# Gymnázium, Praha 6, Arabská 14

Programování

# Ročníková práce

Jan-Matyáš Martinů

Duben 2020

# Gymnázium, Praha 6, Arabská 14

Programování

# Ročníková práce

**Předmět:** Programování

**Téma:** Závodní hra

**Autor:** Jan Matyáš Martinů
Vondrášek

**Třída:** 4. E

**Školní rok:** 2019/2020

**Třídní učitel:** Mgr. Jana Urzová

# Prohlášení

Prohlašuji, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V roce……...dne……..

_____

*vlastnoruční podpis*

# Anotace

## Česky:

Tato ročníková práce se zabývá vytvořením závodní počítačové hry pro jednoho hráče s procedurálně vygenerovaným světem. Rozebírá popis použitých technologií. Dále práce popisuje vývoj samotné hry. V práci jsou popsány základní principy vytváření procedurálních modelů, využití enginu a herní fyzika.

## English:

This work focuses on creating single-player computer game with a procedurally generated world. Analyzes the description of its technologies. Further work describes development of game. The work describes the basic principles of creating procedural models, the use of the engine and game physics.

## Zadání projektu:

Cílem práce je vytvoření 3D závodní hry s procedurálně vygenerovaným světem. Základní myšlenkou projektu je vygenerování herního světa např. jízdní trati, krajiny či mest podle daných pravidel. Herní svět by měl vypadat esteticky dobře. Hráč bude moct jezdit závody na čas.

# Obsah

1. Úvod ..... 5
2. Použité technologie ..... 6
2.1 Unity ..... 6
2.2 Maya ..... 6
2.3 Speedtree ..... 6
2.4 Substance Painter ..... 6
2.5 Visual Studio Code ..... 7
2.6 Photoshop ..... 7
2.7 Houdini ..... 7
2.8 HoudiniEngine ..... 8
3. Herní fyzika ..... 9
3.1 Princip jízdy ..... 9
3.2 Princip zatáčení ..... 10
3.3 Rychlost a zrychlení ..... 11
4. Generování mapy ..... 12
4.1 Princip vytváření trati ..... 12
4.2 Princip vytváření obrubníků ..... 13
4.3 Princip vytváření svodidel ..... 14
4.4 Princip vytváření okolního terénu ..... 15
5. Grafická část projektu ..... 16
5.1 textury/materiály ..... 16
5.2 Modely ..... 16
6. Audio ..... 17
7. Závěr ..... 18
8. Seznam obrázků ..... 19
9. Zdroje ..... 19

# 1. Úvod

V této práci budu popisovat, jak jsem vytvářel svou závodní hru. Již od malička jsem hrál na počítači a jelikož mezi mé nejoblíbenější žánry patří hry závodní, pokusil jsem se vytvořil svou vlastní. Hlavní motivací k výběru tohoto témata bylo lepší porozumění fungování herních enginu a rozšíření vědomostí o způsobech generování herních světů.

Začátek práce pojednává o obecných informacích a využití jednotlivých programů. Dále postupně navazuje na informace o podrobnějším vývoji a funkcionalitě. Na konci se pak můžeme dočíst o grafickém výstupu a zvuku.

Cílem projektu bylo vytvořit plně hratelnou závodní hru s vygenerovaným terénem a tratí. Hra obsahuje herní mód “jízda na čas”, kdy hráč musí po odstartování hry, projet celou tratí v co nejkratším čase. Hráč si v hlavní nabídce může vybrat ze čtyř různých vozů a čtyř různých před generovaných map. Při samotné jízdě hráč vidí v levém horním rohu svůj aktuální čas a v levém dolním rohu svou aktuální rychlost znázorněnou na tachometru. V průběhu hry je také možné měnit pohled kamery či hru celkově pozastavit. Projekt nedělá tolik zajímavým samotná hratelnost hry, ale spíše způsob vývoje, protože byly využity programy jako např. Houdini, Speedtree, Maya, Substance Painter či Unity.

# 2. Použité technologie

## 2.1 Unity

Unity je herní engine, který byl použit při tvorbě tohoto projektu. Unity bylo oficiálně vydáno v roce 2005 pro OS X. Postupem času bylo rozšířeno pro dalších patnáct platforem. Je určeno primárně pro tvorbu 2D a 3D her. Skripty se v Unity píší nejčastěji v jazyce C#, ale je možné použít i JavaScript či UnityScript. V Unity bylo vytvořeno plno celosvětově známých her jako například Pokemon Go nebo Angry Birds.

Protože byl projekt vytvářen ve 3D, bez použití herního enginu by tuto práci nebylo možné stihnout v rozumném časovém úseku. Hlavním důvodem výběru právě tohoto herního enginu byla kompatibilita s programem HoudiniEengine a dále pak možnost používat programovací jazyk C#. Velikou výhodou Unity je také jeho jednoduchost, srozumitelnost a všestrannost. (17)

## 2.2 Maya

Autodesk Maya je profesionální software určený pro vytváření 3D modelů, animací a mnoho dalšího. Ve filmovém průmyslu je Maya standardem pro 3D vizuální efekty. V projektu byla Maya využita pro vytvoření procedurálně negenerovaného terénu. Jako příklad lze uvést hory ohraničující mapu. Výhodou při práci bylo, že Maya podporuje Unity plugin pro přímý export modelu do scény enginu

## 2.3 Speedtree

Speedtree je program zaměřený na vytváření vegetace. Má velmi šikovné nástroje k vytváření a generování stromů. Velice dobře se v něm také vytvářejí věrné fotorealistické modely. Speedtree se běžně využívá ve velkých filmových a herních studiích. V projektu byl využit k vytvoření stromů. Ve hře můžeme najít 4 druhy stromů o různých velikostech.

## 2.4 Substance Painter

Substance Painter je program používaný k malování 3D textur na modely v reálném čase. Nabízí širokou škálu inteligentních materiálů, které se dokážou přizpůsobit modelu tak, aby vykazoval například realistické opotřebení. Program má velice blízko k Photoshopu, až na to že se pracuje ve 3D prostoru. V roce 2019 byl Substance Painter společně se Substance Designerem odkoupen Adobe. V projektu byl použit k texturování některých aut. (15)

## 2.5 Visual Studio Code

Společně z Unity lze využívat jakékoliv vývojové prostředí podporující C#. K projektu bylo využito Visual Studio Code. Sloužilo pouze k psaní a formátování kódu. Jeho výhodou je přehlednost a hezký vzhled. Z osobní zkušenosti bych raději doporučil základní Microsoft Visual Studio, které je dodáváno společně s Unity. V průběhu práce totiž často docházelo k neopodstatněným chybám díky špatné kompatibilitě.

## 2.6 Photoshop

Photoshop byl v projektu využit k navržení jednotlivých tlačítek a obrázků v menu hry.

## 2.7 Houdini

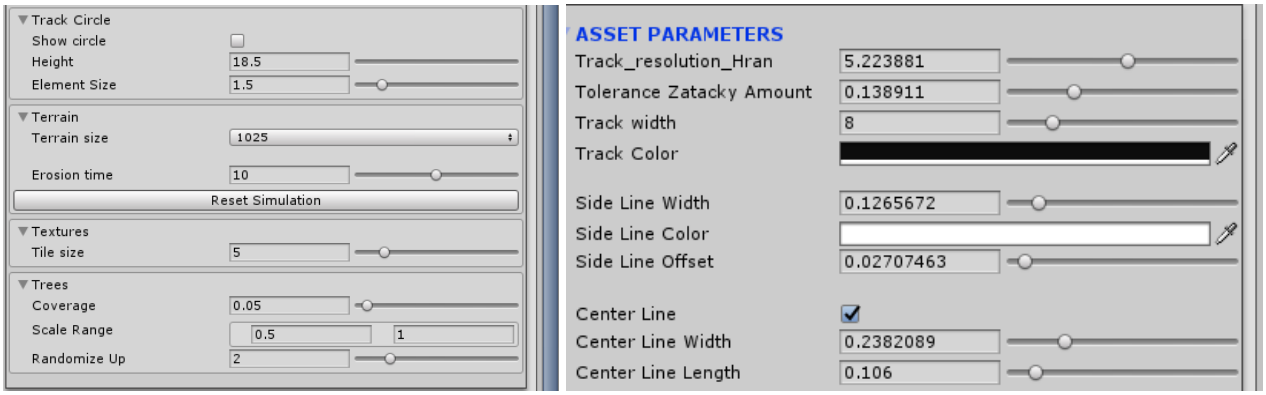
Houdini je 3D software primárně vyvinut pro vytváření speciální efektů do filmů a her. Používá se v každém velkém studiu a je průmyslovým standardem.

Jeho hlavní síla spočívá ve velmi dobře propracovaném systému uzlů, který společně s programovacím jazykem Python a jazykem Vex určeným přímo pro houdini, tvoří dokonalý nástroj pro tvorbu procedurálních objektů. V projektu byl používán velice mnoho, hlavně k vytvoření všech generovaných modelů a také k celému fungování dráhy a okolí hry.

## 2.8 HoudiniEngine

Aby Houdini fungovalo společně s Unity, je potřeba využít Houdini Engine. Houdini Engine je software od stejných vývojářů, který slouží k propojení herního enginu a Houdini. HoudiniEngine momentálně podporuje Unreal engine a Unity. Velkou nevýhodou je, že při propojení musí běžet jak engine tak Houdini, což znamená, že po zkompilování hry již propojení nefunguje.

Po nainstalování HoudiniEngine se v hierarchii Unity vytvoří složka jménem plugins, kde jsou uloženy všechny HDAs. HDA je tzv. *houdini digital asset*. To je speciální formát modelu vytvořený v Houdini, který obsahuje uživatelem vytvořené atributy viz obrátek číslo 1 potřebné pro výslednou úpravu v editoru Unity. Díky těmto atributům lze v projektu upravovat např. tloušťku trati, tvar, délku svodidel či barvu vodorovného dopravního značení.



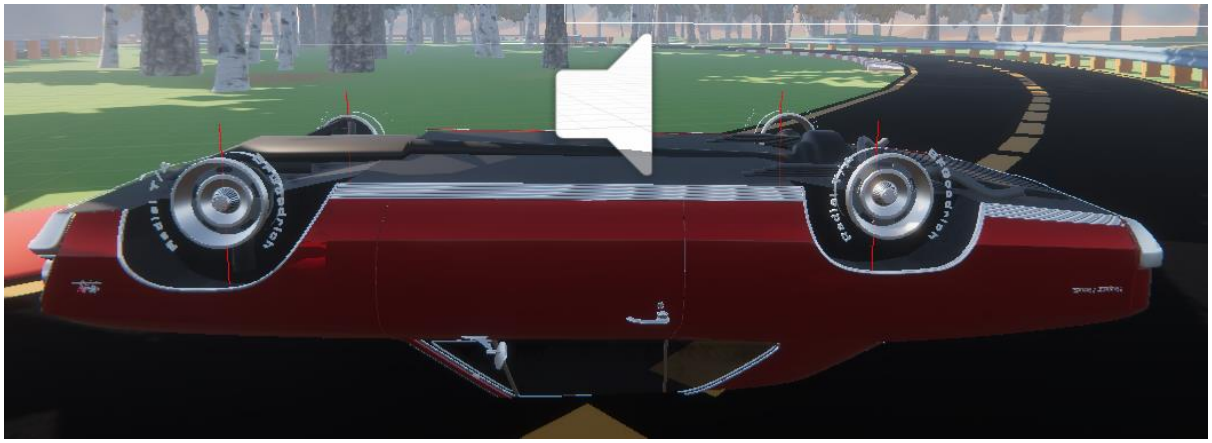

Obrázek 1: Atributy v Unity editoru (29. 04. 2020)

# 3. Herní fyzika

Při programování herní fyziky, bylo cílem vytvořit alespoň z části realistický pocit z řízení jednotlivých aut. Hra obsahuje zatím 4 druhy aut, ale jejich počet není ničím limitován. V budoucím vývoji by jich bylo možné přidat víc pro lepší herní zážitek.

## 3.1 Princip jízdy

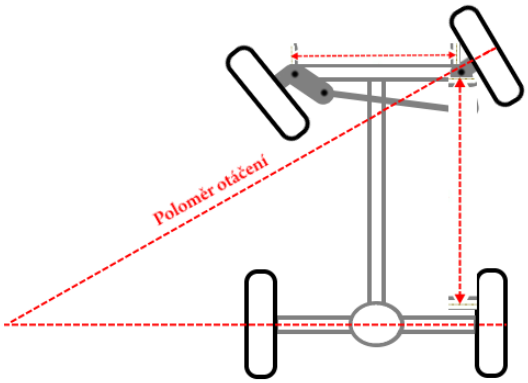
Unity obsahuje třídu *Wheel Collider*, která je určena pro napodobení podvozku a kol auta. Její implementace je velice jednoduchá, ale nenabízí dostatečnou kontrolu nad jízdními vlastnostmi. Proto byl v projektu použit jiný způsob. Ze čtyř bodů viz obrázek číslo 2 je vysílán paprsek směrem do země, nazývaný jako *raycast*. Délka toho paprsku se odvíjí od předdefinovaných parametrů. Tyto parametry imitují Tvrdost pružin, délku pružin a poloměr kol auta. Když na auto poté působí boční síla, podle délky a tvrdosti se auto naklání na stranu. Když auto najede na nerovnost díky paprsku nemůže nikdy dojít k zaseknutí kola. Auto se pouze lehce nadzvedne a nakloní podle velikosti nerovnosti. Velikost jak velkou překážku může auto přejet ovlivňuje vzdálenost pružin a poloměr kol. Důležité je také, že model auta nehraje žádnou roli v jízdních vlastnostech. Záleží pouze na pozici paprsku a nastavených parametrech.



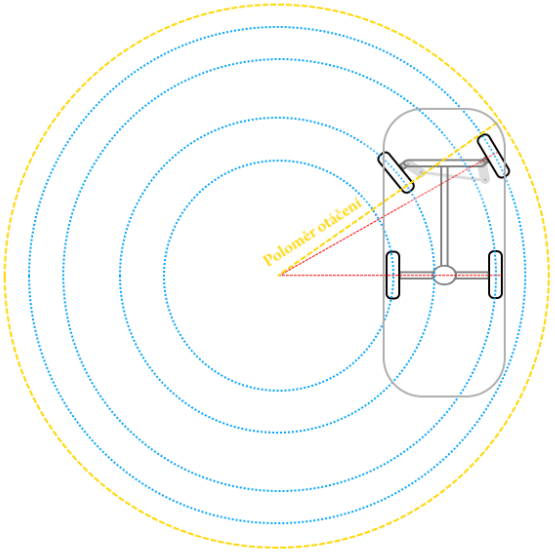
Obrázek 2: Raycast paprsek (29. 04. 2020)

## 3.2 Princip zatáčení

Když auto zatáčí, každé z předních kol se otáčí o určitý úhel. Tento úhel u obou kol není stejný viz obrázek číslo 3. Kolo na vnitřní straně zatáčky se otáčí o trochu více, než na vnější straně. K výpočtu tohoto úhlu potřebujeme znát vzdálenost mezi předními koly viz obrázek číslo 4 a vzdálenost mezi předními a zadními koly. Poté pomocí rádiusu zatáčení daného auta viz obrázek číslo 4 lze goniometrickou funkcí Arkus tangens zjistit dané úhly. Na internetu jsou jednoduše dohledatelné reálné vzdálenosti mezi předními a zadními koly či rádiusu zatáčení. Tyto parametry jsou implementovány v projektu. Pro zatočení auta se poté jednoduše aplikuje síla ve směru kol. (3)



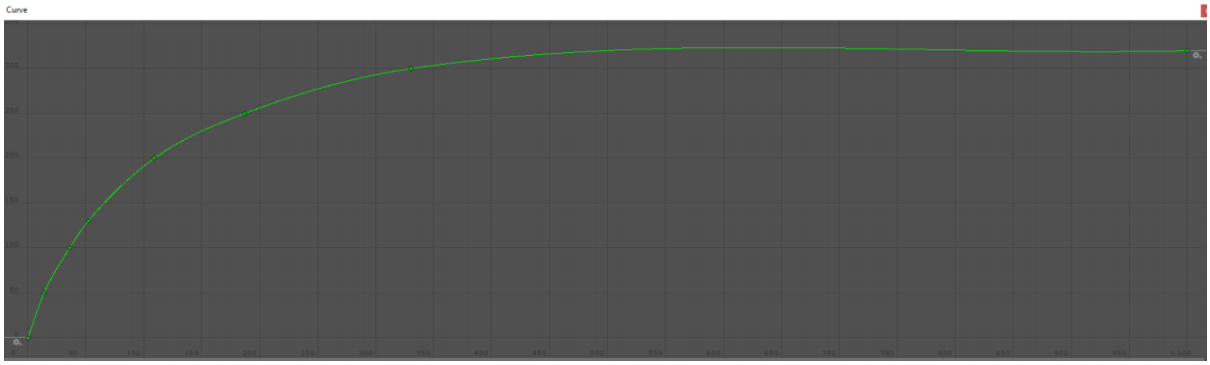

Obrázek 3: vzdálenost kol (29. 04. 2020)




Obrázek 4: rádius otočení (29. 04. 2020)

## 3.3 Rychlost a zrychlení

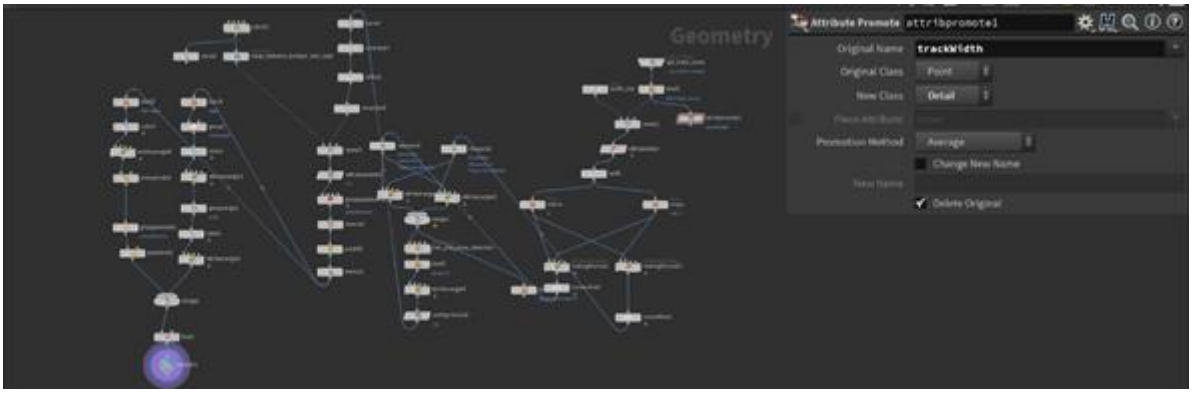
Pro docílení realistického zrychlení, má každé auto ve hře svou vlastní akcelerační křivku. Tato křivka je imitací reálných akceleračních křivek uváděných u jednotlivých modelů aut. Například Porsche 911 GT2 implementované ve hře, zrychluje z 0 km/h na 100 km/h podle akcelerační křivky viz obrázek číslo 5. Na ose x je vyobrazen čas v desetinách sekundy a na ose y je vyobrazena rychlost v km/h. Výhodou tohoto řešení je snadná implementace a obstojná fyzikální přesnost. Při hraní hráč vidí svou rychlost na tachometru, který zobrazuje aktuální rychlost vozu.




Obrázek 5: akcelerační křivka (29. 04. 2020)

# 4. Generování mapy

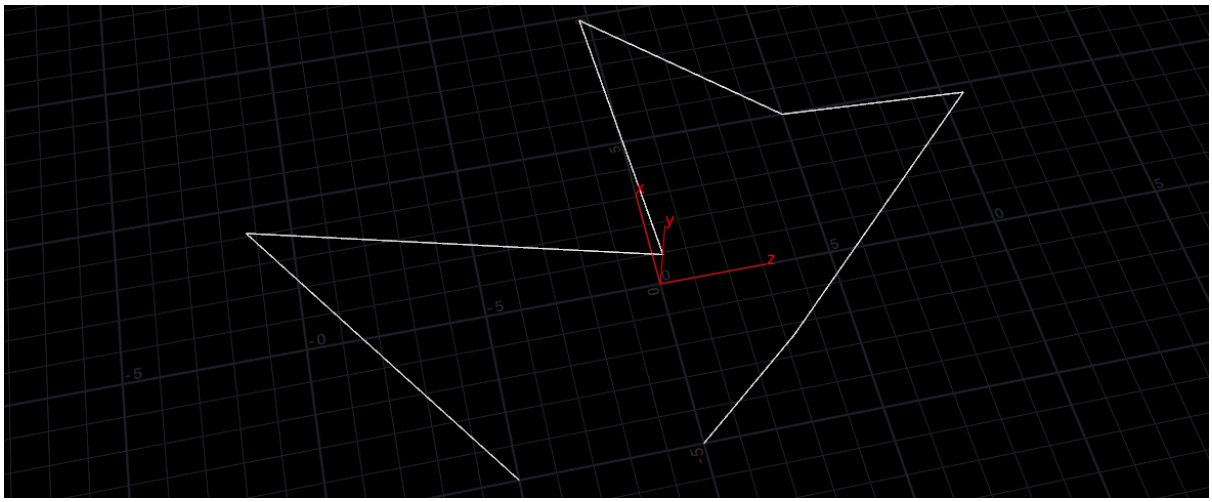
Veškeré generované části mapy byly vytvořeny již ve zmiňovaném programu Houdini. Vytváření geometrie je v Houdini zpracováno pomocí uzlů viz obrázek číslo 6. Díky těmto uzlům je možné vytvářet plně procedurální modely. Celý vývoj velmi ulehčil fakt, že generovaná trať je v rovině.




Obrázek 6: uzly v programu Houdini (29. 04. 2020)

## 4.1 Princip vytváření trati

Nejprve si v prostoru zvolíme osm náhodných bodů v rovině viz obrázek číslo 7. Ty propojíme tak, aby vznikla uzavřená křivka. Pomocí uzlu *Sweep* nakopírujeme ke každému z osmi zvolených bodů kolmou úsečku. Délka této úsečky nám udává šířku trati, a tak ji využijeme jako atribut. Je velmi důležité si hlídat, aby modely nebyly na ruby, protože by se u nich poté nezobrazovala textura ani materiál.




Obrázek 7: body v prostoru (29. 04. 2020)

Pro vytvoření prostřední linky vodorovného dopravního značení využijeme již vytvořenou uzavřenou křivku z osmi bodů. Pomocí uzlu *Carve* rozdělíme křivku na malé úsečky připomínající prostřední linku vozovky.

Lajny na stranách silnice vytvoříme jednoduše z geometrie asfaltové cesty, kterou zkopírujeme a pouze upravíme vzdálenost atributu udávající šířku trati. Z modelu poté extrahujeme okrajové křivky a máme postranní lajny silnice. Na konec stačí pouze přidat materiál a vytvořit potřebné atributy.

Tímto způsobem je docíleno plně procedurálního modelu. Pokud například změníme souřadnice jednoho z osmi zvolených bodů, tvar trati se automaticky upraví a nedojde k žádné deformaci modelu.

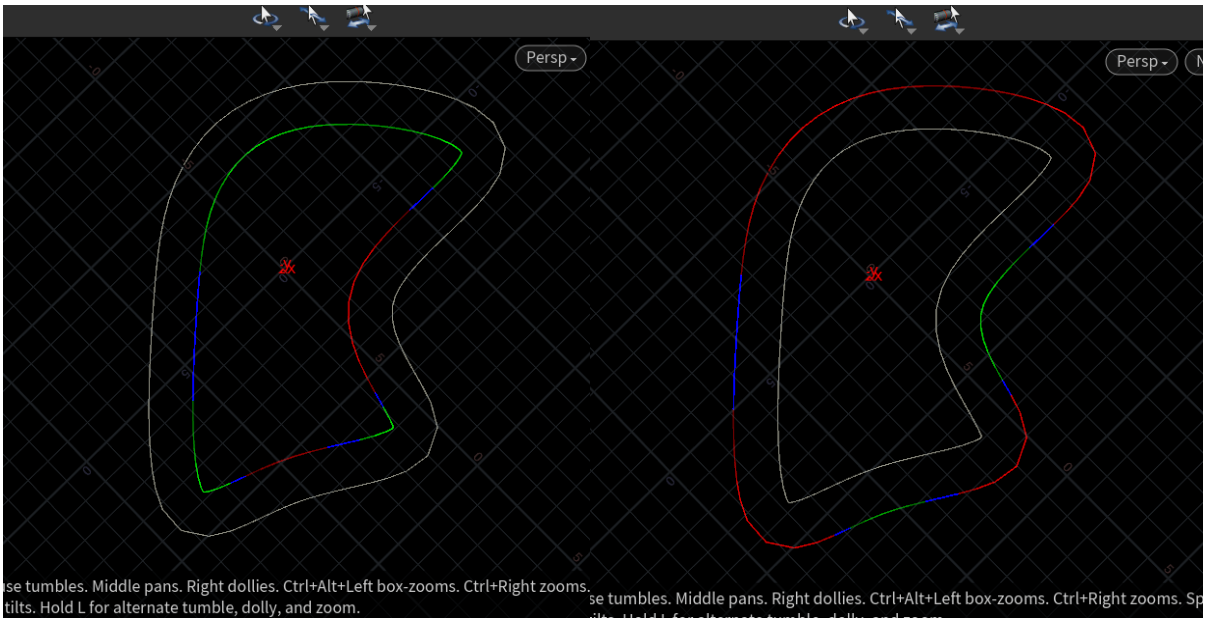
## 4.2 Princip vytváření obrubníků

Z geometrie asfaltové cesty nejprve extrahujeme vnější lajny. Obrubníky jsou na závodních tratích vždy pouze z vnitřních stran zatáček. Musíme tedy nějak rozlišit vnitřní a vnější stranu. Nejprve od sebe oddělíme obě lajny. K rozlišení poté použijeme jednoduchý algoritmus, kdy porovnáváme vzdálenost úseček na obou stranách trati viz obrázek číslo 8.

```
float otherLenght = prim(1,"lenght",@primnum);
float toleranceStraights = chf('toleranceStraights');
float difference = clamp(abs(f@lenght - otherLenght), 0, 1);
f@sdsj = difference;
if(f@lenght>otherLenght)
{
    @Cd = set(difference, 0, 0);
}
else
{
    @Cd = set(0, difference, 0);
}
if(difference < toleranceStraights)
{
    @Cd = set(0,0,255);
}
i@outsideCruve = 1;
```

Obrázek 8: algoritmus detekce zatáček (29. 04. 2020)

Na té straně, kde je úsečka kratší, je vždy vnitřek zatáčky. Všechny vnitřní strany zatáček označíme zelenou barvou viz obrázek číslo 9 a všechny vnější strany zatáček červenou barvou viz obrázek číslo 9.




Obrázek 9: vizualizace zatáček (29. 04. 2020)

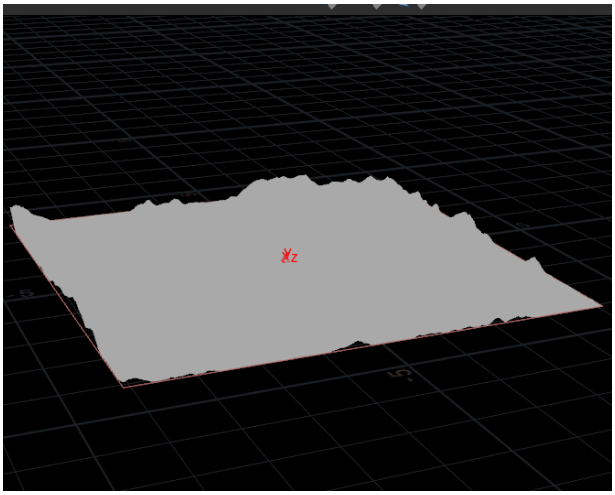
Aby se zatáčky nezobrazovali po celé délce trati, je potřeba vytvořit atribut, který určuje kdy je trať rovná. Veškeré rovné úseky pak vyznačíme modrou barvou. Poté už je potřeba jen rozlišit, zda se obrubník nachází na vnitřní či vnější lajně a podle toho jej řádně otočit správným směrem. Na konec se přidají materiály a vytvoří potřebné atributy.

## 4.3 Princip vytváření svodidel

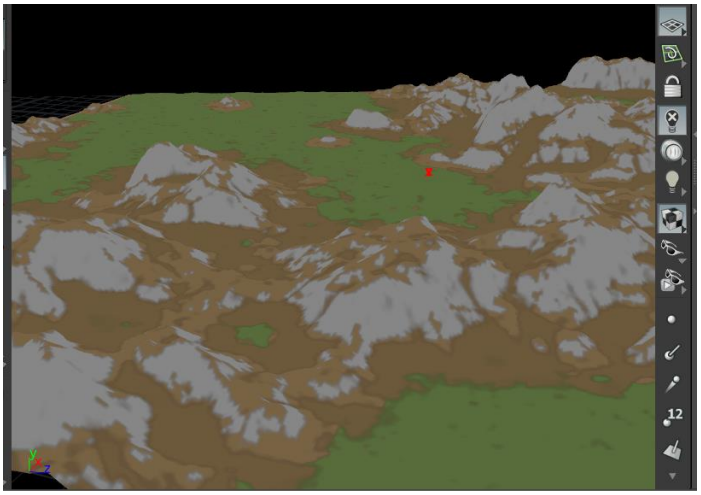
Svodidla bývají vždy z vnějších stran zatáček. Vzhledem k tomu, že vnitřky a vnějšky již máme rozlišené, stačí pouze vytvořit procedurální geometrii svodidel a správně ji natočit. Poté přidáme materiály a vytvoříme potřebné atributy.

## 4.4 Princip vytváření okolního terénu

Houdini obsahuje velmi mnoho uzlů pro ulehčení práce s terénem. Na plochu ve tvaru čtverce aplikujeme *heightfield_noise*. Ten vytvoří hornatý povrch velmi podobný reálnému světu viz obrázek číslo 10. U terénu je vždy největší důraz kladen na textury. V projektu bylo potřeba, aby vrcholky hor byly šedé či bílé a naopak nížiny zelené viz obrázek číslo 11.



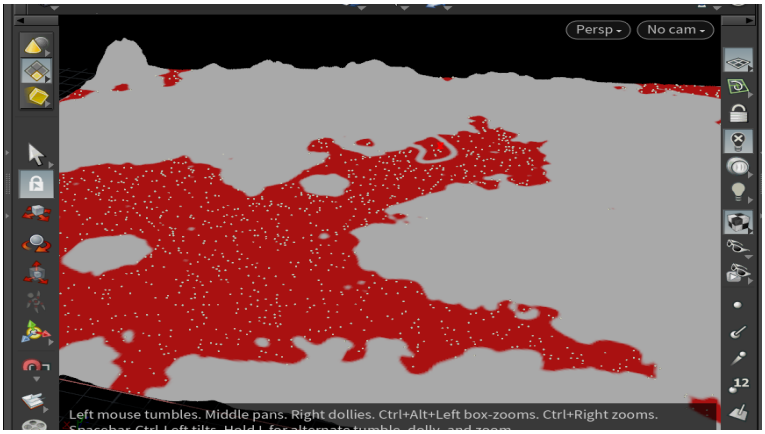
Obrázek 10: scatter node terén (29. 04. 2020)



Obrázek 11: hory s texturou (29. 04. 2020)

Textury se přidávají pomocí masek do vrstev *heightfield_layer*. Výsledkem je pak plně procedurální terén, u kterého se při změně povrchu automaticky upravují jeho textury.

Stromy jsou pomocí uzlu *heighfield_Scatter* náhodně rozmístěné po vrstvě představující nížiny viz obrázek číslo 12. Jejich počet, vzdálenost a velikost jsou atributem, což znamená, že je možné tyto hodnoty libovolně měnit. Stromy jsou vytvořeny v programu Speedtree a importovány do Unity.




Obrázek 12: rozmístění stromů (29. 04. 2020)

# 5. Grafická část projektu

Grafická stránka her, je v dnešní době velmi důležitá. V unity existuje plno nástrojů, jak výsledný vzhled her vylepšit. Mezi ně patří například plugin *PostProccesingStack* či HDRP. Unity podporuje mnoho tzv. *render pipelines*. V tomto projektu byla využita *High Definition Render Pipeline*. Oproti základní *Unity render pipeline* přináší mnohem realističtější a hezčí renderování grafiky. V projektu to můžeme poznat například u světel aut viz obrázek číslo 13, které vypadají, jako kdyby opravdu svítily.

Obrázek 13: brzdová světla (29. 04. 2020)

## 5.1 textury/materiály

Proto aby herní svět nebyl jednobarevný a bylo možné rozlišovat materiály jednotlivých objektů, musí mít všechny modely přidělený materiál a popřípadě i texturu. Materiál určuje barvu a vlastnosti odrazu světla od daného povrchu. Textury používáme, pokud má povrch objektu složitou strukturu. Herní materiály byly vytvořeny v programu Substance Painter, nebo přejaty z volně dostupných zdrojů. Veškeré textury ve hře byly také přejaty.

## 5.2 Modely

Modely můžeme v projektu rozdělit do dvou kategorií na generované a předem vymodelované. Vymodelovaných modelů je ve hře velmi málo. Mezi ně patří modely aut a okolních hor. Všechny modely aut ve hře jsou přejaté.

# 6. Audio

Na zvuk ve hře není kladen velký důraz. Zpracování zvuku by bylo možné při budoucím vývoji určitě vylepšit. Zvuk motoru auta funguje na principu změny výšky opakujícího se zvuku. Výška zvuku přímo závisí na rychlosti auta viz obrázek číslo 14 . Samotný zvuk motoru je přejatý z *Unity AssetStore*.

```
void Awake()
{
    foreach (sound s in sounds)
    {
        s.source = gameObject.AddComponent<AudioSource>();
        s.source.clip = s.clip;

        s.source.volume = s.volume;
        s.source.pitch = s.pitch;
    }
}
void Start()
{

    PlaySound("SongHlavniNabidka");

}

public void PlaySound(string name)
{
    sound s = Array.Find(sounds, sound => sound.name == name);
    s.source.Play();

}
```

Obrázek 14: implementace zvuku (29. 04. 2020)

(1)

# 7. Závěr

Výsledný projekt z velké části splňuje zadání. Byla vytvořena 3D závodní hra s plně procedurálním prostředím. V editoru lze pomocí atributů upravovat plno vlastností mapy jako například délku, tvar, barvu atd. Hráč si může vybrat z různých druhů aut a zazávodit si na čas.

Na závěr bych rád řekl, že tato práce mě velmi obohatila jak z hlediska programování, tak také jako člověka. Není vždy jednoduché si umět správně rozvrhnout čas a pracovat na projektu průběžně a se zápalem.

V budoucnu neplánuji na projektu dále pracovat, ale ani to nevylučuji. Hru by bylo možné dále vylepšovat. Herní zážitek by velmi vylepšilo přidání dalších map, aut a hlavně vytvoření AI protihráče či multiplayer.

# 8. Seznam obrázků

Obrázek 1: Atributy v Unity editoru (29. 04. 2020)……..8
Obrázek 2: Raycast paprsek (29. 04. 2020)……9
Obrázek 3: vzdálenost kol (29. 04. 2020) ……10
Obrázek 4: rádius otočení (29. 04. 2020) ……..10
Obrázek 5: akcelerační křivka (29. 04. 2020) ……..11
Obrázek 6: uzly v programu Houdini (29. 04. 2020) ……12
Obrázek 7: body v prostoru (29. 04. 2020) ……..12
Obrázek 8: algoritmus detekce zatáček (29. 04. 2020) ……13
Obrázek 9: vizualizace zatáček (29. 04. 2020) …… 14
Obrázek 10: scatter node terén (29. 04. 2020) …… 15
Obrázek 11: hory s texturou (29. 04. 2020) ……..15
Obrázek 12: rozmístění stromů (29. 04. 2020) ……..15
Obrázek 13: brzdová světla (29. 04. 2020) ……16
Obrázek 14: implementace zvuku (29. 04. 2020) …… 17

# 9. Zdroje

Při této práci jsem vycházel hlavně z oficiální dokumentace Unity a Houdini. Většina obrázků v dokumentaci je mnou vytvořená.

*Modely:*

1. **cgtrader.** www.cgtrader.com. *Free Car 3D models.* [Online] 2020. [Citace: 15. 2 2020.] https://free3d.com/3d-models/vehicles.

2. **Free3D.** free3d.com. *Free 3D vehicles Models.* [Online] 2020. [Citace: 16. 3 2020.] https://free3d.com/3d-models/vehicles.

*Obrázky:*

3. **G, tonie.** unsplash.com. *unsplash Photos for everyone.* [Online] [Citace: 28. 3 2020.] https://unsplash.com/photos/XP-J5O6A4Ko.

4. **Fzr, Zian.** unsplash.com. *unsplash Photosh for everyone.* [Online] [Citace: 28. 3 2020.] https://unsplash.com/photos/nYlXteQRUHo.

23. **auto.okhelp.** auto.okhelp.cz. *Auto - moto.* [Online] 2020. [Citace: 4. 29 2020.] https://auto.okhelp.cz/rady-navody/vypocet-nejmensi-prumer-otoceni-automobilu.php.

*Tutoriály:*

5. **Unity Technologies.** docs.unity3d.com. *Unity User Manual.* [Online] 22. 04 2020. [Citace: 13. 02 2020.] https://docs.unity3d.com/Manual/index.html.

6. **Brackleys.** www.youtube.com. *youtube.* [Online] 3. 1 2020. [Citace: 24. 3 2020.] https://www.youtube.com/channel/UCYbK_tjZ2OrIZFBvU6CCMiA.

7. **SideFX.** www.sidefx.com. *HOUDINI DOCUMENTATION.* [Online] 2020. [Citace: 25. 2 2020.] https://www.sidefx.com/docs/.

8. **Houdini Kitchen.** /www.houdinikitchen.net. *houdini kitchen.* [Online] 4. 23 2019. [Citace: 12. 3 2020.] https://www.houdinikitchen.net/.

9. **Hibberd, Aaron.** www.youtube.com. *youtube.* [Online] 20. 5 2017. [Citace: 27. 1 2020.] https://www.youtube.com/watch?v=BG7UMUWojik&t=71s.

10. —. www.youtube.com. *youtube.* [Online] 25. 9 2015. [Citace: 18. 3 2020.] https://www.youtube.com/watch?v=kN7Rx3uPBuU.

11. **AxiomaticUncertainty.** www.youtube.com. *youtube.* [Online] 26. 11 2018. [Citace: 19. 2 2020.] https://www.youtube.com/watch?v=8xdXJtu6nig&t=2s.

12. **Brush, Thomas.** www.youtube.com. *youtube.* [Online] 23. 1 2019. [Citace: 22. 4 2020.] https://www.youtube.com/watch?v=vqZjZ6yv1lA.

13. **SpeedTree.** www.youtube.com. *youtube.* [Online] SpeedTree2. 3 2020. [Citace: 17. 4 2012.] https://www.youtube.com/watch?v=sbmQhdQ_2VI.

14. —. www.youtube.com. *youtube.* [Online] 46. 5 2019. [Citace: 26. 2 2020.] https://www.youtube.com/watch?v=S_-HOFEhsg8.

15. **Adobe.** www.substance3d.com. *Substance by Adobe.* [Online] 2020. [Citace: 28. 4 2020.] https://www.substance3d.com/products/substance-painter/.

16. **Autor není uveden.** http://www.tokeru.com/. *JoyOfVex.* [Online] 4. 23 2020. [Citace: 28. 12 2019.] http://www.tokeru.com/cgwiki/index.php?title=JoyOfVex.

17. **McWhertor, Smykill, Jeff Helgason, David.** en.wikipedia.org. *wikipedia.* [Online] 25. 4 2020. [Citace: 15. 4 2020.] https://en.wikipedia.org/wiki/Unity_(game_engine).

18. **watchwellcast.** www.youtube.com. *youtube.* [Online] 13. 5 2013. [Citace: 29. 4 2020.] https://www.youtube.com/watch?v=IqqIV8x82Qc&bpctr=1588113589.

19. **Unity Technologies.** forum.unity.com. *Vehicle Physics Test - WIP*. [Online] 2020 . [Citace: 3. 17 2020.] https://forum.unity.com/threads/vehicle-physics-test-wip.506305/.

20. **Autor není uveden.** http://www.racer.nl/. *Pacejka's Magic Formula.* [Online] 17. 4 2014. [Citace: 7. 2 2020.] http://www.racer.nl/reference/pacejka.htm.

21. **Varaughe, Paulius-Liekis, SuperPingu, Aru3.** answers.unity.com. *what-is-the-math-behind-animationcurveevaluate.* [Online] 14. 9 2018. [Citace: 20. 3 2020.] https://answers.unity.com/questions/464782/t-is-the-math-behind-animationcurveevaluate.html.

*Zvuk:*

22. **Studio, slaczky - Skril.** assetstore.unity. *i6 German - FREE Engine Sound Pack.* [Online] [Citace: 23. 3 2020.] https://assetstore.unity.com/packages/audio/sound-fx/transportation/i6-german-free-engine-sound-pack-106037.

23. **lucvovan.** github.com. *MeasuredMaterialLibraryHDRP*. [Online] 2020. [Citace: 28. 3 2020.] https://github.com/Unity-Technologies/MeasuredMaterialLibraryHDRP.