MULTI-PLATFORM DEVELOPMENT

Documentatie

18 Januari 2019

Aaron Oostdijk

Matthew van Eenenaam - Game Development

Nevyl Mennens - Game Design

Inhoud:

Concept:

Het soort game wat wij voor ogen hadden was het gevoel creëren van (hoge) snelheid. Met behulp van VR wilden wij dit gevoel versterken door de speler in de cockpit te plaatsen van een futuristische raceauto. Omdat de speler in de werkelijkheid ook stil zit, sluit dit goed aan op onze gameplay. De speler kan dan controllers van de VR-set gebruiken om zijn handen als stuur te gebruiken.

Platforms:
- VR
- PC
- Controller*

Kerngebied:
De game zal draaien op PC maar zal onderscheid maken tussen de input en kijken of er gebruik wordt gemaakt van een VR-set of muis en toetsenbord.

Gameplay:
Geinspireerd door bijvoorbeeld series als F-Zero en Mario Kart (“Rainbow Road”), is de bedoeling van het spel om 3 rondes zo snel mogelijk te rijden zonder van het level af te rijden. Als je wel van het level af valt, wordt je met een vertraging weer terug op de racebaan gezet. Op de racebaan bevinden zich poortjes waar de speler doorheen kan gaan. Hiermee wordt zijn “Boost” waarde gevuld en er met een boost de snelheid met 150% toegenomen worden. Dit is handig op rechte stukken of wanneer je van het level bent af gevallen.

Controls:
W = Voorwaarts
S = Achteruit / Remmen
A = Links
D = Rechts
Space = Boost

* wegens tegenslag met de Oculus VR, wordt er in plaats daarvan nu gekeken naar een Xbox360 controller

Proces (Matthew):

We begonnen het project met het concept om een racing game voor Vr en desktop te maken. Omdat ik nog niks kwa VR had gedaan ben ik dus eerst begonnen met tutorial kijken over hoe je oculus met unity werkend krijgt . dat ging de eerste paar weken nog best aardig. Ik had een test object voor het autoje en het stuur in de scene en begon de acties in de game minder direct op de acties van de controls te baseren maar op de posities van objecten in de auto gameobject.

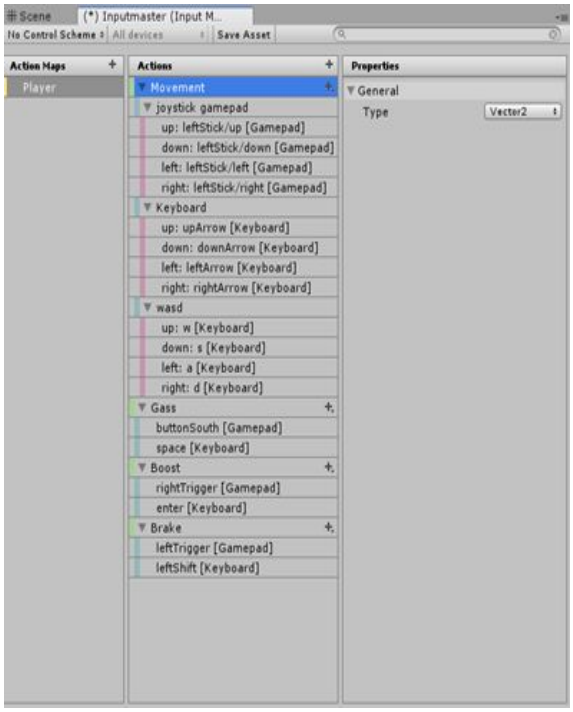
Dit ging allemaal best goed you know totdat het heel fout ging. Op een gegeven moment in ongeveer week 5 kreeg ik allemaal problemen met oculus . Eerst was het steam vr die mijn trackers niet meer kon vinden. Dat losten zich zelf op na een update van steam vr de volgende dag. Maar toen het eenmaal opgelost was wilden oculus en unity niet meer leuk met elkaar spelen. Na een tijd troubleshooten bleek het dat unity mijn headset kwijt was geraakt (misschien door de steam vr update maar not sure). Opgezocht hoe je zoiets weer opnieuw linked . maar geen succes mee gehad.

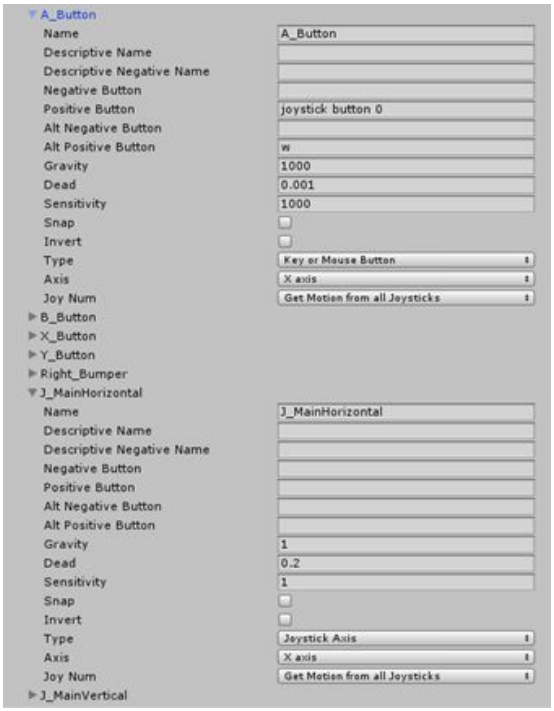
Dit betekende dus dat het concept veranderde moest worde. Je kan natuurlijk geen vr game make zonder VR. Dus toen hebben we het concept getweaked naar iets wat makkelijker was aangezien we nu minder tijd hadden . De game zou nu in plaats van een firstperson racer een third person racer woorden. Die in plaats van VR en Keyboard met Controller en Keyboard wordt bestuurd. De kern van het idee bleef hetzelfde maar we hebben de camera angle veranderend en 1 van de input manieren.

Maar ik wou toch nog iets nieuws proberen en rond deze tijd kwam ik een video van Brackeys over het input systeem voor 2019 unity tegen dat in een early acces state was. Dus ben ik daar mee aan de slag gegaan . in dit systeem kan je hele control schemes voor verschillende input devices heel makkelijk samenstellen en aanroepen in code. Perfect voor dit vak dacht ik. Maar toen liep ik tegen iets aan met verschillende controllers (360, switch pro, dualshock 4) waar hij de stick input nooit als een negatief getal wilt lezen. Het laagste dat die leest is 0 en in het midde ook 0 . dat woord best een probleem als je ooit een keer naar links zou willen in je race game .

dit heb ik best lang geprobeerd te troubleshooten maar omdat het nog zo new was ,was er niet veel info op het internet waar ik wat mee kon zeg maar. En de deadline kwam steeds dichterbij.

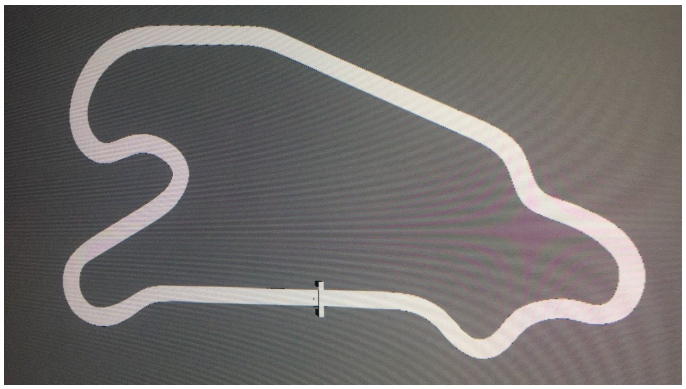
Dus uiteindelijk heb ik maar besloten om terug naar het oude systeem te gaan en daar in te werken.
Daarin heb ik nieuwe axeses in gemaakt die ik return in een input manager script wanneer daar naar wordt gevraagd door het carController script. Dit ging allemaal met redelijk weinig tegenslag.



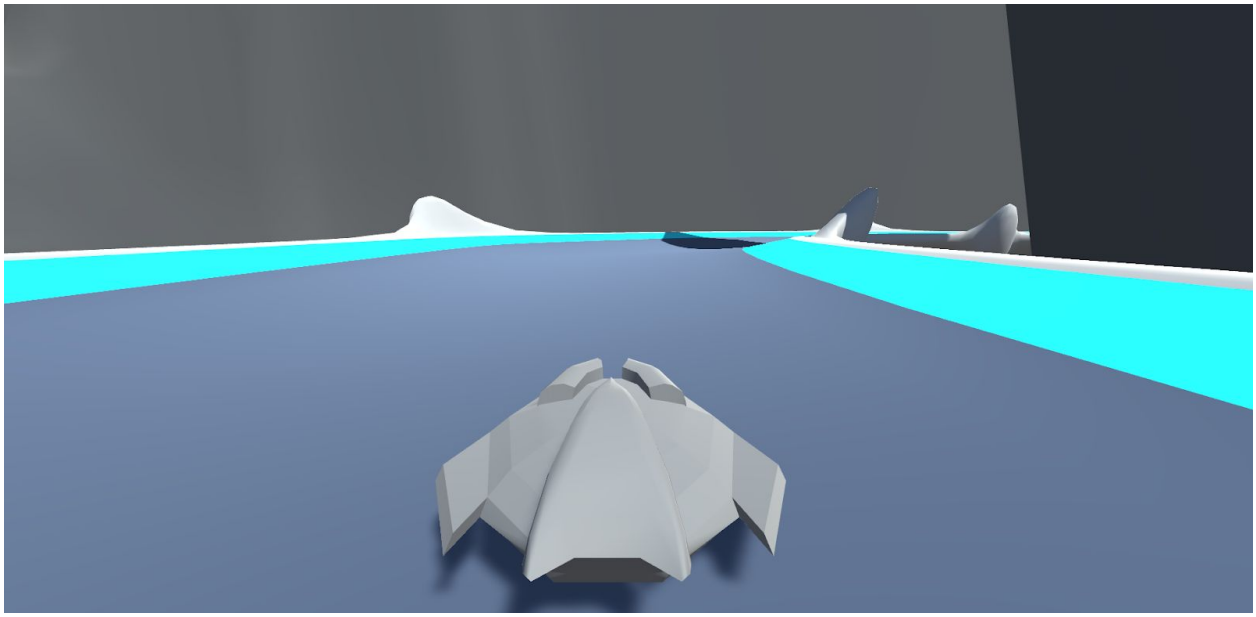


Het enige andere wat hierna nog fout is gegaan is gekloot met rigidbody velocitys. Door de manier waarop wij de car bewegen heeft gravity bijna geen effect op de car wanneer hij van de baan af vliegt. Dit heb ik op een beetje hacky manier opgelost door te zeggen dat hij alleen de forward movement kan uitvoeren als een raycast hit van de car naar de grond binnen een bepaalde range. Hierdoor kan de auto gewoon normaal van de baan af vallen zoals het hoord.

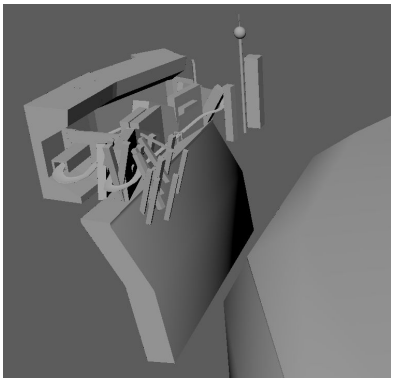
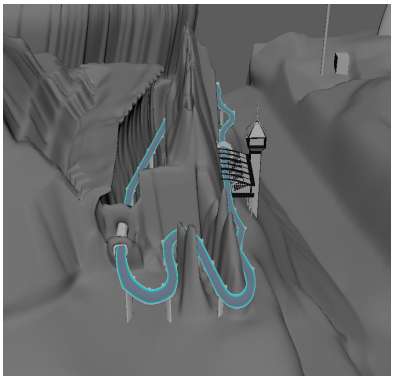
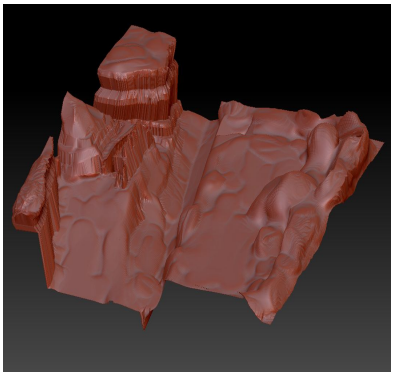
Proces (Nevyl):



Track:
Bij het maken van de track heb ik geprobeerd het simpel te houden maar wel dat er nog een uitdaging in zit. Ik heb geprobeerd het zo af te wisselen dat er ruimte is voor stukjes met een aantal bochten als het kunnen boosten (rechte stukken).



Bochten:
De zijkant van de racebaan heeft een kleine hoek omhoog zodat bochten beter in te schatten zijn. Delen van de baan hebben uitsteeksels zodat de speler een gevoel van snelheid krijgt wanneer hij deze passeert.

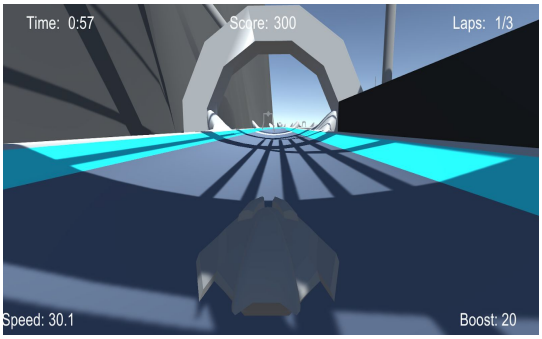




Environment:
Voor het maken van de omgeving heb ik geëxperimenteerd met hoe ik dit het beste aan kan pakken. Ik begon in Unity met een block-out maar bedacht me al snel dat ik dit veel beter in Maya kan doen omdat ik in Maya meer vormen aan kan brengen. Nadat ik mijn block-out af had was mijn eerste poging om het terrein te maken met de Unity Terrain Editor. Naar mijn beleving werkte de terrain brushes niet echt goed mee. Het verhogen en verlagen van het terrein voelde erg inconsistent en was laggy. Zo leek ik de ene keer meer omhoog of omlaag te gaan dan andere keren dat ik klikte waardoor ik veel tijd verloor voor een goede uitwerking.

Omdat ik er naar mijn gevoel erg lang over deed besloot ik om te kijken of er een alternatief hierop is en wat de voor- en/of nadelen zullen zijn. Ik maakte in Maya een enorm vlak en begon met hoogtes te spelen. Al snel merkte ik dat het een stuk aangenamer werkte omdat ik meer controle had over de exacte vorm van het terrein. Om van de hoekige vormen af te komen heb ik geëxperimenteerd door de mesh in ZBrush te gooien en te kijken of het handig is om op deze manier details aan te brengen. In eerste oogopslag leek het erg goed te werken, maar begon het programma erg vervelend te werken zodra ik veel moest inzoomen.

De uiteindelijke basis vorm vind ik er mooi uitgekomen en heeft alles de juiste hoogtes zoals ik het ook in gedachten had, echter raad ik het af om op zo’n grote schaal een map in ZBrush te maken.

Uiteindelijke Ontwerp:



In het input systeem van unity maken we nieuwe axeses aan die we koppelen aan bepaalde acties op het keyboard en de xbox 360 controller.

In de input manager maken we een paar functies aan die uitlezen of de knoppen die we aan de axeses hebben gekoppeld worden ingedrukt . Deze returnen dan een value waar de code in de car script iets mee kan. Als voorbeeld : als de MainHorizontal() een float boven 0.2 geeft dan wordt de turnInput in car.cs zo gezet dat hij naar rechts begint te sturen.

Hierdoor zorgen we dat we niet voor de keyboard en xbox controllers verschillende inputs checks hoeven te schrijven die hetzelfde doen . En kunnen we makkelijk nieuwe controllers toevoegen door hun axeses toe te voegen onder de functies in het inputManager script.

Uml: