

Siege Siege Siege

Kortspelsaktiverad Strid i VR

Linnea Bergman
Rickard Falk
Jonathan Fransson
Felix Grönborg
Simon Johansson
Tobias Pettersson
Sven Thole

Examinator: Karljohan Lundin Palmerius

Sammanfattning

Denna rapport behandlar ett medietekniskt kandidatarbete inom virtuell verklighet. Projektgruppens mål var att utveckla ett spel med ett användargränssnitt i form av materiella spelkort som skulle användas inne i en virtuell värld. Syftet var att öka inlevelsen mellan spelare och spel genom att spelaren fysiskt känner de objekt som han eller hon interagerar med.

Spelet utvecklades i spelmotorn *Unity* med biblioteket *Vuforia* för att hantera spårning av spelkorten. Projektgruppen har arbetat utifrån en agil-utveckling och har följt metoden Scrum för att strukturera arbetet. Arbetsuppgifter delades in i intervall av två veckor, i så kallade sprintar. Varje sprint började med en sprintplanering där projektet granskades i förhållande till projektplanen för att se om utvecklingen låg i fas. Utifrån granskningen valdes ett antal arbetsuppgifter till projektmedlemmarna som skulle slutföras under sprinten.

Spelarens mål i spelet är att eliminera motståndarens bas genom att svepa kort genom ett digitalt slagfält och framkalla undersåtar som då anfaller motståndaren. Spelaren kommer ha tillgång till tre kort som aldrig lämnar handen. Då användaren sveper ett kort uppdateras kortet dynamiskt till en ny slumpmässig undersåte. Anledningen till detta var att användaren inte ska behöva släppa och dra nya kort. Det finns tre olika undersåtar med olika styrkor som strider mot varandra genom ett sten, sax, påse-system.

Resultatet är huvudsakligen spelet, där huvudsyftet var att få det materiella kortgränssnittet att fungera enligt kundens specifikationer. VR-glasögonen som användes för att uppleva den virtuella världen var *HTC Vive*, där en webbkamera monterats på toppen. Begränsningarna i spelet har gjorts för att det ska bli mer användarvänligt, då flera kort kan leda till att användaren upplever spelet som förvirrande eller svårhanterbart. Begränsningar har även gjorts över hur komplexa modeller som använts i spelet, vilket gjordes för att minska prestandakraven.

Typografiska konventioner

- VR - Virtual Reality (virtuell verklighet)
- AR - Augumented Reality (förstärkt verklighet)
- NPC - Non-Player Character (ej spelbar karaktär)
- AI - Artificial intelligence (artificiell intelligens)
- HMD - Head-Mounted Display (huvudburen skärm)

Innehåll

Sammanfattning	i
Typografiska konventioner	ii
Figurer	v
1 Inledning	1
1.1 Bakgrund	1
1.2 Syfte	1
1.3 Frågeställning	1
1.4 Avgränsningar	2
2 Relaterat arbete	3
2.1 TowerVR - ett VR spel för flera spelare	3
2.2 Clash Royale - ett mobilspel för två spelare	3
3 Verktyg, språk och API:er	5
3.1 Unity	5
3.1.1 Programmeringsspråket C#	5
3.2 HTC vive	6
3.3 Vuforia	6
3.4 Google Drive	7
3.5 Trello	7
3.6 Kommunikationsmedel	7
4 Utvecklingsprocessen	8
4.1 Systemutvecklingsprinciper	8
4.1.1 Kravhantering	8
4.1.2 Versionshantering	9
4.1.3 Testningsprinciper och QA	9
4.2 Arbetssätt	9
4.2.1 Mötesrutiner	9

4.2.2	Sprintplanering	10
4.2.3	Dokumentationsprinciper	11
5	Implementation	12
5.1	Kravspecifikation från kund	12
5.2	Spelidé	12
5.3	Utveckling av spelkort	13
5.4	Stridssystem	14
5.5	Klasshierarki	14
5.6	Val av grafisk stil för 3D-objekt i en VR-miljö	16
5.7	Modellering och animering	16
6	Resultat	17
7	Analys och diskussion	20
7.1	Analys av resultatet	20
7.2	Metod	20
7.3	Problem och hinder	21
7.4	Arbetet i ett vidare sammanhang	22
8	Slutsatser	23
Litteraturförteckning		24
A	Individuellt bidrag	25

Figurer

2.1	En scen ur Clash Royale	4
3.1	HTC vive	6
5.1	Till vänster syns korten i verkligheten och till höger korten i virtuell miljö	13
5.2	Soldatkort	13
5.3	Menykort	14
5.4	Undersåtars aggressionsradie visas med en gul cirkel	15
5.5	Klasshierarki för kod gällande levande objekt	15
5.6	3D-objekt med minimalistisk stil	16
6.1	Menyskärmen när applikationen startas	17
6.2	Scen från spelskärmen när spelet är igång	18
6.3	Den högra undersåten håller på att genereras	18
6.4	Skärbilderna som dyker upp vid vinst eller förlust	19

Kapitel 1

Inledning

Denna rapport beskriver utvecklingen av ett VR-spel vars mål var att skapa en ny typ av användarupplevelse genom att använda ett användargränssnitt av materiella kort.

1.1 Bakgrund

Datorspel har sedan dess uppkomst alltid strävat efter att få spelaren att helt leva sig in i spelet och glömma bort verkligheten. Den senaste utvecklingen för en mer verklighetstrogen spelupplevelse är VR (virtual reality). Med VR blir det lättare för spelaren att leva sig in i spelet eftersom att spelaren får vara i spelvärlden istället för att bara titta på en datorskärm. Ett stort problem som bryter inlevelsen är dock att spelaren inte fysiskt känner de objekt som han eller hon interagerar med i spelet. För att försöka öka inlevelsen är det alltså rimligt att utforska hur materiella gränssnitt kan användas i kombination med VR. Kort är ett av de mest populära sätten att spela sällskapsspel på. Exempel på detta är traditionella kortspel som poker och samlarkort-spel. Kort är ett gränssnitt mellan spelare och spel som de flesta redan är vana vid och blir därmed naturliga att hantera.

1.2 Syfte

Projektets mål är att utveckla ett realtidsstrategispel till *HTC Vive*, [1], som använder materiella spelkort som användargränssnitt. Syftet är att via spelkorten försöka öka inlevelsen mellan spelare och spel. Korten används för att styra alla spelets funktioner. Spelet är utformat på ett sätt som uppmuntrar planering och strategiskt tänkande.

1.3 Frågeställning

Problem och vad som behövs lösas inom projektet formuleras nedan som frågeställningar:

- Hur begränsas speldesignen av att använda materiella spelkort som ska spåras som användargränssnitt?
- Kan kameran i en *HTC Vive* användas för att uppfylla kraven som ställs för att utföra spårning av spelkorten?
- Hur ska den grafiska stilen för spelets 3D-objekt väljas för att bäst passa en VR-miljö?

Att styra ett spel genom kort som ska spåras sätter begränsningar för spelarens kontroll över spelet. Detta är ett problem då rörelser som i vanliga spel är lätt att kontrollera via en speldosa kan bli omöjliga att utföra med spelkort. Därmed måste vissa begränsningar tas i hänsyn i speldesignen. VR-utrustningen som ska användas är *HTC Vive*, vilket inkluderar en inbyggd framåtriktad kamera. Klarar denna kamera av att spåra av korten eller måste ett annat alternativ hittas? Valet av den grafiska stilen måste göras med hänsyn med vad som fungerar med VR. Då VR-spel är mycket mer krävande än vanliga spel är det viktigt att modeller inte är för avancerade och då sänker bildfrekvensen i spelet.

1.4 Avgränsningar

Avgränsningar som gjorts inom projektet är att utveckling riktar sig till kunder som äger en PC och VR-utrustningen *HTC Vive*. Detta gjordes då PC är den plattform som i nuläget har bäst hårdvara för att utföra rendering till VR. Hårdvaran som går att finna i konsoler som *Playstation 4* och *Xbox One* är sämre än vad man kan få i en PC. Spelets målgrupp är främst spelare som anses som nybörjare. Det vill säga att spelet ska vara så pass lättförståeligt att vem som helst kan kunna spela det. Det underlättar utvecklingen då spelet inte riktar sig till dedikerade spelare som förväntar sig spelsessioner som kan sträcka ut sig till flera timmar.

Kapitel 2

Relaterat arbete

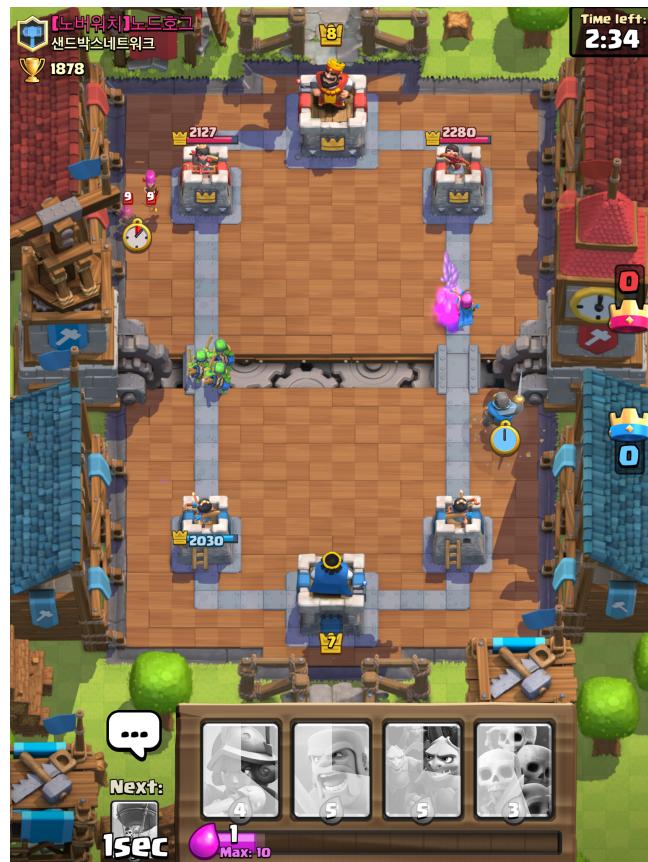
Projektet inleddes med en diskussion där olika tekniker och verktyg jämfördes mot varandra. Flera olika applikationer diskuterades för att komma fram till ett spelkoncept som uppfyllde kundens krav. VR är fortfarande en relativt nytt tekniskt område och det finns många användningsområden och tekniker för VR som ännu ej blivit testade. Användning av materiella spelkort som användargränssnitt är ett av dessa. Däremot finns det applikationer med relaterade tekniker eller koncept. Nedan listas ett par spel som har snarlik spelidé eller använder liknande verktyg.

2.1 TowerVR - ett VR spel för flera spelare

TowerVR, [2], som är ett VR spel för flera spelare som använder AR-teknik. Spelet utvecklades i ett tidigare kandidatprojekt i kursen TNM094 på Linköpings Universitet. TowerVR skapades i *Unity* med *Vuforia* för positionsspårning. Det som gör projektet intressant är att de skapade ett VR-spel som använde sig av position- och rotationsspårning med *Vuforia* som gruppen ska använda sig av i projektet. I TowerVR spåras ett fysiskt objekt som ligger som bas för ett virtuellt torn. På basen staplas sedan virtuella objekt. Den spelare som staplar ett objekt som får tornet att välna förlorar.

2.2 Clash Royale - ett mobilspel för två spelare

Clash Royale, [3], var huvudkällan för spelidén till projektet. Clash Royale är ett mobilspel skapat av Supercell, där två spelare spelar mot varandra. Spelaren som först har sönder den andra spelarens torn vinner och använder kort för att lyckas med detta. Efter ett kort använts dröjer det ett antal sekunder innan ett nytt kort dyker upp. Perspektivet för kameravinkeln från Clash Royale användes som grund och kompletterades för att ge en bättre VR-upplevelse. En bild på spelet ses i Figur 2.1.



Figur 2.1: En scen ur Clash Royale

Kapitel 3

Verktyg, språk och API:er

Under förstudien för projektet valdes verktyg, språk och API:er som sedan skulle användas. Kapitlet redogör varför de språken och API:erna valdes och användes samt hur de tekniska verktygen fungerar och hur de varit till nytta i projektets utveckling.

3.1 Unity

I tidiga stadier av projektet blev det uppenbart att en spelmotor skulle användas för att minska tiden som behövde läggas på att implementera de grafiska användargränssnittet. Spelmotorn som användes för att skapa spelet var *Unity* [4]. Andra anledningar till att *Unity* användes var för att den har inbyggt stöd för VR-utveckling samt att den är kompatibel med diverse tredjeparts API:er, som till exempel *Vuforia* och *OpenCV*. Detta gör den lämplig att använda för att utveckla projektet. Att en spelmotor används gör också att mer fokus kan läggas på till exempel implementation av kortens spårning och spelmekanik. Inom *Unity* finns det även ett versionshanteringssystem som underlättade hur gruppen kunde dela kod mellan gruppmedlemmarna. Utöver detta är även spelmotorn gratis och en gruppmedlem hade tidigare goda kunskaper av *Unity* och blev därmed ett naturligt val av verktyg för projektet.

3.1.1 Programmeringsspråket C#

I *Unity* användes programmeringsspråket C# som är integrerat i spelmotorn och tillåter objekt-orienterad programmering. Programmeringsspråket liknar språk som C++ vilket medför att de tidigare kunskaper som gruppmedlemmarna har inom C++ kan användas för att snabbt kunde bekanta sig med C#. Programmeringen skedde i ett av tredjeparts-API:erna som levereras med *Unity*. Projektmedlemmarna fick individuellt välja vilket API som skulle användas, då detta inte påverkade gruppens övriga medlemmar. *Visual Studio 2015* [5] rekommenderades då det inkluderar en fullt fungerande kompilator och felsöknings-verktyg som är professionellt utvecklat och genomgått flera testningar och uppdateringar. Notera att trots att programmeringen skedde utanför *Unity* så sparades ändå filer i utvecklingsmiljön. Dessa filer var inte bundna till något specifikt objekt inom spelet utan kunde istället kopplas till flera objekt. Detta tillät en effektiv utveckling då användarvänlig kod stod i fokus.

3.2 HTC vive

Ett par VR-glasögon krävdes inom projektet för att man skulle kunna uppleva den virtuella världen. Genom diskussioner bestämdes det att *HTC Vive*, [1], var det bättre valet då den har en inbyggd kamera som skulle användas för att spåra korten. Kameran på *HTC Vive*:s sitter i mitten längst ner på glasögonen vilket syns i Figur 3.1. Även *Oculus Rift* diskuterades men ansågs vara sämre då den var mindre kompatibel för det tänkta projektet.

HTC Vive var också det bättre valet då den är kompatibel med *Steam VR*, [6], som var VR-tillägget som gruppen hade bestämt att de skulle använda, på grund av att det var det tillägget som var mest bekant för gruppen. Tidigt i projektet visade det sig att kameran i VR-glasögonen inte gick att integrera i *Vuforia* och därmed behövde en webbkamera monteras på VR-glasögonen. För att webbkameran skulle sitta stabilt behövde den monteras uppochner. Detta medförde komplikationer för användaren då det blev svårare att veta var man ska hålla kortet för att man skulle kunna spåra det.



Figur 3.1: HTC vive

3.3 Vuforia

För att implementera spårningen av korten användes *Vuforia*, [7], som är ett tredjeparts API som integreras i *Unity*. Anledningen till att *Vuforia* användes var för att gruppen då inte skulle behöva spendera tid på att utveckla en egen spårningsmetod. Därmed kunde tiden istället läggas på att utveckla spelupplevelsen och att förbättra användarvänligheten hos kortgränssnittet. Ytterligare anledningar var att det fanns mycket god dokumentation tillgänglig, vilket underlättat både implementationen av spårningen samt utvecklingen i helhet. *Vuforia* är även gratis, vilket var lämpligt för detta projekt. För att karaktärer skulle skapas hos korten behövdes bilden som skulle spåras läggas in i en databas i *Vuforia* som sedan kopplades ihop med en modell i *Unity*. Därmed kunde korten spåras med hjälp av web-kameran och inne i den virtuella världen skapades modeller då korten kom inom ett visst avstånd till spelplanen.

3.4 Google Drive

Dokumentation, protokoll, gruppkontrakt och, till viss del, modeller lades upp på en delad mapp på *Google Drive*, [8]. I mappen lagrades det som inte gick att lägga in direkt i *Unity*. Förutom olika dokument och protokoll lades även grafiska element upp, som inspiration eller sketcher för hur det skulle se ut och basmodeller som andra i gruppen skulle kunna redigera. Beslut som tagits under projektets gång sparades i mappen. Dokumentansvarig hade som uppdrag att hålla mappen uppdaterad och organiserad. Anledningen till att *Google Drive* användes var för att alla gruppmedlemmar skulle ha god överblick på projektet. Samtidigt blir det enkelt att hålla koll på dokumenten om alla är organiserade och förvaras på samma ställe.

3.5 Trello

Trello, [9], är en applikation som kan användas för att sätta upp mål som användaren vill utföra och ett bra hjälpmittel för projekt som har utvecklats efter Scrum-metoden (vilket detta projekt har gjort). Programmet funkar som ett sätt där man, på ett organiserat sätt, kan ha god koll på sina mål samtidigt som det är lätt att lära sig och är användarvänligt. Delmålen som skapades för projektet lades in i *Trello*, där varje delmål kompletterades med en checklista, för att kunna slutföra det inom sprinten, samt vem som ansvarade för delmålet. I programmet skapades fas-spalter för att hålla koll på hur långt varje delmål hade kommit. I "Att göra"-spalten lades de delmålen som ej var påbörjade än men som antingen skulle göras eller som önskades att göra. "Pågående"-spalten innehåll de delmål som påbörjats men ej var klara. Innan något delmål markerades som klar sattes det upp för att testas och sedan lades detta målet i "Avklarad". För att alltid vara medveten om vilka kravspecifikationer som kund givit, fanns även en spalt där kravspecifikationerna stod.

I början av projektet var delmålen för generalisera, vilket orsakade att det var svårt för gruppmedlemmar att veta när ett delmål faktiskt var klart samt hur långt inom målet gruppen hade kommit. De delmålen som, av olika anledningar, inte blev klara i tid sparades som en framtida arbetsuppgift.

3.6 Kommunikationsmedel

Kommunikation inom gruppen gjordes först och främst via programmet *Slack*, [10], som finns både för dator och mobil. *Slack* använder olika kanaler för att möjliggöra flera diskussioner simultant. Projektgruppen använde till en början tre olika kanaler, en generell, en för möten och en för allmän diskussion. Kanalen för möten skrevs endast när och var ett möte skulle vara. Utöver det så skulle det inte förekomma några ytterligare konversationer i denna kanal, utan skulle det frågas något skedde det i en annan kanal. En viss förvirring uppstod då den generella kanalen och den för allmän diskussion användes på samma sätt. Därför valdes det att ta bort en av kanalerna.

Kommunikation med kund skedde främst mellan kundansvarig via mejl. Att bara en hade kontakt med kund var ett bra sätt att hålla kommunikationen. Detta underlättade när kund skulle kontaktas då kontaktpersonen redan var vald samt minimerade risken att flera kontaktade kund samtidigt eller att ingen tog kontakt.

Kapitel 4

Utvecklingsprocessen

En projektplan över projektarbetet sattes upp innan arbetet på produkten påbörjades. I projektplanen ingick principer, rutiner och krav samt en tidsplan för att underlätta arbetet för projektgruppen. Detta kapitel beskriver de krav och rutiner som projektet följt under utvecklingen.

4.1 Systemutvecklingsprinciper

Detta avsnitt beskriver de systemutvecklingsprinciper som projektet innehållt. Från projektplanen, som beskriver hur projektgruppen ska arbeta med systemutvecklingen, har ett antal principer och rutiner framtagits. Dessa principer och rutiner gäller kravhantering, versionshantering, testning och liknande.

4.1.1 Kravhantering

För att få ett underlag och för att förstå vad som förväntades av projektet skapades kravspecifikationer av kunden. Dessa specifikationer beskrev vad kunden förväntade sig av projektet, till exempel vad för funktioner som skulle finnas, hur användarupplevelsen skulle vara och hur gränssnittet skulle se ut samt spelets generella utseende. De grundläggande kraven som ställdes på projektet inkluderade ett helt digitalt spel i VR som körs i realtid med materiella kort som användargränssnitt. Anledningen till att spelet skulle vara i realtid är att det kommer göra användaren mer aktiv och man behöver inte vänta på att motståndaren ska bli klar med sin tur. Då blir det aldrig tid där användaren måste sitta och vänta utan kan alltid vara med i spelet och svara på motståndarens drag. Hela spelet designades utifrån dessa krav från början och utgör alltså dess grundstenar. Trots kraven så fanns ändå stort utrymme för kreativa idéer då kraven från kunden beskrev systembegränsningar, inte spelets design. För att försäkra sig om att kunden får vad de förväntade sig så var det viktigt att det fanns god kontakt, i form av möten, mellan projektgruppen och kunden. Under dessa möten presenterade gruppen arbetet i dess skede och kollade om det finns något som kunden tyckte skulle ändras. Det var viktigt att de förekom kontinuerliga möten där kunden fick reda på hur projektet gick samt att de kunde ge kritik. Kunden ansåg att många av de idéer projektteamet berättat om passade kundens önskemål. Några fler krav förutom de grundläggande kraven gav inte kunden. Detta innebar att största delen av produktens form och funktionalitet var helt upp till utvecklingsteamet att komma fram till. Idéerna över produktens form och funktionalitet som utvecklingsteamet haft diskuterades till viss mån med kund tidigt i implementationen.

4.1.2 Versionshantering

För att säkerställa högsta kvalitet på ett projekt så krävs ett tydligt sätt att hantera olika versioner och låta olika personer jobba på samma projekt samtidigt. Därmed behövs det ett versionshanteringssystem, där man kan uppdatera och förvara olika versioner av projektet. Ett versionshanteringssystem handlar om att alla utvecklare ska en överblick över projektet samt om att det är ett enklare sätt att dela kod mellan utvecklarna. Då detta projekt utvecklades i *Unity* så användes *Unity collab*, [12], för att hantera versionerna. Dessa verktyg behövdes även för att kunna ha en god överblick på projektet, då det var många som arbetade på det samtidigt. Anledningen till att *Unity Collab* valdes framför andra versionshanteringssystem, som till exempel *GitHub*, var att detta redan var integrerat i *Unity*. Detta gjorde processen att dela kod enkel och snabb och alla i koden kunde direkt se när det blivit ändringar och i vad dessa ändringarna tagit plats.

4.1.3 Testningsprinciper och QA

Quality assurance handlar om att försäkra att projektet håller en hög kvalitetsnivå under utvecklingen. I detta projekt var det viktigaste att korten som användes skulle kunna spåras på ett smidigt sätt samt att användarupplevelsen med kort som gränssnitt skulle vara så bra som möjligt. Det var även viktigt att koden som skrevs skulle vara strukturerad och lätförstådd för att underlätta vidareutveckling av projektet.

För att säkerställa att spelet höll en hög kvalitet krävdes det kontinuerligt testande under projektets gång. Testningen gjordes allt eftersom nya funktioner implementerades och utfördes av personen som utförde implementeringen. När en annan gruppmedlem vidareutvecklade en implementerad funktion granskades koden. Inga tester utfördes av utomstående testare, av anledningen att gruppen inte ansåg att det fanns tid för att utföra detta. Tiden lades istället ned på att implementera viktigare funktioner för att utöka spellogiken hos spelet. I slutet av varje sprint hölls en sprintåterblick där kod som skrivits och nya funktioner gicks igenom och diskuterades i hela gruppen. Under dessa möten säkerställdes att koden var väldokumenterad och följde kodstandarder samt gav övriga gruppmedlemmar chansen att ge kommentarer och konstruktiv kritik. Dessa möten fungerade även som ett tillfälle där gruppmedlemmarna kunde få en förståelse vad implementatören hade gjort.

4.2 Arbetssätt

Projektet utvecklades med verktyg och metoder för en agil arbetsmetodik. Detta avsnitt beskriver mötesrutiner, sprintplanering och dokumentationsprinciper.

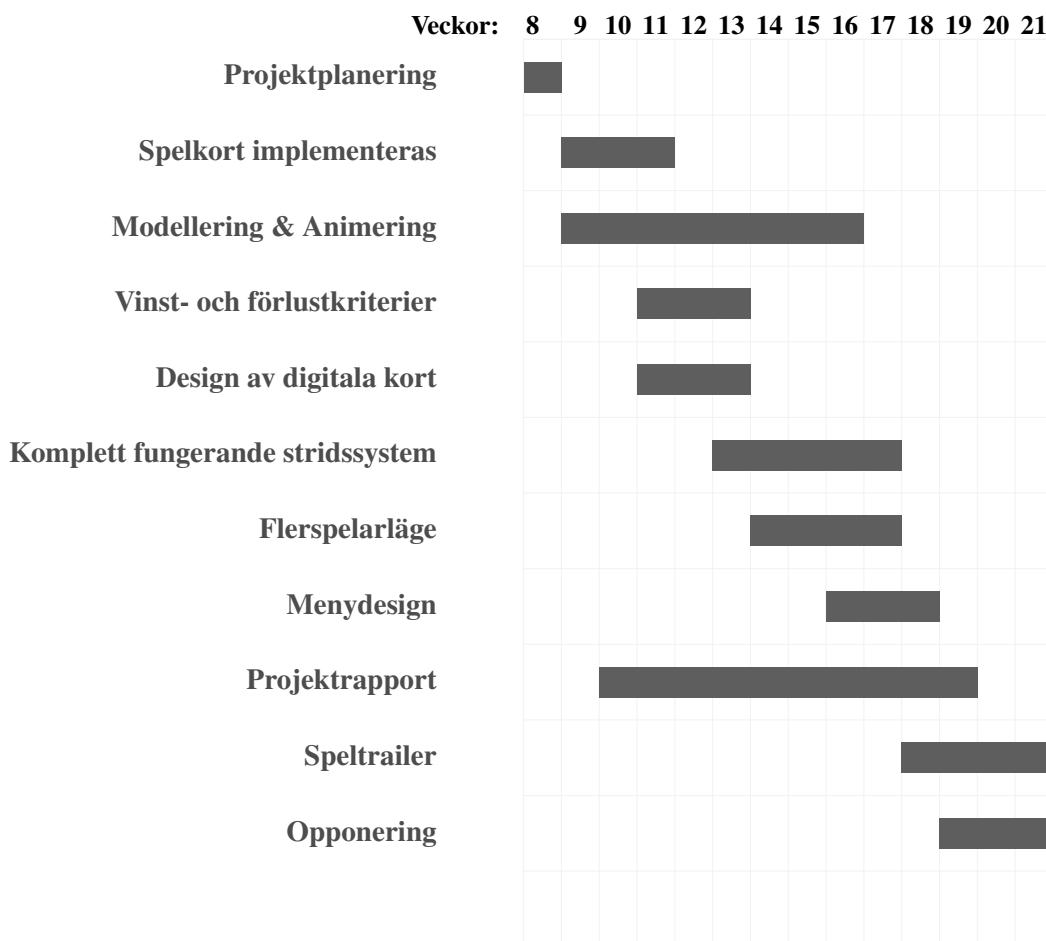
4.2.1 Mötesrutiner

Utvecklingen strukturerades efter den agila arbetsmetoden Scrum [11], vilket innebär att det i början av varje arbetsdag hölls ett kort Scrum-möte. I början av varje sprint hölls en sprintplanering och i slutet ett sprintåterblick. Projektgruppen träffades alltid på samma plats varje morgon för att hålla ett uppstartsmöte och skapa en rutin. Under uppstartsmötet planerades sprinten, Trello fyldes på med uppgifter, ansvarsområden delades ut och arbetspass schemalades. Under sprintåterblicken presenterades och diskuterades de framsteg som gjorts under sprinten. Detta möte gav alla utvecklare möjligheten att få kod förklarat samt ett tillfälle att granska den skrivna koden. Utöver dessa möten hölls även ett lunchmöte varje vecka, vilket huvudsakligen var för beslutstagning och för att diskutera potentiella frågor som dykt upp under projektets gång.

4.2.2 Sprintplanering

Varje sprint inleddes med en sprintplanering. Under sprintplaneringen granskades projektet i förhållande till projektplanen, se tabell 4.1, för att se att det låg i fas. Utifrån granskningen valdes ett antal arbetsuppgifter som skulle slutföras under sprinten. Dessa arbetsuppgifter delades ut mellan projektmedlemmarna. Projektplanen gjordes i början av projektet och har använts som utgångspunkt för sprintarna. Vissa moment i projektet tog längre tid än planerat och vissa moment gick fortare än tänkt, men överlag stämmer tidsplaneringen. Tidsplaneringen ger en överblick över vad som varit fokusområdena.

Tabell 4.1: Tidsplanering



I projektets början var en sprint en vecka lång. Efter sprintgranskningen, som gjordes efter varje sprint, ansågs en-veckors-sprintar vara för korta. Gruppmedlemmarna hann endast komma igång med arbetet när det sedan var dags för nästa sprint med nya arbetsfördelningar. För att skapa mer kontinuitet i arbetet beslutades det att förlänga sprintarna till två veckor. Detta gjorde att gruppmedlemmarna hann sätta sig in i andras implementationer och vidareutveckla dessa under sprintens period. Arbetet blev, med förlängning av sprintar, mer jämn och fick ett bättre flöde.

För att ha en kontinuerlig granskning av implementerat material roterades de olika arbetsområdena mellan gruppmedlemmarna. Efter arbetsuppgifterna var avklarade bestämdes ett arbetschema då gruppen skulle arbeta. I början av projektet schemalades för mycket tid, vilket orsakade oordning inom gruppen. Längre in i projektet utsattes mer tid till andra aktiviteter för att säkerställa att på den schemalagda tiden jobbade alla på projektet.

4.2.3 Dokumentationsprinciper

Under hela projektets gång krävdes det tydlig och kontinuerlig dokumentation av både kod och andra dokument som berör projektet. Välstrukturerad och tydlig dokumentation av koden innebär att man alltid börjar varje kodfil med en förklaring av vad som kommer hända i den filen. Utöver detta krävdes det även kommentarer i koden samt välvalda variabelnamn. Om dokumentationen håller en bra kvalitet så minimeras tiden för andra utvecklare att sätta sig in i koden samt att det även blir enklare att förklara vad som gjorts för kunden. God dokumentation innebär också att variabler och funktioner har tydliga namn som beskriver vad de ska göra och hur de fungerar. Dokumentationen skedde samtidigt som utvecklaren skrev i koden, när en ny fil skapades skrev utvecklaren vad som skulle ske i filen och samtidigt som funktioner skapades beskrevs vad de skulle göra. Utöver att det blir enklare att tolka koden så blir också problemsökningen lättare samt refaktorisering och effektivisering av kod.

Under hela projektet fördes det protokoll över både lunchmötena och sprintplaneringsmötena. Dessa protokoll sparades på Google Drive och var främst till för att förtydliga arbetet och beslut som tagits.

Kapitel 5

Implementation

Inom implementation presenteras utvecklingen av spelprojektet. Det görs genom att framställa idén bakom spelet, utveckling av spelkorten, valet av grafisk stil samt modellering och animering av 3D-objekt.

5.1 Kravspecifikation från kund

Projektet började med ett möte med kund där idéer och tankar kring slutprodukten diskuterades. Utifrån detta möte ställdes en kravspecifikation på produkten upp. Kraven som ingick i specifikationen var att det ska vara ett VR-spel, innehålla en sten, sax, påse liknande stridsmekanik, ha ett flerspelarläge för två spelare, använda fysiska kort som kontroller och att spelet ska spelas i realtid. Dessa krav har satt grunden för hela projektet och genomsyrar hela produkten.

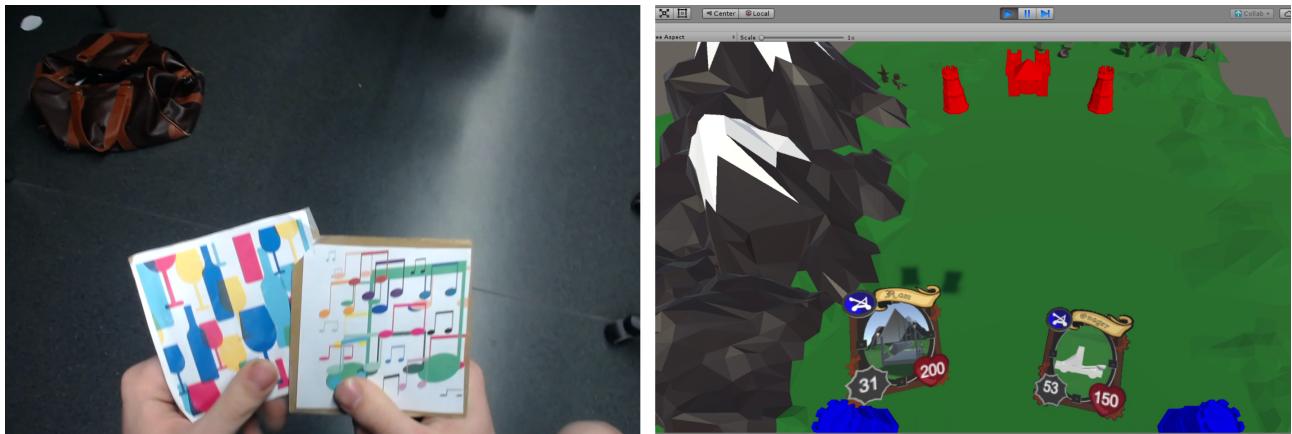
5.2 Spelidé

Första steget i utvecklingen var att skapa en intressant spelidé som passade kraven från kunden. Efter en vecka individuellt arbete av att spåna idéer så träffades gruppen och diskuterade gemensamt en spelidé. Ett val som projektgruppen ansåg var självklart var att utveckla ett strategispel där taktik står i fokus. Detta val gjordes utifrån kundens krav att skapa ett realtids VR-spel med spelkort som användargränssnitt. När det gäller VR-spel, som dessutom har ett ovanligt användargränssnitt, ansåg gruppen att ett strategispel är att föredra då det tillåter en lugn och metodisk spelupplevelse. Ett spel som kräver snabba reaktioner skulle inte fungera tillsammans med spelkorten då kontrollerna, i form av spelkorten, inte känns tillräckligt responsiva. Spelkorten valdes att representera olika typer av undersåtar som kan tas i bruk av de två spelarna som möter varandra. Undersåtarna delas in i tre kategorier, belägringsvapen, trupper och byggnader. Dessa olika typer av undersåtar har fördelar samt nackdelar mot varandra när de möts i strid. Därmed måste spelarna placera sina undersåtar taktiskt för att vinna. Framför spelarna finns ett digitalt slagfält med varsin huvudbas. Spelarnas mål är att eliminera motståndarens bas genom att placera kort på slagfältet och framkalla undersåtar som då anfaller motståndaren. Vid placering framkallas först en undersåte och sedan uppdateras kortet dynamiskt till en ny slumpmässig undersåte, vilket medför att spelaren inte ska behöva släppa och dra nya kort. Istället har varje användare alltid tre kort på hand som dynamiskt uppdateras under spelets gång. För att lära den oerfarne användaren bör även en liten instruktion över hur spelet körs finnas tillgänglig i början av spelet.

5.3 Utveckling av spelkort

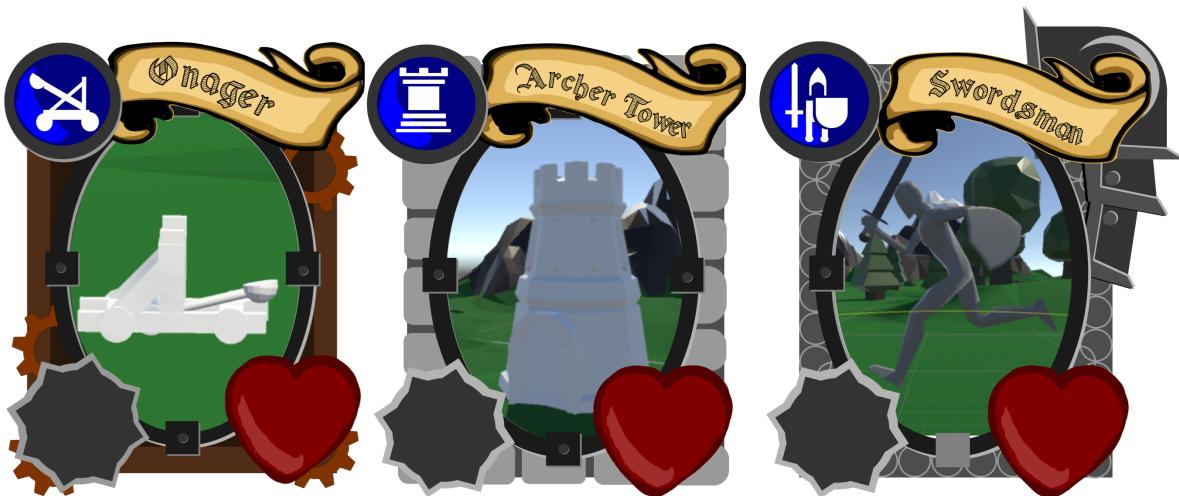
Implementeringen av spelkorten har haft stor fokus i projektet då dessa har stor påverkan på spelupplevelsen. Spelkorten fungerade som hela gränssnittet för spelet och därmed var det viktigt att det var användarvänligt och lätt att förstå.

Utanför den virtuella världen har spelaren kartongbitar med fasttejpade bilder som kan läsas av och läggas in i spelet. I spelet spåras bilden och ett digitalt kort syns. När kortet läggs på spelplanen betraktas det som spelat. Spelet är därför ganska enkelt eftersom att spelkortssystemet inte är lämpligt för en större kontroll av spelvärlden. Eftersom att all interaktion sker med kortet så styrs även menyn med olika kort. Figur 5.1 visar hur korten i verkligheten ser ut i den virtuella miljön



Figur 5.1: Till vänster syns korten i verkligheten och till höger korten i virtuell miljö

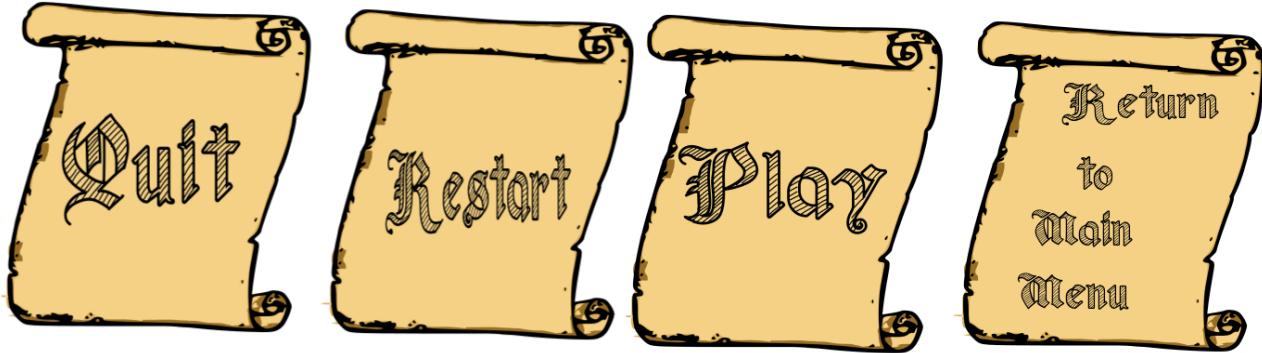
Spelkorten har två olika typer soldatkort och menykort. Soldatkorten i Figur 5.2 är de korten som spelaren använder i spelet för att lägga ut soldater på slagfältet. Dessa kort är indelade i tre olika typer: Siege, Infantry och Building.



Figur 5.2: Soldatkort

Varje kort har ett namn, en ikon, en siffra som visar styrka och en siffra som indikerar hälsa. Namnet anger vilken typ kortet representerade, samt en ikon som visar typen. Siffran som visar styrkan på undersåten visas i den nedre vänstra hörnet, medan siffran som visar hur mycket hälsa undersåten har

visas i nedre högra hörnet. Korten i 5.3 visar basen som används i spelet. Siffrorna dyker upp under spelets gång. Menykorten har en text som beskriver handlingen som de utför, se Figur 5.3.



Figur 5.3: Menykort

Soldatkorten som spelaren har på handen byts ut mot menykort när spelet är slut och när spelaren startar spelet. Alla korten är designade i Scalable Vector Graphics-format SVG-format med programmet *Inkscape*, [13].

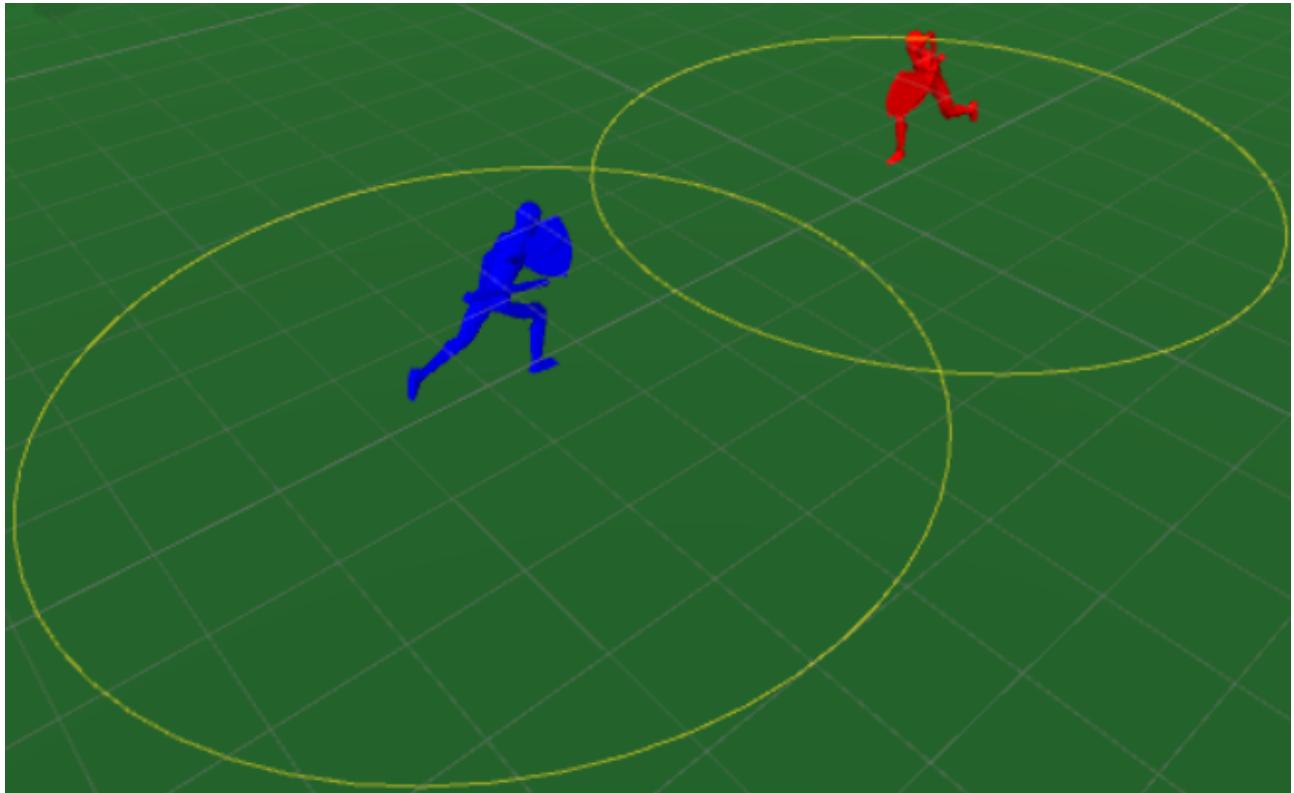
5.4 Stridssystem

Stridssystemet inom spelet fungerar på så vis att varje karaktär har som standard att spring framåt och attackera allt dem ser. Varje karaktär har en aggressionsradie, så kallad aggro range, runt sig som består av en kollision sfär som visas som en gul cirkel, se Figur 5.4.

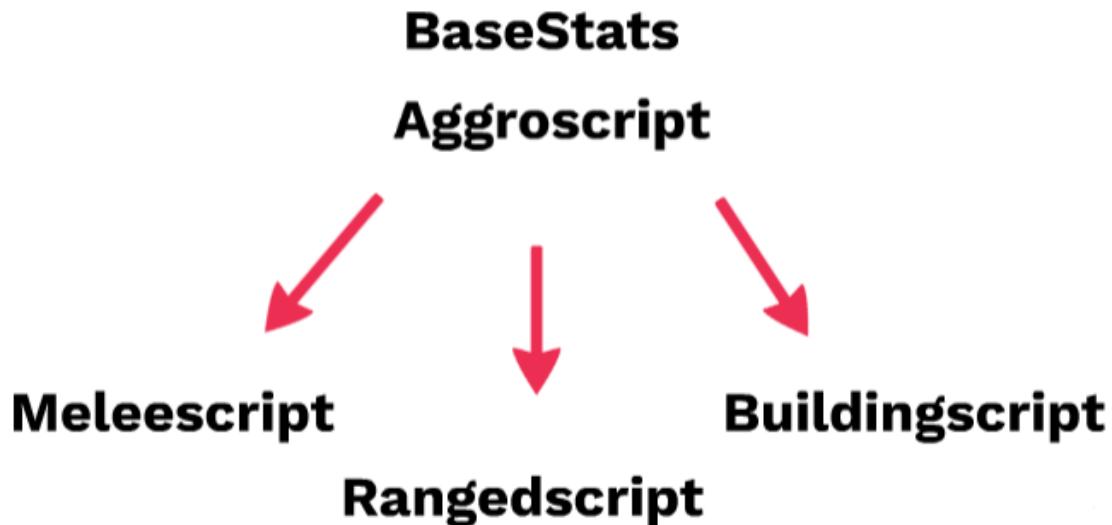
När andra objekt kolliderar med den gula kanten ger det ifrån sig ett event. Vid detta event undersöks vissa kriterier, som till exempel är objektet fientligt eller är karaktären redan upptagen med att slåss med någon annan. Om alla kriterier körs utan att funktionen avslutas så ska karaktären anfalla fienden som gick in i aggressionsradien. Karaktären rör sig då mot fienden och när den kommer tillräckligt nära så börjar den göra skada. Om fienden dör körs en Find Nearest Target-funktion som hämtar närmsta fiende inom aggressionsradien. Om ingen fiende hittas så återställs karaktären till att springa framåt. Stridsmekaniken, enligt kundens krav, fungerar enligt ett sten-sax-påse system. Det betyder att de olika typerna av karaktärer har svagheter och styrkor mot varandra. Trupper är starka mot belägringsvapen, belägringsvapen är starka mot byggnader och byggnader är starka mot trupper.

5.5 Klasshierarki

För att få bättre struktur över koden så följer projektgruppen en hierarki för som visas i Figur 5.5. Den här hierarkin används för alla levande objekt i världen som till exempel svärdsmännen eller vakttornen.



Figur 5.4: Undersåtars aggressionsradie visas med en gul cirkel



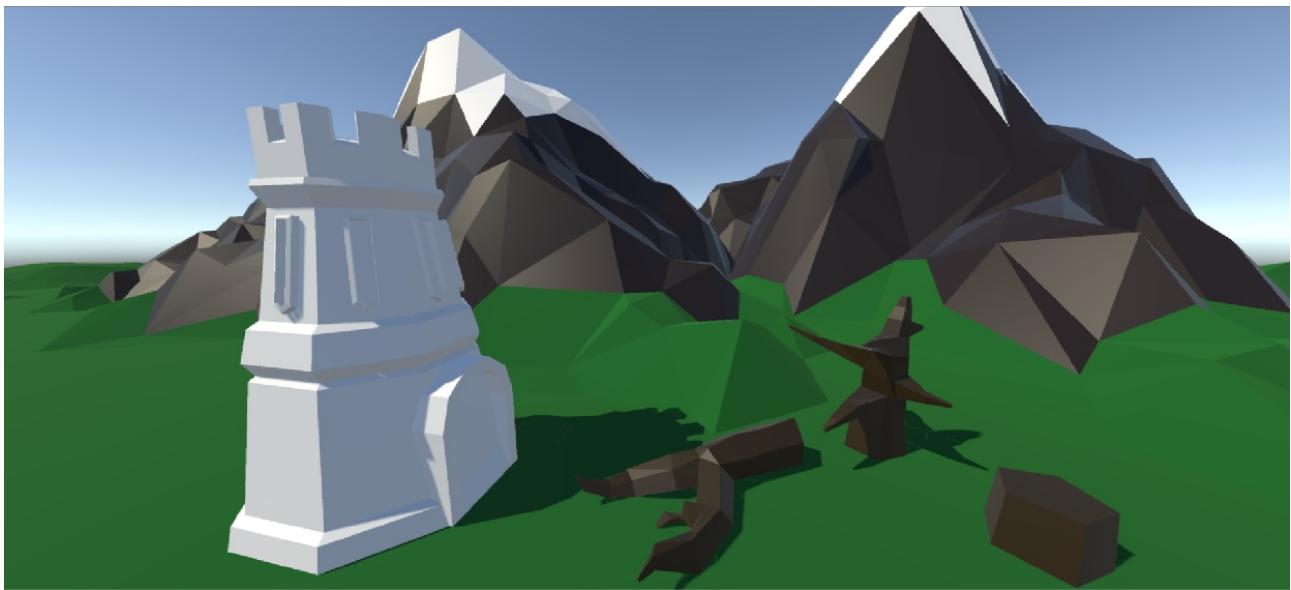
Figur 5.5: Klasshierarki för kod gällande levande objekt

Längst upp i hierarkin är BaseStats och Aggro script. BaseStats innehåller variabler för bland annat liv, skada, vilken spelare karaktären tillhör och längden på aggro range. Det valdes att ha all statistik för karaktärer samlade på ett ställe och låta andra scripts hämta all den data de behövde. Alternativet skulle vara att ha variabler utspridda i olika filer. Det skulle bli lättare att koda, men mycket svårare att organisera. Det andra scriptet längst upp i hierarkin är aggro script. Den används av karaktärer som ska kunna attackera och ger den gula cirkeln som visar räckvidd. Nästa steg i hierarkin är aningen MeleeScript, RangedScript eller BuildingScript. Valet beror på vilken typ av karaktär som ska

skapas. MeleeScript används av svärdsmän och murbräckor. Rangedscript används av katapulter och Buildingscript används av vakttornen och huvudbyggnaden. Dessa tre script skulle även kunna vidareutvecklas att inkludera fler subklasser. Men det var inte nödvändigt för detta projekt. Klasser för att hantera VR, spelkorten och AI använde sig inte av någon hierarki utan bestod av enskilda scripts.

5.6 Val av grafisk stil för 3D-objekt i en VR-miljö

För att VR-teknik ska upplevas realistiskt och ej framkalla illamående hos användaren krävs höga systemspecifikationer [14]. Vid användning av *HTC Vive* rekommenderas att behålla en bildfrekvens på 90 bilder per sekund som visas på två skärmar samtidigt, en för varje öga. För att behålla dessa höga krav utan behovet av extrem hårdvara så skapades spelet med en minimalistisk stil, se Figur 5.6. Det betyder att modeller byggs med få antal polygoner och renderas med enkel ljussättning. Denna enkelhet hos modeller sänker specifikationskraven vid körning av spelet. Kostnaden blir att karakterer kan vara svåruppfattade och se ut att vara orealistiska. Detta kan istället återfås genom att fokusera det artistiska arbetet på att skapa realistiska animationer.



Figur 5.6: 3D-objekt med minimalistisk stil

5.7 Modellering och animering

Modellerna som finns i spelet har modellerats i *3Ds Max*, [15], och *Maya*, [16], och sedan exporterats till *Unity*. Som det nämnts i 5.6 utvecklades modellerna i produkten med ett långt antal polygoner för att inte tappa prestanda. Modellerna texturerades inte för att ge ett bättre helhetsintryck. Enkla färger kombinerat med en enkel ljussättning fick alla polygonerna att synas tydligt i spelet och gav en minimalistisk stil, vilket kan ses i Figur 5.6.

All animering gjordes i *Maya* då det inte fanns stöd för export av animationer mellan *3Ds Max* och *Maya*. Objekten som animerats var svärdsmännen, katapulterna och murbräckan. Animationerna består av minst en gång- och en attackanimation.

Kapitel 6

Resultat

Slutprodukten blev mycket likt det som beskrivs i avsnitt 5.2, med undantaget av flerspelsläget. Spelaren som tar på sig VR-glasögonen möts av en menyskärm när applikationen startas, vilket uppfyller kravet för VR. Menyskärmen innehåller en tom spelplan och två kort, där spelaren kan välja, med hjälp av korten, att antingen starta en spelomgång eller avsluta, se 6.1.

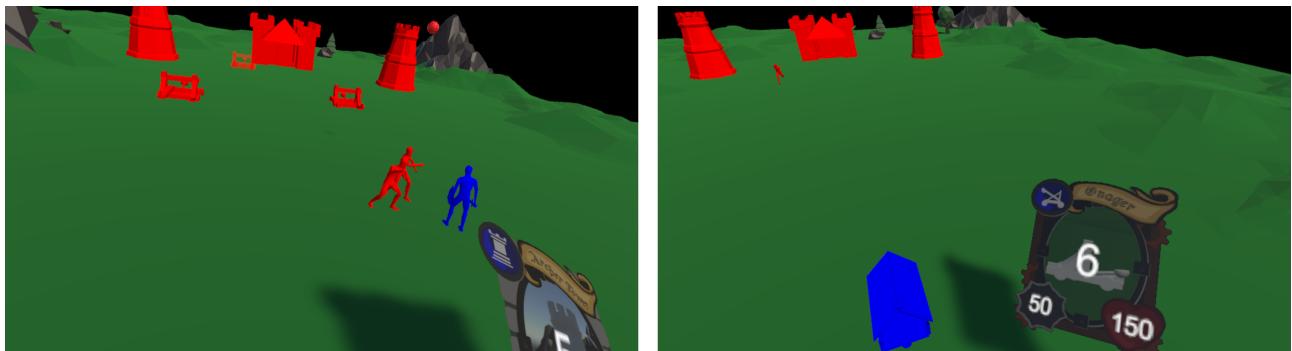


Figur 6.1: Menyskärmen när applikationen startas

Korten dyker upp i VR när de fysiska korten har spårats av kameran på HTC Vive:n. De virtuella korten följer rörelserna från de fysiska korten i VR-miljön. Då de fysiska korten används för att kontrollera spelet uppfylls kravet för att ha fysiska kort som kontroller. I spelvärlden kommer användaren kunna röra sig runt spelplanen och modellerna som skapas kommer se ut som minifigurer som rör sig på spelplanen.

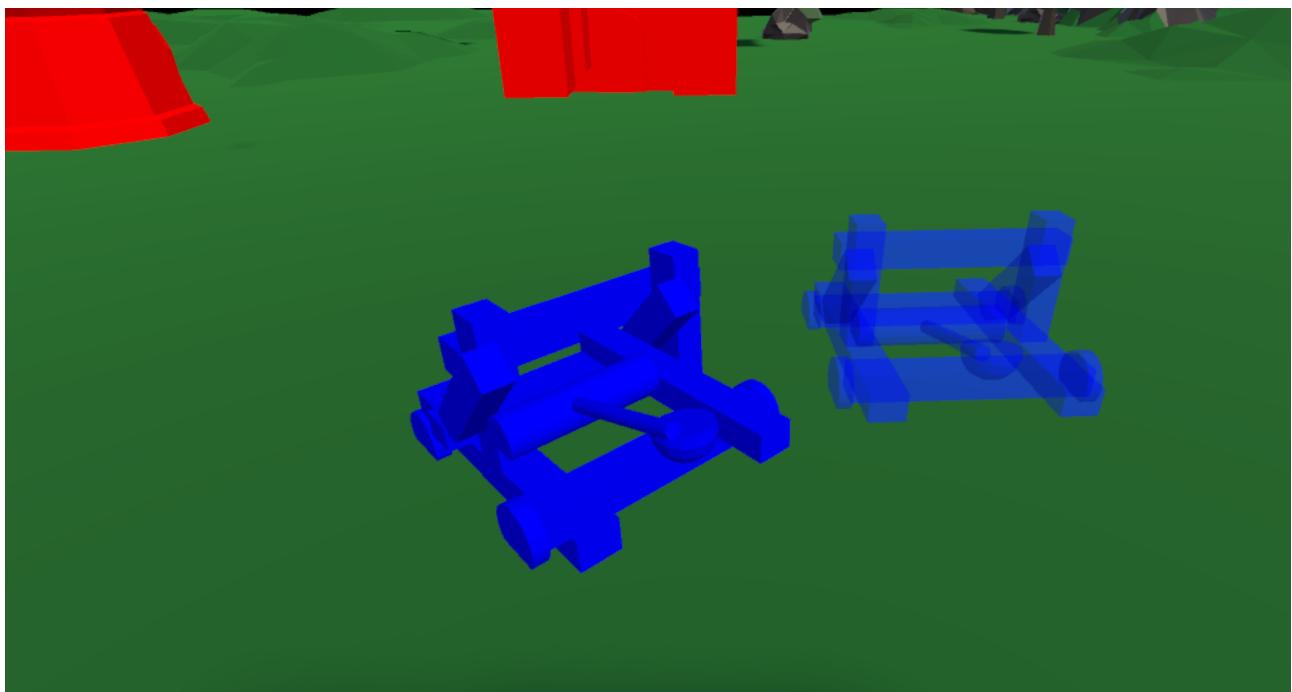
Spelomgången startas när kortet placeras på spelplanen. När en omgång startas dyker det upp två baser, en för spelaren och en för motståndaren. Spelaren har tre kort på hand som slumpas mellan de olika typer av undersåtar som är spelbara.

Vilken sort som kortet representerar syns på bilden som dyker upp, vilket syns i 6.2. På bilden står det även styrka och liv för undersåten. För att generera en undersåte placeras respektive kort på planen.



Figur 6.2: Scen från spelskärmen när spelet är igång

Där kortet placerats dyker en modell av undersåten upp. Undersåten är lätt genomskinlig när den genereras och gradvis blir solid, se 6.3.

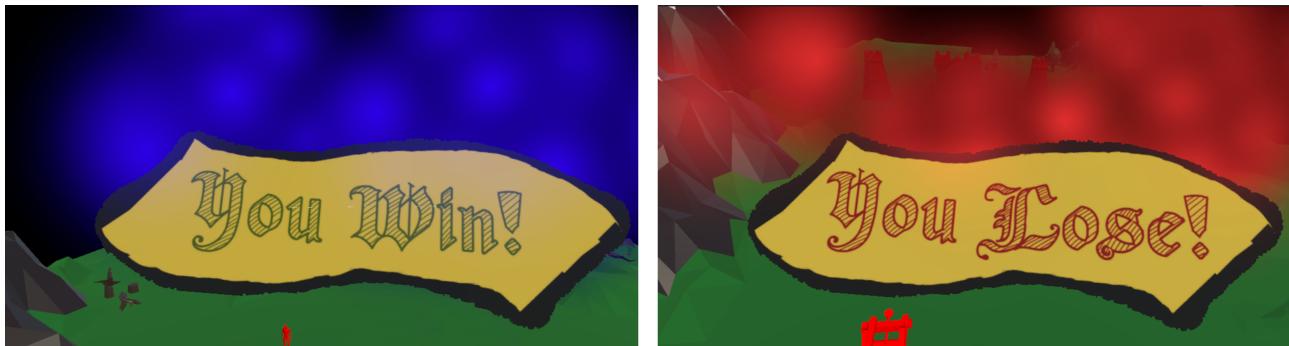


Figur 6.3: Den högra undersåten håller på att genereras

En genererad undersåte kommer per automatik att röra sig mot andra sidan av planen. Kortet som används byter bild och en timer sätts på innan nästa undersåte kan läggas ut på plan. Ju längre tid ett kort är på hand desto starkare blir undersåten när den väl spelas ut. Vart spelaren lägger korten är också en bidragande faktor. En undersåte som läggs långt borta från basen tar längre tid på sig att genereras. Under tiden en undersåte genereras kan motståndarens undersåtar attacker orsaka den skada. De olika typerna av undersåtar är ger olika mycket skada beroende på vilken typ de möter, vilket uppfyller kravet om sten, sax, påse liknande stridsmetodik. Varje undersåte har en attackradie och om en fiende kommer innanför denna radie går undersåten till attack.

Motståndaren som spelaren möter är en AI, en artificiell intelligens. AI:n är programmerad för att försöka ta sönder spelarens bas. När spelaren lägger ut en undersåte kommer AI:n ibland kontra draget med undersåtar som är starka mot de som spelaren lägger ut. På grund av detta blir den en formidabel motståndare. Under utvecklingsfasen av AI:n testades flera olika svårighetsnivåer tills en som upplevdes lagom svår hittades. Svårighetsgraden testades av projektgruppen och kunden. Omgången tar

slut då antingen spelaren eller motståndarens bas har förstörts. Beroende på om spelaren vunnit eller förlorat dyker en vinst- eller förlustskärm upp, se Figur 6.4.



Figur 6.4: Skärmbilderna som dyker upp vid vinst eller förlust

Samtidigt ändras två kort till ett ömstarts-kort och ett gå till menyn-kort. Vid omstart, startas en ny omgång av spelet och det andra kortet leder spelaren tillbaka till menyskärmen.

Spelet går i dagsläget endast att spela ensam vilket orsakar att ett av de initiala kraven ej uppfylls. Huvudorsaken för detta är för att endast en HTC Vive finns tillgänglig att testa med, vilket beskrivs i avsnitt [7.1](#).

Den slutgiltiga produkten renderas i realtid. Detta uppfyller det sista kravet för produkten, om att spelet ska köras i realtid.

Kapitel 7

Analys och diskussion

I detta kapitel analyseras och diskuteras resultatet och spelet diskuteras utifrån etiska och samhälleliga aspekter.

7.1 Analys av resultatet

Vissa ändringar och begränsningar har tillkommit under utvecklingen. Antalet spelkort som spelaren har på handen begränsas av kamerans möjlighet att spåra korten. Grundidén för spelet var att spelaren skulle ha fem kort på handen och på så sätt ha många alternativ och strategier att välja mellan. I det slutgiltiga spelet har spelaren tre kort på handen för att spelaren ska kunna se hela handen. I spelet används korten enbart för att bestämma undersåtarnas position. Om spelaren skulle styra undersåtarnas rörelser och attacker skulle det behövas mycket fler spelkort och spelaren skulle ha svårt att ha koll på hela handen.

Antalet olika undersåtar har begränsats till fyra sorter (murbräcka, svärdsman, torn och katapult). Anledningen till begränsningen är att varje soldat behöver animation, styrkevärdens och modell. En idé som gruppen tidigt hade var att det skulle finnas två olika lag odöda och levande. Lagens skulle ha olika enheter och därför ha egna för- och nackdelar. Eftersom det skulle innebära dubbelt så mycket arbete för modellering och animering så valdes den idén bort. Det skulle även kräva mer av stridsfunktionaliteten i spelet. Tanken med spelet var från början att två spelare skulle kunna spela mot varandra. Eftersom att det bara fanns ett headset att testa med valde vi att en spelare spelade mot en AI istället.

Kameran som finns i en *HTC Vive* har för låg upplösning för att kunna spåra spelkorten. Därför behövs det en webbkamera som limmas fast på headsetet. Webbkameran behöver vinklas rätt för att spelaren ska känna att den representerar dennes ögon annars har spelaren svårt för att placera korten rätt utifrån sitt synfält.

Det slutgiltiga spelet har bra upplösning och bildfrekvens tack vare det låga antalet polygoner hos modeller. Spelet ser mer stiliserat än verklighetstroget ut men spelarens upplevelse påverkas inte av det begränsade antalet detaljer.

7.2 Metod

I början av projektet uppstod flera tillfällen då inte hela gruppen hade uppfattat att ett arbetspass skulle hållas. Detta ledde till att gruppen inte var fulltalig vid så många arbetspass i detta skede av projektet. Till detta finns ingen bättre förklaring än bristande kommunikation. Detta åtgärdades när

det upptäcktes att det inte endast handlade om enskilda fall utan hände ofta. Hela gruppen var efter detta mycket mer noga med denna kommunikation.

Ett annat problem i det tidiga stadiet av utvecklingen var att förseningar var vanligt förekommande. När projektet planerades bestämdes att det inte skulle vara något ”straff” för förseningar och att gruppmedlemmarna skulle ta eget ansvar. Efter flertalet förseningar bestämde sig dock gruppen att försenad gruppmedlem ska bjuda på fika.

I starten av projektet fungerade uppdelningen av arbetsuppgifter mindre bra. För att varje gruppmedlem skulle få chansen att testa på alla arbetsuppgifter så byttes arbetsuppgifter i början av varje sprint. Gruppmedlemmarna fann dock att det var svårt att infinna sig och förbereda sig för en ny roll så ofta. För att motarbeta detta så förlängdes sprintarna och arbetsuppgifterna skiftades inte lika ofta. Detta gjorde arbetet smidigare och förhindrade missförstånd.

En stor del av arbetet skedde individuellt i grupp. Alltså att varje gruppmedlem arbetade med sitt egna ansvarsområde men gruppen satt ändå samlad på samma plats. Detta medförde att det var enkelt att föra en konversation och be om hjälp eller liknande under arbetets gång oberoende av vad varje gruppmedlem arbetade med.

Strukturen på filerna i projektet var mycket bra. Genom sin tydliga struktur förhindrades att någon gruppmedlems arbete skulle påverka någon annans av misstag. Det var enkelt för gruppmedlemmarna att snabbt veta vilka filer som var möjliga att redigera utan att skapa hinder för någon annan. Det gjorde att arbetet i filerna kunde ske smidigt och problemfritt.

7.3 Problem och hinder

Inom projektgruppen uppkom en del logistiska problem som vilka som skulle jobba och när samt förseningar till arbetspassen inom gruppen. I projektet uppstartsfas var det svårt att finna arbetsuppgifter till alla projektmedlemmar, men detta lösades automatiskt då projektet växte i storlek.

I början av projektet beslutades att sprintarnas längd skulle vara en vecka. Ett problem som uppstod då var att utvecklarna inte hade tid att både sätta sig in i problemet och lösa det under en sprint. Detta betydde att många uppgifter lämnades över halvfärdiga till nya personer mellan sprintarna. Arbetet blev således väldigt ineffektivt. För att lösa detta beslutades att sprintarnas längd skulle ökas till två veckor.

På grund av saknad erfarenhet av modellering och animering inom gruppen gjordes misstaget att olika utvecklare använde olika programvara för detta. Det upptäcktes senare att riggningen av modeller inte kunde exporteras mellan dessa modellingsprogram, vilket ledde till logistiska problem. När detta problem uppstod beslutades att hela gruppen skulle använda samma program (*Autodesk Maya*).

Ett av de största problemen som uppstod under projektet var att implementationen av spelkortens spårning tog längre tid än planerat. Detta orsakade att implementationen av flerspelar-läget blev uppskjutet. Denna uppskjutning gjorde att flerspelarläget inte har hunnit testas ordentligt och inte uppnått den nivå som utvecklarna önskade.

Förseningen skedde främst på grund av utvecklarnas bristande erfarenhet med *Unity*. Till en början försökte utvecklarna använda inbyggd funktionalitet för att lösa problemet. Funktioner som behövde användas var ofta antingen ”gömda” bakom *Unitys* användargränssnitt eller obefintliga. Efter många olika försök beslutades att lösa problemet helt via script. Anledningen till att detta inte gjordes från början var för att försöka förenkla hanteringen av de olika koordinatsystemen (i själva verket gjorde det hanteringen mer invecklad).

En annan anledning till att flerspelar-läget inte har testats ordentligt är att endast en *HTC Vive* fanns tillgänglig, och varken tid eller resurser för att införskaffa en till fanns inte.

Från början var det tänkt att spelaren skulle ha fem kort på handen. Men det visade sig att spårningen hade problem med att finna så många kort samtidigt, vilket gjorde det mer besvärligt att hålla koll på korten i spelet. För att lösa detta fick antalet spelkort minskas till tre kort vilket var enklare att spåra och mer hanterligt för spelaren. Fler kort än tre på handen upplevdes som svårhanterligt vid utvecklingen, även fast korten är lite större än vanliga spelkort.

7.4 Arbetet i ett vidare sammanhang

Något som oftast förbises men som är viktigt att diskutera är etiska och samhälleliga aspekter relaterade till arbetet. Detta inkluderar människors liv och hälsa, samhällets funktionalitet, rättssäkerhet, mänskliga fri- och rättigheter samt miljö. Därför, utifrån en säkerhetsfråga, finns slagfältet i spelet endast digitalt och inte som ett verkligt objekt. Detta valdes eftersom vid användning av VR-glasögonen kan inte användaren se verkligheten och riskerar därmed att skada sig själv och sin omgivning vid oförsiktighet. Då spelet riktar sig till ovana spelare är detta särskilt viktigt. I termen ovana spelare inkluderas bland annat barn som målgrupp. Då spelet planeras att innehålla ett flerspelarläge måste några försiktighetsåtgärder tas. En sådan försiktighetsåtgärd är att spelet inte ska innefatta någon chattfunktion då detta kan leda till spridning av diskriminerande kommentarer till och inom målgruppen.

VR-glasögon ses fortfarande som en svåråtkomlig hårdvara på grund av sitt höga pris och prestandan som krävs av datorerna. Prestandan som krävs för VR-glasögonen sänks med nyare och billigare versioner av glasögonen. De billigare versionerna erbjuder fortfarande upplevelsen av att man är i en virtuell värld men har inte samma valmöjligheter och realism som de dyrare alternativen. Att utveckla ett spel till de billigare VR-glasögonen ger en bredare konsumentgrupp men reducerar möjligheten att utveckla avancerade spel.

Skulle arbetet vidareutvecklats finns det ett par saker som gruppen hade velat implementera. Det viktigaste gruppen hade velat implementera är ett flerspelarläge. Då hade spelet blivit mer strategiskt och därmed även att användaren kan svara på motspelarens handlingar. Ännu något som gruppen hade velat implementera är utökad spelmekanik. Alltså göra spelet lite mer komplicerat med fler val för att göra spelet mer intressant. Detta skulle även leda till att man fick ut mer om man ville spela spelet flera gånger.

Kapitel 8

Slutsatser

Projektarbetet svarade på frågeställningarna som ställdes inför arbetet. Detta ger en återkoppling av frågeställningarna.

Hur begränsas speldesignen av att använda materiella spelkort som ska spåras som användargränssnitt?

Till skillnad från vanliga kontroller sätter materiella spelkort ett större krav på spårningen för att det ska fungera. Spelkorten begränsar användaren vilket kan vara bra då det blir lättare att för användaren att förstå funktionaliteterna men samtidigt leder till fler steg för att ändra något. Då ett kort representerar ett alternativ krävs en väl genomtänkt plan för hur de bäst fungerar som kontroller. Vanliga användargränssnitt har fler valmöjligheter än de materiella spelkorten. Det skulle till exempel kräva många fler kort om användaren haft en karaktär att styra med korten.

Att använda kort för att generera karakterer som det görs i projektet är ett intuitivt sätt att använda kort på, då många bräd- och kortspel fungerar på ett liknande vis. Materiella spelkort är inte ett optimalt användargränssnitt för spel eller produkter som kräver snabba rörelser då risken att VR-headsetet förlorar spårningen av korten ökar med snabba och ryckiga rörelser. Att ha för många kort på hand ökar också risken att spårningen av korten förloras, dels för att det blir många kort att spåra och dels för att korten, med största sannolikhet, kommer skymma varandra i hand.

Kan kameran i en *HTC Vive* användas för att uppfylla kraven som ställs för att utföra spårning av spelkorten?

Tidigt i projektet upptäcktes det att kameran i *HTC Vive:n* inte gick att hitta i de valda verktygen. Som en lösning på problemet monterades en extern webbkamera, som var kompatibel med verktygen, fast på VR-glasögonen. Detta beskrivs mer detaljerat i avsnitt 3.2.

Hur ska den grafiska stilens för spelets 3D-objekt väljas för att bäst passa en VR-miljö?

VR-spel bör generellt att spelas upp med en bildfrekvens på 90 bilder i sekunden per öga för att inte orsaka åksjuka. Detta kräver att hårdvaran som används för att rendera bilderna har en bra prestanda. Ett sätt att minska de höga kraven på prestanda är att skapa enklare modeller med färre polygoner, då det blir mindre komplicerade objekt som ska renderas i realtid. Genom att sänka grafikens nivå genom att ha simpla modeller, material, texturer och ljussättning går det att nå upp till en behaglig bildfrekvens. I VR är inte den grafiska stilens i fokus utan snarare känslan av att befina sig i en virtuell värld.

Litteraturförteckning

- [1] *HTC Vive*, HTC Corporation © 2011-2017, hämtad: 2017-02-14
<https://www.vive.com/eu/>
- [2] Adam Alsegård, Jonathan Grangien, Joakim Konac, Love Lidberg, Måns Löglund, Victoria Waldemarsson, Benjamin Wiberg, *TowerVR*, Linköping Universitet, Vårterminen 2016, hämtad: 2017-05-10
http://webstaff.itn.liu.se/~karlu20/courses/TNM094/reports/TNM094-2016-G_TowerVR.pdf
- [3] *Clash Royale*, Supercell, hämtad: 2017-05-10
<https://clashroyale.com/>
- [4] *Unity*, Unity Technologies ©2017, hämtad: 2017-05-10
<https://unity3d.com/webplayer/>
- [5] *Visual Studio 2015 Tools for Unity*, Microsoft ©2017, hämtad: 2017-02-11
<https://marketplace.visualstudio.com/items?itemName=SebastienLebreton.VisualStudio2015ToolsforUnity>
- [6] *SteamVR*, Valve Corporation ©2017, hämtad: 2017-05-10
<http://store.steampowered.com/steamvr>
- [7] *Vuforia Augmented Reality SDK*, PTC Inc ©2017, hämtad: 2017-05-10
<https://www.vuforia.com>
- [8] *Google Drive*, Google ©2017, hämtad 2017-05-10: <https://www.google.se/drive/about.html>
- [9] *Trello*, Atlassian ©2017, hämtad 2017-05-10: <https://trello.com>
- [10] *Slack*, Slack Technologies ©2017, hämtad 2017-05-10: <https://slack.com>
- [11] K. Schwaber och J. Sutherland, *The Scrum Guide*, Scrum.Org and ScrumInc., 2016-07, hämtad: 2017-02-11
<http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-SE.pdf>
- [12] *Unity Collab*, Unity Technologies ©2017, hämtad 2017-05-10:
<https://unity3d.com/services/collaborate>
- [13] *Inkscape*, GNU General Public License ©2017, hämtad 2017-05-10: <https://inkscape.org/en/>
- [14] *Recommended computor specs*, HTC Corporation ©2011-2017, hämtad: 2017-02-08
<https://www.vive.com/us/ready/>
- [15] *3Ds Max*, Autodesk Media and Entertainment ©2017, hämtad: 2017-05-10
<https://www.autodesk.com/products/3ds-max/overview>
- [16] *Maya*, Autodesk Inc ©2017, hämtad: 2017-05-10
<https://www.autodesk.com/products/maya/overview>

Bilaga A

Individuellt bidrag

Nedan följer en grov sammanfattning av vad samtliga gruppmedlemmar har arbetat med under projektets gång.

Linnea Bergman

Agerat som Scrum-master under projektet. Ansvarat och implementerat korten samt basfunktionaliteten för byggnaderna. Skapat menyelement och dess funktionalitet. Modellerat miljöelementen som berg och kullar samt satt ihop spelmiljön. Animerat svärdsmannen både med och utan vapen.

Rickard Falk

Modellerat huvudbyggnad, murbräcka och svärdsmannens svärd. Delaktig i implementationen av kortens spårning. Programmerat spelets AI. Implementerat soldaternas lerpning (så att de vrider sig mjukt när de byter riktning), synkroniserat katapultens animationer med attack och rörelse samt implementerat att kort blir starkare ju längre de hålls på hand.

Jonathan Fransson

Ansvarig för en stor del av implementeringen av spårningen samt kalibreringen av den. Har varit delansvarig i implementeringen av flerspelarläget som sedan togs bort. Animerat murbräckan. Har varit dokumentansvarig och sett till att dokumenteringen har varit i ordning och tillgänglig.

Felix Grönborg

Ansvarig för, och implementerat större delen av flerspelarmöjligheter, vilket senare togs bort på grund av tid- och utrustningsbrist, då det bara fanns en *HTC Vive*-enhet. Modellerat alla träd, stenar och stubbar. Skapat alla bilder och fysiska kort för *Vuforia* och fixat databaser till dessa i *Unity*. Ansvarig för kontakt med kund och övriga intressenter.

Simon Johansson

Projektledare, ser till att alla känner sig inkluderade och får hålla talan. Implementerat *Vuforia* i *Unity* och gjort att korten gick att spåra. Modellerat projektiler och svärdmannens sköld.

Tobias Pettersson

Sekreterare, har skött all dokumentation på de olika mötena. Designat och lagt in korten, modellerat och animerat katapulten, gjort vinst- och förlustmeddelanden samt programmerat rotation för pilarnas

modeller.

Sven Thole

Huvudansvarig för programmering av spellogik, övergripande syn över projektet så att alla moment blir klara i tid och hur produkten utvecklas samt bokning av lunchmöten. Programmerat spelets strids-system, logik för undersåtar, rörelsebana och kollision för projektiler och spelkort för undersåtar. Modellerat vakttorn samt kroppen till svärdsmännen.