Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Nástroj pro editaci hladkých ploch ve virtuální realitě

Plzeň 2018
Jakub Vašta
Místo této strany bude zadání práce.
Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 1. května 2018

Jakub Vaňta
Abstract

In this work a tool for creating and editing of parametric surfaces, which offers basic tools and functions for working with these surfaces, is designed. This software is designed to allow sharing of modeling space with other users. Other topics which are discussed are general and specific characteristics of parametric surfaces.

Abstrakt

V této práci je navržen nástroj pro modelování ve virtuální realitě pomocí parametrických ploch, který pro práci s nimi nabízí základní nástroje a funkce. Nástroj je navržen tak, aby umožňoval sdílet prostor více uživateli. Dále jsou zde zkoumány jak obecné vlastnosti parametrických ploch, tak vlastnosti charakteristické.
Obsah

1 Úvod 7
2 Požadavky na modelovací program 8
3 Virtualní realita 9
4 Vlastnosti parametrických ploch 11
  4.1 Spojitost ......................................................... 12
  4.2 Často požadované vlastnosti ................................. 12
5 Uvažované parametrické plochy 14
  5.1 Bézierova plocha .............................................. 14
    5.1.1 Napojování plátů ........................................ 14
    5.1.2 Výhody .................................................... 15
    5.1.3 Nevýhody .................................................. 15
  5.2 Bézierův trojúhelník ......................................... 15
    5.2.1 Napojování ............................................... 16
    5.2.2 Výhody .................................................... 16
    5.2.3 Nevýhody .................................................. 16
  5.3 B-spline plochy ................................................ 17
    5.3.1 Napojování ............................................... 17
    5.3.2 Výhody .................................................... 17
    5.3.3 Nevýhody .................................................. 18
  5.4 NURBS plochy .................................................. 18
    5.4.1 Výhody .................................................... 18
    5.4.2 Nevýhody .................................................. 18
6 Základní tvary 19
  6.1 Aproximace kruhu Bézierovou křivkou ....................... 19
  6.2 Rotační šablonování ......................................... 20
  6.3 Porovnání ...................................................... 20
7 Rozdělení plátu 21
  7.1 Čtvercové pláty ............................................... 21
    7.1.1 Bézierův plát ............................................ 21
    7.1.2 Coonsův plát ............................................. 22
  7.2 Trojúhelníkový plát .......................................... 22
8 Pohyb řídícím bodem společně s tečnou rovinou

9 Návrh programu
  9.1 Technologie ........................................ 26
  9.2 Plochy k modelování ................................ 26
    9.2.1 Realizace v Unity ............................ 27
  9.3 Ovládání ............................................ 28
  9.4 Konfigurační soubor ................................ 29
  9.5 Program jako celek .................................. 29
  9.6 Hlavní menu ......................................... 29
    9.6.1 Host ........................................... 30
    9.6.2 Klient .......................................... 31
    9.6.3 Informace a nápověda ........................... 31
  9.7 Prostor pro modelování ............................ 31
    9.7.1 Menu pro výběr nástrojů ....................... 32
  9.8 Řídící body ......................................... 33
  9.9 Napojení ............................................ 34
    9.9.1 Bézierův čtvercový plát - hladké napojení .... 34
    9.9.2 Bézierův trojúhelníkový plát - hladké napojení ... 35
  9.10 Uložení do standardního formátu trojúhelníkové sítě ... 36
  9.11 Uložení modelu pro opětovnou editaci ............ 36
    9.11.1 Výčtové typy .................................. 37
    9.11.2 Nahrání souboru .............................. 37
  9.12 Sdílení modelovacího prostoru .................... 38
    9.12.1 Realizace ..................................... 39

10 Testování ............................................. 41
  10.1 Testování použitelnosti ............................ 41
    10.1.1 Scénář ........................................ 41
    10.1.2 Výsledky testování ............................ 42
    10.1.3 Závěr z testování .............................. 43

11 Závěr ................................................. 44
1 Úvod

Virtuální realita přináší pro modelování pomocí parametrických ploch zajímavé možnosti, které tradiční nástroje neumožňují, ale zároveň přináší nové problémy, kterým je potřeba čelit. V této práci bude navrženo uživatelské rozhraní a modelovací nástroj, tak aby v něm byla práce co možná nejpříjemnější a využívalo vlastností VR. Také budou zkoumány obecné vlastnosti parametrických ploch a na základě vlastností specifických parametrických ploch bude rozhodnuto, které plochy je vhodné použít k naplnění požadavků na modelovací nástroj ve VR.
2 Požadavky na modelovací program


Plochy bude možné upravovat pomocí jejich řídících bodů. Bude umožněno manipulovat s pláty jako s celkem (otáčení, posunutí, změna velikosti), pohyb s tečnou rovinou (8), jejich napojování, rozpojování a dělení (7). Všechny tyto vlastnosti budou společné všem typům parametrických ploch, které budou vybrány pro modelování v navrhovaném nástroji.

Dále bude možné výsledný model uložit do standartního formátu trojúhelníkové sítě (9.10), tak aby šel prohlížet i v jiných programech podporujících daný formát. Druhá možnost uložení (9.11) bude sloužit pro opětovné načtení rozpracovaného modelu a další editaci v navrhovaném nástroji pro modelování. Modelovací nástroj bude také umožňovat sdílet modelovací prostor více uživatelů (9.12).
3 Virtualní realita

Virtuální realita (dále označovaná VR) může být definována jako uměle vytvořené digitální prostředí, které můžeme vnímat, a s ním interagovat, jako by bylo skutečné. Je možné rozdělit druhy VR do dvou skupin:

1. **Augmentovaná realita (AR)** je rozšíření reálného prostředí o digitální prvky

2. **Virtuální prostředí** je zcela uměle (digitálně) vytvořené prostředí

V této práci se budeme zabývat virtuálním prostředím a proto je nutno význam slova VR v dalším textu interpretovat právě v tomto smyslu. K vytvoření VR se používají dva displeje, pro každé oko jeden, které jsou ve formě brýlí (Head-Mounted Display) nasazené na hlavu. Iluze 3D prostoru vzniká díky stereoskopii, která funguje na principu zobrazování scény pro každé oko pod mírně jiným úhlem. Stereoskopie spolu s ostatními metodami (stínování, paralaxa) umožňují vytvořit velmi věrohodné virtuální prostředí. Další důležitou součástí příslušenství pro VR jsou ovladače a sledovací zařízení, která sledují pohyb HDM a ovladačů v prostoru [4].

**VR příslušenství** V dnešní době je mnoho příslušenství pro VR, z kterých si lze vybrat. Tato zařízení se liší jak v technických parametrech a vybavení, tak i v ceně. V současnosti (2018) jsou dostupná řešení např. Oculus Rift, HTC Vive, Sony PlayStation VR, Google Daydream View a Samsung Gear VR. Zajímavým projektem je také zařízení sledující pohyb rukou (včetně jednotlivých prstů) Leap Motion, které je v případě nepřítomnosti ovladačů, klasicky dodávaných k VR, zajímavou náhradou. Leap Motion dokonce v mnoha ohledech, například škála a detekce gest, klasické ovladače překoná.

Pro navrhovaný modelovací program bude zapotřebí, aby příslušenství obsahovalo dva ovladače. Na základě těchto požadavků a dostupnosti příslušenství pro VR bude modelovací program vyvíjen primárně pro Oculus Rift s přidanou podporou pro HTC Vive (jedná se v současnosti o dvě nejrozšířenější VR řešení pro PC).

**Práce ve VR** s sebou nese mnoho výhod, mezi hlavní patří snadný pohyb v prostoru a ovládání intuitivními pohyby, popř. gesty. Velmi snadná je také manipulace s objekty pomocí VR ovladačů, jelikož je sledována nejen jejich
poloha, ale i rotace. Bohužel je zde také několik nevýhod, kde jednou z hlavních je, že se uživateli může při určitých pohybech (nebo po určité době) udělat nevolno. Nevolnost je většinou vyvolána ve chvílích, kdy pohyb kamery neodpovídá reálnému fyzickému pohybu uživatele. Jak lze snadno nalézt, nevolnost může tedy vyvolat jak situace, kdy se kamera pohybuje, ale uživatel setrvává v klidu, tak i situace, kdy kamera setrvává v klidu, ale uživatel se pohybuje. Do druhého případu spadá i dostatečně zpoždění pohybu kamery (přibližně víc jak 20 milisekund) [12][13], takzvaný lag. Čas udávající čas potřebný na promítnutí pohybu uživatele na obrazovku se nazývá MTP latency. Dalším faktorem, který by měl být zohledněn pro pohodlnou práci ve VR, která nezpůsobuje nevolnost, je FPS, které by mělo být alespoň 90 [14].

Další problém nastává při zadávání textu. Samozřejmě je mnoho řešení, jak tento problém vyřešit (např. virtuální klávesnice), ale nedosahují komfortu jaký poskytuje klasická klávesnice.
4 Vlastnosti parametrických ploch

Jelikož navrhovaný modelovací program používá parametrické plochy, tak se v této sekci budeme zabývat pojmy a vlastnostmi, které jsou pro všechny parametrické plochy shodné.

**Bodovou rovnici parametrické plochy** $Q(u, v)$ dvou parametrů $u$ a $v$ budeme rozumět funkci

$$Q(u, v) = [x(u, v), y(u, v), z(u, v)],$$

kde $x(u, v), y(u, v)$ a $z(u, v)$ jsou funkce dvou parametrů $u, v \in (0, 1)$. Bod $Q$ o souřadnicích $[x, y, z]$ v trojrozměrném kartézském prostoru má souřadnice $[u, v]$ v prostoru parametrickém. Funkce $x(u, v), y(u, v)$ a $z(u, v)$ jsou obvykle polynomiální, s ohledem na výhodné vlastnosti při modelování a navazování [1].

Plochy jsou zadané řídícími body, kterým může být přiřazena určitá váha (racionální), a bázovými funkcemi. Bázové funkce jsou nejčastěji polynomy, pro jejich dobré vlastnosti (např. jako je jejich snadné vyčíslení a diferencovatelnost). Nejčastěji se používají, díky jejich vhodným vlastnostem, polynomy třetího stupně [1].

Parametrické plochy jsou často zadávány v maticové formě, jak demonstruje následující rovnice.

$$Q(u, v) = \vec{u}^T M_f^T P M_f \vec{v},$$

kde $M_f$ je matice formy, $P$ je matice řídících bodů, $\vec{u} = (u^3, u^2, u, 1)$ a obdobně $\vec{v} = (v^3, v^2, v, 1)$

**Tečný vektor** $\vec{q}_u(u, v)$ ve směru parametru $u$ k ploše $Q(u, v)$ a tečný vektor $\vec{q}_v(u, v)$ ve směru parametru $v$ jsou určeny vztahy [1]:

$$\vec{q}_u(u, v) = \frac{\partial Q(u, v)}{\partial u},$$

$$\vec{q}_v(u, v) = \frac{\partial Q(u, v)}{\partial v}$$
Normála $\vec{n}$ k ploše v bodě $Q$ se vypočte dle následujícího vzorce [1]:

$$\vec{n} = \frac{\vec{q}_u \times \vec{q}_v}{|\vec{q}_u \times \vec{q}_v|}$$

Hlavní křivka plochy ve směru parametru $u$ je každá křivka určená rovnicí $Q(u, k)$ pevného parametru $v = k$ a proměnného parametru $u$ (analogicky hlavní křivka plochy ve směru $v$) [1].

Rohy plochy jsou body $Q(0, 0)$, $Q(0, 1)$, $Q(1, 0)$ a $Q(1, 1)$. Strany plochy $Q(u, v)$ jsou hlavní křivky ve směru $u$ pro hodnoty $v = 0$ a $v = 1$ a ve směru $v$ pro hodnoty $u = 0$ a $u = 1$. Všechny strany plochy dohromady tvoří její okraj [1].

Skládání Plochy stejného typu se dají skládat do větších částí (částím, ze kterých je skládáme, říkáme pláty a samotnému procesu plátování). Inverzní operací, kterou můžeme provádět, je dělení plochy na více plátů [1].

### 4.1 Spojitost

- Dva pláty mají napojení $C^0$, mají-li společnou stranu, která je křivkou třídy alespoň $C^0$ [1][2].

- Dva pláty mají spojité napojení $C^1$, mají-li společnou stranu ($C^0$ napojení) a jsou-li shodné příčné parciální derivace ve vešech bodech společné strany prvního i druhého plátu [1][2].

- Dva pláty mají spojité napojení $C^1$, mají-li společnou stranu ($C^0$ napojení) a jsou-li příčné parciální derivace ve vešech bodech společné strany prvního i druhého plátu lineárně závislé s koeficientem $k > 0$ [1][2]. Toto napojení je méně omezujucí než-li $C^1$ napojení.

### 4.2 Často požadované vlastnosti

- Invariance k lineárním transformacím, která zaručuje, že transformace síť řídících bodů a následné generování plochy má stejný výsledek, jako transformace každého bodu z vygenerované plochy [1].

- Vlastnost konvexní obálky (plocha leží v konvexní obálce tvořené jejími řídícími body)[1].
• Lokalita změn - změnou polohy (u racionálních ploch i váhy) řídícího bodu se mění jen část plochy, nikoliv plocha celá [1].

• Plocha může procházet krajními body sítě řídících bodů [1].
5 Uvažované parametrické plochy


5.1 Bézierova plocha

Bézierovu plochu definujeme předpisem

\[ Q(u,v) = B_m(u)PB^T_n(v), u \in \langle 0,1 \rangle, v \in \langle 0,1 \rangle, \]

kde

\[ B_k(s) = [B^k_0(s), B^k_1(s), ..., B^k_k(s)], \]

\[ B^k_k(s) = [B^k_0(s), B^k_1(s), ..., B^k_k(s)], \]

\[ B_k(s) = \text{vektorová funkce, která parametru } s \text{ přiřazuje vektor, jehož složkami jsou hodnoty jednotlivých Bernsteinových polynomů (11.1) stupně } k, \]

\[ P \text{ je matice řídících bodů. V explicitním tvaru je rovnici Bézierovy plochy možné napsat ve tvaru} \]

\[ Q(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} P_{i,j} B^m_i(u)B^n_j(v) \]


Bézierova plocha prochází rohovými body sítě a okrajové křivky plochy jsou Bézierovými křivkami pro okraj sítě [1][2].

5.1.1 Napojování plátů

Mějme dva Bézierovy pláty \( Q^{(1)}(u,v) \) a \( Q^{(2)}(u,v) \). První z nich je určen sítí řídících bodů \( P_{i,j}^{(1)}, i = 0,..,s, j = 0,.. \) a druhý je určen jako \( P_{i,j}^{(2)}, i = 0,..,t, j = 0,.. \), tzn. počet bodů ve směru \( \vec{v} \) je stejný pro oba pláty a je roven \( m \). Pláty budeme navazovat ve směru \( \vec{u} \) a požadujeme, aby jejich stupeň byl alespoň tři.

Pláty \( Q^{(1)} \) a \( Q^{(2)} \) jsou \( C^0 \) spojitě navážány ve směru \( \vec{u} \), pokud je totožná jejich (alespoň \( C^0 \) spojitá) strana, tj. \( Q^{(1)}(1,v) = Q^{(2)}(0,v) \). Této spojitosti
Bézierových plátů docílíme ztotožnění řídících bodů, které určují příslušnou stranu.

\[ P_{s,j}^{(1)} = p_{0,j}^{(2)}, j = 0, \ldots, m \]

Pláty \( Q^{(1)} \) a \( Q^{(2)} \) jsou \( C^1 \) spojitě navážány ve směru \( \vec{u} \), pokud je jejich společná hraná \( G^1 \) spojitá a jsou-li identické příčné tečné vektory ve směru \( \vec{u} \) podél této strany. \( G^1 \) napojení plátů docílíme shodou \( G^1 \) spojité strany a následujícím vztahem pro body řídících síťí obou plátů

\[ s * (P_{s,j}^{(1)} - P_{s-1,j}^{(1)}) = t * (P_{1,j}^{(2)} - P_{0,j}^{(2)}), j = 0, \ldots, m \]

Pozn.: Řídící body \( P_{s-1,j}^{(1)} \) a \( P_{1,j}^{(2)} \) leží na přímce. Pokud jsou středově souměrné podle \( P_{s,j}^{(1)} = P_{0,j}^{(2)} \) (tzn. \( s = t \)), tak se jedná o \( C^1 \) spojitost [1][2].

### 5.1.2 Výhody

Tyto plochy nabízí poměrně intuitivní modelování ploch, které mimo jiné vyplývá ze skutečnosti, že plocha prochází svými krajními body. Dají se snadno dělit na n menších plátů, při zachování tvaru původního plátu. Existence trojúhelníkových Bézierových ploch (velmi podobné vlastnosti a přístup k modelování) je další výhodou, která by neměla být opomenuta.

### 5.1.3 Nevýhody


### 5.2 Bézierův trojúhelník

Obecný Bézierův trojúhelník má tvar

\[ Q(s, t, u) = (\alpha s + \beta t + \omega u)^n = \sum_{i+j+k=n}^{i,j,k \geq 0} \binom{n}{i,j,k} s^i t^j u^k \alpha^i \beta^j \gamma^k, \quad (5.3) \]
kde $\alpha, \beta, \gamma$ (a jejich kombinace) jsou řídící body a $s, t, u$ jsou barycentrické souřadnice, pro které platí $s + t + u = 1$.

Krajní křivky Bézierova trojúhelníku jsou Bézierovy křivky [1][2].

5.2.1 Napojování

Spojitosti $C^0$ je dosaženo stejně jako v případě Bézierových (čtvercových) ploch.

Bézierovy trojúhelníky jsou $G^1$ hladce napojeny pokud jsou trojúhelníky, které sdílejí body na hraně napojení (viz. Obr. 5.1) v jedné rovině [3].

5.2.2 Výhody

Tyto plochy nabízí poměrně intuitivní modelování, což mimo jiné vyplývá ze skutečnosti, že plocha prochází svými krajními body. Další výhodou je snadné dělení na dvě poloviny, při zachování tvaru původního plátu. Také samotný trojúhelníkový tvar, který dává zajímavé modelovací možnosti, je velmi výhodný. V neposlední řadě je výhoda, že existují čtvercové Bézierovy plochy (velmi podobné vlastnosti a přístup k modelování).

5.2.3 Nevýhody

Při změně jediného řídícího bodu změní svůj tvar celá plocha. Složité hladké navazování více ploch a s ním spojená omezení (viz. 5.1.3). Další nevýhodou je, že pomocí Bézierovy plochy nelze přesně modelovat kuželosečky (6.1). Tyto tvary lze pouze aproximovat.
5.3 B-spline plochy

Obecnou B-spline plochu definujeme předpisem

\[ Q(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} P_{i,j} N_i^m(u) N_j^n(v) \]  \hspace{1cm} (5.4)

Tato plocha je určena:
- řídící síť \((m + 1)(n + 1)\) bodů
- stupněm \(k\) polynomů baze pro parametr \(u\) a stupněm \(l\) polynomů baze \((11.2)\) pro parametr \(v\)
- vektory parametrizace pro parametr \(u\) a \(v\)[1][2]

5.3.1 Napojování

Pokud plochy neprochází svými krajními body [1][2], tak nemá smysl hovořit o \(C^0\) napojení. Pokud ano, tak je opět velmi snadné napojení provést ztotožněním hran, přes které mají být napojeny.

Pozn.: Opakováním bodů lze B-Spline křivku (reps. plochu) protáhnout do krajních bodů[1][2].

B-spline plochy se velice snadno hladce (\(C^1\)) napojují. B-spline plocha n-tého stupně zaručují \(C^{n-1}\) spojitost ve všech svých bodech a není nutné omezovat některé jejich řídící body vnějším podmínkami jako v případě Bézierových ploch.

Navazující B-spline plát se definuje použitím \(m\) x \((n - 1)\) bodů plátu předchozího a přidáním pouze \(m\) dalších bodů.

5.3.2 Výhody

Změnou jediného řídícího bodu měníme vždy tvar pouze části B-spline plochy. Snadné hladké napojování (\(C^1\)). Je snadné tyto plochy dělit na \(n\) menších plátů, při zachování tvaru původního plátu.
5.3.3 Nevýhody

Modelování s B-spline plochami není tak intuitivní jako s Bézierovými (toto platí především v případě, že plochy neprochází svými krajními body). Stále není možné přesně reprezentovat kuželosečky (bude umožněno až při racionalizaci a neuniformním rozdělení), ale pouze aproximovat [11].

5.4 NURBS plochy

Neuniformní racionální B-spline (NURBS) plochy jsou rozšířením B-spline ploch, kde každý řídící bod má navíc svoji váhu a uzlové vektory nemají uniformní rozdělení. Řídící body si můžeme představit jako body zadané v homogeních souřadnicích, kde hodnota \( \omega \) udavá váhu. Podobným přístupem lze samozřejmě získat i racionální Bézierovy plochy [2].

**Obecnou NURBS plochu** definujeme předpisem

\[
Q(u, v) = \frac{\sum_{i=0}^{n} \sum_{j=0}^{m} \omega_{i,j} P_{i,j} N_{i}^{n}(u) N_{j}^{m}(v)}{\sum_{i=0}^{n} \sum_{j=0}^{m} \omega_{i,j} N_{i}^{n}(u) N_{j}^{m}(v)},
\]  

kde \( \omega_{i,j} \) jsou váhy bodů (homogení souřadnice) \( P_{i,j} \) řídící sítě \( P \), \( n \) a \( m \) je počet řídících bodů, \( p \) a \( q \) jsou stupně polynomů a konečně \( N_{i,p}(u), N_{j,q}(v) \) jsou normalizované B-spline bázové funkce. NURBS plocha je dále určena dvěma uzlovými vektory - vektorem \( U \) délky \( n+p+1 \) a \( V \) délky \( m+q+1 \), kde \( n \), resp. \( m \) je počet řídících bodů ve směru \( u \), reps. \( v \) a \( p \), reps. \( q \) je stupeň plochy ve směru \( u \), reps \( v \). Uzlové vektory ovlivňují průběhy jednotlivých bázových funkcí (viz Přílohu) a interval jejich vlivu.

Váhy \( \omega_{i,j} \) určují vliv bodu na plochu. Pokud je váha rovna nule, nemá bod na plochu žádný vliv, s rostoucí váhou se plocha k bodu přimyká a pro hodnotu \( \omega_{i,j} \to \infty \) plocha bodem prochází (B-Spline, Bézierovy plochy mají \( \omega = 1 \), lze tedy nahlédnout, že se jedná pouze o zobecnění) [1].

5.4.1 Výhody

Při změně jednoho řídícího bodu se mění tvar plochy pouze lokálně a toto chování lze dále upravovat pomocí váhy bodů. NURBS plochu umožňují přesně reprezentovat kuželosečky [1].

5.4.2 Nevýhody

Modelování s NURBS plochami není tak intuitivní jako s Bézierovými.
6 Základní tvary


6.1 Aproximace kruhu Bézierovou křivkou

Kruh je možné aproximovat pomocí $n$ Bézierových křivek. V tomto případě pro optimální aproximaci, ve smyslu že prostřední a krajní body Bézierovy křivky leží na kružnici, platí, že krajního a k němu příslušejícího řídícího bodu je $\frac{4}{3}\tan\left(\frac{\pi}{2n}\right)$, kde $n$ je počet segmentů kružnice (počet Bézierových křivek, které tvoří kružnici) [5][6].

Obrázek 6.1: Aproximace kružnice
6.2 Rotační šablonování

Jak již z názvu vychází, tak rotační těleso (základní tvar koule, kužel a válec) získáme rotačí křivky kolem osy (v našem případě osy z), tato rotovaná (profilová) křivka může být libovolného tvaru. Při dalším postupu budeme vycházet z použití profilové křivky typu NURBS \( P(v) \) stupně \( k \) v rovině \( xz \) zadanou body \( P_i \) a uzlovým vektorem \( \vec{V} \).

Profilovou křivku podrobíme otáčení kolem osy \( z \). Řídicí síť výsledné rotační plochy získáme tak, že každý řídicí bod \( P_i = [X_i, Y_i, Z_i, W_i] \) profilové křivky okopírujeme v rovině \( xy \) ve výšce \( Z_i \) sedmkrát tak, aby vzniklé řídicí body tvořily v rovině \( xy \) řídicí polygon kružnice.

Váhy \( W_{i,j} \) nově vzniklých bodů se vypočítají z váhy \( W_i \) bodu \( T_i \) takto

\[
W_{i,j} = \{W_i, W_i/2, w_i/2, W_i, w_i/2, w_i/2, W_i\}; j = 0, ..., 6.
\]

Rovnice rotační plochy má tvar:

\[
Q(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{6} P_{i,j} R_{i,3}(u)R_{j,k}(v) \quad (6.1)
\]

s uzlovými vektory \( U = \{0, 0, 0, 1/4, 1/2, 1/2, 3/4, 1, 1, 1\} \) (kružnice) a \( V \) (určen profilovou křivkou) [1].

6.3 Porovnání

Mohlo by se zdát, že vyjádřit kružnici přesně pomocí NURBS křivek (repos. vyjádřit rotační těleso pomocí NURBS ploch) je oproti aproximaci velmi výhodné, avšak je potřeba si uvědomit, že samotný hladký povrch námi uvažovaných ploch je v programu reprezentován trojúhelníkovou síťí. Jinými slovy, hladký povrch ploch také pouze aproximujeme. Dále by bylo vhodné udržovat v modelovacím programu plochy pouze určitého stupně (a se stejnými uzlovými vektory v případě B-spline), abychom je mezi sebou mohli snadno napojovat. Při rotačním šablonování by však vznikaly různé NURBS plochy. Z těchto poznatků vyplývá, že dobrá aproximace křivky (povrchu) je pro účely tohoto modelovacího nástroje zcela dostatečující a nezavádí do programu mnoho druhů ploch s různými stupně.
7 Rozdělení plátu

Rozdělením plátu je v tomto případě myšleno vytvoření daného počtu menších plátů stejného typu, jako je plát který dělíme, tak aby byl zachován původní tvar. Díky této operaci se mimo jiné zvýší možnosti editace daného plátu.

7.1 Čtvercové pláty

Čtvercové (obecně obdélníkové) pláty budou děleny na čtyři menší pláty. Křivky ve směru $\vec{u}$ (resp. $\vec{v}$) budou rozděleny na polovinu a následně se rozdělí nově vzniklé křivky ve směru $\vec{v}$ (resp. $\vec{u}$), také na polovinu. Tímto postupem se získají čtyři nové čtvercové pláty, které zachovávají tvar plátu původního. Nyní se zaměříme na dva druhy čtvercových parametrických ploch, u kterých bude uveden podrobný postup.

7.1.1 Bézierův plát

K rozdělení Bézierovy křivky lze s výhodou využít algoritmu De Casteljau [1][2].

$P_0^0 = P_i, i = 0, ..., n$

$P_i^j = P_{i-1}^{j-1}(1 - t_0) + P_{i+1}^{j-1}t_0, i = 0, ..., n - j, j = 1, ..., n$

Během několika málo iterací je tento algoritmus schopen nalézt prostřední bod na křivce (neboli bod pro parametr $u = 0.5$ resp. $v = 0.5$). Navíc tímto postupem se získají řídící body dvou nově vzniklých křivek a to následovně:

První křivka: $P_0^0, P_1^1, P_2^2, P_3^3$
Druhá křivka: $P_3^3, P_2^3, P_1^3, P_0^3$

Tímto postupem bychom mohli Bézierovu křivku rozdělit i v jiném poměru. Lze snadno nahlédnout, že pokud jsme schopní dělit Bézierovy křivky, tak jsme také schopní dělit Bézierovy (čtvercové) plochy.
7.1.2 Coonsův plát

Pro dělení Coonsova plátu budeme postupovat jiným způsobem. Vyjdeme z následujících rovnic, které platí pro Coonsovu kubickou křivku.

\[
Q(0) = \frac{P_0 + 4P_1 + P_2}{6} \\
Q(1) = \frac{P_1 + 4P_2 + P_3}{6} \\
Q(0)dt = \frac{P_2 - P_0}{2} \\
Q(1)dt = \frac{P_3 - P_1}{2}
\]

Je tedy zřejmé, že je potřeba najít počáteční a koncový bod křivky i s jejich tečnými vektory (tyto informace je možné získat z původní křivky). Následně je možné dopočítat řídící body nově vzniklé křivky. Lze opět snadno nahlédnout, že pokud jsme schopni dělit Coonsovy křivky, tak jsme také schopni dělit Coonsovy plochy.

7.2 Trojúhelníkový plát

Je několik možností, jak lze trojúhelníkový plát rozdělit [3]. V tomto případě bude Bézierův trojúhelníkový plát dělen na dvě poloviny podle dané hrany. Tento postup může sice vést k vytvoření nových trojúhelníkových plátů s velmi ostrými úhly (tomtu problému se může uživatel vyhnout vhodným dělením). Na druhou stranu je tento způsob dělení nenáročný na výpočet a výpočetní výkon.
Následující rovnici lze využít pro výpočet poloviny Bézierova trojúhelníku s prvním rohem v $\alpha^3$, druhým v polovině Bézierovy křivky mezi $\alpha^3$ a $\beta^3$ a třetím v $\gamma^3$.

$$
\begin{bmatrix}
\alpha^3' \\
\alpha^2\beta' \\
\alpha\beta^2 \\
\beta^3' \\
\alpha^2\gamma' \\
\alpha\beta\gamma' \\
\beta^2\gamma' \\
\alpha\gamma^2 \\
\beta\gamma^2 \\
\gamma^3
\end{bmatrix} =
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{4} & \frac{3}{4} & \frac{1}{4} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{8} & \frac{7}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\times
\begin{bmatrix}
\alpha^3 \\
\alpha^2\beta \\
\alpha\beta^2 \\
\beta^3 \\
\alpha^2\gamma \\
\alpha\beta\gamma \\
\beta^2\gamma \\
\alpha\gamma^2 \\
\beta\gamma^2 \\
\gamma^3
\end{bmatrix}$$

Tímto postupem je možné získat dva nové Bézierovy trojúhelníkové pláty, které zachovávají tvar původního (děleného) trojúhelníkového plátu [9].

**Na závěr** by mělo být zmíněno, že v případě racionalního rošíření ploch (tzn. i křivek), mohou být stále tyto algoritmy použity. Pouze pro řídící body budou použity homogení souřadnice, kde hodnota $\omega$ určuje váhu daného bodu.
U rohových bodů plochy bychom chtěli umožnit pohybování spolu s tečnou rovinou. Tento požadavek může být splněn poměrně jednoduše, stačí při transformaci rohového bodu aplikovat transformace i na body, které určují tečnou rovinu plochy v tomto rohovém bodě.

Zůstává tedy pouze otázkou, které body společně s námi zvoleným rohovým bodem tuto tečnou rovinu určují. Pomocí vztahů platících pro parametrické plochy tyto body nalezneme pro námí uvažované plochy. Výpočet je uveden v příloze.

Výsledky pro bod $P_{0,0}$ jsou uvedeny na následujících obrázcích.

Obrázek 8.1: Bézierova plocha
V případě Coonsovy plochy je funkce pohybu s tečnou rovinou experimentální, jelikož plocha neprochází svými rohovými body a proto se nedá zcela hovořit o pohybu s tečnou rovinou rohového bodu.

Obrázek 8.3: Bézierův trojúhelníkový plát
9 Návrh programu

Program musí být navržen tak, aby splňoval požadovky uvedené v kapitole (2) a byl snadno ovladatelný i pro uživatele, kteří nemají s VR předchozí zkušenosti. Program musí být bezpodmínečně navržen tak, aby se běžnému uživateli při standardní práci s programem nedělalo špatně a byl schopný v programu pracovat po delší dobu.

9.1 Technologie

Jako nástroj pro tvorbu programu pro modelování parametrických ploch je zvoleno Unity, konkrétně Unity 2017.4.0f1 (64-bit). Tento nástroj se využívá především pro tvorbu her, ale poskytuje nástroje vhodné a potřebné k realizaci tohoto projektu (například i nástroje pro sdílení po síti) a je pro osobní využití zcela zdarma. V Unity je možné používat jako skriprovací jazyk C# nebo JavaScript. Pro tento projekt je využíván C#. Vývojáři Oculus Rift i HTC Vive poskytují předpřipravená řešení jak integrovat jejich VR příslušenství do projektů tvořených v Unity. Je použit balík SteamVR, kvůli jeho schopnosti podporovat jak Oculus Rift, tak i HTC Vive, který zavádí sadu abstrakcí nad ovladači. Samozřejmě by bylo možné si vybrat i jiné nástroje, jménem např. Unreal Engine, s kterými by bylo možné dosáhnout podobného výsledku. Dále je potřeba vytvářet 3D modely objektů, které budou použity v programu. K tomuto účelu jsem zvolil modelovací nástroj Blender, konkrétně 2.79, který je zdarma dostupný a má modelovací možnosti na profesionální úrovni.

9.2 Plochy k modelování

Pro modelování byly zvoleny Bézierovy plochy. Tyto plochy byly zvoleny z důvodů uživatelské přívětivosti, která vyplývá především z faktu, že plochy procházejí svými krajními body a lze velmi rychle pochopit reakci plochy (jejího tvaru) na pohyb řídícími body. Další výhodou jsou dva tvary, které mohou být uživateli nabídnuty, čtvercové a trojúhelníkové, při téměř shodných modelovacích vlastnostech. V neposlední řadě také zůstává skutečnost, že se lze běžně s Bézierovými křivkami setkat v mnoha současných grafických nástrojích. Nevýhodou zůstává obtížnější hladké napojování křivek než u jiných typů ploch. Do modelovacího nástroje byl také přidán Coonsův
plát, jako zástupce jiného druhu plátu, který se snadno hladce napojuje, ale
neprochází svými krajinmi body. Každá plocha bude mít specifickou barvu
jak řídících bodů, tak i vlastního povrchu, pro lepší vzájemné rozlišení.

9.2.1 Realizace v Unity

Plochy jsou v modelovacím nástroji reprezentované jako GameObject, které
jsou rodičovským objektem pro řídící body dané plochy. Každá plocha má
komponentu Mesh, která reprezentuje povrch plochy a skript, který definuje
její chování (samozřejmě obsahuje i další strandartní komponenty Unity).
Skript definující chování plochy musí splňovat rozhraní IParametricSurface
(Obr. 9.1). Do programu tedy lze velmi snadno přidat další plochy, aniž by
byla ovlivněna současná funkcionalita.

```
<<Interface>>
IParametricSurface
+ SID: int
+ Create(Vector3[]): void
+ Create(IControlPoint[]): void
+ Remove(): void
+ Split(EdgesType): Vector3[]
+ Join(IParametricSurface, EdgesType, EdgesType, ConnectionType): bool
+ Free(IParametricSurface, EdgesType, EdgesType): bool
+ Translate(Vector3): void
+ Rotate(Vector3): void
+ Scale(Vector3): void
+ TangentMove(IControlPoint): IControlPoint
+ GetEdge(EdgesType): IControlPoint
```

Obrázek 9.1: Unity surface shader

Parametrické plochy jsou vykreslovány jako trojúhelníkové sítě pomocí
surface shaderu, který nabízí Unity (Obr. 9.2). Tyto výpočty jsou prováděny
na GPU, kvůli rychlosti (GPU je pro takový typ výpočtů optimalizovaný).
Do jisté úrovni jemnosti trojúhelníkové sítě by bylo možné využívat i CPU,
ale při větší jemnosti by byl vliv na výkon již znatelný.
Unity surface shader (Obr. 9.2) umožňuje pomocí modulu Vertex modifier měnit pozici vrcholů tvořících trojúhelníkovou síť, která tvoří parametrický povrch. Každému bodu je při vytvoření povrchu přiřazena \((u,v)\) souřadnice, která je dosazena ve Vertex modifier modulu do rovnice parametrického povrchu, a tím je získána pozice daného vrcholu (vzhledem k řídícím bodům povrchu). Dále je možné upravit povrch parametrické plochy (stínování) v dalším modulu, konkrétní Surface function (Obr. 9.2)[10].

9.3 Ovládání

Ovládací prvky VR příslušenství standardně obsahuje dva ovladače. V modelovacím programu bude jeden ovladač sloužit k navigaci a vybírání položek v menu (dále nazývan ovladač nástrojů). Druhý ovladač bude sloužit k samotnému modelování (dále nazýván aktivní ovladač) a pohybu po prostoru. Veškerá interakce v rámci modelovacího nástroje bude probíhat pomocí těchto dvou ovladačů a k ovládání programu budou nezbytně nutné. Program nebude umožňovat gesta, aby se unsnadnila možnost používat jiné VR ovladače.

Pohyb Jak již bylo zmíněno, pohyb ve VR může působit nepříjemné pocity (někdy dokonce i nevolnost), pokud se virtuální scéna hýbe, ale ve skutečnosti uživatel se pohybuje v klidu a naopak. Pro uživatele je mnohem příjemnější změnit pozici skokově. Z těchto důvodů bude v programu zavedena možnost teleportace na zvolené místo. Místo k teleportaci zvolíme pomocí virtuálního
laserového ukazovátka. Pro skokový pohyb existuje i obyčejnější důvod a tím je u většiny VR příslušenství připojení HMD pomocí kabelu omezené délky k počítači. Zbytek pohybů bude zajišťovat sledovací systém VR (přirozený pohyb).

V návrhu programu se budeme snažit vyhnout potřebě po zapnutí programu zadávat textový vstup pomocí klasické klávesnice, protože by se rázně snížila použitelnost a pohodlnost (uživatel by si musel sundat HMD, napsat text a opět si HMD nasadit).

9.4 Konfigurační soubor

Aplikace bude obsahovat konfigurační soubor, v kterém budou nastaveny všechny potřebné informace pro běh modelovacího nástroje. Konfigurační soubor je zaveden, aby nebylo nutné zdlouhavě zadávat textový vstup ve VR. Bohužel, konfigurační soubor znepříjemní práci uživatelům, kteří nejsou zběhlí v práci s počítačem, proto je vhodné vytvořit klasický desktopový program, který je uživatelsky přijatelným způsobem. Bude možné nastavit, který ovladač (pravý, levý) je nástrojový a který aktivní, cesty ke složkám k ukládání modelu, které bude program zároveň procházet v případě, že bude požadavek z programu pro modelování ve VR nahrát již existující modelovací prostor. Dále zde bude také možné nastavit údaje potřebné k vytvoření sdíleného modelovacího prostoru a výchozí údaje o modelovacím prostoru, ke kterému se bude možné připojit.

9.5 Program jako celek

Program je rozdělen do modulů, které plní specifické úkoly, tak aby bylo možné jednotlivé moduly mezi sebou zaměňovat v případě potřeby. Všechny části zastřešuje v modelovacím prostoru modul Správce Prostoru (SpaceManager), který spravuje všechny vytvořené plochy v modelovacím prostoru.

9.6 Hlavní menu

Po spuštění modelovacího programu se uživatel ocitne v hlavní místnosti, která obsahuje nápovědu a hlavní menu. Toto menu bude obsahovat v základním tvaru tři prvky: klient, host a konec (v tomto pořadí). Možnost konec ukončí program.
Hlavní menu bude uspořádáno do kruhu kolem uživatele (Obr. 9.3). Tento typ menu je v aplikacích využívajících VR velmi oblibený a využívá vlastnosti lidského vnímání nazývané periferní vidění. Jedna z možností, jak se v menu navigovat, bude samozřejmě otáčením hlavou a výběrem položky menu pomocí ukazovátka. Abychom se vyhnuli nutnosti příliš otáčet hlavu, což by nemuselo být příjemné, tak bude umožněno menu otáčet pomocí šipek v daném směru (nejsou zakresleny v ilustrativním obrázku 9.3) a poté nástrojovým ovladačem vybrat aktivní položku menu.

9.6.1 Host

Bude vytvořen klient a zároveň server (9.12), podle nastavení v konfiguračním souboru (9.4), ke kterému se budou moci připojit ostatní uživatelé. Pouze při startu hosta (serveru) bude umožněno načíst uložený modelovací prostor.

Vybrání uloženého souboru k načtení je v klasické desktopové aplikaci snadný úkol. V případě VR čelíme potřebě se vyhnout zadávání textu a samozřejmě není k dispozici klasický průzkumník souborů. Navržené řešení je menu ve tvaru zdi, kde jednotlivé panely představují jednotlivé soubory. Uživatel si vybere soubor pomocí ukazovátka nebo nástrojového ovladače. Soubory se načítají ze složky nadefinované v konfiguračním souboru (9.4).
9.6.2 Klient

Bude vytvořen klient, který se pokusí podle parametrů nadejinovaných v konfiguračním souboru (9.4) připojit k serveru, který hostuje modelovací prostor. Jako klient není možné načíst uložený modelovací prostor.

9.6.3 Informace a nápověda

Pro pohodlné používání bude uživateli zobrazována nápověda a užitečné informace. Pro nápovědu bude sloužit určité místo, konkrétně tabule, kde se budou zobrazovat textové informace podle akcí, které bude uživatel provádět. Dále v nutných případech bude ukazována non-diagetic informace (zobrazení textové informace před uživatelem v určité vzdálenosti), aby bylo zajištěno, že uživatel informaci zaregistrouje.

9.7 Prostor pro modelování

V tomto prostoru bude jeden z ovladačů vyměněn za kruhové menu nástrojů (9.7.1) (diagetic menu k danému ovladači - menu se bude pohybovat v závislosti na nástrojovém ovladači), druhý ovladač bude sloužit k samotnému modelování a pohybu po modelovacím prostoru. Dále bude použito i non-diagetic menu, které bude zobrazovat pro uživatele důležité informace (9.6.3).
9.7.1 Menu pro výběr nástrojů

Toto menu bude obsahovat všechny potřebné a dostupné funkce pro práci v modelovacím prostoru. Uživatel se v něm bude pohybovat pomocí joysticku na nástrojovém ovladači, který bude toto menu otáčet ve zvoleném směru. Položka menu, na které se uživatel nachází, se barevně zvýrazní a vždy se bude nacházet na dvanácté hodině. Pokud bude položka vybrána, tak bude zabarvena barvou signalizující výběr a bude možné s aktivním ovladačem používat její funkci. V případě že bude vybrána položka menu, která musí být dále specifikována, tak se menu přemění na submenu s danými položkami. Pokud uživatel vybere určitou možnost v submenu, tak nebude vrácen do prvotního menu. Z tohoto menu se bude můžně dostat pomocí šipky zpět.

Předtím než budou popsány jednotlivé funkce, které modelovací nástroj podporuje, budou rozebrány principy, které platí pro modelování ploch obecně.

- Základní jednotkou pro manipulaci je řídící bod a pouze skrze ně je možné s plochami manipulovat. Jinými slovy zvolením bodu, hrany nebo celého plátu rozumíme zvolení příslušných řídících bodů.
- Vždy když budeme muset vybrat určité řídící body splňující nějakou podmínku (např. pouze hrany, určitý druh plátu ... ), tak budou vysvíceny body, které je možné vybrat (splňují danou podmínku).

**Funkce**, které je možné v modelovacím prostoru využívat:

- Pohybat řídícím bodem
- Vytvořit novou plochu
- Pohybovat objektem
  - Pohybovat s celou plochou
  - Pohybovat s tečnou rovinou
- Rozdělit povrch
- Smazat povrch
- Napojit dva povrchy
  - Spojit hrany
  - Napojit hladce
9.8 Řídící body

Každý řídící bod je GameObject obsahující skript definující jeho chování, který je potomkem nějakého parametrického plátu. Každý skript určující chování řídícího bodu musí splňovat rozhraní IControlPoint (Obr. 9.5).

![Interface]

Obrázek 9.5: IControlPoint

9.9 Napojení

Každý parametrický povrch si udržuje pole o velikosti počtu jeho hran, v kterém si udržuje informace o napojených plátech na dané hraně. K vytvoření vazeb je využit návrhový vzor Observer, který podporuje řídící body parametrických ploch (9.8).

Každý bod si drží seznam řídících bodů, které jsou pevně svázané (List<IControlPoint>). To znamená, že při změně polohy bodu se stejně pohnou i body s ním takto pevně svázané. Díky této funkcionalitě je realizováno ne-hladké napojování Bézierových čtvercových i trojúhelníkových plátů a hladké napojení Coonsových plátů.

9.9.1 Bézierův čtvercový plát - hladké napojení

Problém bude vysvětlen na napojení dvou Bézierových křivek. Postup pro napojení dvou čtvercových Bézierových plátů je totožný, pouze budou takto napojeny čtyři křivky, které k sobě náleží.

Nejprve budou ztotožněny krajní body, na kterých má proběhnout hladké napojení, jako v případě ne-hladkého napojení. Dále jsou body A a B vyrovnané (viz. Obr 9.6), tak aby ležely v přímce (pozn.: jelikož se nacházíme v trojrozměrném prostoru, tak je použit parametrický tvar přímky). Při pohybu s body A, M nebo B je nutné ostatním oznámit změnu jejich polohy, aby se opět zarovnaly do přímky. Za tímto účelem je těmto třem bodům zaveden seznam (List<IControlPoint>) bodů, kterým musí znám své polohy označovat. Dále je zaveden slovník (Dictionary<IControlPoint, IControlPoint, t>), díky kterému je upozorněný bod schopen přepočítat svoji pozici tak, aby ležel stále v přímce s body s kterými je svázan. Klíčem k tomuto slovníku je bod, který posílá upozornění o změně své pozice. V našem případě tedy vypadá propojení následovně.

<table>
<thead>
<tr>
<th>Bod</th>
<th>Seznam k upozornění</th>
<th>Slovník spojení</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>&lt;B, {M, ta}&gt;</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
<td>&lt;A, {M, tb}&gt;</td>
</tr>
<tr>
<td>M</td>
<td>A, B</td>
<td>prázdny</td>
</tr>
</tbody>
</table>

kde $ta = \frac{Tb}{Tb_a}$ (parametr $t$ je vypočítán při sestrojení napojení a pokud nenastanou zvláštní okolnosti, tak je stále stejný). Obdobně je vypočítán parametr $tb$. 

34
9.9.2 Bézierův trojúhelníkový plát - hladké napojení

Napojení dvou Bézierových trojúhelníkových ploch bude téměř totožné jako napojení Bézierových čtvercových ploch, pouze budou body rovnány do plochy místo přímky. Bude použita parametrická rovnice roviny. Souřadnice \([x,y]\) budou prohlášeny za neměnné a z rovnice plochy se vypočtou parametry \(u\) a \(v\) (viz. Obr. 9.7), které budou dále udržovány.

Obrázek 9.7: Napojení Bézierových trojúhelníkových ploch

Problémy při udržování více hladce navázaných Bézierových ploch (jak čtvercových, tak trojúhelníkových) mohou vzniknout v rohových bodech plátů, kde se schází více hladce napojených plátů (čtyři a více). Potíže v těchto místech působí kruhové závislosti (počáteční bod - bod kterým se pohybuje - ovlivňuje body, které po určitém počtu kroků chtějí opět ovlivnit počáteční bod). V takovýchto případech se nebude lpět na udržování para-
metru \( t \) (popřípadě parametrů \( u \) a \( v \)) a bude snaha o nalazení jakéhokoliv parametru, který by nám umožnil napojení udržet. V případě, že to nebude možné, tak bude uživatel upozorněn na porušení hladkého navázání.

**9.10 Uložení do standardního formátu trojúhelníkové sítě**

K uložení vytvořených modelů je možné využít mnoho formátů, mezi známé a často používané patří např. OBJ (Wavefront), PLY nebo STL [7],[8]. Všechny tyto formáty umožňují uložit trojúhelníkovou síť jako seznam vrcholů a trojúhelníků. Jelikož modely budou uloženy právě tímto způsobem (nebudeme ukládat žádné jiné informace), tak lze použít jakýkoliv z dříve navržených formátů. Aplikace pro modelování bude používat formát OBJ.

Jelikož je plocha vykreslována pomocí unity surface shaderu (9.2.1), který vypočítává pozice vrcholů trojúhelníkové sítě pouze pro vykreslování, tak je zapotřebí před uložením tyto pozice opět vypočítat. IO operace (zápis do souboru) jsou samy o sobě časově náročné, proto je potřeba výpočet co nejvíce optimalizovat.

Čtvercové parametrické plochy počítáme dle následující rovnice:

\[
Q(u, v) = \begin{pmatrix} u^3 & u^2 & u \end{pmatrix} \cdot M_f^T \cdot P \cdot M_f \cdot \begin{pmatrix} v \end{pmatrix},
\]  

(9.1)

První optimalizací je výpočet \( M_f^T \cdot P \cdot M_f \) předem a pouze jednou. Toto předpočítání je možné, jelikož se během výpočtu \( M_f^T \cdot P \) ani \( M_f \) nemění. Dále je možné u čtvercových ploch předpočítat \( \bar{u} = (u^3, u^2, u, 1) \), pro \( u = 0, \frac{1}{k}, \frac{2}{k}, \ldots, \frac{k-1}{k}, 1 \), kde \( k \) je počet dělení parametru \( u \). Jak je možné nahlédnout, tak v případě stejných kroků parametru \( u \) a \( v \) stačí vypočítat pouze \( u \) (resp. \( v \)).

**9.11 Uložení modelu pro opětovnou editaci**

Pro tento druh uložení je možné použít libovolný formát. Z důvodů použitelnosti jinými programy by bylo vhodné použít nějaký ze standartních formátů. V úvahu připadají formáty XML a JSON, jelikož do těchto formátů umí Unity data serializovat a existuje mnoho způsobů jak pohodlně data parsovat. Zvolen byl formát JSON.

36
Výčtové typy

- **Typy ploch** ("type"): Bézierova čtvercová (0), Coonsova (1), Bézierova trojúhelníková (2)

- **Typy hran** ("connections.oEdge", "connections.tEdge"): spodní (0), levá (1), pravá (2), horní (3)

- **Typ spojení** ("connections.type"): \( C^1 \) (0), \( C^0 \) (1)

Nahrání souboru

JSON soubor je rozparsován a jsou vytvořeny dané parametrické plochy, které jsou následně uloženy do slovníku pod svým SID (klicem je SID a hodnotou odkaz na danou plochu). Posledním krokem je propojení jednotlivých ploch. Při propojování je možné díky již zmíněného slovníku rychle hledat plochy s daným SID.
9.12 Sdílení modelovacího prostoru

Unity přináší možnosti jak vytvářet projekty sdílené více uživateli (klasicky se jedná o multiplayer ve hrách) díky svému vysokoúrovňovému API (The High Level API). HLAPI je vystavěné na nízkoúrovňové transportní vrstvě, kterou je také možné využívat a v případě potřeby si vytvořit libovolnou nadstandardní funkionalitu, kterou HLAPI neposkytuje. Pokud jsou však vyžadovány standardní prostředky, pak HLAPI zcela postačuje. Transportní vrstva může využívat protokol UDP nebo WebSockets pro WebGL. Díky Unity službě Multiplayer je propojení serveru a klientů velmi snadné [10].

Unity umožňuje aplikaci běžet v různých módách: host, server nebo klient. Host představuje server a klienta v jednom procesu. Tato možnost se využívá v modelovacím programu, pokud chceme pracovat sami, nebo vytvořit modelovací prostor, ke kterému se ostatní mohou připojit. V případě severu je nutné specifikovat port, na kterém má aplikace poslouchat a v případě klienta je ještě zapotřebí specifikovat IP adresu serveru. Obě tyto nastavení musí být vyplněny v konfiguračním souboru (9.4). Pro vytvoření hosta nebo připojení klienta k serveru budeme využívat komponentu NetworkManager [10].

Host využívá speciální typ klienta, který se nazývá místní, narodil od klasických klientů, kteří jsou vzdálení. Díky spojení místního klienta a serveru (mód host) probíhá komunikace přes přímé volání funkcí a fronty zpráv, navíc spolu sdílí scénu. Na rozdíl od místního klienta ti vzdálení komunikují přes síť (Obr. 9.8)[10].

Jednoduchost systému Unity spočívá v tom, že kód zůstává pořád stejný.
Stačí pouze rozlišit, kdo má daný kód vyvolat. Pomocí systému autorit lze zjistit, zda máme autoritu nad daným objektem (klasicky má klient autoritu pouze nad hráčským objektom, ale od verze Unity 5.2 je možné vytvořit objekt s autoritou klienta, viz dokumentaci [10]). Je potřeba, aby objekty vytvářel server a udržoval synchronizaci mezi klienty, popřípadě objekty mazal. Této role se ujme správce modelovacího prostoru (SpaceManager), který bude udržovat informace o všech parametrických plochách v modelovacím prostoru, tvořit je a mazat. Při tvorbě objektů, které mají být sdílené, je potřeba kromě metody Instantiate() volat i metodu Spawn(). Místo klasické třídy MonoBehaviour musí objekty, které chtějí využívat síťovou komunikaci, dědit třídu NetworkBehaviour ze jmenného prostoru UnityEngine.Networking, a používat pro objekty komponenty potřebné k synchronizaci pohybu (NetworkTransform), získání identity (NetworkIdentity) a případně další.

Je možné používat dva druhy volání vzdálených akcí (RPC). Jedny jsou nazývány Commands a druhé ClientsRPC. Commands jsou procedury volané z klienta na serveru a ClientRPC jsou naopak volány ze serveru na klientu (Obr. 9.9). Díky těmto funkčním probíhá komunikace serveru a klientů.

9.12.1 Realizace

V modelovacím nástroji je zaveden objekt reprezentující lokálního "hráče". Jedině přes tento objekt je možné volat funkce SpaceManagera, který je
umístěn na serveru a spravuje celý modelovací prostor. Každý objekt, který chce využívat služby serveru musí implementovat rozhraní INetworkAware (Obr. 9.10), aby mohl být notifikován o výsledku svého požadavku. Hráčský objekt si uchovává ve slovníku reference na objekty (INetworkAware), které mají požadavek na server, aby jim mohl sdělit výsledek jejich požadavku. Pokud chce uživatel pohybovat s plochou nebo řídícími body, tak musí nejprve server požádat o přidělení autority nad danými objekty. Autoritu nad objektem může mít pouze jeden uživatel. Z tohoto důvodu je žádost o přidělení autority nad danými objekty odmítnuta pokud autoritu nad daným objektem (popř. objekty) drží jiný uživatel. V případě, že je objekt (rozuměj řídící bod) v držení jiného uživatele, tak je zabarven červeně a není možné s ním pohybovat (respektive získat nad ním autoritu). Pouze server (SpaceManager) si udržuje záznamy o vzájemném propojení ploch a udržuje daná napojení.
10 Testování

Program byl testován funkčními testy a následně byly provedeny uživatelské testy použitelnosti. Na tomto místě je vhodné zmínit, že Unity poskytuje nástroje k jenotkovým i integračním testům [10]. Testování jednotkovými testy ustoupilo požadavku poměrně rychlého vytvoření funkčního prototypu programu, který demonstruje možnosti modelování ve VR.

10.1 Testování použitelnosti

Pro testování použitelnosti programu byl vytvořen scénář, podle kterého se v každém testovacím případu postupovalo. Jako množina testerů byla zvolena skupina uživatelů, kteří jsou schopni pracovat s počítačem alespoň na běžné úrovni (tzv. využívají ho pro svoji práci, studium, nebo pouze ve volném čase). Nebyla požadována znalost problematiky parametrických ploch ani zkušenosti s počítačovými modelovacími programy.

10.1.1 Scénář

Nejprve byla testerům vysvětlena práce s programem a jednotlivé funkce, které je možné použít. Tyto instrukce by se u jednotlivých testerů neměly lišit (pokud ano, tak pouze v maličkostech), jelikož instruktor postupoval podle předem stanovených bodů. Následně bylo testerům umožněno s nástrojem pro modelování libovolně pracovat a dotazovat se instruktora, který program důvěrně zná, na případné nejasnosti. V momentě kdy se tester cítil připraven, byl požádán, aby splnil následující úkoly:

1. Přesuňte se k tabuli, vytvořte Bézierův čtvercový plát a poté ho smažte.
2. Přesuňte se doprostřed modelovacího prostoru, vytvořte dva Bézierovy trojúhelníkové pláty a hladce je napojte.
4. Vytvořte dva Bézierovy trojúhelníkové pláty a pomocí dostupných funkcí z nich vytvořte model ve tvaru padáku.
5. Vytvořte tvar podobný krychli.
6. Vytvořte tvar podobný trojbokému jehlanu.

Tento scénář byl zvolen tak, aby bylo možné využít všech dostupných funkcí, a aby nebylo nijak obtížné jej splnit pro uživatele, který nemá velké znalosti v oblasti geometrie ani počítačového modelování. Jak je možné si povšimnout, tak do testování nebyly zařazeny Coonsovy plochy, jelikož jsou pouze doplňkovým typem ploch a pokud nemají uživatelé znalosti o jejich chování, tak se s nimi nemodeluje tak přirozeně jako s plochami Bézierovými. Po provedení zadaných úkolů testeři odpověděli na následující otázky:

1. Dělalo se Vám při práci v modelovacím nástroji nevolno?
2. Pracovalo se Vám s nástrojem pohodlně, nebo Vám příjemnou práci něco znemožňovalo? Pokud Vám příjemnou práci něco znemožňovalo, tak co?
3. Chybělo Vám v modelovacím nástroji něco, co by Vám práci zpříjemnilo? Pokud ano, tak co?
4. Je pohyb v menu přirozený a pohodlný? Pokud ne, tak popište co Vám vadilo?
5. Měl(a) jste problém splnit úkol č.1? Pokud ano, tak proč?
6. Měl(a) jste problém splnit úkol č.2? Pokud ano, tak proč?
7. Měl(a) jste problém splnit úkol č.3? Pokud ano, tak proč?
8. Měl(a) jste problém splnit úkol č.4? Pokud ano, tak proč?
9. Měl(a) jste problém splnit úkol č.5? Pokud ano, tak proč?
10. Měl(a) jste problém splnit úkol č.6? Pokud ano, tak proč?

10.1.2 Výsledky testování

Celkem se testů použitelnosti programu zúčastnili čtyři testeři ve věku 22-25 let. Každý tester strávil s testováním přibližně 30 minut. Testování probíhalo na počítači s Windows 10 Home x64, Intel(R) Core(TM) i5-7400 CPU @ 3.00GHz 3.00GHz, 16G RAM. Nyní budou shrnuty jejich odpovědi na jednotlivé otázky.
• Otázka č.1: Žádnému z testerů práce s nástrojem nezpůsobovala pocit nevolnosti.

• Otázka č.2: Všem testerům se s nástrojem pracovalo pohodlně, s výjimkou několika připomínek. Jeden tester zmínil špatnou čitelnost nápovědy umístěné na tabulí (text problikává). Virtuálním laserovým ukazovátkem není možné ukázat na zdi. Poslední tester také uvedl, že mu znepříjemňovala práci délka kabelu k HMD (příliš krátký).

• Otázka č.3: Všichni testeři uvedli, že by se jim líbila funkce, kdy by s výsledným modelem (např. vytvořenou krychlí) bylo možné pohybovat jako s jedním objektem a ne s každou plochou zvlášť.

• Otázka č.4: Jeden tester uvedl, že po vrácení ze submenu by chtěl být navrácen na pozici, kde v menu před vstupem do submenu skončil, a ne na první položku.

• Otázky č.5 - 10: Testeři neměli se splněním úkolů problémy. Jediná připomínka, která se vyskytla u více testerů, se týkala obtížnosti vybrat dva body, které jsou téměř na sobě. Jeden z testerů také zmínil, že je obtížné vybírat body, které jsou schované za plochou (popř. uvnitř objektu).

10.1.3 Závěr z testování

11 Závěr

Z testování použitelnosti vyplývá, že byl vytvořen nástroj pro modelování parametrických ploch, který umožňuje pohodlné modelování ve VR. Tento nástroj nezpůsobuje při práci nevolnost, umožňuje uložit vytvořený model a dokonce je možné svůj modelovací prostor sdílet s více uživateli. Možnosti tohoto nástroje jsou stále omezené a pro plnohodnotné modelování by bylo potřeba zaavést mnoho dalších funkcí. S přibývajícím počtem funkcí by bylo potřeba přehodnotit vzhled a chování menu, aby se stále udržel komfort při práci. Další rozšíření, která se pro modelovací nástroj nabízí, je zavedení NURBS ploch, které by do modelování vneslo nové možnosti a automatická tvorba základních tvarů z NURBS ploch nebo aproximací. Navržený a následně vytvořený nástroj pro modelování s prametrickými plochami je vhodný pro pochopení chování parametrických ploch a základní modelování, které nevyžaduje velkou přesnost.
Přehled zkratek

- VR - virtuální realita
- AR - augmentovaná realita
- HMD - Head-Mounted Display - brýle pro VR
- NURBS - neuniformní racionální B-Spline
- Ovladač nástrojů - ovladač pro interakci v menu
- Aktivní ovladač - ovladač pro pohyb v modelovacím prostoru a pro samotné modelování
- HLAPI - High Level API od Unity
- MTP Latency - Motion To Photon Latency
- FPS - Frames Per Second
Literatura


Přílohy
Bernsteinovy polynomy

Přesněji baze prostory Bersteinových polynomů, jsou definované jako

\[ B^n_k(t) = \binom{n}{k} t^k (1-t)^{n-k} \]

Pro barycentrické souřadnice \( \alpha, \beta, \gamma \) definujeme zobecněný Bersteinův polynom vztahem

\[ B^n_{i,j,k}(\alpha, \beta, \gamma) = \frac{n!}{i!j!k!} \alpha^i \beta^j \gamma^k, \] (11.1)

\[ [1][2] \]

B-spline baze

Označme \( T = (t_0, \ldots, t_m) \) tzv. vektor parametrizace. Platí: \( t_0 \leq t_1 \leq \ldots \leq t_m \)

B-spline baze je tvořena funkcemi (polynomy) \( N_i^k \) stupně \( k \) definovanými předpisem:

- pro \( k = 0 \)

\[ N_i^0(t) = \begin{cases} 1 & \text{pro } t \in (t_i, t_{i+1}) \\ 0 & \text{jinak} \end{cases} \]

- pro \( k > 0 \)

\[ N_i^k(t) = \frac{t-t_i}{t_{i+k}-t_i} N_i^{k-1}(t) + \frac{t_{i+k+1}-t}{t_{i+k+1}-t_{i+1}} N_{i+1}^{k-1}(t), \]

Je nutné vzít v úvahu, že v tomto výrazu mohou vzniknout výrazy typu \( \frac{0}{0} \), které definitoricky položíme rovny nule.

B-spline baze je tedy charakterizována:

- stupněm \( k \) polynomů
- vektorem parametrizace, tj.
  - číslem \( m \) - vektor parametrizace má \( (m+1) \) složek
  - složkami \( t_0 \leq t_1 \leq \ldots \leq t_m \)
- číslem \( j \) - počet funkcí tvořících bazi.

Musí platit \( m \geq k + j \), stačí však volit \( m = k + j \), tj. počet složek parametrického vektoru je roven součtu stupně B-spline baze a počtu funkcí baze \([1][2]\).
**Spojitost křivky**

Říkáme, že křivka \( Q(t) \) je třídy \( C^n \), má-li ve všech bodech spojité derivace podle parametru \( t \) do řádu \( n \). Ozačení \( C^n \) se nazývá parametrická spojitost stupně \( n \).

Geometrická spojitost \( G^n \) v daném bodě je definována nezávisle na parametru \( t \). Za předpokladu, že obě křivky jsou v místě spojení diferencovatelné, pak \( Q_1 \) a \( Q_2 \) splňují podmínku geometrické spojitosti \( G^n \), pokud jsou v bodě \([x_0, y_0, z_0]\) \( G^{n-1} \) spojité a platí

\[
\left[ \frac{\partial^n Q_1}{\partial x^n}, \frac{\partial^n Q_1}{\partial y^n}, \frac{\partial^n Q_1}{\partial z^n} \right]_{[x_0, y_0, z_0]} = h \cdot \left[ \frac{\partial^n Q_2}{\partial x^n}, \frac{\partial^n Q_2}{\partial y^n}, \frac{\partial^n Q_2}{\partial z^n} \right]_{[x_0, y_0, z_0]}, \quad n, h > 0.
\]

Ze subjektivního hlediska zaručuje \( G^1 \) spojitost "skoro stejnou" hladkost jako \( C^1 \), z hlediska použití bývá daleko snažší zaručit spojitost \( G^n \) nežli \( C^1 \). Spojitost \( C^1 \) implikuje \( G^1 \) s výjimkou jediného případu, kdy vektor rychlosti v místě spojení dvou segmentů je \((0, 0, 0)\) [1][2].

**Výpočet bodů ovlivňujících tečnou rovinu v krajních bodech**

**Bézierovy plochy**

Mějme Bézierovu kubickou plochu zadanou jako

\[
Q(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} P_{i,j} B^m_i(u) B^n_j(v)
\]

(11.2)

, kde \( m = n = 3; v, u \in <0, 1> \)

Bersteinovy polynomy jsou

\[
B^3_0(t) = (1 - t)^3 \\
B^3_1(t) = 3t(1 - t)^2 \\
B^3_2(t) = 3t^2(1 - t) \\
B^3_3(t) = t^3
\]

(11.3)

Nejprve budeme hledat \( \alpha(u, v) = \frac{\partial Q(u, v)}{\partial u} \)
Upravíme li rovnici popisující Bézierovu plochu získáme následující vyjadření plochy

$$(Q(u, v) = (1 - u)^3 \cdot [P_{0,0}(1 - v)^3 + P_{0,1}3v(1 - v)^2 + P_{0,2}3v^2(1 - v) + P_{0,3}v^3] + 3u(1 - u)^2 \cdot [P_{1,0}(1 - v)^3 + P_{1,1}3v(1 - v)^2 + P_{1,2}3v^2(1 - v) + P_{1,3}v^3] + 3u^2(1 - u) \cdot [P_{2,0}(1 - v)^3 + P_{2,1}3v(1 - v)^2 + P_{2,2}3v^2(1 - v) + P_{2,3}v^3] + u^3 \cdot [P_{3,0}(1 - v)^3 + P_{3,1}3v(1 - v)^2 + P_{3,2}3v^2(1 - v) + P_{3,3}v^3]$$

Pro usnadnění další práce uvažujme substituce

$$a = P_{0,0}(1 - v)^3 + P_{0,1}3v(1 - v)^2 + P_{0,2}3v^2(1 - v) + P_{0,3}v^3$$

$$b = P_{1,0}(1 - v)^3 + P_{1,1}3v(1 - v)^2 + P_{1,2}3v^2(1 - v) + P_{1,3}v^3$$

$$c = P_{2,0}(1 - v)^3 + P_{2,1}3v(1 - v)^2 + P_{2,2}3v^2(1 - v) + P_{2,3}v^3$$

$$d = P_{3,0}(1 - v)^3 + P_{3,1}3v(1 - v)^2 + P_{3,2}3v^2(1 - v) + P_{3,3}v^3$$

$$\alpha(u, v) = \frac{\partial Q(u, v)}{\partial u} = -3(1 - u)^2 \ast a + 3(u - 1)(3u - 1) \ast b + (6u - 9u^2) \ast c + 3u^2 \ast d$$

Dále budeme hledat $\beta(u, v) = \frac{\partial Q(u, v)}{\partial v}$

$$Q(u, v) = (1 - v)^3 \cdot [P_{0,0}(1 - u)^3 + P_{1,0}3u(1 - u)^2 + P_{2,0}3u^2(1 - u) + P_{3,0}u^3] + 3v(1 - v)^2 \cdot [P_{0,1}(1 - u)^3 + P_{1,1}3u(1 - u)^2 + P_{2,1}3u^2(1 - u) + P_{3,1}u^3] + 3v^2(1 - v) \cdot [P_{0,2}(1 - u)^3 + P_{1,2}3u(1 - u)^2 + P_{2,2}3u^2(1 - u) + P_{3,2}u^3] + v^3 \cdot [P_{0,3}(1 - u)^3 + P_{1,3}3u(1 - u)^2 + P_{2,3}3u^2(1 - u) + P_{3,3}u^3]$$

Pro usnadnění další práce uvažujme substituce

$$e = P_{0,0}(1 - u)^3 + P_{1,0}3u(1 - u)^2 + P_{2,0}3u^2(1 - u) + P_{3,0}u^3$$

$$f = P_{0,1}(1 - u)^3 + P_{1,1}3u(1 - u)^2 + P_{2,1}3u^2(1 - u) + P_{3,1}u^3$$

$$g = P_{0,2}(1 - u)^3 + P_{1,2}3u(1 - u)^2 + P_{2,2}3u^2(1 - u) + P_{3,2}u^3$$

$$h = P_{0,3}(1 - u)^3 + P_{1,3}3u(1 - u)^2 + P_{2,3}3u^2(1 - u) + P_{3,3}u^3$$

$$\beta(u, v) = \frac{\partial Q(u, v)}{\partial v} = -3(1 - v)^2 \ast e + 3(v - 1)(3v - 1) \ast f + (6v - 9v^2) \ast g + 3v^2 \ast h$$

Nyní dosadíme do námí nalezených derivací v jednotlivých směrech a získáme body řídící sítě, které ovlivňují tečnou rovinu Bézierovy kubické
plochy v rohových bodech

\[ \alpha(0, 1) = -3a + 3b = -3P_{0,3} + 3P_{1,3} \]
\[ \beta(0, 1) = -3g + 3h = -3P_{0,2} + 3P_{0,3} \]

\[ \alpha(1, 1) = -3c + 3d = -3P_{2,3} + 3P_{3,3} \]
\[ \beta(1, 1) = -3g + 3h = -3P_{3,2} + 3P_{3,3} \]

\[ \alpha(1, 0) = -3c + 3d = -3P_{2,0} + 3P_{3,0} \]
\[ \beta(1, 0) = -3e + 3f = -3P_{3,0} + 3P_{3,1} \]

\[ \alpha(0, 0) = -3a + 3b = -3P_{0,0} + 3P_{1,0} \]
\[ \beta(0, 0) = -3e + 3f = -3P_{0,0} + 3P_{0,1} \]

**B-spline plochy**

Mějme kubickou B-spline plochu zadanou jako

\[ Q(u, v) = \frac{1}{36} \sum_{i=0}^{m} \sum_{j=0}^{n} P_{i,j} N_i^m(u) N_j^n(v) \]  

(11.4)

kde \( m = n = 3; v, u \in <0, 1> \)

Zvolíme bazové funkce pro Consův B-spline

\[ N_0^3(t) = -t^3 + 3t^2 - 3t + 1 \]
\[ N_1^3(t) = 3t^3 - 6t^2 + 4 \]
\[ N_2^3(t) = -3t^3 + 3t^2 + 3t + 1 \]
\[ N_3^3(t) = t^3 \]  

(11.5)

Nejprve budeme hledat \( \alpha(u, v) = \frac{\partial Q(u,v)}{\partial u} \)
Upravíme li rovnici popisující B-spline plochu získáme následující vyjádření plochy

\[ Q(u, v) = \frac{1}{36} \left\{ (-u^3 + 3u^2 - 3u + 1) \ast [P_{0,0}(-v^3 + 3v^2 - 3v + 1) + P_{0,1}(3v^3 - 6v^2 + 4) + P_{0,2}(-3v^3 + 3v^2 + 3v + 1) + P_{0,3}v^3] + (3u^3 - 6u^2 + 4) \ast [P_{1,0}(-v^3 + 3v^2 - 3v + 1) + P_{1,1}(3v^3 - 6v^2 + 4) + P_{1,2}(-3v^3 + 3v^2 + 3v + 1) + P_{1,3}v^3] + (-3u^3 + 3u^2 + 3u + 1) \ast [P_{2,0}((-v^3 + 3v^2 - 3v + 1) + P_{2,1}(3v^3 - 6v^2 + 4) + P_{2,2}(-3v^3 + 3v^2 + 3v + 1) + P_{2,3}v^3] + u^3 \ast [P_{3,0}((-v^3 + 3v^2 - 3v + 1) + P_{3,1}(3v^3 - 6v^2 + 4) + P_{3,2}(-3v^3 + 3v^2 + 3v + 1) + P_{3,3}v^3] \right\} \]

Pro usnadnění další práce uvažujme substituce

\[ a = P_{0,0}(-v^3 + 3v^2 - 3v + 1) + P_{0,1}(3v^3 - 6v^2 + 4) + P_{0,2}(-3v^3 + 3v^2 + 3v + 1) + P_{0,3}v^3 \]
\[ b = P_{1,0}(-v^3 + 3v^2 - 3v + 1) + P_{1,1}(3v^3 - 6v^2 + 4) + P_{1,2}(-3v^3 + 3v^2 + 3v + 1) + P_{1,3}v^3 \]
\[ c = P_{2,0}(-v^3 + 3v^2 - 3v + 1) + P_{2,1}(3v^3 - 6v^2 + 4) + P_{2,2}(-3v^3 + 3v^2 + 3v + 1) + P_{2,3}v^3 \]
\[ d = P_{3,0}(-v^3 + 3v^2 - 3v + 1) + P_{3,1}(3v^3 - 6v^2 + 4) + P_{3,2}(-3v^3 + 3v^2 + 3v + 1) + P_{3,3}v^3 \]

\[ \alpha(u, v) = \frac{\partial Q(u, v)}{\partial u} = \frac{1}{36} \left\{ (-3u^2 + 6u - 3) \ast a + (9u^2 - 12u) \ast b + (9u^2 + 6u + 3) \ast c + 3u^2 \ast d \right\} \]

Dále budeme hledat \( \beta(u, v) = \frac{\partial Q(u, v)}{\partial v} \)

Upravíme li rovnici popisující B-spline plochu získáme následující vyjádření plochy

\[ Q(u, v) = \frac{1}{36} \left\{ (-v^3 + 3v^2 - 3v + 1) \ast [P_{0,0}(-u^3 + 3u^2 - 3u + 1) + P_{1,0}(3u^3 - 6u^2 + 4) + P_{2,0}(-3u^3 + 3u^2 + 3u + 1) + P_{3,0}u^3] + (3v^3 - 6v^2 + 4) \ast [P_{0,1}(-u^3 + 3u^2 - 3u + 1) + P_{1,1}(3u^3 - 6u^2 + 4) + P_{2,1}(-3u^3 + 3u^2 + 3u + 1) + P_{3,1}u^3] + (-3u^3 + 3u^2 + 3u + 1) \ast [P_{0,2}((-u^3 + 3u^2 - 3u + 1) + P_{1,2}(3u^3 - 6u^2 + 4) + P_{2,2}(-3u^3 + 3u^2 + 3u + 1) + P_{3,2}u^3] + v^3 \ast [P_{3,0}((-u^3 + 3u^2 - 3u + 1) + P_{3,1}(3u^3 - 6u^2 + 4) + P_{3,2}(-3u^3 + 3u^2 + 3u + 1) + P_{3,3}u^3] \right\} \]

Pro usnadnění další práce uvažujme substituce

\[ e = P_{0,0}(-u^3 + 3u^2 - 3u + 1) + P_{1,0}(3u^3 - 6u^2 + 4) + P_{2,0}(-3u^3 + 3u^2 + 3u + 1) + P_{3,0}u^3 \]

53
$f = P_{0,1}(-u^3 + 3u^2 - 3u + 1) + P_{1,1}(3u^3 - 6u^2 + 4) + P_{2,1}(-3u^3 + 3u^2 + 3u + 1) + P_{3,1}u^3$

$g = P_{0,2}(-u^3 + 3u^2 - 3u + 1) + P_{1,2}(3u^3 - 6u^2 + 4) + P_{2,2}(-3u^3 + 3u^2 + 3u + 1) + P_{3,2}u^3$

$h = P_{0,3}(-u^3 + 3u^2 - 3u + 1) + P_{1,3}(3u^3 - 6u^2 + 4) + P_{2,3}(-3u^3 + 3u^2 + 3u + 1) + P_{3,3}u^3$

$\beta(u, v) = \frac{\partial Q(u, v)}{\partial v} = \frac{1}{36} [(-3v^2 + 6v - 3)\epsilon + (9v^2 - 12v)\epsilon f + (9v^2 + 6v + 3)\epsilon g + 3v^2 \epsilon h]$

Nyní dosadíme do námi nalezených derivací v jednotlivých směrech a získáme body řídící sítě, které ovlivňují tečnou rovinu kubické B-spline plochy v rohových bodech

$\alpha(0, 1) = \frac{1}{36}(-3a + 3c) = \frac{1}{36}[-3(P_{0,0} + 4P_{0,1} + P_{0,2}) + 3(P_{2,0} + 4P_{2,1} + P_{2,2})]$

$\beta(0, 1) = \frac{1}{36}(-3e + 3g) = \frac{1}{36}[-3(P_{0,0} + 4P_{1,0} + P_{2,0}) + 3(P_{0,2} + 4P_{1,2} + P_{2,2})]$

$\alpha(1, 1) = \frac{1}{36}(-3a + 3c) = \frac{1}{36}[-3(P_{0,1} + 4P_{0,2} + P_{0,3}) + 3(P_{2,1} + 4P_{2,2} + P_{2,3})]$

$\beta(1, 1) = \frac{1}{36}(-3f + 3h) = \frac{1}{36}[-3(P_{0,1} + 4P_{1,1} + P_{2,1}) + 3(P_{0,3} + 4P_{1,3} + P_{2,3})]$

$\alpha(1, 0) = \frac{1}{36}(-3b + 3d) = \frac{1}{36}[-3(P_{1,1} + 4P_{1,2} + P_{1,3}) + 3(P_{3,1} + 4P_{3,2} + P_{3,3})]$

$\beta(1, 0) = \frac{1}{36}(-3f + 3h) = \frac{1}{36}[-3(P_{1,1} + 4P_{2,1} + P_{3,1}) + 3(P_{1,3} + 4P_{2,3} + P_{3,3})]$

$\alpha(0, 0) = \frac{1}{36}(-3b + 3d) = \frac{1}{36}[-3(P_{0,0} + 4P_{0,1} + P_{0,2}) + 3(P_{3,0} + 4P_{3,1} + P_{3,2})]$

$\beta(0, 0) = \frac{1}{36}(-3e + 3g) = \frac{1}{36}[-3(P_{1,0} + 4P_{2,0} + P_{3,0}) + 3(P_{1,2} + 4P_{2,2} + P_{3,2})]$

54
Bézierovy trojúhelníky

Mějme kubický Bézierův trojúhelník zadaný jako

\[ Q(s, tu) = (\alpha s + \beta t + \gamma u)^3 = \beta^3 t^3 + 3\alpha \beta^2 s^2 t + 3\beta^2 \gamma t^2 u + 3\alpha \beta \gamma s^2 t + 6\alpha \beta \gamma s t u + 3\beta \gamma^2 t u^2 + \alpha^3 s^3 + 3\alpha^2 \gamma s^2 u + 3\alpha \gamma^2 s u^2 + \gamma^3 u^3 \]

(11.6)

Řídící body sítě si přepíšeme do následujícího tvaru

\[
\begin{align*}
P_0 &= \beta^3 \\
P_7 &= \alpha \beta^2, P_8 = \beta^2 \gamma \\
P_4 &= \alpha^2 \beta, P_5 = \alpha \beta \gamma, P_6 = \beta \gamma^2 \\
P_0 &= \alpha^3, P_1 = \alpha^2 \gamma, P_2 = \alpha \gamma^2, P_2 = \gamma^3
\end{align*}
\]

Pro body ve tvaru \( P_i \) přepíšeme rovnici Bézierova trojúhelníku do tvaru

\[
Q(s, t, u) = P_0 s^3 + 3P_7 s t^2 + 3P_8 s t^2 u + 3P_4 s^2 t + 6P_5 s t u^2 + P_6 s^3 + 3P_1 s t^2 + P_3 s t u^2 + P_3 u^3
\]

Nyní budeme hledat \( \frac{\partial Q(s, t, u)}{\partial s} \), ale jelikož jsou argumenty \( s, t, u \) závislé musíme přepsat \( Q(s, t, u) \) na \( Q(s, t, 1-s-t) \), abychom mohli provést parciální derivaci ve směrech \( s \) a \( t \) a například do tvaru \( Q(s, 1-s-u, u) \), abychom mohli provést parciální derivací ve směru \( u \).

\[
Q(s, t, 1-s-t) = P_0 s^3 + 3P_7 s t^2 + 3P_8 s t^2 (1-s-t) + 3P_4 s^2 t + 6P_5 s t (1-s-t) + 3P_1 s t^2 + 3P_2 s (1-s-t) + 3P_3 (1-s-t)^3
\]

\[
A(s, t, 1-s-t) = \frac{\partial Q(s, t, 1-s-t)}{\partial s} = 3P_7 t^2 + 3P_8 t^2 + 3P_4 s^2 t + 6P_5 s t (1-s-t) - 6P_5 s t - 6P_5 t (1-s-t) + 3P_1 s^2 + 6P_1 s (1-s-t) - 3P_1 s^2 + 3P_2 (1-s-t)^2 - 6P_2 s (1-s-t) - 3P_3 (1-s-t)^2
\]

\[
B(s, t, 1-s-t) = \frac{\partial Q(s, t, 1-s-t)}{\partial t} = 3P_0 t^3 + 6P_7 s t + 6P_8 s t (1-s-t) - 3P_3 t^2 + 3P_4 s^2 + 6P_5 s (1-s-t) - 6P_5 s t + 6P_5 (1-s-t)^2 - 6P_5 t (1-s-t) - 3P_3 s^2 + 6P_2 s (1-s-t) - 3P_3 (1-s-t)^2
\]
\[ Q(s, 1 - s - u, u) = P_9(1 - s - u)^3 + 3P_7s(1 - s - u)^2 + 3P_8(1 - s - u)^2u + \\
3P_4s^2(1 - s - u) + 6P_5s(1 - s - u)u + 3P_6(1 - s - u)u^2 + \\
P_0s^3 + 3P_1s^2u + 3P_2su^2 + P_3u^3 \]

\[ C(s, 1 - s - u, u) = \frac{\partial Q(s, 1 - s - u, u)}{\partial u} = -3P_9(s, 1 - s - u, u)^2 - 6P_7s(s, 1 \\
- s - u, u) + 3P_8(s, 1 - s - u, u)^2 - 6P_8u(s, 1 - s - u, u) \\
- 3P_4s^2 + 6P_5s(s, 1 - s - u, u) - 6P_5su + 6P_6u(s, 1 - s - \\
u, u) - 3P_6u^2 + 3P_1s^2 + 6P_2su + 3P_3u^2 \]

Nyní se opět vrátíme k parametrům \( s, t, u \)

\[ A(s, t, u) = 3P + 7t^2 - 3P_3t^2 + 6P_4st + 6P_5tu - 6P_6tu + 3P_0s^2 + 6P_1su \\
- 3P_1s^2 + 3P_2u^2 - 6P_2su - 3P_3u^2 \]

\[ B(s, t, u) = 3P_9t^2 + 6P_7st + 6P_8stu - 3P_8s^2 + 3P_4s^2 + 6P_5su - 6P_5st + 3P_6u^2 - \\
6P_6tu - 3P_1s^2 - 6P_2su - 3P_3u^2 \]

\[ C(s, t, u) = -3P_9t^2 - 6P_7st + 3P_8s^2 - 6P_8stu - 3P_4s^2 + 6P_5st - 6P_5su + 6P_6ut \\
- 3P_6u^2 + 3P_1s^2 + 6P_2su + 3P_3u^2 \]

Nyní dosadíme do námí nalezených derivací v jednotlivých směrech a získáme body řídící sítě, které ovlivňují tečnou rovinu Bázierova kubického trojúhelníku v rohových bodech

\[ A(0, 0, 0) = 3P_0 - 3P_1 \]

\[ B(0, 0, 0) = 3P_4 - 3P_1 \]

\[ C(0, 0, 0) = 3P_1 - 3P_4 \]

\[ A(0, 1, 0) = 3P_7 - 3P_8 \]

\[ B(0, 1, 0) = 3P_9 - 3P_8 \]

\[ C(0, 1, 0) = 3P_8 - 3P_9 \]

\[ A(0, 0, 1) = 3P_2 - 3P_3 \]

\[ B(0, 0, 1) = 3P_6 - 3P_3 \]

\[ C(0, 0, 1) = 3P_3 - 3P_6 \]
Uživatelská příručka

Program *VRModeler* lze spustit dvěma způsoby. Standardní cesta je spuštění *VRModeler.exe*. Druhou možností, pokud máte k dispozici projekt z Unity, je do Unity projekt naimportovat a program spustit přes toto vývojové prostorejší.

Ovladače

V programu jsou k dispozici dva ovladače, jeden aktivní a druhý nástrojový (viz. sekce konfigurační soubor). Je doporučováno používat ovladače Oculus Rift, ale je možné využít i ovladače HTC Vive. Na ovladačích lze využít následující tlačítka.

<table>
<thead>
<tr>
<th>Název v příručce</th>
<th>Oculus Rift (Obr. 15.1)</th>
<th>HTC Vive (Obr. 15.2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hlavní tlačítko</td>
<td>Primary/Secondary/thumbTrigger</td>
<td>7</td>
</tr>
<tr>
<td>Joystick</td>
<td>Primary/Secondary/thumbStick</td>
<td>2</td>
</tr>
</tbody>
</table>

Nástrojový ovladač slouží k pohybu v menu. Aktivní ovladač slouží k modelování, pohybu po prostoru a k ovládání virtuálního laserového ukazovátku.

Obrázek 11.1: Ovladače Oculus Rift [10]
Obrázek 11.2: Ovládeč HTC Vive [10]

**Konfigurační soubor**

Konfigurační soubor se nachází ve složce se spustitelným souborem VR-Modeler.exe a je pojmenován vrmodeler.conf (pokud nebude tento soubor nalezen, tak bude vytvořen nový s výchozím nastavením. V případě, že do složky nelze zapisovat, tak bude program pouze spuštěn s výchozím nastavením). Tento soubor je ve formátu JSON a je možné skrze něj upravovat konfiguraci programu.

```plaintext
# VR MODELER SETTINGS
# Every setting must be within one line

# SETTINGS OF SAVING FOLDERS

# Folder where meshes are stored (absolute path)
MeshSaveFolder = C:/Users/vasta/Documents/SavedMeshFiles/

# Folder where models which can be loaded and edited
# are stored (absolute path)
SpaceSaveFolder = C:/Users/vasta/Documents/SavedSpaceFiles/
```
# SETTINGS OF OBJECTS

# Active controller hand R = right, L = left
ActiveController = R

# Heightening of camera (if the number is negative camera
# will be decreased)
CameraHeightening = 0.0

# SETTINGS OF NETWORK

# IPv4 address of server (host)
ServerAddress = localhost

# Port the server (host) is listening on
ConnectPort = 7777

# Port created server (host) will listen on
ListenPort = 7777

Pohyb

V nástroji se lze pohybovat pomocí teleportace. Stačí namířit laserovým
ukazovátkem (aktivuje se zmáčknutím joysticku na aktivním ovladači) na
pozici (na zemi) kam se chcete přemístit. Po uvolnění joysticku na aktivním
ovladači budete na toto místo přeneseni.

Hlavní menu

Po spuštění se budete nacházet v hlavním menu (Obr. 15.3). V tomto menu
se můžete pohybovat pomocí joysticku na nástrojovém ovladači (doprava,
doleva) nebo vybírat položky pomocí ukazovátka aktivního ovladače. Pokud
nezvolíte jinou položku menu pomocí ukazovátka, tak je aktivní ta, která
se nachází mezi šipkami (Obr 15.3) (obecně je aktivní položka jasnější než
položky ostatní). Pro zvolení dané položky ji musíte neprve vybrat a poté
zmáčknout hlavní tlačítko na nástrojovém ovladači.
Obrázek 11.3: Hlavní menu aplikace

Položky menu (Obr. 15.3)

1. Client pokusí se připojit k modelovacímu prostoru nadefinovaném v konfiguračním souboru.

2. Host založí nový modelovací prostor sdílený s ostatními uživateli, podle nastavení v konfiguračním souboru. Po zvolení se dostaneme do dalšího menu (Obr. 15.4), kde si lze vybrat zda chcete založit nový soubor, nebo nahrát model uložený v souboru ze složky definované v konfiguračním souboru (výchozí složka je %USER%/Dokumenty/SavedSpaceFiles).

3. Exit ukončí modelovací program

Obrázek 11.4: Výběrové submenu HOST

Modelovací prostor

Nástrojový ovladač se přemění na nástrojové menu (Obr. 15.5). Toto menu v základní formě nenabízí menu možností (Obr 15.5). Pokud položka menu funkci možností využívá, tak ji lze aktivovat pomocí zmáčknutí joysticku na nástrojovém ovladači. Možnosti fixace pohybu v daných osách jsou v
horní řadě (Obr. 15.5 a, b, c) a možnosti fixace rotace v daných osách jsou ve spodní řadě (Obr. 15.5 c, d, e). Možnosti fixace jsou tlačítka, proto je pro jejich aktivaci nutné na ně zmáčknout aktivním ovladačem. Pokud je tlačítko aktivované, tak ho deaktivujete opětovným zmáčknutím aktivním ovladačem.

**Výběr**

Manipulace s plochami probíhá výhradně přes jejich řídící body.

- **Výběr bodu** pomocí aktivního ovladače (místo výběru je označeno koulí) najedně na bod, který chcete vybrat. Pokud je bod připraven k výběru, tak změní svoji barvu. Pro výběr zmáčkněte a držte hlavní tlačítko na aktivním ovladači. Pro zrušení výběru uvolněte hlavní tlačítko na aktivním ovladači.

- **Výběr plochy** vyberte bod dané plochy.

- **Výběr hrany** vyberte jeden z prostředních bodů hrany.

**Položky menu**

Pokud položka menu podporuje využití menu možností, tak je za ní v závorce uvedeno M.

![Obrázek 11.5: Nástrojové menu](image)

1. **Pohyb s jedním řídícím bodem** (M)
2. **Vytvořit novou plochu** (Obr. 15.6) přesuňte aktivní ovladač na místo, kde chcete vytvořit danou plochu. Pro vytvoření plochy zmáčkněte hlavní tlačítko na aktivním ovladači.

   (a) Bézierova čtvercová plocha
   (b) Coonsova plocha
   (c) Bézierova trojúhelníková plocha
   (d) Zpět do hlavního nástrojového menu

3. **Pohyb s objektem** (M) (Obr. 15.7)

   (a) Pohyb s celou plochou (M) vyberte plochu
   (b) Pohyb s tečnou rovinou (M) vyberte rohový bod plochy
   (c) Zpět do hlavního nástrojového menu

4. **Rozdělit plochu** vyberte bod plochy, kterou chcete rozdělit. Pokud se jedná o trojúhelníkový plát, tak lze vybráním specifické hrany rozdělit plochu právě podle této hrany.

5. **Smazat plochu** vyberte bod plochy, kterou chete smazat.

6. **Spojit hrany** (Obr. 15.8) vyberte první hranu pro napojení a poté vyberte druhou hranu pro napojení.

   (a) Ostré napojení
   (b) Hladké napojení
   (c) Zpět do hlavního nástrojového menu

7. **Rozpojit hrany** vyberte hranu, která je napojená.

8. **Zvětšování/zmenšování** vyberte plochu, kterou chcete zvětšovat/zmenšovat a táhnutím plochu zvětšíte/zmenšíte.

9. **Soubor** (Obr. 15.9)

   (a) Uloží vytvořený model jako trojúhelníkovou síť do formátu OBJ do složky definované v konfiguračním souboru (výchozí složka pro uložení je %USER%/Dekoumenty/SavedMeshFiles)
   (b) Uloží vytvořený model s možností opětovné editace do složky definované v konfiguračním souboru (výchozí složka pro uložení je %USER%/Dekoumenty/SavedSpaceFiles)
(c) Zpět do hlavního menu celé aplikace (Obr. 15.3). POZOR pokud si model neuložíte, tak o něj přijdete!

(d) Zpět do hlavního nástrojového menu

Obrázek 11.6: Submenu vytvoř novou plouchu

Obrázek 11.7: Submenu pohyb s objektem
Obrázek 11.8: Submenu napojení hran

Obrázek 11.9: Submenu soubor