Project: 3D to 2D Animation tool

(Afstudeer Scriptie)

Datum voltooid: 14-1-2014
Auteur: Vince Houbraken
Opdrachtgever: Gert-Jan Brok, Founder of Fantazm
Stage Begeleider: Rens Lensvelt, Developer Fantazm
Docent Begeleider: Jan Oonk, Docent Fontys Hogeschool Eindhoven

Versie: 1.0
Status: Final

Titelblad

Studentgegevens:

Naam: V.P.J.M. Houbraken, Vince

Studentnummer: 2178499

Afstudeerrichting: Voltijd

Afstudeerperiode: September t/m Januari, 85 Werkdagen

Bedrijfsgegevens:

Naam: Fantazm

Afdeling: Productie

Plaats: Den Bosch

Bedrijfsbegeleider: L.G.A Lensveld, Rens

Docentbegeleider:

Naam: J. Oonk, Jan

Verslag gegevens:

Titel: 3D to 2D Animation Tool

Datum van Uitgifte: 14-1-2014

Vertrouwelijk: Nee

Getekend en gezien door bedrijfsbegeleider:

Datum:

De bedrijfsbegeleider,

Rens Lensvelt

Voorwoord

Deze scriptie is tot stand gekomen tijdens mijn afstudeerstage bij Fantazm. Het dient verantwoording van mijn werkzaamheden af te leggen aan de examencommissie van Gamedesign & Technology aan Fontys Hogeschool Eindhoven. In dit document wordt de opdracht, het onderzoek, het product en de uiteindelijke conclusies en aanbevelingen doorgenomen en besproken.

Ik ben bij Fantazm uitgekomen na een lang proces van selectie. Ik zocht een bedrijf dat niet te groot was, bekend was met Unity3D en bovendien haalbaar was in reistijd. Dit beperkte mijn selectie enorm, maakte wel mogelijk dat ik elk bedrijf goed kon beoordelen op hun eigenschappen en uitstraling. Fantazm maakte op mij een positieve indruk. Fantazm was dan ook een van de eerste bedrijven die ik aanschreef voor een mogelijke stage. Het was het eerste bedrijf dat interesse toonde en ook direct het eerste bedrijf dat mij uitnodigde voor een intake gesprek. Tijdens dit gesprek werd er gekeken naar het werk dat ik meegenomen had. Beide school en hobby projecten. Er werd gekeken naar inhoud en motivatie, er werd geen document met vragen voorgelegd zoals dit bij andere bedrijven gebeurt. Dit alles gaf mij een positief gevoel bij het bedrijf en een stage bij Fantazm werd een feit.

Er waren meerdere opdrachten voor mij beschikbaar, elk met een duidelijk doel en elk even interessant. Na overleg met school is gekozen voor de opdracht met de omschrijving: “Maak een goede 2D animatie engine”. Een interessante uitdaging op een terrein waarmee ik nog niet bekend ben. Ik heb veel nieuwe dingen geleerd over Unity3D, en de interne werking van Unity3D als ontwikkelomgeving.

Graag zou ik op dit moment een paar mensen willen bedanken die deze stage voor mij mogelijk gemaakt hebben. Voor ondersteuning en motivatie van school uit, Jan Oonk en Mark van Kuijk. Zij hebben mij opgeleid samen met hun collega’s tot wie ik ben als software engineer/Game Designer en mij gewezen op het plezier en het nut van vele aspecten. Zonder hen was ik waarschijnlijk nooit zo ver gekomen. Dank gaat ook uit naar Ton van Rheenen, voor zijn inzet en hulp bij het op orde stellen van mijn stage. De planningshulp en de vraagbaak die hij mij geboden heeft was van grote waarde in het verder ontwikkelen van mijn vaardigheden tijdens deze stage.

Verder gaat mijn dank uit naar iedereen bij Fantazm. De grote baas, Gert-Jan Brok, voor het aanbieden van mijn stage en het opzetten van een open en persoonlijk game gericht bedrijf. Developer en bedrijfsbegeleider, Rens Lensvelt, voor zijn motivatie en waardevolle tijdsbesteding aan zijn stagiaires wanneer dit nodig was, zelfs al was het een erg drukke periode. En developer Koen van Lin, voor het delen van zijn expertise met mij en de andere stagiaires.

Verder zou ik graag iedereen willen bedanken die indirect mijn opleiding en stage mogelijk maakt. En mijn naasten die mij moreel gesteund hebben in mijn laatste en drukste schooljaar.

Hierbij wens ik de lezer veel plezier in het lezen van deze scriptie.

Inhoudsopgave

1 Samenvatting

1.1 Nederlands

Het bedrijf Fantazm maakt spellen met het software programma Unity3D. Ondanks de naam van deze software, gebruik Fantazm het ook om 2D spellen te ontwikkelen. Het animeren van 2D karakters wordt echter niet goed ondersteund door Unity3D zelf. Fantazm gebruikt een plugin om dit te verhelpen, genaamd SmoothMoves. Deze tool voldoet maar is beperkt, te langzaam en gebruik van de tool is ongebruiksvriendelijk.

Samen met Fantazm is een lijst van eisen en wensen opgesteld waaraan een 2D animatietool moet voldoen in het oog van Fantazm. Op basis van deze lijst is het erg duidelijk dat Fantazm een tool zoekt die gebruiksvriendelijk is, maar vooral snel werkt. De onderzoeksvraag luid dan ook: “Op welke wijze kan het animatie proces bij de ontwikkeling van 2D games versneld worden?”

In het vooronderzoek is beschreven hoe de huidige werkwijze van Fantazm is. Door de gemaakte lijst van eisen en wensen kan er gekeken worden of er bestaande tools zijn die aan de eisen van Fantazm voldoet.

Van de bestaande tools komt uiteindelijk Puppet2D als beste kandidaat naar voren en kiest Fantazm ervoor om te gaan werken met Puppet2D. Aan Puppet2D ontbreken echter nog wel een aantal features.

De meeste tijdswinst kan worden behaald door een feature te schrijven dat 3D animaties omzet naar 2D animaties. Het zorgt er voor dat animaties die voorheen uren duurden om te maken, nu in enkele minuten kunnen worden gegenereerd.

Deze feature is door mij ontwikkeld gedurende de stageperiode. De nieuwe feature kan een willekeurige 3D animatie die Unity3D kan uitlezen, omzetten voor een gewenst 2D karakter.

1.2 English

The company Fantazm creates games in the Unity3D software program. Despite the name of the software, Fantazm also uses it to create 2D games. Animating 2D characters in Unity3D is not well supported by default. As a result of this, Fantazm uses a Plugin named SmoothMoves. The tool does the job but is very limited, very slow and not user friendly at all.

Together with Fantazm a list of demands and wishes was created. The list contains all the features that Fantazm would want in a 2D animation tool. Based on this list it is easy to conclude that Fantazm is looking for a tool that is user friendly and helps shortening the develop time of 2D animation. Therefor the research question is: “How can the animation process of 2d games be improved to speed up development?”

In the pre-research we described the current workflow of Fantazm. Due to the existing demands and wish list, research can be done towards existing tools that might comply with the demands of Fantazm.

From the existing 2D animation tools, Puppet2D was the best candidate to use. Fantazm chose to work with Puppet2D as tool to use. However Puppet2D is still missing some features.

The most time that can be saved in development is by developing a feature for Puppet2D which can convert 3D animation into 2D animations. It would make it so that animations that would take hours to create normally, then would only take a few minutes to generate.

This feature is developed by me during the internship. The new feature can take any 3D animation that Unity3D can read, and convert it for use on a 2D character.

2 Verklarende woordenlijst

Woord	Uitleg
Baked Animations	Baked Animations zijn 'ingebakken' animaties, gebruikt in de terminologie rondom Puppet2D. Baken animations houdt in dat een animatie in zijn geheel kan werken in Unity3D, zonder de externe ondersteuning van Puppet2D. Dit is soms nodig voor performance verbeteringen binnen een project.
Bone Animation	Bone animation is een animatietechniek die gebruik maakt van botten net als bij 3D animatie. 2D plaatjes kunnen direct gekoppeld worden aan een gewricht (onderarm aan de elleboog bijvoorbeeld) of een plaatje van de gehele arm kan opgezet worden naar een mesh, waar hij aan meerdere botten wordt bevestigd. Vanaf dat punt werkt de techniek hetzelfde als bij 3D botten.
Child/Parent	In de context van Unity3D zijn Child en Parent een relatie vorm. Een parent-object kan 0,1 of meerdere child-objecten bevatten. Alles wat een parent-object beïnvloedt, beïnvloed ook de-child objecten. Een child heeft echter geen effect op zijn parent. Een child-object kan op zijn beurt ook een parent object zijn voor andere child-objecten. Dit zorgt voor een simpele boom structuur van objecten die Unity3D gebruikt bij het opbouwen van een scene of object.
Curves	Curves worden in dit verslag gebruikt in de context van een AnimationClip. Binnen een AnimationClip heeft elke variabele een curve waarin deze opgeslagen wordt. Dit opslaan gebeurt op specifieke momenten genaamd keypoints. Deze keypoints kunnen worden uitgezet worden over een tijdlijn, waarbij door al deze keypoints een lijn kan worden getrokken. Het resultaat is een curve.
Euler Rotaties	Euler rotaties zijn rotaties over 3 assen; X, Y en Z roteren tussen 0 en 360 graden.
Interpolatie	Interpoleren is een methode om nieuwe variabelen te genereren tussen 2 of meerdere bekende variabelen. Het resultaat bij lineaire interpolatie kan gezien worden als een soort gemiddelde nemen.
Inverse Kinematics	Inverse kinematics is een berekening voor beweging. In 3d animaties word de techniek vaak toegepast op een karakter zijn arm of been. Men kan dan de voet bewegen naar een bepaalde positie, en inverse kinematics berekend de juiste positie en rotatie voor die knie en de heup van het karakter.
Jitter	Het storen van een signaal. Een abnormale afwijking van het gewenste signaal. In ons geval komt dit voor in Curves.
Keypoints	Keypoints zijn letterlijk punten op een tijdlijn. Een keypoint wordt gebruikt om punten van verandering aan te duiden binnen een animatie. Elk punt bevindt zich op een specifieke tijd met een waarde van een variabelen voor die specifieke tijd. Meerdere keypoints van een variable vormen een Curve.
Mesh	Een mesh is een een verzameling van punten, randen, en vlakken die samen een 3D model vormen. Deze hoeven echter niet altijd 3D te zijn. In de context van dit verslag zijn de meshes altijd vlak, en betekend het simpelweg dat een plaatje omgezet is naar een verzameling punten, randen en vlakken. Dit maakt het mogelijk om het plaatje te vervormen aan de hand van de punten in de mesh.

Motion Capture Library	Een collectie van opgenomen animaties. Unity3D heeft een grote gratis collectie online beschikbaar gesteld voor menselijke 3D karakters.
Plane	Een plane binnen Unity3D is vergelijkbaar met een grote digitale vierkante plaat. Een plane heeft een locatie en rotatie, de locatie bepaalt waar het 0 punt van de plane is en de rotatie bepaalt hoe de vlakke plaat geroteerd is in de 3D wereld om zich heen. Een plane kan gebruikt worden voor specifieke berekeningen om bijvoorbeeld te bekijken hoe een set 3D posities er zou uitzien als zij op de plane worden gezet, en de plane vervolgens wordt gedraaid of gespiegeld.
Quaternion	Een alternatieve methode voor het noteren van rotaties. Quaternion rotatie gebruikt 4 variabelen om rotaties uit te drukken, X Y Z en W. Het grote voordeel van Quaternions is dat een computer ze gemakkelijker kan gebruiken om transformaties toe te passen. Ondanks dat Unity3D intern gebruik maakt van quaternions, gebruikt het visueel vaak Euler Angles om een duidelijke rotatie te tonen aan gebruikers.
Up/Down-sampling	Up sampelen is het gebruik van interpolatie op een variabele, in ons geval een curve. Het resultaat is meer keypoints die worden opgeslagen. Down sampelen is juist het opslaan van minder keypoints, het resultaat vrijwel altijd in verlies van detail en precisie.

3 Inleiding

In de huidige wereld van Triple A spellen, multithreaded particle systemen en prerendered gameplay videos, is er gelukkig af en toe nog eens vraag naar simpele maar niet minder interessante 2D games. Het is op deze momenten prettig als een klein maar origineel bedrijf als Fantazm voor je klaar staat om het idee piekfijn uit te werken.

Het bedrijf Fantazm werkt graag met bedrijven samen in het opzetten van 2D of 3D projecten waarbij het andere bedrijf als opdrachtgever fungeert. Vaak gebeurt dit in een nauwe samenwerking om er voor te zorgen dat het project exact zo uit de verf komt als de klant het voor ogen had.

Omdat Fantazm relatief vaak wordt aangesproken om 2D spellen te maken, is dit een groot deel van hun werk. Het nadeel aan de 2D stijl is dat veel processen nog steeds erg tijdsintensief zijn. Een groot deel van die tijd zit in het juist animeren en laten bewegen van karakters. In veel gevallen worden deze animaties nog met de hand gemaakt.

Deze animaties werden vroeger frame by frame geanimeerd. Nu werkt men met tools die interpolatie mogelijk maakt en hoeven er alleen maar keyframes gemaakt te worden. Het werk is dus in de jaren wel versimpeld, maar een werknemer is toch snel een aantal uren kwijt om een karakter op te zetten, en deze te voorzien van verschillende animaties. Vaak loopt het aantal animaties per karakter zelfs in de 10-tallen, en elk van deze animaties heeft op zijn beurt weer 10-tallen keypoints waar op het karakter herpositioneerd moet worden.

Fantazm is op het moment dat mijn stage begon op zoek naar een tool die deze tijdsintensieve taak kan verlichten. Indien mogelijk zouden sommige aspecten geautomatiseerd moeten worden zodat er niet steeds dubbel werk is voor elk nieuw karakter of elke kleine wijziging in animaties.

In deze scriptie worden de volgende hoofdstukken in de hierna genoemde volgorde behandeld:
In Hoofdstuk 4 is het bedrijf beschreven waar de afstudeer opdracht voor gemaakt wordt.
Hoofdstuk 5 beschrijft de opdracht.
In het daarop volgende hoofdstuk, 6 doe ik verslag van mijn onderzoek voor deze opdracht.

In de daarop volgende hoofdstukken 7, 8 en 9 wordt het implementatieplan beschreven, de uitwerking van het beschreven plan en de uiteindelijke resultaten.

In hoofdstuk 10 volgen een aantal conclusies en aanbevelingen voor het vervolg van dit project.
In hoofdstuk 11 kijk ik terug op mijn periode bij Fantazm en de ervaringen die is opgedaan.

4 Het bedrijf Fantazm

Fantazm is een gezellig klein bedrijf dat gestart is in het jaar 1998. Sinds 2004 richt Fantazm zich alleen nog maar op games. Fantazm is dus al 10 jaar bezig met de ontwikkeling van games maar is bescheiden gebleven in omvang. Op het moment heeft Fantazm drie vaste werknemers en delen zei een kantoor met het bedrijf NoMoreMondays.

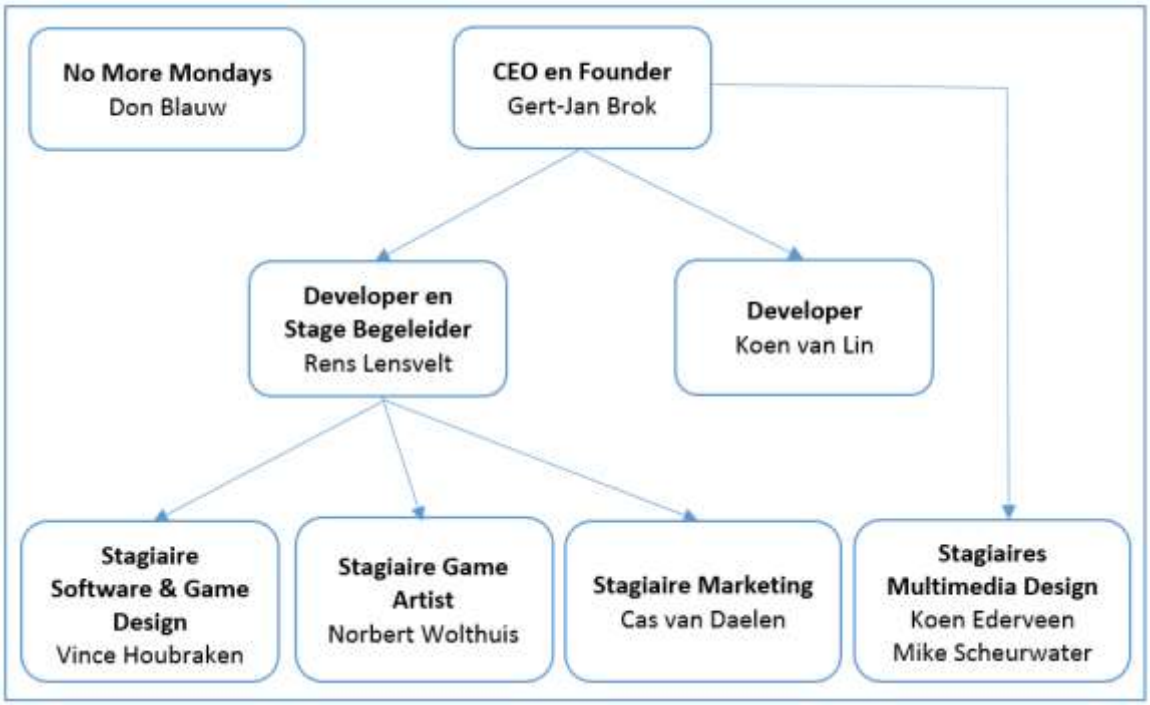
Fantazm maakt voornamelijk games voor andere bedrijven, waaronder simulatiegames. Verder maken zij games in eigen beheer. Een eerste, genaamd Dikkie Durf, is al een tijd op de markt en een tweede is in ontwikkeling.

Figuur 1 - Fantazm oude website banner (Fantazm, 2010)

Door de achtergrond die Fantazm heeft in Flash, werken ze graag aan 2D projecten. Het bedrijf is echter al een paar jaar geleden overgestapt naar Unity3D als ontwikkel omgeving.

Fantazm doet graag zo veel mogelijk met eigen mensen, maar huurt waar nodig externe experts in.

In de gezellige omgeving van Fantazm kan altijd een grapje gemaakt worden en is niemand te beroerd eens een nieuwe gadget of juist een oude reliek te demonstreren aan de groep. Soms ligt de werkdruk hoog en is er geen tijd voor afleiding, maar ook dan laat Fantazm een sterke kant zien en worden projecten snel en efficiënt afgewerkt.



Figuur 2 - Organigram van Fantazm

Het organigram van Fantazm.

5 Opdrachtformulering

5.1 Opdracht aanleiding

Zoals eerder vermeld ontwikkelt Fantazm veel 2D-projecten voor externe opdrachtgevers. Fantazm wil graag een goede 2D animatie tool om het ontwikkelproces in dergelijke projecten te versnellen en te verbeteren. Fantazm heeft duidelijke ideeën over de eisen waar aan een dergelijke tool aan zou moeten voldoen. De meeste eisen hebben betrekking op het versnellen van het ontwikkelproces, onder meer door een gebruiksvriendelijke tool te vinden die het werk van de ontwikkelaars goed ondersteunt.

Daarnaast is het van belang dat de tool geïntegreerd moet kunnen worden in de ontwikkelomgeving die Fantazm gebruikt om games te ontwikkelen: Unity3D.

Figuur 3 - Unity3D Logo (Unity3D, 2011)

Op dit moment wordt de animatie tool Smoothmoves gebruikt voor 2D animatie in Unity3D. De tool is niet erg gebruiksvriendelijk. Simpele shortcuts zoals men die gewend is in software (ctrl-c, ctrl-z, etc.), worden niet ondersteund of zijn ongemakkelijk geïmplementeerd. Hierdoor is het niet mogelijk om een snelle workflow te krijgen naarmate je vaker met de software werkt. Een paar simpele shortcuts kunnen het animeren veel gemakkelijker, sneller en gebruiksvriendelijker maken.

5.2 Onderzoeksvraag

Fantazm wil het ontwikkelproces van 2D games versnellen. Fantazm denkt dat de grootste winst te behalen is bij het animatieproces. Met de huidig gebruikte tool Smoothmoves, is het proces erg arbeidsintensief. De onderzoeksvraag luid dan ook:

“Op welke wijze kan het animatie proces bij de ontwikkeling van 2D games versneld worden?”

Bij deze hoofdvraag horen de volgende deelvragen:

1) “Wat is de huidige werkmethode van Fantazm?”
2) “Wat zijn de eisen en wensen van Fantazm?”
3) “Is het nodig een 2D animatie tool te schrijven, of is er een bestaande tool die volstaat?”
4) “Hoe kan het 2D animatie proces verder worden geoptimaliseerd voor tijdswinst?”

5.3 Onderzoeksaanpak

In dit vooronderzoek wordt eerst de huidige werkwijze van Fantazm in kaart gebracht. (Deelvraag 1) Ook worden alle eisen van Fantazm geïnventariseerd die Fantazm stelt aan de gewenste tool voor 2D animatie. (Deelvraag 2)

Vervolgens wordt onderzocht of er tools op de markt zijn die aan de belangrijkste eisen van Fantazm voldoen. (Deelvraag 3)

In de conclusie van het onderzoek wordt gekeken welke opties de meeste tijdswinst op zouden leveren voor Fantazm tijdens het animatie proces. (Deelvraag 4)

Het resultaat van het onderzoek wordt voorgelegd aan Fantazm. Fantazm zal op basis van het onderzoek een keuze maken en een traject definiëren waarin deze keuze wordt gerealiseerd.

6 Het vooronderzoek

6.1 De huidige situatie

Op dit moment maakt Fantazm gebruik van Smoothmoves. Het programma werkt binnen Unity3D, en heeft een eigen interface. Smoothmoves is erg specifiek met het aanleveren van karakters. Het programma wil elk onderdeel van een karakter als een los plaatje aangeleverd krijgen. Fantazm zou graag meer vrijheid hebben in het aanleveren van karakters.

De interface van Smoothmoves is onoverzichtelijk en niet intuïtief. Het maken van een animatie vereist veel handelingen waardoor men uren bezig is om een karakter op te bouwen en te animeren. Ook zijn animaties voor deze karakters niet opnieuw toepasbaar voor nieuwe karakters.

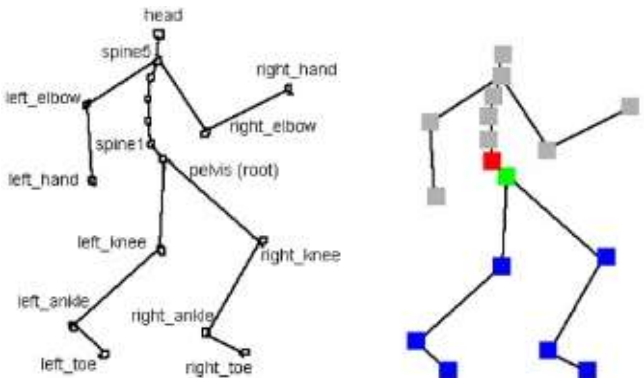
Een voordeel van Smoothmoves is dat het animaties maakt met behulp van Unity3D Animation Clips. Dit houdt in dat wanneer animaties klaar zijn, ze wel kunnen worden gebruikt zonder al te veel problemen.

Fantazm maakt bij de animatie van 2D ook nog gebruik van sprite animatie. Sinds versie 4.5 van Unity3D heeft sprite animatie echter voldoende ondersteuning en hoeft er dus niets aan dit proces gewijzigd te worden.

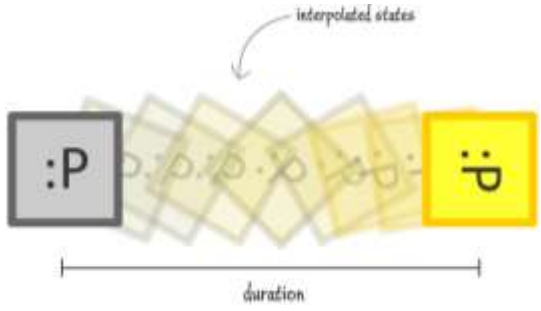
6.1.1 2D Animeren

Fantazm beschrijft het animeren van een 2D-karakter als een tijdsintensief proces. In principe zijn er op dit moment drie technieken om dit proces te versnellen: *bone animation, inverse kinematics* en *interpolatie*.

Bij *bone animation* wordt er tijd gewonnen door een 2D-karakter een bone structuur te geven. Deze techniek werkt hetzelfde als de 3D-animation variant. Men gebruikt een skelet en beweegt het karakter door de botten te manipuleren. Het gebruik van *bone animation* maakt arbeidsintensief frame by frame tekenen voor sprite animaties overbodig. Men kan een 2D-karakter tekenen in losse onderdelen, en daar *bone animation* op toepassen.



Figuur 4 - Skelet voor animatie (Butterfield, 2005)



Figuur 5 - Visualisatie van interpolatie (Chinnathambi, 2013)

Bone animation wordt ondersteund door *inverse kinematics*. Inverse kinematics zorgt er voor dat indien een hand of voet bewogen wordt, de rest van de arm of het been automatisch wordt meebewogen aan de hand van een formule. Met deze techniek hoeft de animator maar 6 punten te animeren in plaats van 12 of meer botten.

Ten slotte is er de *interpolatie* techniek. *Interpolatie* is een relatief simpele techniek waarbij de animator een aantal poses maakt op key frames, en vervolgens de software de tussenliggende animatie frames berekend. Dit is een lineair proces voor beide rotaties en locaties. Voorbeeld: Frame

0 heeft waarde 10, en frame 1 heeft waarde 20. Als er dan een frame wordt opgenomen die tussen 0 en 1 valt, zeg frame 0,5. Dan is de waarde 15. Een gemiddelde tussen frame 0 en frame 1.

Het grootste probleem van het animatie proces ligt in het feit dat één karakter animeren met een tiental animaties uren kan duren ondanks alle bestaande technieken. Vervolgens kan de animatie niet gebruikt worden voor een ander karakter, tenzij deze een exact gelijke botten structuur heeft.

6.2 Eisen en wensen van Fantazm

Samen met Fantazm is een lijst van eisen en wensen opgesteld. Deze lijst is vertaald naar een set features die de animatie tool zou moeten bevatten en is er een prioriteit toegekend aan elke feature: Hoe belangrijk is de feature voor Fantazm. Deze features zijn in het Engels beschreven omdat Fantazm een voorkeur heeft voor Engelse documentatie.

Feature Naam	Prioriteit Fantazm	Motivatie
Mobile Compatible	Demand	Het is voor Fantazm van belang dat het eindresultaat zonder problemen gebruikt kan worden op mobiele platforms.
Multi Layer Support	Demand	Het gebruiken van meer dan 1 laag voor een karakter is erg belangrijk bij bone animations.
Mesh Deformation	Demand	Mesh deformation houdt in dat men met een enkel plaatje van een geheel been, toch de buigbewegingen kan maken in de knie, belangrijk maar niet vereist bij bone animations.
Multi Bone Influence	Demand	Het beïnvloeden van bovenstaande feature over meerdere botten is nodig om een deformatie vloeiend te maken.
Cut Scene Animation	Demand	Om hele scenes te animeren is het handig om hier een soort van tijdlijn of hulptool te hebben die het proces versimpelt of structureert. Indien goed uitgevoerd is deze feature goed voor veel tijdswinst.
Re-Use Animations	Desired	Indien een ander karakter met hetzelfde skelet exact dezelfde animaties kan gebruiken is dit een makkelijke manier van tijdswinst door hergebruik van de animaties.
Re-Use Animations Altered Skeleton	Desired	Indien de animaties ook op een aangepast skelet kan werken, is dit nog beter dan bovenstaande feature.
Smooth Transitioning by Interpolation	Desired	Indien een 2D karakter wisselt tussen 2 animaties is het mooi als dit met interpolatie gebeurt net als bij de 3D variant.
3D Animation to 2D Animation	Desired	Als een 3D animatie kan worden gebruikt op een 2D karakter, kan er veel tijdswinst gemaakt worden door dat er veel minder geanimeerd hoeft te worden. Ook zijn er veel 3D animaties gratis beschikbaar.
Scaling Animated Skeleton	Desired	Een 2D karakter kunnen schalen na dat deze geanimeerd is in sommige situaties voor scene animatie nodig.
Unity Timeline for Editing	Desired	Het gebruik van de Unity3d tijdlijn is gemakkelijk om alles geïntegreerd te houden. Ook zijn Fantazm developers al gewend aan deze tijdlijn en zijn functies.
Sprite Swapping	Desired	Het kunnen uitwisselen van hoofden of kledingstukken tijdens het maken of aanpassen

Feature Naam	Prioriteit Fantazm	Motivatie
		van een animatie staat veel meer vrijheid en mogelijkheden toe tijdens game development.
Runtime Sprite Swapping	Desired	Het kunnen uitwisselen van kleding tijdens het spelen van een spel. Sprite Swapping tijdens het spelen van een spel.
Audio Synchronization	Desired	De mogelijkheid tot synchronisatie van audio of andere acties op basis van een animatie is belangrijk in games. Denk aan het gebruik van voetstappen tijdens het lopen.
Drawing Support	Desired	Het aanpassen van je tekeningen van karakters binnen dezelfde tool als waarmee je ze animeert zou heel fijn kunnen zijn. De prioriteit voor deze feature ligt echter laag door de hoge moeilijkheidsgraad van de feature.
Undo	Desired	Het kunnen terug draaien van een fout.
Shortcut Keys	Desired	Het kunnen gebruikmaken van shortcuts is een gebruiksvriendelijkheids feature. Als een ontwikkelaar of animator langer met de tool werkt en meer shortcuts weet, kan hij sneller zijn werk doen.
Inverse Kinematics	Preferred	Bij het animatie proces is inverse kinematics een gemakkelijk hulpmiddel wat veel tijdswinst oplevert voor de animator.
Facial Animation	Preferred	Het animeren van het gezicht kan erg lastig zijn voor 2D animatie, en er zijn meerdere manieren voor. Speciale ondersteuning om dit gemakkelijk te doen is gewenst.
Ragdoll Physics	Preferred	Het uitoefenen van externe krachten op het 2D karakter. Voornamelijk om een 2D karakter correct te laten reageren bij zijn dood, of bij een explosie.
Unity Integrated	Preferred	Het werken met een tool binnen Unity3D is altijd gemakkelijker dan een tool buiten Unity3D.
Trial Version Available	Preferred	Gebruik kunnen maken van het programma voor test doeleinden.
Runtime Playback Type	Unity Preferred	Het afspelen van animaties uit een Unity3D animatie file is gemakkelijk en snel. Het liefste wordt er gebruik gemaakt van deze bestaande manier zonder gebruik te maken van extra ingewikkelde playback scripts.
Save Format	Unity Preferred	De manier waarop animaties worden opgeslagen is relevant voor de snelheid en gebruiksvriendelijkheid van de tool. Over het algemeen geldt hier dat de Unity standaard manier van opslaan het gemakkelijkste, snelste en universeelste is.
Export Image Format	Unity Spritesheet Preferred	Exporteren van de animatie in plaatjes. De eis is hier dat wanneer het gebeurt, het in een sprite sheet formaat gebeurt waarmee Unity kan werken.
CPU usage	Lowest Possible Preferred	Voor mobile gebruik van de animaties is CPU gebruik het liefste zo laag mogelijk. Door dat dit moeilijk meetbaar is beoordeelt Fantazm achteraf het resultaat.

6.3 Onderzoek naar bestaande tools

Er zijn 4 tools geselecteerd die op het eerste gezicht relevant zijn:

- Puppet2D, een populaire tool uit de Unity Asset Store
- Smoothmoves, ook een populaire tool
- Sprites and bones, de open source tool
- Spine, een tool die niet binnen Unity werkt

Fantazm heeft veel ervaring met Flash en op aanvraag van Fantazm is ook Flash meegenomen in dit onderzoek: alles exporteren vanuit flash is namelijk ook een mogelijkheid.

Tool	Versie	Homepage
Puppet2D	1.10	http://www.puppet2d.com/
Smoothmoves	2.5.0d	http://echo17.com/smoothmoves.html
Sprites and Bones	1.2.2	http://banbury.bootes.uberspace.de/blog/?p=sprites-and-bones
Spine	1.9.17	http://esotericsoftware.com/
Flash	12.0.2.529	http://www.adobe.com/products/flash.html

Referenties: (Niman, 2013) (echo17, 2014) (TheRealBanbury, 2014) (Esoteric Software, 2014) (Adobe Systems Incorporated, 2014)

In een kort vooronderzoek zijn de tools vergeleken met de eisen van Fantazm. Hierbij vielen Smoothmoves, Spine en Flash snel af. Een nadere vergelijking tussen Puppet2D en Sprites and Bones maakte duidelijk dat Puppet2D beantwoorde aan de meeste wensen van Fantazm. De details van het vooronderzoek zijn terug te vinden in bijlage B.

Puppet2D heeft echter nog wel de volgende ontbrekende features:

- *Cut Scene Animation*
- *Re-Use Animation*
- *Drawing Support*
- *Ragdoll Physics*
- *3D Animation to 2D Animation*

Uit deze ontbrekende features is kort bekeken welke van de eisen het meeste zouden bijdragen aan de tijdswinst bij het animeren. De bevindingen zijn hieronder beschreven per feature.

6.3.1.1 Cut Scene Animation

De mogelijke tijdswinst van een Cut Scene animation tool is niet echt duidelijk. In theorie is het gemakkelijk om een tool te hebben die cut scenes gemakkelijker maakt, zodat wanneer een animatie gewijzigd wordt, alle overige animaties automatisch worden aangepast. Dit zou tijd kunnen besparen, mits de wijzigingen correct zijn. De mogelijke is tijdswinst onbekend omdat er geen implementatie van een dergelijke feature bekend is.

6.3.1.2 Re-Use Animatie

Het hergebruiken van animaties op andere karakters zou tijdswinst opleveren omdat de animator geen dubbel werk hoeft te doen voor verschillende skeletten. Het nadeel aan het realiseren van deze feature is dat deze aanpassingen in de code van Puppet2D moeten worden doorgevoerd en dat de Puppet2D codebase onbekend is.

6.3.1.3 Drawing Support

Het toevoegen van tekenen binnen Unity3D is een lastig toe te voegen feature waarbij toch veel gebruik gemaakt zou moet worden van tools buiten de Unity3D omgeving omdat de kwaliteit van gespecialiseerd tekentools nooit geëvenaard kan worden.

6.3.1.4 Ragdoll Physics

Het hebben van ragdoll physics draagt niet bij in het efficiënter creëren van 2D animaties.

6.3.1.5 3D Animation to 2D Animation

Het converteren van 3D animaties naar 2D animaties is zo ver bekend nog nooit gebeurt voor dit doeleinde. Het is in theorie mogelijk, maar er zijn veel randvoorwaarden waar rekening mee gehouden moet worden. De randvoorwaarden bepalen bijvoorbeeld tot hoe ver men diepte bewegingen mee neemt in het 2D karakter. Wat in hele specifieke gevallen heel erg belangrijk is om te hebben, maar extra ontwikkeltijd vergt bij het maken van de tool. Het maken van deze feature zou veel tijdswinst op kunnen leveren omdat veel animaties niet meer hoeven worden gemaakt. Ze kunnen simpelweg geïmporteerd worden uit een bestaande 3D animatie library, of zelf worden opgenomen met een Kinect camera.

6.4 Conclusie Fantazm

Fantazm heeft de resultaten van het vooronderzoek uit bijlage B bekeken. Op basis hiervan is besloten dat er gewerkt gaat worden met Puppet2D en dat Puppet2D uitgebreid zal worden met een deel van de missende features die in hoofdstuk 6.3 zijn genoemd. Drawing Support is de uitzondering, deze feature is niet redelijk om uit te voeren binnen Unity3D.

Bij het realiseren van deze missende features zal de prioriteit liggen bij het omzetten van 3D animaties naar 2D animaties. De motivatie achter deze keuze is dat er een grote gratis library met animaties bestaat voor Unity3D. Deze library is direct en gratis te gebruiken. Als deze animaties omgezet kunnen worden naar 2D animaties, zou het zelf ontwikkelen van animaties vrijwel onnodig maken. Dit zou een enorme tijdbesparing opleveren.

7 Implementatie

In dit hoofdstuk wordt besproken hoe de door Fantazm gekozen feature wordt uitgewerkt.

Eerst wordt in hoofdstuk 7.1 de uitgangssituatie beschreven.

Vervolgens wordt in hoofdstuk 7.3 en 7.4 de aanpak voor het realiseren van de feature beschreven.

De aanpak wordt voorgelegd bij Fantazm wat leidt tot de uitwerking in hoofdstuk 8.

7.1 Beschrijving nieuwe situatie

Door het voorgaande onderzoek en de conclusie van Fantazm is er een nieuwe situatie ontstaan. Het is niet nodig om een 2D animatie tool te ontwikkelen, maar tegelijkertijd zijn de bestaande alternatieven onvoldoende om aan de behoefte van Fantazm te voldoen. Puppet2D is uit zichzelf een zeer gestructureerde tool die vele features al heeft, en daarbij ook erg flexibel is. Op basis van de keuze voor Puppet2D moeten bepaalde features worden toegevoegd aan Puppet2D. Fantazm heeft besloten om daarbij de prioriteit te geven aan de feature waarin 3D animaties worden omgezet naar 2D animaties.

Dit hoofdstuk gaat nader in op realisering van de feature om 3D animaties om te zetten naar 2D. Eerst gaan we in op de vraag wat een dergelijke feature allemaal moet kunnen doen om 3D animaties om te zetten naar 2D animaties.

7.2 Een eerste analyse

Het gebruiken van 3D animaties op een 2D manier, betekent in feite dat er een dimensie verdwijnt (de diepte). Meer specifiek betekent dit dat het 3D object vanaf een specifieke hoek bekeken wordt, en dat vanuit deze hoek de 3de dimensie van de 3D positie en rotatie wordt weggefilterd. Het resultaat hiervan zou dan kunnen worden gebruikt om een 2D skelet te animeren mits het dezelfde opbouw heeft.

7.3 Oplossingsplan

Geen van de onderzochte tools of bestaande software heeft een feature om 3D animaties naar 2D om te zetten. Er is dus geen voorbeeld of inspiratiebron voor de te realiseren feature.

Als uitgangspunt nemen we het idee van een filmcamera tijdens het opnemen van een film. We gebruiken een virtuele camera die de afgespeelde 3D animatie vastlegt op een 3D AnimationClip.

We volgen dan ook dezelfde stappen als bij het opnemen van een film:
Stap 1 – De 3D animatie on-demand afspelen.
Stap 2 – De 3D animatie opnemen met een virtuele camera.
Stap 3 – Het resultaat van de camera opslaan naar een formaat zodanig dat het op elk 2D skelet gebruikt kan worden.

Deze aanpak is voorgelegd aan Fantazm en deze heeft besloten dat onderzoek naar alternatieve methodes overbodig is en de voorgelegde stappen doorlopen gaan worden om de feature te gaan uitwerken.

Stap 1 is het gecontroleerd afspelen van een 3D animatie clip. Om opnames perfect te kunnen uitvoeren is het noodzakelijk dat een animatie frame voor frame kan worden afgespeeld. Door de manier waarop Unity3D animaties geïnterpoleerd terugspeelt is het dan ook direct mogelijk om te up- of down-samplen. Het up-sampelen heeft in de context van onze feature niet veel zin omdat het enkel meer ruimte in zou nemen dan nodig is zonder toegevoegde waarde. Het down-sampelen kan

echter een kleine performance verbetering opleveren. Down-sampelen kost echter wel detail in animatie waardoor subtiele en snelle bewegingen verloren gaan of minder zichtbaar worden.

Stap 2 is een orthografische camera maken die de animatie kan opnemen vanuit een instelbare hoek en omzet naar een 2D plane. Een orthografische camera is een lineaire camera, niet de gebruikelijk camera met een kijkhoek. Dit is als het ware de cameraman bij het maken van een film. Men kan instellen onder welke hoek het karakter gezien moet worden voor de opname. Vervolgens neemt de camera elk 3D punt van het model, en zet dit orthografisch op een vlakke plane. Deze plane kan vervolgens gebruikt worden voor stap 3.

Stap 3 is de 2D plane opslaan in een formaat dat op elk skelet met ongeveer gelijke botstructuur afgespeeld kan worden. Hierdoor kan men de animaties hergebruiken zonder dat deze opnieuw wordt opgenomen; het wordt dan een soort conversie van de plane naar een Unity3D tijdlijn. Deze stap is het minst bekend en het meest experimenteel.

In het vervolgtraject worden deze stappen in de genoemde volgorde uitgewerkt. Omdat voor een groot deel Puppet2D niet benodigd is voor de ontwikkeling, wordt alles uitgewerkt in een geïsoleerde scene, om vervolgens de koppeling te maken naar Puppet2D.

8 Uitwerking

8.1 Afspelen van animatie

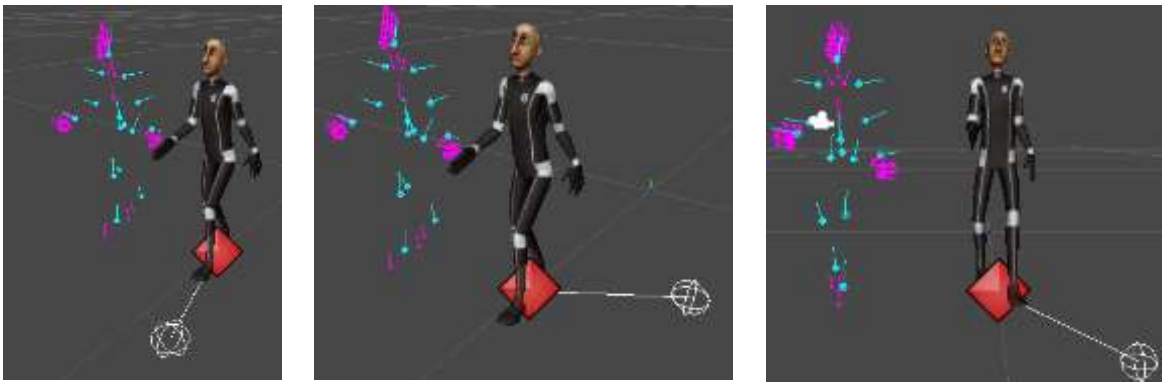
De eerste stap in hoofdstuk 7.3 beschrijft het afspelen van de animatie. Hierbij is het belangrijk dat het afspelen van de animatie zo gecontroleerd mogelijk gebeurd. De best mogelijke manier om dit te volbrengen is elke gewenste frame van de animatie op elk gewenst moment te kunnen oproepen.

Om deze controle te verkrijgen is er gezocht in de Unity3D Scripting Manual gezocht naar mogelijke code die dit kan realiseren. De Unity3D Scripting Manual bevat beschrijvingen van vrijwel elk stuk code dat Unity3D standaard aanbiedt.

Unity3D slaat animaties op in een component genaamd AnimationClip. We moeten in Stap 1 dus een AnimationClip kunnen afspelen. Hiervoor kent Unity3D twee manieren van afspelen: Animation en Animator.

Animation is een verouderd component die enkele animaties kan afspelen en heeft geen gecontroleerde manier van afspelen, het speelt alleen clips af vanaf het begin.

Animator is het nieuwe component met meer mogelijkheden. De Animator heeft een methode genaamd Animator.Play. Deze methode maakt het mogelijk om elke gewenste frame in een animatie aan te roepen met behulp van een normalizedTime. Het is dus mogelijk om met Animator.Play elk gewenst frame van een animatie aan te roepen, mits de genormaliseerde tijd die bij de frame hoort berekend kan worden. Hierdoor is Animator dus het gewenste component voor het afspelen van animaties.



Figuur 6 - Het standaard Unity3D karakter wat animaties afspeelt met een visuele representatie van elk bot in de animatie. De cyaan gekleurde punten gaan opgenomen worden, de roze punten worden niet gebruikt in de opnames. De witte bol met de lijn naar het karakter dient als aanduiding voor de opname vanaf de camera.

Een nadeel aan het gebruik van de Animator is dat om een animatieclip af te spelen deze op 2 locaties handmatig moet worden ingevoerd. De animatieclip moet ingevoerd worden in de Mecanim Animator, en in het camera script van de tool. Het dubbel invoeren is nodig omdat Mecanim een relatief nieuw onderdeel is van Unity3D en nog niet alle functies van Mecanim zijn in versie 4.5 aan te roepen vanuit code. Dit is echter in Unity 4.6 en Unity 5.0 verholpen. Door toevoegingen aan de code is het nu mogelijk om direct de Mecanim Animator aan te spreken om de gewenste animaties op te vragen. Dit zorgt er voor dat de gebruiker de AnimationClip niet meer dubbel in hoeft te voeren. Aangezien de huidige tool uiteindelijk ook mee gaat naar Unity 4.6, is besloten er nu geen extra tijd aan te besteden.

Het enige wat benodigd is om de gewenste frame af te spelen, is een formule die de genormaliseerde tijd kan vinden van een frame. Genormaliseerd wil in dit geval zeggen dat de

waarde tussen 0 en 1 is, waarbij 0 het begin van de animatie is en 1 het einde van de animatie. Stel een animatie is 10 seconden lang, op 5 frames per seconde. Dan heeft de gehele animatie 50 frames. Bij de genormaliseerde waarde 0.5, spreek ik frame 25 van de animatie aan.

Het normaliseren van de tijd is dus relatief gemakkelijk. Men pakt de animatie lengte, en vermenigvuldigt deze met het aantal frames per seconde om het totale aantal frames te krijgen. Vervolgens deelt men het totaal aantal frames door het gewenste frame nummer om de genormaliseerde tijd te vinden.

De volgende formule wordt gebruikt om de gewenste frame te berekenen:
*normalizedTime = currentFrame / (AnimatieLengteSeconde * FramesPerSeconde) ;*
Met het voorbeeld:
*normalizedTime = 25/(10*5);*
normalizedTime = 25/50;
normalizedTime = 0.5;

Hierdoor is het doel om maximale controle te hebben over de animatie voltooid. Elke gewenste frame kan worden omgerekend naar een genormaliseerde tijd, waardoor de Animator deze exacte frame toont.

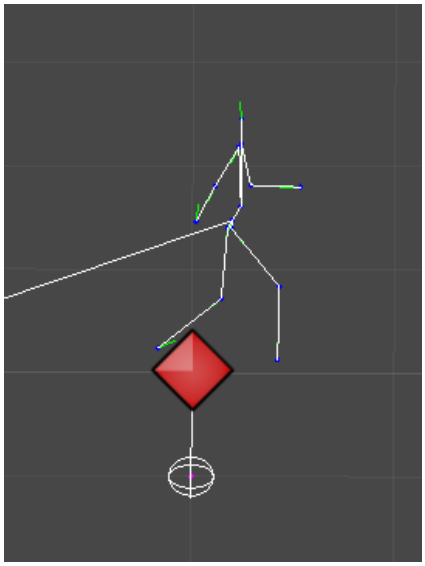
8.1.1 Animatie terugspelen verbeteren

Het terugspelen van animaties is tweemaal uitgevoerd. Eerst is gebruik gemaakt van de Unity3D-editor Update die een vaste 100fps heeft. Opnemen van animaties met deze Update is niet gegarandeerd accuraat doordat de opname snelheid niet gelijk loopt aan de editor snelheid. Hierdoor ontstonden afwijkingen in de opgenomen animaties: de frame rate was niet perfect en ook de synchronisatie tussen afspelen en opname klopte niet.

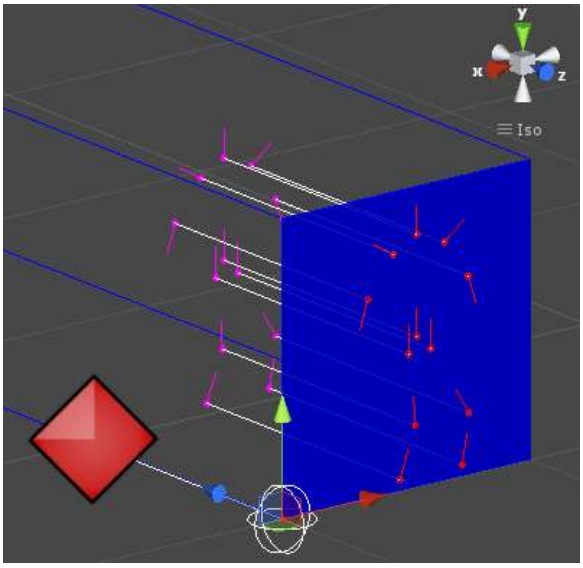
Puppet2D gebruikt een methode die niet afhankelijk is van de editor Update, maar gebruik maakt van de animatiemodus van Unity3D's. Door deze modus aan te zetten kunnen alle animatie activiteiten versneld uitgevoerd worden en kan een 3D animatie op elk gewenst moment zijn bijbehorende skelet herberekenen. Hierdoor is het mogelijk om een animatie in 1 enkele frame op te nemen en er dus geen sprake is van mogelijke verstoring door frame rate of frame drops. Door deze betere methode is het nu mogelijk om de 3D animaties terug te spelen voor het opnemen van de 2D animaties. Gekozen is om deze laatste aanpak te volgen. Met dit resultaat is Stap 1: Het afspelen van de animatie, afgerond. De volgende stap is het opnemen van de animatie.

8.2 Camera en opname

Als camera fungeert een simpel visueel punt dat om het centrum van het karakter kan draaien. Dit centrum is gevisualiseerd als een rode ruit, en de camera als een witte bal. Tussen deze punten bevind zich een lijn van de camera naar het skelet die aangeeft op welke manier op dat moment wordt gekeken. Met 2 sliders kan de developer nu de camera draaien op de X en Z-as zodat het skelet uit elke gewenste hoek kan worden opgenomen.

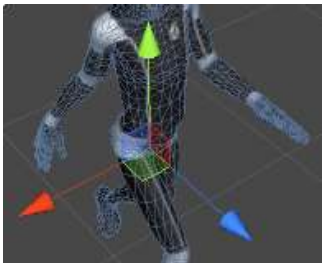


Figuur 7 - Visualisatie van de camera die naar het karakter kijkt, en het uiteindelijke resultaat van de plane. De groene lijntjes visualiseren de rotatie van de ledematen.



Figuur 8 – Het blauwe vierkant visualiseert het onzichtbare plane van de camera. In werkelijkheid is de plane oneindig groot in alle richtingen. Positie (0,0,0) van de plane is op het kruispunt van de pijlen. Links onder van het blauwe vierkant.

In werkelijkheid is het camerapunt het begin punt van een plane. Deze plane is een onzichtbaar assenstelsel waarvan de locatie (0,0,0) gelijk is aan de wereldpositie van het camerapunt. De rotatie van dit plane is gelijk aan de rotatie van het camerapunt. Het script gaat alle botten van het skelet af en kijkt bij elk bot hoe het punt op de plane komt te staan.



Figuur 9 - Oriëntatie van een object, blauw voorwaarts, rood rechts, groen boven.

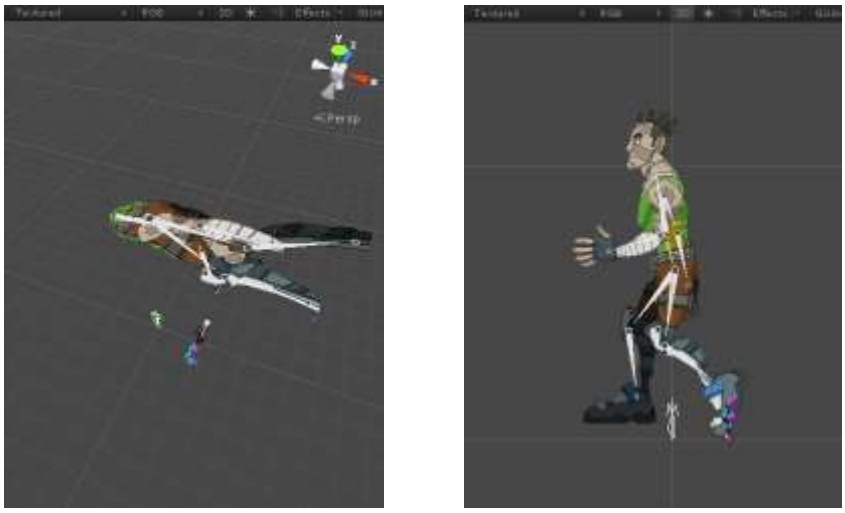
De exacte berekening om een bot op de plane te zetten staat op de Unity3D wiki (Bit Barrel Media, 2014) maar komt in het kort hierop neer: De methode die gebruikt wordt heet ProjectPointOnPlane. Om ook de rotatie van de punten mee te nemen op de plane, wordt de voorwaartse richting van de bone opgeteld bij de locatie van het bot. Dat hulppunt bevindt zich dan direct voor de rotatie van het bot. Dit hulppunt wordt ook geprojecteerd op de plane met ProjectPointOnPlane. Vervolgens kan het originele botpunt op de plane geroteerd worden, zodat het naar het hulppunt kijkt. Hierdoor krijgt het originele botpunt de correcte 2D rotatie. Vervolgens kan het hulppunt voor deze rotatie verwijderd worden. Nadat alle botten deze bewerkingen zijn ondergaan wordt de gehele plane terug geroteerd naar een (0,0,0) rotatie en op de (0,0,0) locatie gezet. Het eindresultaat is een platte 2D versie van onze originele 3D animatie.

8.2.1 Quaternion rotaties in opname

Omdat de oriëntatie van het originele 3D skelet niet gelijk was aan die van het 2D skelet ontstonden er in de rotaties van de opname ongewenste hoeken (Zie figuur 10). Deze rotaties waren in theorie correct, maar stonden op een verkeerde as. Rotaties waren gewenst op de Z-as, maar de voorzijde van de afbeeldingen stonden naar de Y-as in plaats van de Z-as.

Om dit te verhelpen is eerst geprobeerd de quaternions te corrigeren tijdens het opslaan van de animatie (dus tijdens stap 3). Dat bleek te complex. Uiteindelijk is het probleem opgelost bij het opnemen van de rotaties (Stap 2). Het voorkwam hierdoor de ongewenste draaiing in de voorzijde van de afbeeldingen maar behouden de rotaties op de Z-as waardoor het karakter correct animeerde.

Stap 2: Het filmen, is hierbij afgerond. De volgende stap is het opslaan van de opnames.



Figuur 10 - Voor en na rotatie correcties, alles bewoog op een verkeerde as.

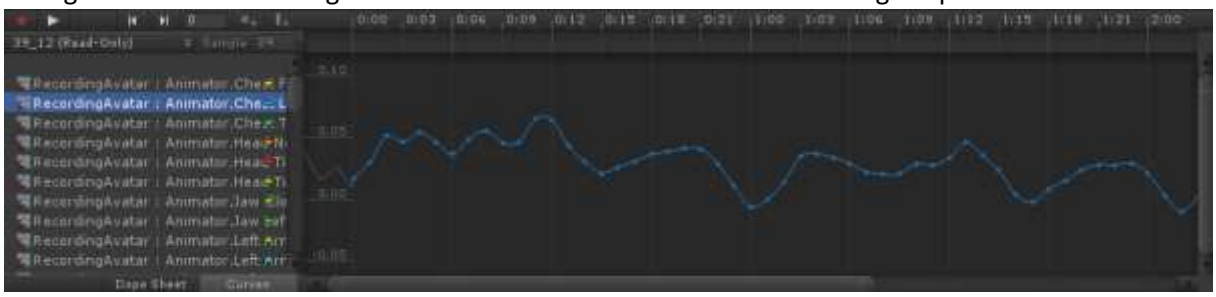
8.3 Opslaan van de opnames

Het gebruik van de Unity3D AnimationClips is de enige manier om animaties correct terug te spelen binnen Unity3D. Externe opslag is mogelijk, maar voor terugspelen is het nodig om naar een AnimationClip te vertalen vanuit de externe opslag. Helaas zijn AnimationClips niet erg flexibel. Ze zijn naamgevoelig, child/parent relaties van het skelet zijn relevant en het aanpassen van één naam of child/parent relatie maakt dus de gehele animatie onbruikbaar.

Om deze redenen is het onmogelijk om een universele 2D AnimationClip te maken. Daarom wordt er voor nu gebruik gemaakt van een doelskelet tijdens opname. Dit doelskelet zorgt ervoor dat de code weet hoe de doel animatie opgeslagen moet worden. Dit maakt animaties opslaan voor meerdere karakters/skeletten iets lastiger omdat men meerdere doel-skeletten moet aanduiden. Hiermee is met de gekozen oplossing een redelijk compromis gevonden waarbij toch veel tijdwinst wordt behaald.

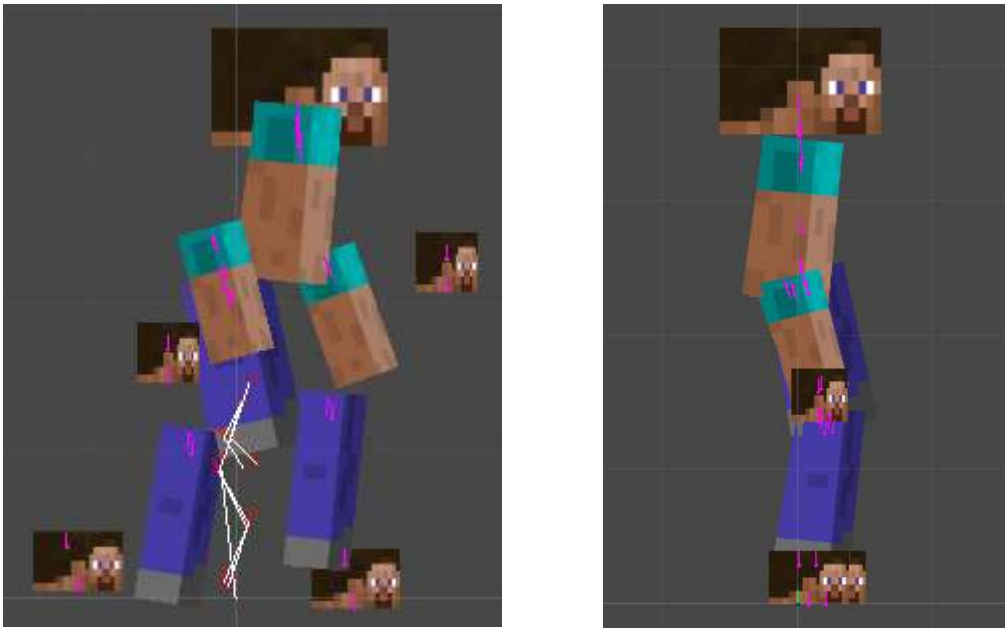
Voor het opnemen van een animatie is het dus nodig om een AnimationClip aan te maken. De opname snelheid voor deze AnimationClip wordt als eerste ingesteld in frames per seconde. Deze snelheid is ideaal gezien hetzelfde als de animaties die opgenomen gaan worden. Maar door bij het afspelen van de originele animatie gebruik te maken van *interpolatie*, kan men op elke gewenste snelheid opnemen met weinig tot geen verlies van informatie. Interpolatie zorgt er namelijk voor dat tussen 2 animatie frames een gemiddelde genomen kan worden bij het opnemen. Voor elk skeletpunt worden 3 positie en 4 rotatie curves aangemaakt. In elke curve kan slechts 1 variabele opgeslagen worden per frame. Een *curve* is nodig om een variabele op te slaan in een AnimationClip.

Elke curve slaat 1 variabelen op specifieke punten in een tijdlijn. Deze punten worden in een curve doorgetrokken en worden gebruikt om later een vloeiende animatie terug te spelen.



Figuur 11 - Animation Curve. De blauwe lijn is de curve, boven in staat de verlopen tijd, en links staat de variabele waarde. De blauw punten zijn de keypoints van de animatie.

Om de animatie op te nemen wordt de originele animatie frame by frame afgespeeld zoals beschreven in hoofdstuk 8.1. Na elke frame worden alle 7 variabelen opgenomen als een punt in de curve. Uiteindelijk worden de voltooide curves opgeslagen in de voorheen gemaakte AnimationClip, en krijgt de AnimationClip een nieuwe naam met _2D aan het einde. De animatie is nu gereed voor gebruik op een 2D skelet.

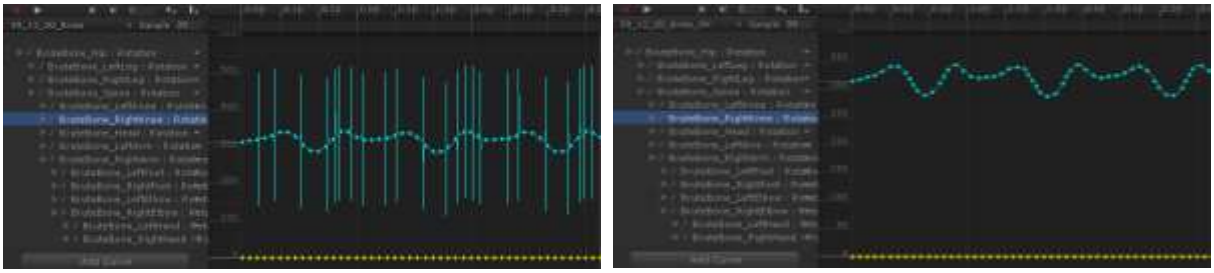


Figuur 12 - Links: Oud testkarakter waarbij rotaties nog niet werden opgenomen en botlengte niet relevant was. Rechts: Oud testkarakter waarbij deze features wel relevant waren.

Hierbij is ook het opslaan voltooid en het maken van de feature: 3D naar 2D feature bereikt.

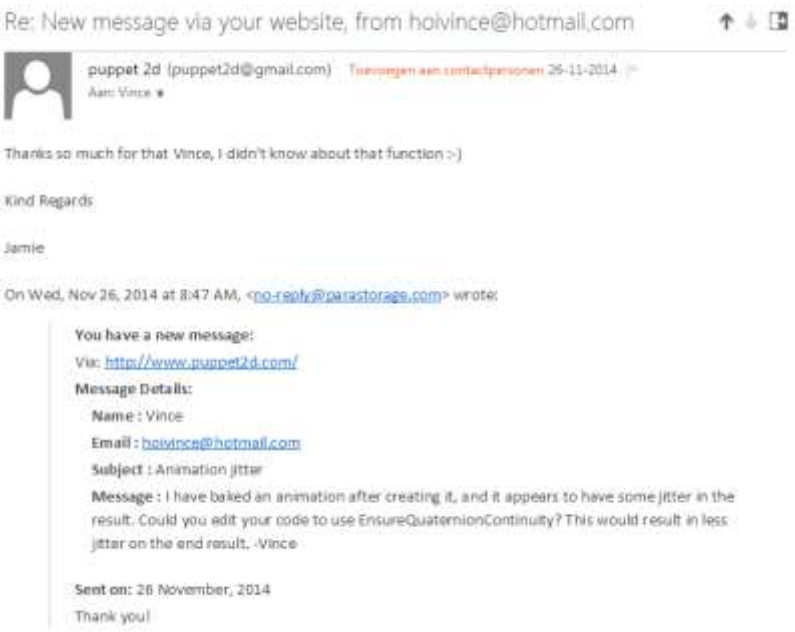
8.3.1 Animatie Jitter verwijdering

Na opnames bleek dat er rare jitter in de animaties verscheen. Aangezien bij testen bleek dat de input en de bewerkingen zelf deze jitter niet veroorzaakten, leek de oorzaak bij Unity3D te liggen. Een oplossing werd gevonden bij de methode EnsureQuaternionContiniuity in de referentiepagina van Unity AnimationClip. (Unity Technologies, 2014) Deze methode verhelpt deze jitter op te animatie door quaternions correct door te laten lopen. Onderstaand resultaat van deze bewerking is aanzienlijk, en het probleem was verholpen.



Figuur 13 - Voor en na het gebruik van EnsureQuaternionContiniuity

Puppet2D staat het toe zijn animaties ook op te slaan in een soortgelijke Unity3D AnimationClip in de feature genaamd “Bake Animation”. Deze baked animations hadden echter hetzelfde probleem als mijn eigen gemaakte animaties. Omdat de oplossing relatief eenvoudig was en Puppet2D deze nog niet geïmplementeerd had, heb ik de oplossing doorgespeeld aan de developer van Puppet2D om deze op de hoogte te stellen van het probleem en de mogelijke oplossing.



Re: New message via your website, from hoivince@hotmail.com

puppet 2d (puppet2d@gmail.com) Toevoegen aan contactpersonen 26-11-2014

Aan: Vince

Thanks so much for that Vince, I didn't know about that function :-)

Kind Regards

Jamie

On Wed, Nov 26, 2014 at 8:47 AM, <no-reply@parastorage.com> wrote:

You have a new message:

Via: http://www.puppet2d.com/

Message Details:

Name : Vince

Email : hoivince@hotmail.com

Subject : Animation jitter

Message : I have baked an animation after creating it, and it appears to have some jitter in the result. Could you edit your code to use EnsureQuaternionContinuity? This would result in less jitter on the end result. -Vince

Sent on: 26 November, 2014

Thank you!

Figuur 14 - Reactie email van Jamie Niman, Developer van Puppet2D

9 Resultaten

Na de uitwerking van de opdracht volgens het voorgaande hoofdstuk is het mogelijk om animaties op te nemen op basis van een standaard Unity3D skelet. Dat betekent dat elke 3D animatie die gebruikt kan worden in Unity3D, ook gebruikt kan worden door mijn tool.

Na het opzetten van een 2D skelet kan de gebruiker een prefab 3D skelet in de scene zetten en de camera hoek configureren. Hierna koppelt de gebruiker de 3D skelet punten aan de 2D skelet punten en geeft het 3D skelet alle animaties voor de opname. Nu kan een opname gemaakt worden van het 3D skelet naar het 2D skelet.

De gemaakte opnames kunnen nu gebruikt worden op het 2D skelet in combinatie met alle baked animaties die het 2D skelet mogelijk al bezat. Met een extra script zou het mogelijk moeten zijn om te wisselen tussen originele Puppet2D animaties type en de Puppet2D baked animaties, maar zelfs zonder dit zijn mijn animaties nu al zo flexibel dat zij hetzelfde kunnen worden gebruikt als Puppet2D's originele baked animaties.

10 Conclusies en aanbevelingen

De onderzoeksvraag luidde: “Op welke wijze kan het animatie proces bij de ontwikkeling van 2D games versneld worden?” Door het onderzoek en de realisering van een van de features voor de tool Puppet2D heeft Fantazm nu een belangrijke versnelling in het ontwikkelen van 2D games tot haar beschikking. De nieuwe feature voor Puppet2D zal het animatie proces aanzienlijk versnellen. Echter, de tijd ontbrak om de laatste puntjes op de i te zetten, omdat de feature nog niet onder alle omstandigheden is getest. Ook zou er nog tijd gestoken moeten worden in het robuuster en gebruiksvriendelijker maken van de feature en de feature moet nog worden geïntegreerd in Puppet2D zodat het geheel perfect kan samenwerken.

10.1 Aanbevelingen

Om het project geheel te integreren met Puppet2D zijn de volgende stappen nodig:

- Bestaande feature test
- Automatisch skelet binden
- Integreren opname knop
- Onderzoek compatibiliteit
- 3D Rotatie invloed
- Optimalisatie opnames

Ik ga nader in op deze punten in de volgende paragrafen.

10.1.1 Bestaande feature test

De bestaande tool is niet uitvoerig getest met alternatieve skeletten, niet-menselijke skeletten en animaties van een andere bron dan de aangeleverde Motion Capture Library. Deze situaties moeten getest worden om vast te stellen of ook in deze situaties de tool werkt of aangepast moet worden.

10.1.2 Automatisch skelet binden

Op dit moment moet de gebruiker de skeletpunten zelfstandig binden naar het doelskelet. Men zou gebruik kunnen maken van specifieke namen of een hiërarchie waardoor dit proces geautomatiseerd kan worden. Dit zou flink bijdragen aan de gebruiksvriendelijkheid van de feature.

10.1.3 Integreren opname knop

Het integreren van de opnameknop in de Puppet2D GUI zou ook de gebruiksvriendelijk verbeteren. Omdat het 2de menu overbodig wordt en het geheel dan beter aansluit op de originele Puppet2D GUI.

10.1.4 Onderzoek compatibiliteit

Bij het ontwikkelen van de tool is het grotendeels los van Puppet2D ontwikkeld. Hierdoor is niet onderzocht of de feature probleemloos samenwerkt met alle features van Puppet2D. Er is een (kort) onderzoek nodig naar alle features van Puppet2D om te achterhalen of de compatibiliteit van de extra feature voldoende is.

10.1.5 3D Rotatie invloed

Tijdens de ontwikkeling is in overleg met de bedrijfsbegeleider ter sprake gekomen dat de 2D animaties één belangrijk punt missen: Indien de originele 3D animatie zijn arm uitsteekt richting de camera, komt deze na de omzetting in 2D vreemd over. Dit komt omdat de arm normaal gesproken naast het lichaam blijft. Het uitsteken zou de zijkant van het plaatje laten zien. Er is onderzoek nodig naar het herkennen van dit probleem in de rotaties, zodat de gebruiker bijvoorbeeld een sprite swap kan doen op het moment dat een arm een X aantal percentage afwijkt van zijn normale

diepterotatie. Wanneer dan de arm normaal naar buiten zou komen naar de camera, gebruikt men exact dezelfde 2D rotatie, maar een ander sprite voor de arm. Dit zou het rotatieprobleem in de diepte vrijwel geheel wegnemen en de animatiekwaliteit erg verbeteren. Er zal echter wel rekening gehouden moeten worden met de huidige aanname dat de arm altijd een vaste lengte heeft wanneer dit geïntegreerd wordt.

10.1.6 Optimalisatie opnames

Op het moment wordt er bij elk frame elke rotatie en positie opgenomen. Het is kennelijk zo dat elke opgeslagen keyposition een heel klein beetje vertraging veroorzaakt. Het is daarom misschien zinnig om geen keypositions op te slaan voor posities of rotaties die niet veranderd zijn. Onderzoek hiernaar is nodig.

11 Evaluatie

Ik had gewenst de opdracht meer volledig af te ronden en aan te leveren aan het einde van deze stageperiode. Helaas is dit door vele vertragingen niet gelukt. Veel tijd is er ook gaan zitten in opdrachten van Fantazm. Daar heb ik persoonlijk veel kennis en ervaring opgedaan. Helaas kan deze ervaringen niet geuit worden in dit verslag omdat deze niet direct relevant waren aan mijn stageopdracht.

Bij sommige onderdelen van mijn opdracht zoals het Quaternion probleem wat eerder genoemd is, heb ik achteraf gezien niet op tijd hulp ingeschakeld. Als ik bij dit probleem eerder mijn begeleider had laten meekijken was sneller naar boven gekomen dat ik het probleem verkeerd bekeek en was het probleem dus sneller opgelost. Meerdere keren is gebleken dat ik soms een probleem verkeerd aanpak of bekijk en vervolgens geen andere oplossing bereik. Op deze momenten moet ik mijzelf niet op het probleem blijven focussen, maar een teamgenoot inroepen en zijn visie op het probleem vragen.

Ik heb ondervonden dat mijn werk tempo relatief laag ligt in vergelijking met de vaste werknemers van Fantazm. Dit is als stagiaire nog geen probleem, later in de praktijk wel. Ik zal dit moeten verbeteren.

12 Afbeeldingen lijst

13 APA Referenties

Adobe Systems Incorporated. (2014, October). *What's New*. Retrieved from Adobe Flash Professional CC: http://www.adobe.com/products/flash/features.html

Bit Barrel Media. (2014, Oktober 16). *3D Math functions*. Retrieved from Unify Community: http://wiki.unity3d.com/index.php?title=3d_Math_functions

Butterfield, J. (2005). *Animadead Documentation*. Retrieved from Animadead: A skeletal animation library: http://animadead.sourceforge.net/docs/

Chinnathambi, K. (2013, Februari 5). *Introduction to Animation in HTML*. Retrieved from Kirupa.com: http://www.kirupa.com/html5/introduction_to_animation_html.htm

echo17. (2014, December 2). *ChangeLog*. Retrieved from echo17: http://www.echo17.com/forum/index.php?PHPSESSID=b12933720f4478ab91a50eb2960c8331&topic=2.15

Esoteric Software. (2014, August 30). *Changelog*. Opgehaald van esoteric software: http://esotericsoftware.com/spine-changelog

Fantazm. (2010). *Home*. Retrieved from Fantazm: http://fantazm.com/2014/pages/index2.php

Niman, J. (2013). *Release Notes*. Retrieved from Puppet 2D: http://www.puppet2d.com/#!release-notes/c10lk

TheRealBanbury. (2014, October 21). *Unity Sprites And Bones - 2D skeleton animation*. Opgehaald van Unity Forum: http://forum.unity3d.com/threads/release-free-unity-sprites-and-bones-2d-skeleton-animation.219915/

Unity Technologies. (2014). *Unity Documentation | AnimationClip.EnsureQuaternionContinuity*. Retrieved from Unity - Scripting API: http://docs.unity3d.com/ScriptReference/AnimationClip.EnsureQuaternionContinuity.html

Unity3D. (2011). *Unity Logo*. Retrieved from Unity3D: http://unity3d.com/

14 Bijlage A – PID

PROJECT: 2D ANIMATION TOOL / INITIATIEFASE

(Project Initiation Document)

Datum voltooid: 30-9-2014
Auteur: Vince Houbraken

Versie: 1.1
Status: Compleet

Documenthistorie

Revisies

Versie	Status	Datum	Wijzigingen
1.0	concept	16-9-2014	Initiële opzet
1.1	compleet	30-9-2014	Kleine correcties

Goedkeuring

Dit document behoeft de volgende goedkeuringen:

Versie	Datum goedkeuring	Naam	Functie	Paraaf

Distributie

Dit document is verstuurd aan:

Versie	Datum verzending	Naam	Functie
1.0	15-9-2014	Gert-Jan Brok, Rens Lensvelt	
1.1	30-9-2014	Jan Oonk	

Managementsamenvatting

Aanleiding voor het project

Fantazm is op zoek naar een 2D animatie tool. Na het proberen van meerdere tools is de conclusie getrokken dat geen enkele voldeed aan Fantazm's eisen. Problemen lagen in de workflow en missende features. Fantazm wenst een betere tool voor 2D animaties.

Het doel is dat men onderzoekt of de beoordeling van Fantazm klopt en er geen tools zijn die alle features al bezitten. Indien dit klopt, wordt er een tool ontwikkeld.

Zie Bijlage B voor een volledige lijst van eisen.

Globale aanpak van het project

Voor de ontwikkeling wordt het 10 stappen plan (TSP) gevolgd. Dit is een methode gericht op de gehele stage procedure, van bedrijf oriëntatie tot het afstuderen en levert de nodige structuur voor analyse, planning en ontwikkeling van het project.

Voor onderzoek worden de eisen van Fantazm vastgelegd en deze vergeleken met de bestaande 2D tools. Indien blijkt dat er geen voldoende tool is, wordt een eigen tool ontwikkeld.

Tijdens ontwikkeling maakt men gebruik van Github voor versie beheer en Tickspot voor urenverantwoording.

De tool wordt ontwikkeld van scratch en itereert niet op een bestaand project. Echter wordt er wel informatie vergaard van bestaande tools om het ontwikkelingsproces zo spoedig mogelijk te laten verlopen.

Oplevering van het project

Het eindproduct wordt een los staande Unity asset. Deze asset wordt in C# gerealiseerd. Er wordt eventueel een minimaal demo project gerealiseerd, zodat alle features gemakkelijk kunnen worden getoond aan de gebruiker als voorbeeld. Deze moet voldoen aan zo veel mogelijk van de eisen en alle kwaliteitscriteria. (Zie bijlage B)

Doorlooptijd van het project

De doorlooptijd van het project is officieel 85 dagen, met een totaal van 680 uur. De officiële looptijd is van 1 september tot 23 januari.

Voor gedetailleerde planning van de doorlooptijd, zie bijlage A.

Inhoudsopgave

1 Inleiding

Stage periode en bedrijfsinformatie

Dit PID gaat over het project 2D animatie tool. Dit is een opdracht voor het stage bedrijf Fantazm. De stageperiode loopt van 1 september tot ongeveer 23 januari. In deze periode zal ik een project draaien genaamd "2D Animatie Tool". Deze tool wordt gemaakt in opdracht van het bedrijf Fantazm, onder leiding van Gert-Jan Brok en Rens Lensvelt.

PID

Dit Product Initiatie Document dient er voor om alle geïnteresseerden op de hoogte te brengen van de aanpak en de opstart van het project. Verder wordt er ook informatie geboden over de planning, uren budget en het doel van het project.

Tot stand komen van de opdracht

Fantazm is op zoek naar een 2D animatie tool die voldoet aan hoge eisen. Deze eisen zijn niet alleen features, maar ook flexibiliteit, stabiliteit en gebruiksgemak van de tool. Gezien deze eisen erg hoog liggen en er veel ontbrekende features zijn in de verschillende bestaande tools, gaat Fantazm zelf de tool ontwikkelen. In dit document leest u hoe dit gaat verlopen.

Unieke features

Naast bestaande features die al eerder uitgevoerd zijn in andere tools, heeft Fantazm ook een paar unieke features op het oog. Een goed voorbeeld is het converteren van 3D animaties naar 2D animaties. Er zijn al veel 3D animaties beschikbaar op de markt, maar 2D animaties worden vaak nog zelf gemaakt en zijn niet tot nauwelijks herbruikbaar. Het idee is dat de tool van Fantazm de bestaande 3D animaties om kan zetten naar werkende 2D animaties zonder al te veel moeite.

2 Projectdefinitie

2.1 Projectdoelstellingen

Het bedrijf Fantazm is op zoek naar een 2D animatie tool. Na het proberen van vele tools is naar voren gekomen dat veel van de tools een onhandige workflow hebben of niet de features hebben die Fantazm nodig heeft om kwalitatieve 2D games gemakkelijk te ontwikkelen.

Het doel van dit project is om te onderzoeken of er geen tool is die aan Fantazm's eisen kan voldoen. Indien geen enkele tool aan de eisen voldoet is Fantazm geïnteresseerd in het ontwikkelen van deze tool.

De voordelen aan een eigen tool is dat deze een goede workflow bevat en alle features die Fantazm nodig heeft voor hun productie ingebouwd heeft. Dit maakt het ontwikkelen van 2D games gemakkelijker en biedt zelfs mogelijkheden om de tool te verkopen aan andere Unity3D gebruikers.

De tool dient gemakkelijk te zijn in de omgang, dus ook een goede workflow te bevatten, vele onderdelen zoals animaties dienen herbruikbaar te zijn. Ook moet het voldoen aan de vele eisen die gesteld zijn (Zie Bijlage B voor een overzicht van de eisen).

2.2 Gekozen oplossing of aanpak

Voor de ontwikkeling volg ik het 10 stappen plan (TSP). Dit is een methode gericht op de gehele stage procedure, van bedrijf oriëntatie tot het afstuderen, maar levert wel de nodige structuur voor analyse, planning en ontwikkeling van het project.

Eerst dient er onderzoek te worden gedaan naar de bestaande tools. Om deze te kunnen vergelijken moeten ook de eisen van Fantazm worden vastgelegd. Vervolgens zullen de features en eisen van de beste tools naast elkaar worden gelegd. Als hier uit blijkt dat geen enkele tool aan de juiste eisen voldoet, is dit voldoende aanleiding om verder te gaan kijken naar het produceren van een eigen tool.

Tijdens ontwikkeling zal er gebruik gemaakt worden van versie beheer op Github en urenverantwoording via Tickspot.

De tool wordt vanaf niets opgebouwd en itereert dus niet op een bestaande tool. Men wil niet afhankelijk zijn van andere tools. De bestaande tools worden echter wel gebruikt om informatie te vergaren over bekende oplossingen en methodes om over te nemen.

Voor de werkstructuur wordt er eerst een soort klassen diagram uitgeschreven zo dat alle features die men wil hebben in het project gedefinieerd zijn als losse classes. Vervolgens probeer je deze classes te groeperen en aan elkaar te linken voor communicatie. Wanneer dit correct wordt toegepast kan men in een korte tijd een groot deel van het project vast leggen, maar toch snel aanpassingen maken. Uiteindelijk krijgt men een overzicht met daarop alle benodigde classes en relaties tussen deze classes.

2.3 Scope van het project

Het project wordt opgeleverd in de vorm van een tool. Deze tool is verpakt in een Unity Package. De tool wordt geschreven in C#. De package bevat 3 onderdelen, de tool, de runtime minimal requirements en een demo map met eventuele voorbeelden. Alle code en commentaar in de code zijn in het Engels.

Eventuele tutorial of guide wordt opgeleverd in het Engels, wanneer deze nodig zijn om het eindproduct begrijpelijk te maken.

Het geheel wordt opgeleverd aan de testgroep (Zie Organisatiestructuur), en daarna officieel aan de opdrachtgever. Het marketing deel en het verkopen van het eindproduct is een onderdeel waar de opdrachtgever zich over ontfermt.

2.4 Productbeschrijvingen op projectniveau

Zie bijlage B.

2.5 Projectbudget

Het project loopt tijdens mijn gehele stage. Inclusief introductie en het schrijven van dit document. Ik heb voor mijn stage 85 Dagen van 8 uur. Het budget is dus 680 uur. Van deze werk uren gaan nog een week introductie af samen met de tijd die nodig is om benodigde documentatie te schrijven. Voor een accurate planning, zie de bijlage Projectplan. (Zie bijlage A)

Indien de planning uit loopt door ziekte of andere onvoorziene omstandigheden, is er een mogelijkheid om extra dagen te draaien om de benodigde uren alsnog te halen. Zie hiervoor de escalatie procedure.

Er zijn verder geen directe kosten gerelateerd aan dit project, en er wordt hier dus verder geen financieel deel besproken.

2.6 Benodigde Resources

Humanitaire Resources:

- Opdrachtgever
- Stage Begeleider
- Stage Coördinator
- Online hulpbronnen (Indien nodig, communicatie met experts via het internet)

Hardware Resources:

- Development PC op locatie
- Test PC/Mac op locatie (Uitsluitend tijdens test voor eindgebruikers)

Software Resources:

- Unity3D (Mogelijk Pro licenced)
- Unity3D 5.0 Beta (Onderzoek)
- Toegang tot Fantazm projecten (Werkprocedure)
- Github toegang
- Tickspot

2.7 Uitsluitingen

Na officiële oplevering is de uitvoerder niet meer verantwoordelijk voor het eindproduct tenzij hier nieuwe afspraken over worden gemaakt in de toekomst. Dit houdt in dat tijdens de overdracht alle informatie van belang wordt overgedragen. Hiermee is ondersteuning voor toekomstige software en andere software matige problemen na oplevering, niet de verantwoordelijkheid van de uitvoerder.

Asset levering is niet de verantwoordelijkheid van de uitvoerder. Dit houdt in dat eventuele afbeeldingen die benodigd zijn voor het project, naast schermafdrukken, niet de verantwoordelijkheid zijn van de uitvoerder, maar moeten worden aangeleverd voor een compleet resultaat.

2.8 Afhankelijkheden

Het project is afhankelijk van het Unity3D platform waarvoor het eindproduct ontwikkelt wordt. In geval van aanpassingen in een nieuwere versie van Unity3D, is het mogelijk dat het eindproduct aangepast moet worden.

Ook is de uitvoerder afhankelijk van alle voorgenoemde Resources. De uitvoerders werk wordt belemmerd of zelfs tot een (tijdelijk) halt geroepen indien er resources ontbreken.

2.9 Randvoorwaarden

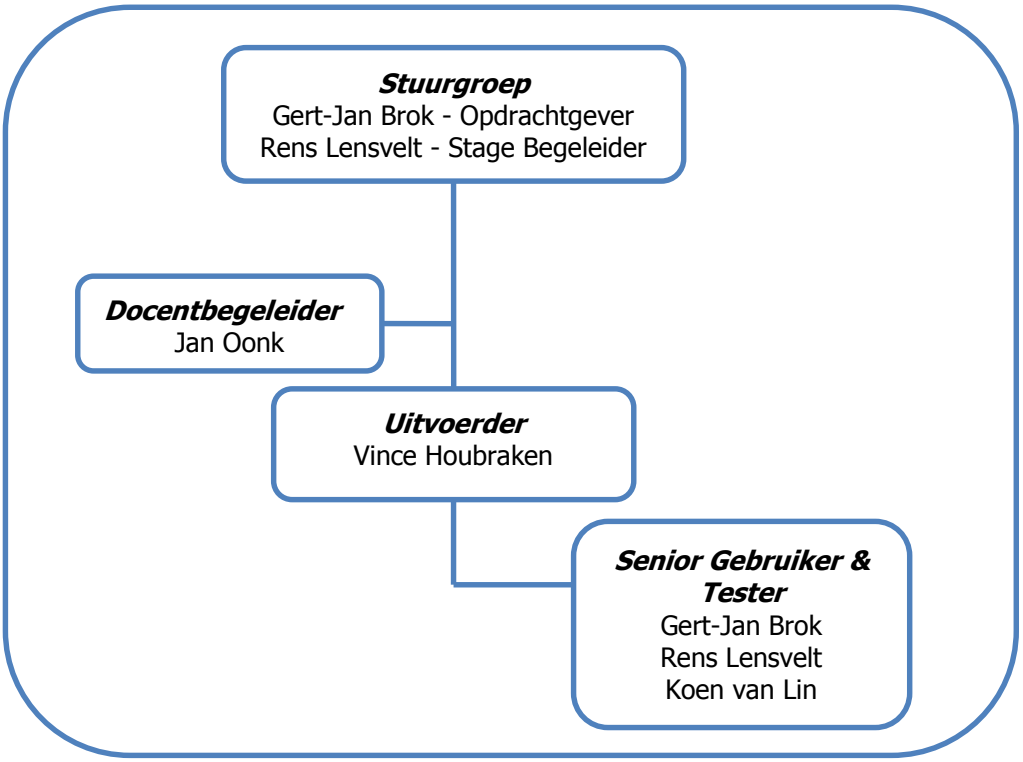
Alle voorheen genoemde benodigde resources zijn aanwezig of worden beschikbaar gesteld indien nodig.

2.10 Aannames

Er wordt vanuit gegaan dat de benodigde humanitaire resources tijd hebben en/of maken indien nodig. Hierbij gaat men er ook vanuit dat gemaakte afspraken worden nagekomen en dat iedereen handelt in belang van het project en het eindverslag wat daar bij hoort en dat iedereen bij Fantazm zijn kennis en expertise vrijwillig deelt.

Voor software resources wordt er uitgegaan dat de ontwikkeling van Unity 5 binnen de looptijd van het project wordt beëindigd en het eindproduct wordt uitgegeven.

3 Projectorganisatiestructuur



3.1 Opdrachtgever

Rolbeschrijving

De opdrachtgever is Gert-Jan Brok, Directeur van Fantazm

Projectgerelateerde taken

De verantwoordelijkheid van de opdracht gever is het duidelijk delen van zijn mening. Alle vorm van input van de opdrachtgever is belangrijke informatie om mee te nemen in het project. De sturing van de opdrachtgever en de eisen die hij stelt, geven het project een vorm en een doel.

Specifieke verantwoordelijkheden

Het begeleiden van het project met zijn meningen en eisen.

Het beoordelen van het eindproduct.

3.2 Stage Begeleider

Rolbeschrijving

Begeleid de stagiaire tijdens het project en controleert zijn voortgang. Helpt eventueel met problemen die zich voortdoen binnen het project.

Projectgerelateerde taken

De stage begeleider binnen het bedrijf zorgt voor ondersteuning op de werkvloer. Indien er vragen of onduidelijkheden zijn, worden deze eerst bij de stage begeleider neergelegd. Deze werkt dan mee om de vraag te verhelderen en op te lossen.

Specifieke verantwoordelijkheden

Het begeleiden van de stagiaire.

Het beoordelen van het eindproduct.

Het controleren van de uitvoerder zijn voortgang.

3.3 Docentbegeleider

Rolbeschrijving

De docentbegeleider begeleidt de stage vanuit school. Hij bekijkt het project, de voortgang en de documentatie vanuit de belangpunten van Fontys.

Projectgerelateerde taken

Tijdens het verloop van het project zijn er documenten die naar school toe worden opgeleverd. De docentbegeleider bekijkt deze documenten en beoordeelt of deze voldoen aan de eisen van het Fontys. Indien nodig stuurt de coördinator het project bij, om beter te voldoen aan deze eisen. Uiteindelijk dient het Fontys een correct en onderbouwd eindproduct te zien te krijgen, waar de stage coördinator mee naar toe werkt.

Specifieke verantwoordelijkheden

Controle van alle documentatie op de door Fontys gestelde eisen.

Mede beoordeling van het eindproduct aan het einde van de stage periode namens Fontys.

3.4 Uitvoerder

Rolbeschrijving

De uitvoerder voert het gehele project uit en is mede de project manager, het is zijn verantwoordelijkheid dat het project correct wordt afgerond, volgends de gestelde regels loopt en aan de gestelde eisen voldoet.

Projectgerelateerde taken

De uitvoerder moet alle eisen van de opdracht gever en de stage coördinator inventariseren. Op basis van deze eisen zet hij de opdracht op, en voert hij de opdracht uit. Ook is de uitvoerder mede verantwoordelijk voor de voortgangsbewaking. Hij dient op tijd alle documentatie en het eind product op te leveren.

Specifieke verantwoordelijkheden

Het inventariseren van eisen.

Het opzetten en uitvoeren van de opdracht.

Voortgang bewaken.

3.5 Senior gebruiker & Tester

Rolbeschrijving

De senior gebruiker is een ervaringsdeskundige. Hij test het eindproduct op gebruiksgemak en technische eisen. Ook kijkt hij uit voor mogelijk tekortkomingen vanuit het gebruikers oogpunt.

Projectgerelateerde taken

De senior gebruiker heeft ervaring met het eindproduct zijn bestaande alternatieven en weet hoe het eindproduct moet werken om een duidelijke en fijne workflow te bereiken. Dit inzicht is erg belangrijk in het laatste stadium van het project, waarbij gebruiksgemak en functionaliteit van het project belangrijk is voor de oplevering.

Specifieke verantwoordelijkheden

Testen van gebruiksgemak.

Aanduiden van tekortkomingen van het eindproduct.

Inzicht geven door middel van ervaring naar de uitvoerder toe.

4 Projectbeheersing

4.1 Voortgangsbewaking

Voor de voortgangsbewaking worden meerdere maatregelen genomen. De uitvoerder houdt dagelijks zijn uren bij en schrijft op wat hij in die uren gedaan heeft. Dit wordt niet alleen op papier gedaan maar ook op Tickspot.

De uitvoerder maakt wekelijks een korte blog post. Deze blog post bevat de werkzaamheden van de afgelopen week samen met uitleg indien dit van toepassing is. De posts bevatten mogelijk afbeeldingen ter verduidelijking van de situatie.

Verder zijn er nog vaste besprekingen. Deze besprekingen zijn bekend gemaakt aan de uitgever aan het begin van het studie jaar en hebben vaak nog geen vaste datum. De uitgever wordt op de hoogte gesteld van exacte datums wanneer deze naderen. De uitvoerder is verantwoordelijk voor het nakomen van al deze datums en dient de geschikte voorbereidingen te treffen.

4.2 Hoe en met welke frequentie wordt de voortgang bewaakt?

Voor sommige besprekingen moet eerst een afspraak gepland worden. Deze besprekingen worden aangeduid met het week nummer waarin de afspraak gemaakt moet worden, gevolgd door de eerste datum van de aangeduide week.

Overleg	Aanwezig	Frequentie	Tijdstip	Doel	Onderwerpen	Notulen
PID bespreking	Stage Begeleider, Uitvoerder, Stage Coördinator	Eenmalig	Week 5 P15 (29 September)	Review en beoordeling PID	PID	Uitvoerder
Tussen Presentatie	Uitvoerder, Stage Coördinator, Medestudenten	Eenmalig	Week 9 P15 (2 November	Presenteren van de stage opdracht voortgang	Stage Opdracht	Uitvoerder
Stage Verslag	Uitvoerder, Stage Coördinator	Eenmalig	Week 6 P16 (5 Januari)	Eindverslag van de stage opdracht opleveren	Stage Opdracht	NVT
Eind Beoordeling	Opdrachtgever, Stage Begeleider, Uitvoerder, Stage Coördinator	Eenmalig	Week 7/8 P16 (12 Januari)	Beoordelen van het gehele stage project	Stage Opdracht	Uitvoerder
Eind Presentatie	Uitvoerder, Stage Begeleider, Stage Coördinator, Overige belangstellenden	Eenmalig	Week 9/10 P16 (2 Februari)	Uiteindelijke beoordeling van de stage opdracht	Stage Opdracht	Uitvoerder, Stage Begeleider
Uren Verantwoording	Uitvoerder	Dagelijks	Elke middag	Uren verantwoording bijhouden	Dagelijkse bezigheden van de uitvoerder noteren	NVT
Blog	Uitvoerder	Wekelijks	Elke vrijdag	Voortgang bewaking en uren verantwoording en stage coördinator inzicht geven op werkzaamheden	Delen van de dagelijkse/wekelijkse bezigheden van de uitvoerder	NVT
Onderling Overleg	Uitvoerder, Opdracht Gever, Stage Begeleider	Continu	Dagelijkse check door stagebegeleider of uitvoerder	Het beantwoorden van vragen en onduidelijkheden	Huidige stand van zaken en eventuele problemen of onduidelijkheden	Uitvoerder indien een besluit genomen wordt

4.3 Toleranties

Alle deadlines voor dit project zijn definitief, met uitzondering van de blog posts. De blog posts kunnen namelijk ook in het weekend worden uitgevoerd. Alle andere deadlines moeten worden nageleefd, en eventuele tolerantie zal afhankelijk zijn van de uitvoerder. Wenst hij extra tijd te hebben voor correcties op zijn eindverslag, dient deze ruim voor de deadline opgegeven te worden voor review.

4.4 Risicomanagement

Risico: Tijdsgebrek

Maatregel: Aanhouden van de planning, voortgangsbewaking gebruiken, escalatieprocedure indien nodig.

Risico: Project Schaal

Maatregel: De schaal van dit project kan erg groot uitpakken. De schaal kan zo groot worden dat tijdgebrek al ontwikkelt voor het project begonnen is, omdat het uren budget het werk niet toe laat. In het geval dat dit gebeurt, wordt er een escalatie procedure gebruikt om dit te verhelpen.

Risico: Er blijkt een tool te zijn die aan alle eisen van Fantazm voldoet

Maatregel: Vroegtijdig onderzoek laten uitwijzen of dit een feit is. Indien dit probleem voor komt moet er een nieuwe opdracht geformuleerd worden met samenwerking van de opdrachtgever.

Risico: Uitkomen van de Unity3D 5.0 feature, De sequencer

Maatregel: De komst van de sequencer zou geen probleem moeten geven voor het nut van dit project. Veel van de eisen zijn namelijk niet sequencer gerelateerd. Indien mogelijk wordt deze sequencer gebruikt in het project vanaf het punt dat Unity 5.0 uit is.

4.5 Kwaliteitsbewaking

Zie kwaliteitscriteria (Bijlage B)

4.6 Wijzigingsprocedure

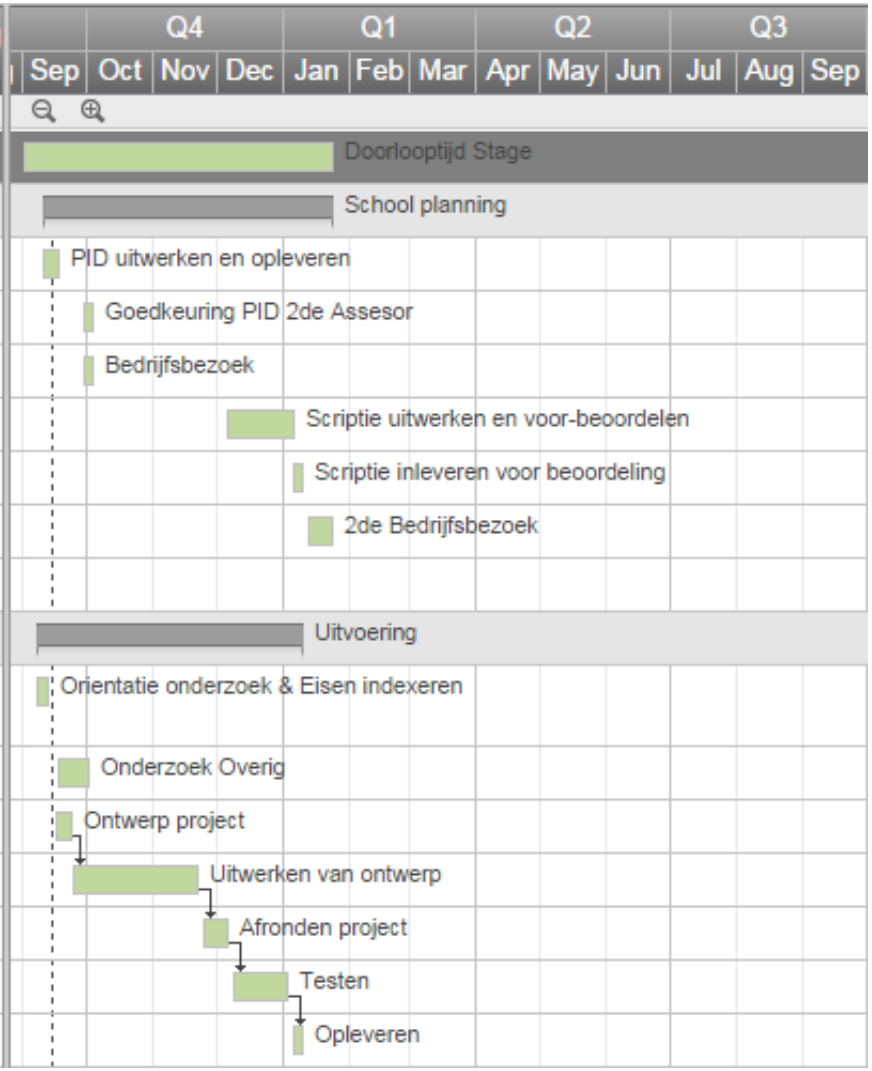
Indien er wijzigingen zijn, dan worden deze besproken met de opdrachtgever en/of de stage begeleider. Er wordt dan een besluit genomen over wat de beste oplossing is voor de wijziging. Deze aanpassingen worden doorgevoerd en genoteerd. Uiteindelijk komen ze ook op het Blog te staan, waar alle geïnteresseerden op de hoogte gehouden worden.

4.7 Escalatieprocedure

In het geval dat het project escaleert, door tijdnood of door schaal problemen, wordt het project netjes afgesloten aan het einde van de stage periode. Dit houdt in dat er met de opdrachtgever en/of de stage begeleider om de tafel gezeten wordt en men bespreekt hoe het project opgeleverd gaat worden. Men stelt minimale eisen voor de (incomplete) oplevering, waarna het project wordt afgerond op deze aangepaste oplever afspraken. Indien mogelijk zet productie zich verder na het afstuderen van de uitvoerder. In ieder geval moet het project goed overdraagbaar zijn, en correct gedocumenteerd door middel van het eindverslag.

Bijlage A: Projectplan (Smartsheet)

Task Name	Start Date	End Date	Assig To
Doorlooptijd Stage	**01-09-14**	**23-01-15**	
School planning	**10-09-14**	**23-01-15**	
PID uitwerken en opleveren	10-09-14	17-09-14	
Goedkeuring PID 2de Assesor	29-09-14	03-10-14	
Bedrijfsbezoek	29-09-14	03-10-14	
Scriptie uitwerken en voor-beoordelen	05-12-14	05-01-15	
Scriptie inleveren voor beoordeling	05-01-15	09-01-15	
2de Bedrijfsbezoek	12-01-15	23-01-15	
Uitvoering	**07-09-14**	**09-01-15**	
Orientatie onderzoek & Eisen indexeren	07-09-14	12-09-14	
Onderzoek Overig	17-09-14	01-10-14	
Ontwerp project	16-09-14	23-09-14	
Uitwerken van ontwerp	24-09-14	21-11-14	
Afronden project	24-11-14	05-12-14	
Testen	08-12-14	02-01-15	
Opleveren	05-01-15	09-01-15	



Bijlage B: Product beschrijving

Doel

Fantazm heeft als doel een 2D animatie engine die gemakkelijk in de omgang is. Bestaande tools zijn deels voorzien van alle features die Fantazm voor ogen heeft. Hieronder vallen de tools Puppet2D, Spine, Sprites and Bones en Adobe Flash. Ook zijn sommige tools erg vervelend in hun omgang, en er dus een efficiëntere manier van werken is. Ook dit efficiëntie probleem wil Fantazm in een 2D animatie engine graag verholpen zien. Zodat animeren gemakkelijk wordt, en er minder tijd in gestoken kan worden voor een gelijk resultaat.

Samenstelling

Hier volgt een lijst met eisen zoals deze nu bekend zijn. Het kan blijken dat in ontwikkeling extra eisen worden gesteld, die dan tijdens productie worden toegevoegd.

Must have

- Bone animation
- Mesh deformation
- Weight Painting (Mutli-Bone Deformation)
- Mobile compatible (Multi-Platform)
- Scalable animation
- Runtime playback standalone (Tool should not be required for playback)

Should have

- Re-use of animation (with bone alterations)
- Interpolate transitioning
- 3D skeleton use in 2D animation (3d bones/animation to 2d bones/animation)
- Use the unity Timeline (No new timeline system)
- Sprite swapping in animation
- Runtime sprite swapping and adding (IE. Adding clothing to characters at runtime)
- Full undo support
- Full shortcut support
- Standardized animation format for saving

Could have

- Inverse Kinematics
- Face Animation (Sprite swapping or bone related)
- Bones to Ragdoll (Runtime support preferred)
- Export to sprite sheet (Possible performance gain for end product)

Naast deze eisen wenst Fantazm ook nog een mogelijkheid tot cut scene animatie. Voor dit doeleinde wordt er nadat het product is gerealiseerd, een cut scene animator toegevoegd. Deze is echter compleet afhankelijk van het originele product en wordt mogelijk niet binnen dit project gerealiseerd.

Bronnen

Er is geen specifieke bron voor informatie. Veel kennis zal worden vergaard via het Internet. Alle inhoudelijk informatie, hardware, software en eisen worden aangeleverd door Fantazm. Verder wordt het programma van de grond af aan opgebouwd.

Formaat

Het eindproduct wordt een los staande Unity asset. Deze asset wordt in C# gerealiseerd. Er wordt eventueel een minimaal demo project gerealiseerd, zodat alle features gemakkelijk kunnen worden getoond aan de gebruiker als voorbeeld.

Producent

De uitvoerder is verantwoordelijk voor alle beslissingen die worden doorgevoerd. Alle beslissingen worden overlegd met de opdrachtgever voor dat deze worden doorgevoerd.

Kwaliteitscriteria

Bruikbaarheid en herbruikbaarheid van de tool
De tool moet toe te passen zijn in nieuwe projecten dan wel bestaande projecten. Voor een test wordt de tool toegepast in 2 bestaande projecten en 1 nieuw project. Ook moeten de 3 testers de tool zonder problemen kunnen gebruiken.

Flexibiliteit van de tool
De tool moet toepasbaar zijn in meerdere versies van Unity. Vanaf Unity 4.3 tot de huidige release 4.5.4 en mogelijk ook de Beta 5.0 versie, als deze uitkomt voor het einde van het project. Indien de tool compatible is met 4.3 en de nieuwste versie, kan er van uit gegaan worden dat alle versies daar tussen ook functioneren.

CPU verbruik tijdens runtime
Omdat CPU gebruik erg belangrijk is bij mobile devices, wordt er een minimale eis gesteld aan dit CPU verbruik. Op een Ipad 1 moet met behulp van de tool 5 animaties gebruikt kunnen worden met 12 bones elk, zonder dat deze onder de 30 FPS loopt.

Stabiliteit
Tijdens ontwikkeling met de tool is het een eis dat deze niet vast loopt of crashed tijdens gebruik.

Verkoopbaarheid
Verkoopbaarheid is een kwaliteitscriteria waaraan uiteindelijk voldaan moet worden. Deze criteria is hier vermeld zodat hij niet vergeten wordt, maar dient uitgewerkt te worden naar inzicht van de opdrachtgever na het opleveren van het project.

Methode van toetsing

Bruikbaarheid en herbruikbaarheid van de tool
Als de tool in alle 3 de projecten (2 bestaande, 1 nieuw) functioneert zonder stabiliteit of workflow problemen van de testers, wordt deze criteria voldoende beschouwd. Dit houdt in dat de testers na een simpel voorbeeld, de tool volledig zelfstandig kunnen gebruiken.

Flexibiliteit van de tool
Wanneer de tool werkt binnen Unity versie 4.3 en de huidige release versie (4.5.4), gaat men er van uit dat alle tussenliggende versies ook functioneren. Indien dit het geval is, kan deze criteria voldoende worden beschouwd.

CPU verbruik tijdens runtime
CPU gebruik wordt gemeten op een Ipad 1. Indien de Ipad 1 met minimaal 5 animaties, met elk 12 bones, op 30 FPS kan afspelen, is deze criteria geslaagd.

Stabiliteit
Indien de tool tijdens het testen van de bruikbaarheid niet crasht of vastloopt, wordt deze criteria als voldoende beschouwd.

Verkoopbaarheid
Zie Kwaliteitscriteria.

15 Bijlage B – Vooronderzoek bestaande tools

In deze bijlage wordt beschreven hoe alle tools naast de eisen van Fantazm zijn gelegd. En mijn persoonlijke aanbevelingen over deze tools. Deze gehele bijlage is uiteindelijk bij Fantazm neergelegd voor een uiteindelijke beoordeling over welke richting de opdracht moest gaan.

15.1 Tools

15.1.1.1 Smoothmoves

Smoothmoves is een tool dat binnen Unity3D werkt. Het heeft zijn hele eigen editor wat in eerste opzicht een positief punt is. Maar Smoothmoves scoort erg laag op gebruiksvriendelijkheid: Shortcuts werken niet of nauwelijks, de eigen gemaakte tijdlijn is onlogisch in gebruik en de animaties claimen met bones te werken, maar in werkelijkheid zie je weinig terug van de voordelen die dat met zich mee zou moeten brengen. Het resultaat is een erg onoverzichtelijke tool met weinig features die animeren gemakkelijker maken voor de gebruiker. Om deze redenen is Smoothmoves niet nader onderzocht.

Figuur 15 - Smoothmoves Logo (echo17, 2014)

15.1.1.2 Spine

Spine is een tool die niet binnen Unity3D werkt. Het heeft een runtime plugin zodat Unity3D de animaties die in spine gemaakt zijn, kan afspelen. Dit maakt Spine erg ongemakkelijk in gebruik omdat elke bewerking of aanpassing weer opnieuw geïmporteerd moet worden naar Unity3D. Verder is het programma niet uit te breiden met extra features. Ook is Spine erg duur in aanschaf tegen over de andere tools en kan de trial versie niet exporteren naar Unity3D. Het testen van de tool is hierdoor ook vrijwel onmogelijk. Op de basis van het feit dat Spine niet uitgebreid kan worden en het niet aan belangrijke eisen voldoet is Spine niet meegenomen in het vervolgonderzoek.

Figuur 16 - Spine Logo (Esoteric Software, 2014)

15.1.1.3 Flash

Op verzoek van Fantazm is Flash meegenomen in het onderzoek. Flash is niet in staat te exporteren naar Unity3D en er is geen mogelijkheid om animaties te maken die in Unity3D af te spelen zijn via bone animation. Om deze reden is Flash niet meegenomen in het vervolgonderzoek.

15.1.1.4 Puppet2D en Sprites and Bones

De tools Puppet2D en Sprites and Bones lijken qua features veel op elkaar: Ze maken beide gebruik van bone animation en integreren vrijwel alles met de Unity3D interface en gebruiken Unity3D file types. Ook zijn beide in een formaat waarin aanpassingen kunnen worden gemaakt. Op basis van deze voordelen worden deze tools verder meegenomen voor een diepere vergelijking met de eisenlijst van Fantazm.

Figuur 17 - Puppet2D Logo (Niman, 2013)

15.1.2 Vervolg onderzoek resterende tools

Puppet2D en Sprites and Bones zijn nader geanalyseerd op basis van de eisen van Fantazm. De volgende matrix geeft de uitkomst van de analyse van beide tools weer.

Features	Demanded System	Puppet2D	Sprites and Bones
Mobile Compatible	Demand	Yes	Yes
Multi Layer Support	Demand	Yes	Yes
Mesh Deformation	Demand	Yes	Yes
Multi Bone Influence	Demand	Yes	Yes
Cut Scene Animation	Demand	No	No

Re-use Animation	Desired	Yes	Yes
Re-use Animation Altered Skeleton	Desired	No	No
Smooth Transitioning (Interpolate)	Desired	Yes	No
3D skeleton to 2D	Desired	No	No
Scaling Animation	Desired	Yes	Yes
Unity Timeline	Desired	Yes	Yes
Sprite Swapping	Desired	Yes	Yes
Runtime Sprite Swapping	Desired	Yes	Yes
Audio Synchronization	Desired	Yes	No
Drawing Support	Desired	No	No
Undo	Desired	Yes	Yes
Shortcut keys	Desired	Yes	Yes
Inverse Kinematics	Preferred	Yes	Yes
Face animation	Preferred	Yes	No
Physics (Ragdoll)	Preferred	No	Yes
In Unity Editor	Preferred	Yes	Yes
Trial version	Preferred	No	Free!
Runtime Playback	Unity Preferred	Unity Animation	Unity Animation
Save Format	Unity Preferred	Unity Animation	Unity Animation
Export Formats	Unity Spritesheet	None	PNG (low Q for non-pro)
CPU usage	Lowest Possible	Low enough	Low enough

15.2 Conclusie vooronderzoek

Uit een vergelijking van Puppet2D en Sprites and Bones blijkt dat Puppet2D overal een streepje voor heeft. Uiteindelijk is in alle aspecten wel te merken dat Puppet2D beter ontwikkeld is. Puppet2D heeft meer features dan Sprites and Bones, een betere GUI en maakt beter gebruik van de integratie met Unity3D elementen. Beide tools zijn ook geschreven en aangeleverd in C# scripts. Dit maakt de tools gemakkelijk aan te passen en uit te breiden. Zo kunnen de gebrekkige features worden toegevoegd. Het enige nadeel is dat Puppet2D een betaalde tool is en Sprites and Bones een gratis tool.