTurnOn* je oblika $b_1b_2.....b_kb_{k+1}....b_n$ gde je *k* redni broj kugle i samo b_k = *1*, svi ostali nula. U main-u se koristi bitmaska *activityCheck* koja se OR-uje sa ovom bitmaskom svake kugle koja se kreće. Kada kugla prestane da se kreće tada se AND-uje sa bmaskTurnOff koji se dobija obrtanjem bitova *bmaskTurnOn-a*. Poređenje *activityCheck*-a sa nulom je tada ekvivalentno proveri da li sve kugle stoje
- **int** *idx* - indeks kugle u vektoru *balls* unutar main-a. Nema neku specijalnu svrhu, korisno za debagovanje

Metode: logički podeljene u četiri grupe.

1. Konstruktori:

- *Ball ()* - prazan konstruktor
- *Ball(***Vec2** *position,* **Vec2** *velocity,* **double** *radius,* **GLfloat** *r,* **GLfloat** *g,* **GLfloat** *b,* **int** *i)* - konstruktor koji inicijalizuje sva relevantna polja

2. Geteri, seteri i sl. - pogledati u kodu
3. Pametne metode:

- **void** *drawSelf()* - iscrtavanje kugle. Prilikom crtanja kugle pretpostavljamo da je koordinatni sistem već transliran uvis. Razlog je izbegavanje velikog broja pozivanja funkcije *glPushMatrix()* zato što će se kugle crtati u petlji
- **void** *updateSelf(***unsigned int** ***** *activity)* - dodaje vektor brzine na vektor pozicije čime se simulira kretanje. Množi vektor brzine sa 0.99 čime se simulira trenje. Proverava da li se kugla zaustavila / pokrenula i u zavisnosti od toga ažurira *activity* (objašnjeno iznad)
- **void** *collideWith(***Ball &** *b,* **int** *what)* - Zasnovano na elastičnim kolizijama. Pogledati članak http://www.vobarian.com/collisions/2dcollisions2.pdf. Celobrojni parametar *what* daje mogućnost da se uradi ili samo provera (JUST_CHECK) ili da se zapravo menjaju brzine kugli

- **void** *cushionCollide(***double** *limUp***, double** *limDown***, double** *limLeft***, double** *limRight)* - proverava da li se kugla sudarila sa ivicom stola i u zavisnost od toga, i vrednost *what,* koja ima isto značenje kao i u prethodnoj funkciji, ažurira vektor brzine
- **void** *pocketCollide*(**double** *limUp*, **double** *limDown*, **double** *limLeft*, **double** *limRight*, **double** *pocketRadius*) - proverava da li je kugla upala u neku od rupa

4. Debug metode - samo jedna *toString* metoda

main.cpp

Globalne promenljive:

- **vector<Ball>** *balls* - vektor koji sadrži kugle
- **GLUquadric*** *asd* - *quadric* objekat za crtanje cilindara.
- **static GLuint** *tex_names[1]* - teksture

Funkcije: logički podeljene u pet grupa.

1. *Hendleri događaja:*
   - **static void** *on_display*(**void**) - podešava matricu i poziva funkcije iscrtavanja (*directCamera, drawTable, drawAmbient, drawBalls, drawAim, darwCueballIndicator*)
   - **static void** *on_reshape*(**int**, **int**) - podešava *viewport* prilikom promene veličine prozora
   - **static void** *on_keyboard*(**unsigned char** *key*, **int** *x*, **int** *y*) - upravlja kontrolama tastature koje su navedene u *README.md*
   - **void** *mainTimerCallBack*(**int**) - tajmer callback funkcija za animaciju. Prolazi kroz sve kugle, obrađuje kolizije jedne s drugom, kolizije sa ivicama i rupama. Proverava *activityCheck* i na osnovu njega prekida ili nastavlja animiranje
2. *Funkcije za inicijalizaciju:*
   - **void** *initAll(double)* - podešava veliki broj početnih vrednost od kojih se najveći deo nalazi u *shconsts.h*
   - **void** *fillCluster()* - Popunjava vektor balls kuglama sa pozicijama na početku bilijarske igre (trougao)
   - **void** *initTexture()* - inicijalizacija teksture površine stola
3. *Funkcije za iscrtavanja:*
   - **void** *directCamera()* - podešava pogled
   - **void** *drawTable()* - iscrtava bilijarski sto, uključujuci nogare, osnovu, površinu i rupe
   - **void** *drawAmbient()* - iscrtava sobu
   - **void** *drawCoord()* - iscrtava koordinatni sistem
   - **void** *drawBalls()* - iscrtava sve kugle koje su na stolu
   - **void** *drawAim()* - iscrtava nišan, ukoliko je igrač u režimu udarca
   - **void** *drawHud()* - ispisuje informacije relevantne za igrača
   - **void** *drawCueballIndicator()* - iscrtava torus oko bele kugle ukoliko je igrač u režimu postavljanja bele kugle rukom
4. *Nesvrstane funkcije:*
   - **bool** *anyBallsMoving()* - proverava li se ikoja kugla pomera, koristi se u *mainTimerCallback*-u

- **Vec2** *getViewDirection()* - vraća normalizovan vektor pogleda koji se kasnije množi sa jačinom udarca i time se dobija inicijalni vektor brzine bele kugle
- **void** *setStandardLimitsAndVals()* - podešava vrednosti i ograničenja u standardnom režimu igre
- **void** *setShotModeLimitsAndVals()* - podešava vrednosti i ograničenja u režimu udarca
- **void** *toggleFineTune()* - uključuje/isključuje režim preciznijeg ciljanja
- **void** *toCtlModePlaceCueball()* - prebacuje se u režim postavljanja bele kugle rukom (početak partije i svako ubacivanje bele kugle)
- **void** *toCtlModeRegular()* - vraća se iz režima postavljanja bele kugle rukom

5. *Funkcije za pisanje teksta:*
   - **void** *output*(**double** *x*, **double** *y*, **float** *r*, **float** *g*, **float** *b*, **void** ***** *font*, **string** *s*) - funkcija preuzeta iz man stranica. Koristi se za ispisivanje teksta na ekranu
   - **string** *getShotStrengthString*() - vraća string koji će biti ispisan za jačinu udarca
   - **string** *getFineTuneString*() - vraća string koji će biti ispisan za režim preciznijeg ciljanja
   - **string** *getShotModeString*() - vraća string koji će biti ispisan za režim udarca