DEFEND THE CASTLE

Predmet: Računalniška grafika in teorija iger

Študijsko leto: 2011/2012

Avtorja:

Alan Lukežič (63090089)

Miran Levar (63090369)

KAZALO

[**Koncept** 3](#_Toc314249428)

[**Liki in okolje** 3](#_Toc314249429)

[Zgodba 3](#_Toc314249430)

[Igranje 3](#_Toc314249431)

[Glavni lik 3](#_Toc314249432)

[Sovražniki 4](#_Toc314249433)

[Kontrole 5](#_Toc314249434)

[Okolje 5](#_Toc314249435)

[**Realizacija in opis dela** 5](#_Toc314249436)

[Liki in ostali modeli 5](#_Toc314249437)

[Igra 7](#_Toc314249438)

[*Premikanje* 7](#_Toc314249439)

[*Detekcija trkov* 7](#_Toc314249440)

[*Delovanje orožja* 8](#_Toc314249441)

[*Kamera* 8](#_Toc314249442)

[*Izris* 8](#_Toc314249443)

[*Zaslon* 8](#_Toc314249444)

[*Potek igre* 9](#_Toc314249445)

[**Zaključek** 9](#_Toc314249446)

## Koncept

Osnovna ideja igre je branjenje gradu pred sovražniki, ki poizkušajo priti vanj in ga uničiti. Igralec nadzoruje glavni lik in se z njim bojuje proti sovražnikom. Točke dobiva glede na vrsto in število sovražnikov, ki jih uniči. Konec igre predstavlja sovražnikovo zajetje gradu oziroma dokončanje vseh stopenj.

## Liki in okolje

Zgodba

V generičnem kraljestvu pogumni heroj brani grad svoje izbranke pred napadi zlobnega čarodeja, ki mu izbranko skuša ugrabiti. Čarodej je v svoji silni zlobi ugrabil živali iz bližnjih gozdov in kmetij ter jih modificirane poslal nad heroja, le-ta pa bo moral za zmago nad zlom pokazati vse svoje veščine in znanje, ter svojo izbranko obvarovati.

Igranje

Glavni lik je postavljen v 3D prostor, v katerega iz robov vdirajo sovražniki. Sovražniki poizkušajo priti do gradu in ga uničiti, glavni lik pa jim mora to preprečiti tako, da jih ubije. To mu omogočajo orožja, ki jih ima na razpolago. Igralec dobiva točke, glede na to, koliko sovražnikov uspe uničiti. Igra je razdeljena na stopnje, ki se med seboj razlikujejo po število vseh sovražnikov in v hitrosti njihovega pojavljanja. Igra se uspešno konča, ko heroj dokonča vse stopnje ali pa neuspešno, če sovražniki vdrejo v grad.

Glavni lik

Igralec nadzoruje glavni lik s tipkovnico in miško. Pogled kamere predstavlja pogled glavnega lika skozi oči, torej gre za prvoosebno igro. Pogled omogoča tudi prepoznavanje orožja, ki ga lik trenutno uporablja. Tipkovnico potrebuje za premikanje lika po prostoru, miško pa za premikanje kamere in uporabljanje orožja. Igralec ima omejeno število življenj, prav tako tudi grad, ki ga brani. Glavni lik ima dve vrsti orožij, to sta meč in lok. Meč je namenjen uničevanju sovražnikov v bližini, njegova prednost je hitro zaporedno uporabljanje. Lok se uporablja za sovražnike ki so bolj oddaljeni, v primerjavi z mečem ga ni mogoče tako hitro po izstrelitvi puščice zopet uporabiti.

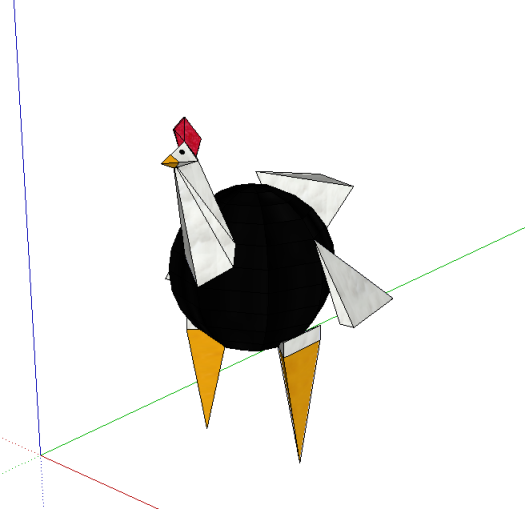


Slika 2: Pogled glavnega igralca - orožje lok

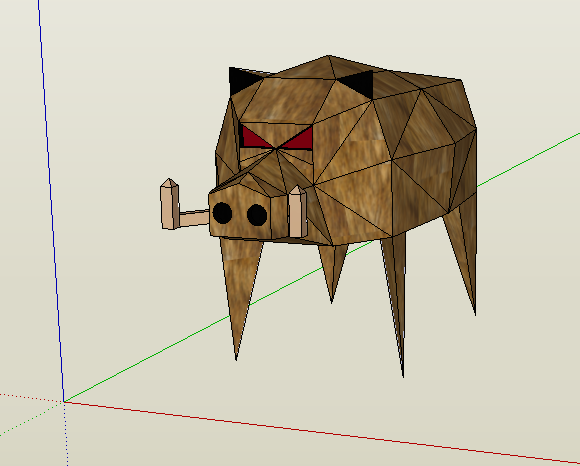
Slika 1: Pogled glavnega igralca – orožje meč

Sovražniki

Glavni cilj sovražnikov je uničenje gradu. V igri obstajajo trije različni sovražniki, ki se med seboj razlikujejo po hitrosti premikanja v prostoru in po škodi, ki jo naredijo na gradu. Sovražniki so poleg glavnega lika edini osebki v igri. Igralec lahko z njimi interaktira tako, da jih uniči, sovražniki pa imajo možnost povzročanja škode na gradu. Prva vrsta sovražnikov so kokoši, ki imajo namesto telesa tempirano bombo. Kokoši se premikajo najhitreje, naredijo pa najmanj škode na objektu. Druga vrsta so merjasci z izrazitimi čekani. Merjasci predstavljajo standardnega sovražnika, ki je srednje hiter in povzroča zmerno škodo. Tretja vrsta sovražnika, ki je najpočasnejši, a naredi največ škode, je medved. Ker so medvedi za grad najbolj nevarni mora biti igralec na njih najbolj pozoren. Sovražnike lahko igralec uniči tako, da jih zadane s puščico, ki jo izstreli z lokom ali pa tako, da zamahne z mečem. Posebno vrsto predstavljajo poglavarji. To so liki, ki se od navadnih ločijo po tem da so večji, težje jih je uničiti in ob morebitnem dosegu grada to pomeni konec igre.



Slika 3: Sovražnik št.1: kokoš



Slika 5: sovražnik št. 3: medved

Slika 4: sovražnik št.2: merjasec

Kontrole

Za premikanje lika so zadolžene tipkovnične tipke WSAD, kar je pogosto pri večini iger. Za interakcijo z uporabnikom se uporablja tudi miška, in sicer za premikanje kamere, ter menjanje orožja, za kar poskrbi miškin kolešček ali števila. Obe orožji uporabimo z levim miškinim gumbom. Z ESC je dostopen meni.

Okolje

Okolje je predstavljeno z ravno ploskvijo, ki predstavlja travnik pred gradom. Na eni strani je postavljen grad, katerega moramo braniti, na drugi pa gozd. Tile elementi nam predstavljajo tudi meje okolje, tako da igralec ne mora skozi. Prav tako je ob straneh postavljen gozd, vendar je realiziran kar s pomočjo tekstur. Celoto zaključuje nebo z oblaki, ki ustvarja vtis pravega okolja. Za dodatno zadostitev zgodbe je na koncu, za gozdom, dodan še sovražnikov grad.

## Realizacija in opis dela

Liki in ostali modeli

Modela glavnega lika nismo realizirali, ker je pogled kamere tak, da se nam je to zdelo nesmiselno. Potrebno pa je bilo narediti le roko, ki je vidna pri uporabi orožja, ter orožje samo. Za izdelavo teh modelov smo uporabili Google Sketchup, ter jih uvozili v naš program s pomočjo pretvornika modelov v datoteke .obj. Roka, ter obe orožji – lok in meč sta vidni na slikah 1 in 2. Pri loku je prisoten še model puščice, ki se ob uporabi tega orožja izstreli in je nekaj časa nevidna, tako da orožja tisti čas ni mogoče uporabiti. Pri uporabi meča je vidna preprosta animacija, ki predstavlja zamah meča z roko.

Za izdelavo modelov živali smo prav tako uporabili Sketchup, ter nanje nalepili teksture, tako da so živali čim bolj podobna resničnim. Vidimo jih lahko na slikah 3,4 in 5. Gibanje živali nadziramo iz programa, tako da jim spreminjamo koordinate postavitve v prostoru.

Celotno okolje je tudi uvoženo iz Sketchup-a in predstavlja ploskve na katere so nalepljene ustrezne teksture, ter grad, ki je sestavljen iz preprostih poligonov, ter struktur kamenja, tako da daje vtis realnosti. So.vražniki so v igri predstavljeni z instancami razredov, tako da imajo kot atribut referenco na določen model. S takim načinom predstavitve smo dosegli enotno možnost evidence za vse like in enostavno možnost računanja položajev objektov v okolju



Slika 6: Model našega gradu – pogled iz Google Sketchup

Na sliki 1 lahko vidimo še pogled proti drugi strani – gozdu in sovražnikovemu gradu in sicer je slika zajeta iz igre. Del gozda je tudi sestavljen iz modelov, del pa je realiziran kar s teksturami (Vidno na slikah 1 in 6).

Igra

Za izvedbo same igre smo uporabili OpenGL, in sicer knjižnico LWJGL. Zato je bilo potrebno celotno funkcionalnost in logiko igre sprogramirati »na roke«. Za začetek je bilo potrebno modele pravilno uvoziti in vpeljati v igro. To smo storili s kreacijo razreda za vsako vrsto sovražnika, ki je imel referenco na ustrezen model. Sledila je implementacija ustrezne logike in funkcijonalnosti.

#### Premikanje

Premikanje je razdeljeno v dve skupini in sicer: premikanje glavnega lika in premikanje sovražnikov. Za premikanje glavnega lika poskrbi igralec, ki ga nadzira s tipkovnico in miško. Njegova dejanja se v vsaki iteraciji obdelajo in spremembe se izrišejo na zaslon. Drugi sklop predstavlja premikanje živali, za kar poskrbi logika v programu. Vsak primerek objekta ima kot atribute koordinate, ki predstavljajo postavitev v prostoru, hitrost s katero se premika, ter končno koordinato, ki predstavlja cilj živali. Ta je seveda v gradu, saj je naloga vsakega sovražnika zajetje gradu. Glede na trenutno in končno pozicijo se v vsaki iteraciji izračuna nov položaj živali. Pri obeh izračunih novega položaja – tako glavnega junaka, kot tudi posameznega sovražnika je potrebno upoštevati tudi trke, ter jih tudi pravilno obravnavati.

#### Detekcija trkov

Delimo jo lahko v tri skupine, in sicer trki med glavnim likom in okoljem, med glavnim likom in sovražniki, ter med sovražniki samimi. Prvo skupino trkov zaznavamo tako, da se v vsakem koraku naredi »poskusna poteza« v želeno smer in potem preveri trk, torej gre za računanje trka vnaprej. Če se ob novem položaju pojavi trk se poteze ne izvede, v obratnem primeru se rezultat poskusne poteze izriše. Ker so stene okolje poševne so predstavljene z enačbo premice in glede na položaj igralca levo oziroma desno od premice se določi trk. Trke med glavnim junakom in sovražniki zaznavamo s krogi. V vsaki iteraciji se preveri naš trenutni položaj, ter položaj vseh sovražnikov. Če se »navidezni krog«, ki si ga lahko predstavljamo okrog vsakega lika (njegov radij je shranjen v atributu objekta) seka, se spremeni smer premikajočih živali tako, da se ne prekrijejo (žival zaobide heroja). Zadnja skupina trkov, ki se dogajajo samo med sovražniki in nanje nimamo od zunaj nič vpliva se detektirajo podobno kot trki med glavnim likom in sovražniki, in sicer z navideznimi krogi. Okoli vsake živali si zamislimo krog, ter preverjamo, če se ti krogi sekajo, kar je enostavno, saj preverimo le, če je razdalja med dvema objektoma večja od vsote njunih radiev. Tudi v tem primeru se spremeni smer živali da ne pride do prekrivanja.

Vse trke računamo t.i. »bruteforce« torej poteka primerjanje med vsemi možnimi pari objektov. Za tak pristop in ne za kakšnega naprednejšega sva se odločila, ker misliva, da v igri ne bo hkrati tako veliko sovražnikov, da bi potrebovali zelo »pameten« algoritem, ki bi skrbel za detekcijo trkov, ter zaradi preprostosti objektov (prekrivanje preverjamo kar s krogi), to nam prihrani kompleksne račune, zato je lahko le-teh več.

#### Delovanje orožja

Delovanje orožja je realizirano tako, da se ob vsaki njegovi uporabi preveri, če je orožje naredilo kakšno spremembo, ter da se v vsakem primeru pokaže ustrezno animacija. Če ustrelimo z lokom je potrebno preveriti, če vektor, ki kaže naravnost v smeri pogleda gledalca in je definiran skupaj s trenutno postavitvijo kamere, »zadane« kakšnega sovražnika. V tem primeru se sovražnika uniči in odstrani iz igre. Ob smrti živali se zgodi manjša animacija, žival pade postrani in ostane v okolju dokler se stopnja ne konča. V primeru da uporabimo meč je potrebno vektor smeri orožja ustrezno omejiti glede na lastnost meča koliko meri njegov doseg. V primeru da meč »zadane« žival, le-ta izgubi dve točki zdravja. Tako heroju omogočimo taktično izbiro pri boju s sovražniki, saj bo živali, ki so mu bližje in potrebujejo več udarcev, lažje pokončal z mečem kot lokom. Delovanje orožja je lahko definirano tudi kot problem detekcije trkov, saj v bistvu pregledujemo sekanje smeri orožja z določenim sovražnikom, vendar smo ga zaradi specifičnosti opisali posebej.

#### Kamera

Kamera predstavlja osnovni pogled igralca. Ker OpenGL ne nudi podpore za kamero, smo jo morali realizirati sami. To smo storili z implementacijo razreda, ki je predstavljal igralca – njegovo pozicijo v prostoru, ter smer kamor je obrnjen. Glede na te parametre se tudi spreminja izris okolja. V pomoč nama je bil vodič <http://www.lloydgoodall.com/tutorials/first-person-camera-control-with-lwjgl/>, ki nama je omogočil, da sva razumela delovanje prvoosebne kamere ter jo nato priredila za svoje potrebe.

#### Izris

Izris poteka v zanki, tako da se v vsaki znova izračunajo potrebni parametri objektov, glede na te parametre pa se izriše vse kar je potrebno. Parametre je potrebno računati za objekte, ki se premikajo (sovražniki), ter za parametre igralca. Izračun potega na podlagi atributov vsakega objekta, igralčevih interakcij, ter morebitnih trkov z drugimi objekti. Za okolje, ki je statično teh parametrov ni potrebno računati, ampak se objekti samo izrišejo.

#### Zaslon

Na zaslonu so prikazane le najosnovnejše informacije, ki jih igralec potrebuje za nemoteno igranje igre. V zgornjem levem kotu lahko vidimo na kateri stopnji se trenutno nahajamo, v zgornjem desnem kotu pa koliko »življenj« je še ostalo gradu. Ta številka se zmanjša vsakič, ko uspe sovražniku priti v grad. Napisi so realizirani s HUD – om, ki smo ga obravnavali tudi na vajah. Igra vsebuje tudi meni, ki je prav tako realiziran s HUD – om, elemente menija pa izberemo z numeričnimi tipkami. Funkcije, ki jih meni ponuja so le najosnovnejše in sicer možnost nadaljevanja igre (izhod iz menija), začetek nove igre, ter izhod iz igre, ki zapre glavno okno in zaključi izvajanje programa.

#### Potek igre

Igra se prične s prvo stopnjo, tako da se sovražniki premikajo proti gradu. V vsaki stopnji je vnaprej določeno skupno število vseh sovražnikov, ter njihovo pojavljanje v okolju. Vodi se tudi evidenca o tem koliko sovražnikov je že uničenih, ter koliko jih je uspelo priti v grad. Ko so vsi predvideni sovražniki bodisi ubiti, oziroma v gradu je stopnje konec. Sledi prehod na novo stopnjo, ki je od prejšnje seveda težja v smislu da se v njej generira več sovražnikov, ki se tudi pojavljajo bolj pogosto. Ko se zaključijo vse predvidene stopnje je igre konec in to predstavlja zmago igralca. Vsa ta logika je vgrajena v programu in je igralec ne more spreminjati. Tudi za grad je potrebno voditi evidenco o sovražnikih, ki so ga že uspeli doseči, ter jo primerjati z največjim dovoljenim številom »pobeglih« sovražnikov. Obe številki sta predstavljeni kot atributa igralca, ter tudi prikazani na zaslonu.

## ZAKLJUČEK

Pri seminarski nalogi sva se spoznala z osnovniki koncepti načrtovanja igre, ter njene implementacije. Vsi problemi, ki so s tem povezani se nama zdijo zelo zanimivi, obenem pa tudi zelo obsežni. Človek ki se s takimi problemi nikoli ne sreča si morda misli, da je izdelava igre preprosta, vendar se je že v najinem primeru pokazala kot zahtevna, ter predvsem obsežna. V igro sva vključila nekaj osnovnih konceptov, ki smo jih obdelovali pri predmetu. Med izdelovanjem igre se nama je odprlo veliko problemov, ki sva jih bolj ali manj uspešno rešila. Kot tiste ki so nam vzeli največ časa bi veljalo izpostaviti začetek projekta, ter detekcijo trkov. V začetku nam je največji problem predstavljalo to, kje in kako se problema sploh lotiti. Ko sva si razjasnila nekaj osnovnih konceptov je stvar stekla in tudi napredek je bil kar hitro viden. Kot drugi največji problem smatrava detekcijo trkov in sicer smo poizkušali realizirati nekaj naprednejših algoritmov za detekcijo, vendar sva se na koncu odločila da bo igra tekoča tudi že z naivnim algoritmom. Pri detekciji trkov se nam je tudi postavljalo vprašanje kako se odzvati na detektirane trke. To sva rešila s preprostim upočasnjevanjem določenih objektov.

Kot tisto, kar misliva da sva najbolje realizirala je vizualen izgled igre, kar vključuje predvsem modele, ki sva jih implementirala v programu za modeliranje Google sketchup.