Projektityön dokumentti

Joonas Palosuo

477743

Bioinformaatioteknologia

2014

6.5.2016

Yleiskuvaus

Toteutettiin vuoropohjainen strategiapeli tekoälykkäällä tietokonevastustajalla. Pelissä komennetaan erilaisia hahmoja. Peli sisältää monipuolisesti muokattavia hahmoja sekä erittäin intuitiivisen kentänluomismahdollisuuden. Käyttöliittymältään peli on erittäin pelattava, vaikkei graafinen toteutus olekaan kaikkein hiotuin. Mielestäni tehtävä toteuttaa vaikeustason vaativa.

Käyttöohje

Käynnistääksesi pelin avaa komentoikkuna (cmd, terminal, tai vastaava) ja kirjoita siihen “python3 main.py”. Tämä avaa pelin, jossa voit valita joko pelin aloituksen tai pelin sulkemisen. Aloitettuasi pelin voit valita sotilaita klikkaamalla niitä hiiren vasemmalla painikkeella, käskeä niitä hyökkäämään tai liikkumaan oikealla painikkeella, liikuttaa näkymääsi nuolinäppäimillä ja pausettaa pelin esc-näppäimellä. Liikuttele joukkojasi ja tuhoa vastustajan armeija, niin voitat pelin. Kun olet käyttänyt vuorosi kaikki liikkeet, “next turn” -nappi muuttuu vihreäksi. Voit luovuttaa vuorosi aiemminkin, jos niin haluat.

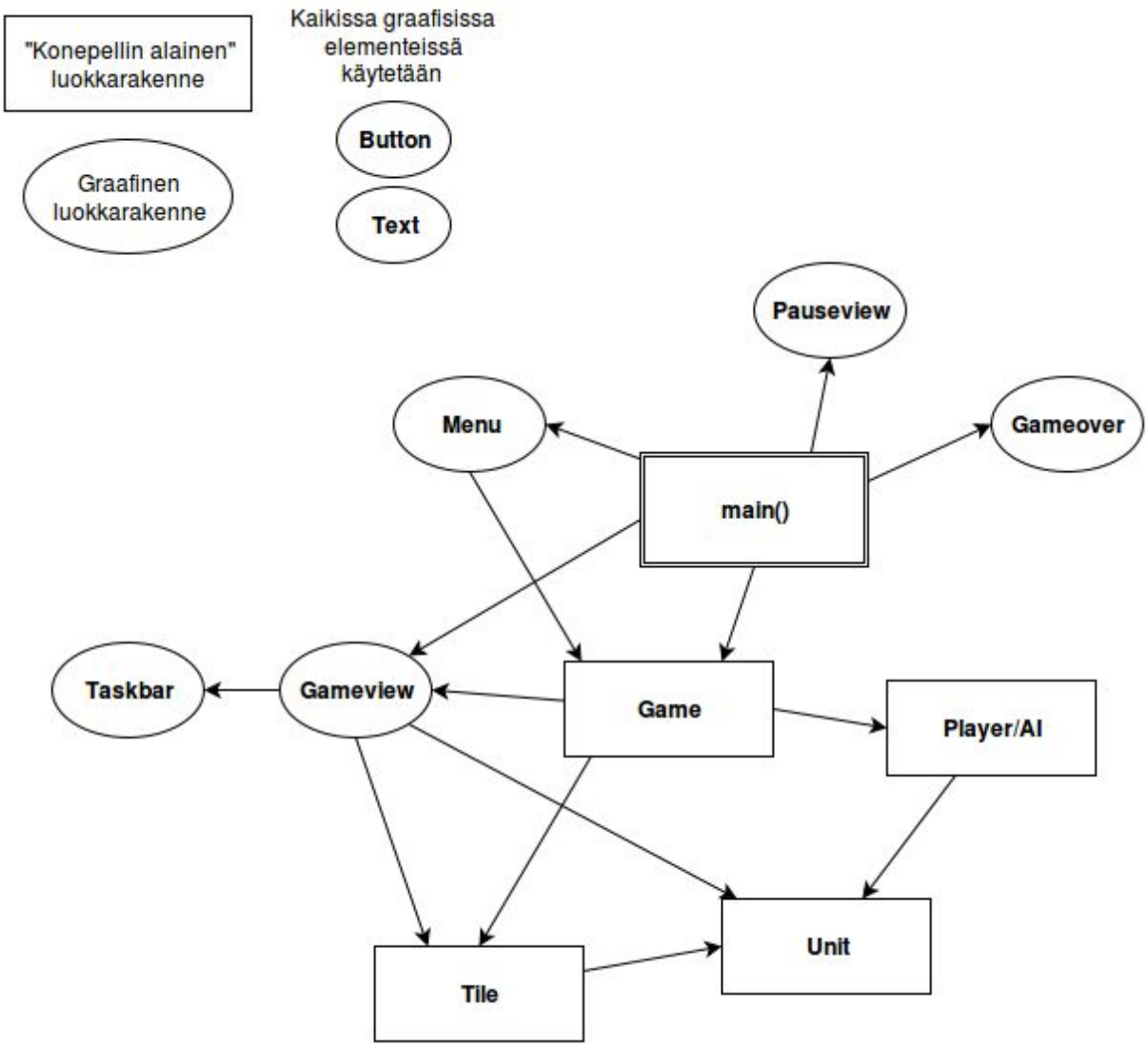
Mikäli tahdot muokata peliä, voit avata setup.txt:n valitsemallasi tekstieditorilla. Sieltä löydät muun muassa graafisia asetuksia, ladattavan kenttätiedoston, sekä kentissä käytettävät grafiikat ja syntaksit. Uusia sotilaita voit luoda tekemällä uuden tekstitiedoston {sotilastyypin nimi}.txt ja lisäämällä niitä pelaajakohtaisiin player#.txt -armeijakompositiotiedostoihin. Lisäksi tarvitset graphics/sprites/ ja graphics/portraits/ -kansioihin vapaamuotoiset .png -kuvatiedostot.

Uusia karttoja voit luoda maps/ kansioon. Karttojen täytyy olla neliskulmaisia, ja niissä on käytettävä setup.txt:ssä mainittua syntaksia. Voit itse määrittää uusia maastotyyppejä, mutta niihinkin on luotava omat grafiikkatiedostot itse.

Mikäli pelin käynnistyksessä tapahtuu virhe, log.txt sisältää tietoa siitä, mikä meni mönkään.

Ohjelman rakenne

Graafinen esitys, jossa käy ilmi, mitkä luokat sisältävät/kutsuvat/linkittyvät mihinkin luokkiin:



Luokkien esittely:

- **Game**: itse peli, sisältää kaiken tiedon pelaajista, hahmoista ja kentästä sekä metodeja niiden luomiseen ja käsittelyyn, muun muassa generateMap, initializeUnits, moveUnit, dealDamage, switchTurn.
- **Player/AI**: sisältää tiedot pelaajasta sekä pelaajan hahmoista sekä metodeja näiden käsittelyyn, kuten resetUnits, actionsLeft. **AI** sisältää lisäksi tekoälyn ja kaikki sen kutsumat metodit, kuten processTurn ja approachEnemy
- **Tile**: sisältää tiedot yksittäisestä pelikentän ruudusta sekä metodin ruudun klikkauksen käsittelyyn (click)

- **Unit**: sisältää tiedot yksittäisestä pelihahmosta, sekä metodin pelihahmon liikkeiden nollaamiseen (resetMoves)
- **Menu**: avautuu pelin käynnistyessä, näyttää pelin nimen ja tarjoaa käyttäjälle painikkeet pelin käynnistämiseen ja sulkemiseen. Keskeisimpänä metodina, kuten kaikissa graafisissa elementeissä, draw.
- **Gameview**: piirtää pelikentän ja hahmot eli kaiken mitä käyttäjä näkee **Game**sta. Tarkkailee myös kenttää klikkausten varalta. Keskeisin metodi moveView, jolla käyttäjä liikuttaa näkymäänsä kentästä.
- **Taskbar**: piirtää pelinäkymän alapalkin, eli valitun hahmon tiedot.
- **Pauseview**: piirtää taukoikkunan pelin päälle ja keskeyttää sen prosessoinnin, kun käyttäjä painaa esc-näppäintä.
- **Gameover**: piirtyy näkyviin kun toiselta pelaajalta loppuvat hahmot eli peli on ohi.
- **Button**: geneerinen painike, piirtää halutun värisen nelikulmion tekstin kera, jota klikkaamalla suoritetaan haluttu funktio.
- **Text**: piirtää tekstiä näkyviin.

Lähes kaikki luokkarakenteet ovat itseselitteisiä ja selkeitä. Graafiset elementit ovat jokainen omana luokkanaan, jotta niitä on helppo luoda lisää ja käyttää modulaarisesti tarpeen tullen. Peliä prosessoidaan eteenpäin käytännössä ainoastaan **Gameview**in piirtyessä, jolloin mm. pelin tauottaminen onnistuu niinkin helposti kuin vaihtamalla aktiivisesti piirtyvää näkymää.

Algoritmit

Peli käyttää simppeleitä algoritmeja, kuten yksiköiden asettelu kentälle ja hieman mutkikkaampia, kuten reitin löytäminen kentällä sekä tekoälyn päätöstenteko. Yksiköt asetetaan kentälle satunnaisille koordinaateille pelaajan aloituspisteen ympärille, kokoajan satunnaisaluetta tasaisesti kasvattaen. Tekoäly lähestyy pelaajaa tasaisesti, muodostaa kaukana vihollisesta olevista sotilaista pieniä ryhmiä ja hyökkää lähimpiin vastustajiin.

Kun pelaajan armeijan tiedostoa luetaan, yksiköitä asetetaan yksi kerrallaan aloituspisteen ympärille. Kun vapaita ruutuja ei löydy, laajennetaan satunnaisluvun arpomien koordinaattien aluetta, kunnes kaikki sotilaat ovat saaneet paikkansa kentällä.

Tekoäly laskee keskimääräisen etäisyyden jokaisen sotilaan ja vihollisen armeijan arvioidun sijainnin välillä. Tätä se käyttää päättäessään, tuleeko hahmon lähteä siirtymään vihollista kohti, johdattamaan muita hahmoja vai hyökkäämään lähimpään vihollisyksikköön.

Reitin löytämiseen käytetään ahnetta best first search -algoritmiä, joka on hieman ajoittain nopeampi kuin perinteisempi A*, muttei löydä aina tehokkainta reittiä määränpäähänsä. [1] Tähän algoritmiin päädyttiin sen implementaation helppouden sekä keskimääräisen laskennallisen nopeuden kannalta. Niin kauan kun kenttä ja sen mahdolliset esteet ovat helposti ohitettavissa, algoritmi suoriutuu kiitettävästi hyvän reitin löytämisessä.

Tietorakenteet

Pelissä ei tarvittu monimutkaisia tietorakenteita. Pelikenttä tallennetaan kaksiulotteiseen listaan, hahmot yksiulotteisiin listoihin ja tiedostoista luetut attribuutit sanakirjoihin. Hahmoja ja kenttää ei tarvitse käydä läpi muuten kuin suoraan indeksoimalla, ja listaan on helppo lisätä ja poistaa hahmoja niitä luodessa tai niiden kuollessa. Tiedostosta luetut rivit on käytännöllistä tallentaa sanakirjoihin avainten kera, jotta tarvittavat parametrit saadaan helposti kaivettua ulos niitä käsitellessä.

Tiedostot

Kaikki pelin grafiikka on tallennettuna .png kuvina, sillä ne tukevat tarpeen tullen läpinäkyvyyttä ja ovat tiedostokooltaan pieniä ja häviöttömiä. Kaikkiin tekstitiedostoihin voi kirjoittaa kommentteja aloittamalla rivin risuaidalla (#), ja melkein mitä vain voidaan muuttaa ulkoisten tiedostojen avulla.

Setup.txt pitää sisällään graafisia asetuksia, kuten resoluution, pelaajien värit ja pelin ruutujen koon. Lisäksi sieltä spesifioidaan mikä kenttä peliin ladataan, ja siellä on määriteltynä minkälaisia ruutuja kenttä voi pitää sisällään.

maps/ kansion tiedostot sisältävät pelikenttiä, joita voi muokata tai luoda itse uusia. Kenttien tulee olla nelikulmioita, niiden tulee sisältää “.” ja “,” -ruudut eli pelaajien 1 ja 2 aloitusruudut sekä tarpeeksi tilaa molempien pelaajien armeijoille. Muuten käyttäjällä on vapaat kädet muokata ja luoda täysin haluamansalaisia kenttiä.

units/ kansion player1.txt ja player2.txt sisältävät molempien pelaajien armeijat eli niiden sisältämät sotilaat. Uusia sotilaita voi luoda template.txt:n ohjeiden mukaan.

Testaus

Ohjelmaa testattiin pääosin käsin. Yksikkötestejä ei nähty tarpeelliseksi luoda, sillä pelin sisällä käyttäjän on mahdotonta saada ohjelmaa jumiin komentojensa avulla. Visuaalisia elementtejä seuraamalla ja komentorivin tulosteita tarkkailemalla nähtiin, että kaikki toimi kuten pitikin. Erilaisilla testitiedostoilla nähtiin myös, että tiedostojen oikeaoppinen luku toimi, ja virheellisten tiedostojen tullessa vastaan joko siirryttiin oletusarvoisiin asetuksiin, tai suljettiin ohjelma kirjoittaen virheen syy ulkoiseen tekstitiedostoon.

Ohjelman tunnetut puutteet ja viat

Erittäin pienissä kentissä tekoälyn on ajoittain vaikeaa päästä hyökkäämään pelaajan hahmoihin, mikäli pelaaja ei tee mitään vuorojen välissä. Normaalisti pelatessa kuitenkin teköäly onnistuu liikkumaan, lähestymään ja hyökkäämään pelaajaan kuten kuuluukin.

Tekoälyyn olisi voinut ohjelmoida enemmän ja monimutkaisempia piirteitä, kuten heikentyneiden yksiköiden perääntymisen tai yksiköiden muodostelmiin järjestäytymisen riippuen niiden hyökkäysetäisyydestä. Ensimmäinen ei olisi kuitenkaan vaatinut muuta kuin perääntymisliikkeen laskemisfunktion ja joka hahmon päätöksen teon alussa osumapisteiden tilanteen tarkistamisen.

Hahmoihin kohdistuvia väliaikaisvaikutuksia ei ole, mutta nekin olisi ollut helppo toteuttaa esimerkiksi hahmokohtaisella “statusEffect” -muuttujalla, joka olisi voinut olla esimerkiksi vuorokohtainen osumapisteiden parantuminen / heikkeneminen tai damage-bonus.

Maastot tukevat tällä hetkellä vain liikkuvuutta ja täyttä estettä, välimaastoa ei ole. Jokaiseen ruutuun olisi kuitenkin voinut tallentaa liikkumakertoimen, eli kuinka monta liikepistettä ruudun yli liikkuminen olisi vaatinut. Tämä olisi myös ollut helppo implementoida reitinlöytämisalgoritmiin, sillä tällä hetkellä se painottaa vain jokaisen ruudun liikkumasakoksi yhden liikkumapisteen.

3 parasta ja 3 heikointa kohtaa

Parhaita kohtia ohjelmassa ovat tiedostoilla muokattavuus, ohjelman modulaarisuus ja käyttäjäystävällisyys.

Lähes jokaista pelin osa-aluetta voidaan kontrolloida ulkoisilla tiedostoilla, ja nekin harvat asiat jotka ovat koodattuna suoraan lähdekoodiin voidaan muuttaa yhden muuttujan arvon avulla. Tiedostoformaatit ovat yksinkertaisia ja itseselitteisiä, joten käyttäjän on helppo muokata ohjelmaa omiin tarkoituksiinsa sopivaksi.

Ohjelma on myös rakennettu erittäin modulaariseksi, joten ominaisuuksien lisääminen myöhemmin on erittäin helppoa. Yksinkertaisimpiin graafisiin elementteihin löytyy jo valmiit pohjaluokat, ja ohjelman rakenne tukee helppoa siirtymistä eri näkymien välillä. Myös tekoälylle voi lisätä uusia käyttäytymisfunktioita suhteellisen vähällä vaivalla.

Heikkouksia ovat valmiiden ominaisuuksien vähäisyys, yksiköiden yksinkertaisuus ja graafinen simppeliys.

Grafiikka ei ole päätä huimaavaa millään osa-alueella, vaikka se onkin yksinkertaisuudessaan toimivaa. Visuaalisia efektejä ei tueta ollenkaan, ja niiden lisääminen saattaisi olla huomattavan hankalaa. Yksiköt eivät toistaiseksi tue mitään sekundääritoimintoja liikkumisen ja hyökkäämisen lisäksi, ja yksiköitä ei ole kovin montaa, eivätkä maasto tai tekoäly ole kovinkaan monipuolisia.

Poikkeamat suunnitelmasta

Peli ei tue pelinsisäistä tiedostojen latausta tai uudelleenkäsittelyä, eikä myöskään pelitilan tallennusta tai latausta. Tekoälyn debuggaaminen osoittautui ajoittain huomattavan haastavaksi. Ajankäyttö sekä osa-alueiden toteuttamisjärjestys pitivät kuitenkin pääosin paikkansa, vaikka aika tahtoikin loppua kesken muiden kiireiden takia.

Toteutunut työjärjestys ja aikataulu

1. Pelin näkymättömät osa-alueet: 10h
2. Ensimmäiset graafiset käyttöliittymän osat: 5h
3. Tiedostojen käsittelyä ja peliin lukemista: 10h’
4. Tekoälyn kehittämistä: 15h
5. Pelin käyttöliittymän navigointia (menut jne): 5h
6. Mahdollisimman monen muuttujan siirtäminen tiedostoista luettavaksi: 3h
7. Tekoälyn käyttäytymisen debugaamista ja muokkaamista: 10h
8. Kommentointia, hiomista, viimeistelyä: 3h

Arvio lopputuloksesta

Ohjelma toimii erittäin hyvin. Se lukee suurimman osan asioista tiedostoista, on mielekäs pelattava ja helposti muokattava. Pelillisiltä ominaisuuksiltaan ei kovinkaan rikas, mutta niitä pystyy halutessaan lisäämään jälkikäteen. Tekoäly on toimiva ja reilu, muttei erityisen monipuolinen. Luokkajako on toimiva ja havainnollistava. Enemmällä ajalla peliin olisi saanut helposti lisättyä valmiita ominaisuuksia.

Viitteet

[1] http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html
http://pygame.org/docs/
https://docs.python.org/3.4/

Liitteet

Ai.py
graphics/
Main.py
Menu.py
Setup.txt
Ui.py
units/
Game.py
Log.txt
maps/
Player.py
README.md
Tile.py
Unit.py