VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

WEBOVÉ APLIKACE PRO PODPORU VÝUKY POČÍTAČOVÉ GRAFIKY

WEB APPS SUPORTING EDUCATION IN COMPUTER GRAPHICS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — Bc. Denis Jalovecký

AUTHOR

VEDOUCÍ PRÁCE — prof. Mgr. Pavel Rajmic, Ph.D.

SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Telekomunikační a informační technika**

Ústav telekomunikací

Student: Bc. Denis Jalovecký ***ID:*** 203569

Ročník: 2 ***Akademický rok:*** 2023/24

NÁZEV TÉMATU:

Webové aplikace pro podporu výuky počítačové grafiky

POKYNY PRO VYPRACOVÁNÍ:

V jazyce JavaScript vytvořte tři webové aplikace, které poslouží jako interaktivní podpora výuky v kurzech zaměřených na vytváření a zpracování obrazových signálů. První aplikace bude demonstrovat parametrickou definici křivky v 3D prostoru (x,y,z). Druhá aplikace se bude věnovat ilustraci 2D barycentrických souřadnic trojúhelníku, což jsou souřadnice hojně využívané ve 2D i 3D grafice. Třetí aplikace bude ukazovat tzv. Bayerovo schéma při typickém způsobu pořízení digitální fotografie. Při tvorbě aplikací se zaměřte na názorné podání vysvětlovaných jevů a na funkčnost pro potřebu výuky.

DOPORUČENÁ LITERATURA:

[1] ŽÁRA, Jiří et al. Moderní počítačová grafika. Brno: Computer Press, 2004, 609 s. ISBN 80-251-0454-0.

[2] PIHAN, Roman. Mistrovství práce s DSLR. Praha: Institut digitální fotografie, 2006, 230 s. ISBN 80-903210-8-9.

Termín zadání: 5.2.2024 ***Termín odevzdání:*** 21.5.2024

Vedoucí práce: prof. Mgr. Pavel Rajmic, Ph.D.

prof. Ing. Jiří Mišurec, CSc.
předseda rady studijního programu

ABSTRAKT

Diplomová práca sa zameriava na vytvorenie troch dynamických a interaktívnych aplikácií v jazyku Javascript, ktoré budú nástrojmi pri výučbe kurzov zameraných na vytváranie a spracovanie obrazových signálov. Prvá aplikácia bude reprezentovať parametrickú krivku v 3D priestore. Druhá aplikácia sa bude zaoberať barycentrickými súradnicami 3D trojuholníka a tretia bude zobrazovať Bayerov filter pri vytváraní digitálnej fotografie.

KĽÚČOVÉ SLOVÁ

počítačová grafika, barycentrické súradnice, Bayerov filter, Parametrická sústava, webové stránky, Javascript, HTML, CSS

ABSTRACT

Master's Thesis focuses on creating three dynamic and interactive applications in JavaScript, serving as educational tools for courses centered around the creation and processing of image signals. The first application will represent a parametric curve in 3D space. The second application will explore barycentric coordinates of a 3D triangle, and the third will illustrate the Bayer filter in the context of digital photography creation.

KEYWORDS

computer graphics, barycentric coordinates, Bayer filter, parametric system, websites, Javascript, HTML, CSS

Vysadené pomocou balíčku `thesis` verzie 4.09; `https://latex.fekt.vut.cz/`

JALOVECKÝ, Denis. *Webové aplikace pro podporu výuky počítačové grafiky*. Diplomová práca. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2023. Vedúci práce: prof. Mgr. Pavel Rajmic, Ph.D.

Vyhlásenie autora o pôvodnosti diela

Meno a priezvisko autora:	Bc. Denis Jalovecký
VUT ID autora:	203569
Typ práce:	Diplomová práca
Akademický rok:	2023/24
Téma závěrečnej práce:	Webové aplikace pro podporu výuky počítačové grafiky

Vyhlasujem, že svoju záverečnú prácu som vypracoval samostatne pod vedením vedúcej/cého záverečnej práce, s využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej záverečnej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto záverečnej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákonníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podpisuje iba v tlačenej verzii.

POĎAKOVANIE

Rád by som poďakoval vedúcemu diplomovej práce prof. Mgr. Pavlovi Rajmicovi Ph.D. za pomoc pri práci, konzultácie a za návrhy na zlepšenie. Tiež by som chcel poďakovať mojej rodine za podporu popri štúdiu.

Obsah

Zoznam obrázkov

Zoznam výpisov

Úvod

Je užitočné vytvárať pre študentov efektívne interaktívne nástroje, pomocou ktorých sa dá preberanej látke lepšie porozumieť. Témy, ktorým sa táto práca bude venovať sa zdajú ako zložité koncepty, kde interaktívne nástroje ukážu, že to až tak zložité nakoniec nie je. V tejto diplomovej práci sú vytvorené tri interaktívne webové aplikácie v jazyku Javascript, ktoré budú slúžiť ako podpora výuky v kurzoch zameraných na problematiku jednotlivých tém popísaných nižšie.

Prvá z aplikácií demonštruje parametrickú definíciu krivky v 3D priestore, čo študentom poskytne intuitívny náhľad do sveta trojrozmerných priestorových kriviek. Druhá aplikácia je zameraná na ilustrácií barycentrických súradníc trojuholníka, ktoré sú veľmi dôležité pri konceptoch v 2D a 3D grafike. Tretia aplikácia je zameraná na Bayerov filter, ktorý sa používa pri vytváraní digitálnych fotografií, čo umožní lepšie pochopiť procesy spojené s digitálnym obrazom.

Hlavným cieľom tejto práce je vytvorenie troch funkčných aplikácií. Súčasťou toho je nutné porozumieť problematike daných tém a nakoniec tiež programovaniu týchto aplikácií v jazyku Javascript. Pre programovanie v Javascripte je vhodné použiť naprogramované knižnice, ktoré uľahčia prácu a tvoria súčasť tejto diplomovej práce.

1 Programovacie jazyky použité pri vytváraní webových aplikácií

V nasledujúcich podkapitolách sú uvedené informácie o programovacích jazykoch, ktoré sa používajú pri programovaní webových stránok a sú použité aj v praktickej časti tejto práce pri programovaní webových appletov.

1.1 HTML

HTML (Hyper text Markup Language) je názov značkovacieho jazyka, ktorý sa používa pri tvorbe obsahu webových stránok. Obsah webových stránok môže byť rôzny, ako napríklad texty, tabuľky, obrázky a iné multimediálne prvky. Webové stránky sú prepojené medzi sebou hypertextovými odkazmi [5, 6].

HTML jazyk patrí medzi hlavné jazyky, pomocou ktorých sa vytvárajú stránky v systéme WWW. Umožňuje na Internete publikovať rôzne dokumenty [5, 6].

Pôvodne vznikol tento jazyk z univerzálneho jazyka SGML (Styandard generalized Markup Launguage), ktorý je veľmi rozsiahly. Jazyk HTML sa neustále vyvíja s vývojom webových prehliadačov a platí to aj naopak, s vývojom webových prehliadačov sa vyvíja jazyk HTML [5, 6].

1.1.1 Verzie jazyka HTML

HTML jazyk prešiel veľkým vývojom od svojho prvého vzniku. V roku 1990 ešte nebol jazyk HTML štandardizovaný. Užívateľom nestačili základné funkcie a prehliadače si vytvárali svoje vlastné rozšírenia HTML, ktorých obsahom bolo formátovanie stránok farebným textom, prípadne vkladanie obrázkov. Tieto rozšírenia neboli kompatibilné medzi prehliadačmi a fungovali len na tom prehliadači, pre ktorý boli vytvorené. Práve kvôli tomu prebehla štandardizácia jazyka v roku 1995. Táto verzia obsahovala hlavne neočíslované a číslované zoznamy, formátovanie textu, pridávanie obrázkov a prvky pre formuláre [7].

Verzia HTML 3.0 nebola štandardizovaná. Bola zverejnená tiež v roku 1995 a nebola podporovaná ani prehliadačmi. V tejto verzií fungovala aj spätná kompatibilita s verziou 2.0. Podporovala kaskádové štýly, tabuľky, matematické výrazy a ďalšie funkcie. Keďže nebola podporovaná prehliadačmi tak v roku 1997 bola schválená verzia 3.2. Táto verzia prebrala veľa funkcií z verzie 3.0 ale niektoré funkcie sa vynechali. Podporovala veľké množstvo funkcií na úpravu textu z dôvodu, že nepodporovala kaskádové štýly. Obsahovala aj funkcie na obtekanie textu, prípadne formátovanie tabuliek a applety [7].

Na konci roku 1997 bola vyvinutá nová verzia HTML 4.0, ktorá bola štandardizovaná až v apríli v roku 1998. Medzi hlavnú funkciu patrila podpora kaskádových štýlov. Táto verzia podporovala aj applety, rôzne fonty a matematické funkcie. Tiež podporovala vytváranie multimediálneho obsahu väčšieho rozsahu. To umožňovalo vytvárať interaktívne stránky [7].

Verzia HTML 4.01 bola len opravou niektorých chýb vyskytnutých vo verzií 4.0. Organizácia W3C, ktorá vyvíjala HTML jazyk sa rozhodla, že už ďalej jazyk nebude vyvíjať, ale sa bude venovať vyvíjaniu jazyku XHTML [5, 7].

Posledná verzia HTML 5.0 bola štandardizovaná v roku 2014. Pracovali na nej spoločnosti Mozilla a Opera a Apple. Nakoniec sa znova pridala aj organizácia W3C. Základom tejto verzie bolo, aby podporovala CSS a Javascript. Táto verzia priniesla veľmi veľa funkcií ako lepšie editovanie formulárov, 2D a 3D grafiku, audio, video, lokálne úložisko, lokálne databázy, webové aplikácie a podobne. Táto verzia sa používa do dnes s malými úpravami vo verziách 5.1, 5.2, 5.3 [5, 7].

1.1.2 Štruktúra jazyka HTML

Každá informácia v tomto jazyku je jasne definovaná - má svoj význam, ktorý určuje konkrétny element (prvok). Každý prvok má začiatok a koniec. To určuje značka (tag). V značka, ktorá je umiestnená na začiatku sa môžu definovať atribúty, ktoré majú svoju hodnotu a príslušný názov. Jedná sa napríklad o atribúty class, id, a pod., v ktorých sa môžu nachádzať aj ďalšie informácie [8].

Vo výpise 1.1 je zobrazená základná šablóna dokumentu HTML. V prvom riadku je určený typ dokumentu. V druhom riadku je určený jazyk, ktorý sa používa v textoch na webových stránkach. Jazyk sa určuje pomocou skratiek: sk, en, cz, de [8].

Štruktúra dokumentu HTML sa delí na hlavičku a telo. Hlavička (head) a telo (body) má svoj začiatok a koniec. V hlavičke je definovaná použitá znaková sada, v tomto prípade je to utf-8. Ďalej na riadku 5 je definovaný názov stránky. V tele dokumentu sa definujú nadpisy rôznych úrovní, textové odstavce a polia a tiež tam môžu byť definované aj iné prvky, ktoré budú viditeľné na webovej stránke.

Výpis 1.1: Ukážka HTML šablóny

```
    <!doctype html>
<html lang="sk">
    <head>
        <meta charset="utf-8">
        <title>Názov web stránky</title>
    </head>
    <body>
        <h1>Nadpis úrovne jedna (Hlavný nadpis)</h1>
        <p>Odstavec textu</p>
    </body>
</html>
```

V tabuľke 1.1 sa nachádza zoznam základných značiek používaných v jazyku HTML. Tabuľka obsahuje aj významy jednotlivých značiek a aj párovosť, čo znamená, či musia byť aj ukončené, alebo nie. Je to len malý prehľad spomedzi veľkého množstva značiek. Veľmi užitočná je napríklad aj značka `div`, ktorá vymedzuje univerzálny blokový kontajner [8].

1.1.3 Programy pre tvorbu kódu a spúšťanie jazyka HTML

Syntax jazyka HTML umožňuje aj bežný textový editor. Umožňujú napríklad farebné rozlišovanie jednotlivých častí kódu, prípadne dokážu pomocou tabulátora napovedať značky. Medzi najznámejšie textové editory, v ktorých sa programujú webové stránky patrí Notepad++, Atom alebo VS Code [5].

V prípade potreby interpretácie kódu sa môže použiť ľubovoľný prehliadač. Postup spracovania kódu sa nazýva parsovanie [5].

1.2 CSS

Kaskádové štýly (CSS) je programovací jazyk pre grafickú úpravu webový stránok. Káskádové štýly, pretože je možné definície štýlov vrstviť, ale platí len tá, čo je posledná. programovací jazyk CSS vznikol v roku 1996. Hlavnou ideou tohto jazyka je oddeliť vzhľad dokumentu od jeho štruktúry a obsahu. Existuje viac verzií, ktoré boli vydané [9, 10].

Prvá verzia CSS 1.0 bola vydaná na konci roku 1996. Obsahovala základné formátovacie prvky pre webové stránky. Verzia 2.0 bola vydaná v máji v roku 1998.

Tab. 1.1: Tabuľka základných značiek v dokumente HTML. Prevzaté a upravené z [8].

Značka	**Význam**	**Párovosť**
<!-- -->	poznámka	ano
b	tučné písmo	ano
i	kurzíva	ano
code	výpis kódu	ano
var	formátovanie premennej	ano
q	citácie	ano
p	odstavec	nepovinné
h1	nadpis 1. úrovne	ano
h2	nadpis 2. úrovne	ano
hr	vodorovná čiara	ano
header	záhlavie stránky	ano
footer	pätička stránky	ano
nav	navigácia	ano
figure	obalenie obrázku s jeho súvisejúcim obsahom	ano
ol	číslovaný zoznam	ano
ul	odrážkový zoznam	ano
img	obrázok	nie
a	hypertextový odkaz	ano
table	tabuľka	ano
form	formulár	ano

Podporovala formátovanie tabuliek a základné určenie pozícií. Verzia 2.1 bola vydaná v roku 2011. Jej cieľ bol opraviť chyby, zlepšiť presnosť a zjednodušiť formátovanie. Je to prvá verzia, ktorú podporuje veľký okruh webových prehliadačov. Následne sú vydané verzie 3 a 4, ktoré ale nie sú konkrétne, ale vyvíjajú sa nezávisle na sebe. Tieto verzie umožňujú animácie, prechody, lepší dizajn. Webové prehliadače podporujú rôzne špecifikácie a ich rozsah, to znamená, že nie sú všetky špecifikácie podporované všetkými webovými prehliadačmi. Verzia CSS 3 je plne kompatibilná s HTML 5.0 [9, 10].

1.2.1 Štruktúra jazyka CSS

Vo výpise 1.2 je zobrazený príklad tvorenia štruktúry jazyka CSS. Na prvom riadku je možné vidieť názov elementu, ktorého formát sa nastavuje. V tomto prípade je to telo dokumentu HTML. Vyhradené miesto formátovania značiek je vymedzené

množinovými zátvorkami. V druhom riadku je zvolený font písma Arial, v prípade nedostupnosti tohto fontu sa použije font sans-serif. v treťom riadku je nastavenie zobrazenia tela dokumentu na flexbox model. V ďalšom riadku je nastavené centrovanie v rámci flexboxu. Piaty riadok určuje centrovanie flexboxu v rámci stránky na stred. V šiestom riadku je nastavenie výšky elementu, čo v tejto štruktúre znamená 100 % výšky viditeľnej obrazovky. V poslednom riadku formátovania tela sa nastavujú predvolené okraje. Ak je zadaná hodnota 0, predvolené okraje okolo elementu sa odstránia. To znamená, že tam nebudú žiadne medzery medzi okrajom stránky a obsahom. Blok štýlov `h2` nastavuje len farbu textu pre nadpisy druhej úrovne na červenú.

Výpis 1.2: Ukážka štruktúry CSS

```
1  body {
2      font-family: Arial, sans-serif;
3      display: flex;
4      align-items: center;
5      justify-content: center;
6      height: 100vh;
7      margin: 0;
8    }
9  h2 {
10     color: red;
11   }
```

V štylizácií dokumentu môžu byť použité rôzne jednotky dĺžky ako percentá, palce, centimetre, štvorčíky. Záleží len na užívateľovi, aké jednotky dĺžky si zvolí [10].

S použitím jazyka CSS sa nedefinujú len fonty písma ale aj veľkosť písma, farby, štylizácia a plno ďalších funkcií [10].

1.3 Javascript

Javascript je objektovo orientovaný programovací jazyk, ktorý sa používa pri tvorbe webových stránok a doplňuje tak jazyky HTML a CSS. Patrí medzi skriptovacie jazyky. Pôvodne sa tento programovací jazyk volal Mocha a neskôr LiveScript. Až pri predstavení verejnosti bol vybratý názov JavaScript, pretože programovací jazyk Java bol v tedy veľmi populárny. Je mu síce veľmi podobný, no nie je vyvíjaný podľa jazyka Java ale vývojári sa inšpirovali jazykom Self. Bol vydaný v roku 1995. Nižšie sú popísané výhody a nevýhody tohto programovacieho jazyka [11].

Výhody [12]:

- objektovo orientovaný – používa objekty webových prehliadačov,
- funguje vo väčšine prehliadačov,
- case sensitive – rozlišuje veľké a malé písmená,
- podobá sa programovacím jazykom Java a C,
- nie je nutná kompilácia.

Nevýhody [12]:

- používateľ môže JavaScript v prehliadači zakázať,
- funguje len v prehliadačoch,
- webové prehliadače majú vlastné modifikácie, čo môže viesť k nezrovnalostiam a chybám,
- používa cookies, ostatné dáta nevie ukladať,
- nevie pristupovať k systémovým objektom.
- potenciálne zneužitie naprogramovaného kódu.

Obecne tento jazyk slúži na tvorbu interaktivity na webových stránkach a na tvorbu príjemnejšieho užívateľského rozhrania [11, 12].

1.3.1 Štruktúra dokumentu

Vo výpise 1.3 je zobrazená ukážka štruktúry dokumentu JavaScript. V tejto ukážke je javascript v súbore `index.html`, no pri programovaní zložitejších stránok je nutné pre javascript vytvoriť ďalšie súbory. Podstatným obsahom tejto ukážky je definovanie premenných na riadku 9 až 12 a následným výpisom na obrazovku v riadkoch 13,14. Tento príklad len jednoducho spočíta počty guličiek a vypíše ich na obrazovku. V riadku 15 je HTML tag, ktorý pridáva prázdny riadok. V riadkoch 16 až 19 sú definované reťazce, názvy farieb a v riadkoch 20 a 21 sú vypísané na obrazovku, kde medzi nimi sú medzery je znak &. Všetky tieto operácia sa robia v tele dokumentu, nie v hlavičke! Skript je nutné vyznačiť, kde začína a kde končí značkami `<script>` a `</script>`, pretože je to párová značka.

Výpis 1.3: Ukážka štruktúra jazyka HTML a Javascript

```
<!DOCTYPE html>
<html lang="sk">
 <head>
        <meta charset="utf-8" />
        <title>Ukážka štruktúry dokumentu</title>
 </head>
 <body>
     <script>
      let modre = 15;
      let cervene = 16;
      let zlte = 5;
      let suma = modre + cervene + zlte;
      document.write("Počet guličiek dohromady je: ");
      document.write(suma);
      document.write("<br />");
      let farba1 = "biela";
      let farba2 = "čierna";
      let farba3 = "hnedá";
      let farby = farba1 + " & " + farba2 + " & " farba3;
      document.write("Spojenie reťazcov farieb: ");
      document.write(farby);
     </script>
 </body>
</html>
```

1.4 Knižnice programovacieho jazyka Javascript

Knižnica v Javascripte je súbor, ktorý obsahuje veľké množstvo funkcií. Tieto funkcie stačí aplikovať vo svojom webovom projekte. Niektoré knižnice sú voľne dostupné, iné sú spoplatnené, nachádzajú sa v databázach. Pre prístup k nim, je nutné použiť absolútny odkaz URL, aby obsahoval presnú cestu. Ďalší možný postup je, že sa knižnica stiahne do počítača a dá sa otvoriť pomocou lokálnej cesty k nej [13].

Najpoužívanejšie knižnice, ktoré sa dajú využiť pri tvorbe tejto semestrálnej práce sa zameriavajú na počítačovú grafiku a nižšie budú vymenované a popísané [13].

Prvou knižnicou je `Tree.js`. Je to knižnica, ktorá funguje vo viacerých prehliadačoch. Slúži na zobrazenie 3D počítačovej grafiky za pomoci knižnice WebGL (Web

Graphics Library) vo webovom prehliadači. Taktiež je možné vytvárať trojrozmerné počítačové animácie a zobrazovať ich v prehliadači. Zdrojový kód tejto knižnice je uložený v databáze GitHub [14].

Ďalšou knižnicou je najobsiahlejšia knižnica `jQuery`, ktorá je vytvorená pre jednoduchšiu manipuláciu so stromovou štruktúrou kódu. Nazýva sa to aj DOM (Document Object Model). Knižnica tiež umožňuje vytvárať rôzne animácie, efekty a widgety na veľmi vysokej úrovni. Firma Microsoft zahrnula túto knižnicu aj do programu Visual Studio. Táto knižnica začala byť populárna už v roku 2015, kde až 62,7 % stránok z jedného milióna použilo práve túto knižnicu. Dnes ju používa niečo cez 80 % webových stránok. Podporujú ju prehliadače Firefox, Safari, Edge a aj Chrome [15].

Nasledujúcou veľmi zaujímavou knižnicou je `Verge3D`, ktorá slúži na vytváranie 3D interaktívnych prvkov s užívateľom v prehliadačoch. Podporuje grafiky ako Blender a Maya. Na vykresľovanie grafických prvkov používa WebGL ako aj knižnica `Tree.js`, z ktorej používa aj nejaké časti a funkcie [16].

Poslednou knižnicou, ktorú je dobré spomenúť je knižnica `D3.js` (Data-Driven Documents). Podporuje vektorovú grafiku (SVG), HTML5 a CSS. Používa sa na interaktívne, dynamické zobrazovanie grafických prvkov prípadne na pridávanie rôznych prechodov a efektov [17].

Toto je len pár knižníc, ktoré boli vybraté z desiatok ďalších. Záleží na užívateľovi, ktorú si vyberie a aplikuje vo svojom projekte [13].

2 Parametrická definícia krivky v 3D priestore

V teoretickej časti sú zhrnuté definície pojmov, ktoré sa využívajú pri vytváraní a spracovaní obrazových signálov. Prvá časť obsahuje definície a rozdelenie vyjadrení kriviek a ich matematický popis. Následne je podrobne rozobraté parametrické vyjadrenie krivky, spojitosti a modelovanie kriviek.

V ďalšej časti je zobrazený popis jazykov HTML a CSS využitých v tejto diplomovej práci a popis možnosti výberu knižníc v jazyku Javascript. Tieto knižnice slúžia na uľahčenie programovania matematickej a grafickej časti aplikácií.

V poslednej časti tejto kapitoly sa nachádza praktická časť, ktorá ukazuje programovanie appletu s podrobným popisom.

2.1 Krivky, ich rozdelenie a možnosti napojenia

Existujú tri druhy vyjadrenia krivky, ktoré sa používajú pri ich popise:

1. explicitné,
2. implicitné,
3. parametrické.

V počítačovej grafike sú krivky vyobrazené ako sústava parametrov nejakej rovnice (parametrické vyjadrenie), ktorá je následne vyobrazená [1].

2.1.1 Vyjadrenie krivky explicitne

Explicitná krivka má tvar funkcie, ktorý je následovný:

$$y = f(x). \tag{2.1}$$

V 3D má zvyčajne tvar analytickej rovnice:

$$z = f(x, y),\ z = g(x, y). \tag{2.2}$$

Táto rovnice popisuje povrch, kde hodnota `z` je daná funkciou dvoch premenných `x,y`. Plocha môže byť vyjadrená v rôznych smeroch, pričom hodnota `z` narastá alebo klesá v závislosti od hodnôt `x,y`. Znamená to, že pre každú dvojicu hodnôt `x,y` z definičného oboru funkcie zodpovedá práve jedna hodnota `z` [1].

2.1.2 Vyjadrenie krivky implicitne

Implicitná krivka je vyjadrená ako

$$F(x, y, z) = 0,\ G(x, y, z) = 0. \tag{2.3}$$

Zadanú krivku v tomto tvare je ťažko zobraziť v porovnaní vyjadrenia krivky explicitne alebo parametricky. Ďalším dôležitým poznatkom je, že takto vyjadrená krivka neumožňuje postupný výpočet [1].

Napríklad kružnica so stredom $S = [0, 0, 0]$ a polomerom r, ktorá leží v rovine x,y má implicitné vyjadrenie:

$$x^2 + y^2 + r^2 = 0,\ z = 0. \tag{2.4}$$

Tento tvar rovníc vyjadruje kružnicu ako prienik dvoch plôch. Rotačná valcová plocha, ktorá má rovnicu $x^2 + y^2 + r^2 = 0$. Druhá plocha vyjadruje vlastne rovinu a tou je rovnica $z = 0$ [1].

Implicitne zadaná krivka nie je veľmi vhodná pre prácu v počítačovej grafike, pretože postupný výpočet krivky neumožňuje [1].

2.1.3 Vyjadrenie krivky parametricky

Parametrické vyjadrenie krivky sa najčastejšie používa práve v počítačovej grafike. Parametrickému vyjadreniu sa rozumie dráha pohybujúceho sa bodu, ktorého parametre (súradnice) sú funkciami času, čo vyjadruje parameter t. Parametrické zadanie krivky má tvar [1, 2]:

$$x = x(t),\ y = y(t),\ z = z(t). \tag{2.5}$$

Z rovníc 2.5 je možné vyjadriť ako vektorovú 2.6, tak aj bodovú rovnicu 2.7 krivky [1, 2]:

$$\vec{v}(t) = (x(t), y(t), z(t)), \tag{2.6}$$

$$V(t) = [x(t), y(t), z(t)]. \tag{2.7}$$

Veľkosť polohového vektora $\vec{v}(t)$ sa rovná vzdialenosti bodu $V(t)$ od začiatku. Ako bolo spomenuté vyššie, parametrický zápis závisí na parametre t, ktorý vyjadruje čas. Tento spôsob umožňuje popísať rôzne tvary kriviek a trajektórií vo viacrozmernej (napr. 3D) sústave. Krivka tiež môže v priestore viackrát prechádzať rovnakými bodmi a to v rôznych časových okamžikoch, môže byť uzavretá alebo sa môže krížiť [1, 2].

2.1.4 Naviazanie kriviek

Pomocou derivácií krivky, ktorá je vyjadrená parametricky, je možné vyjadriť dotyčnicový vektor v bode $V(t_0)$ po jednotlivých zložkách x', y', z' [1, 2]:

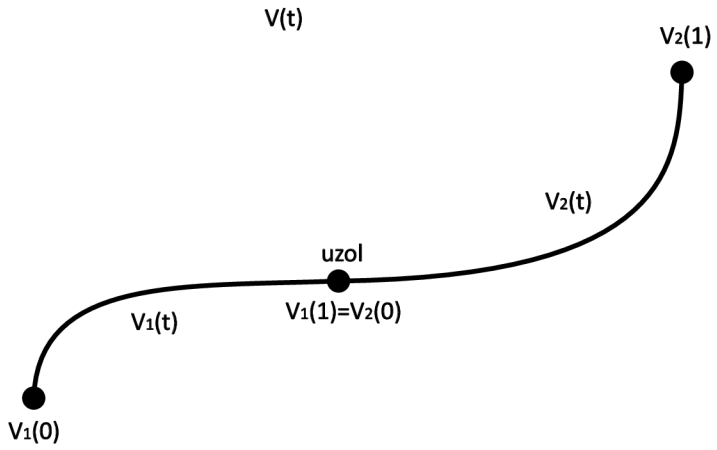
$$\vec{v'}(t_0) = (x'(t_0),\ y'(t_0),\ z'(t_0)) = (\frac{\mathrm{d}x(t_0)}{\mathrm{d}t},\ \frac{\mathrm{d}y(t_0)}{\mathrm{d}t},\ \frac{\mathrm{d}z(t_0)}{\mathrm{d}t}). \tag{2.8}$$

Ďalším dôležitým pojmom je rovnica dotyčnice v bode, v ktorom sa dotýka s krivkou. Tento parameter je možné vypočítať pomocou bodu na krivke a dotyčnicového vektoru, ktorý sa nazýva aj ako smerový vektor priamky. Rovnica dotyčnice v bode sa vypočíta ako:

$$P(u) = V(t_0) + \vec{v'}(t_0). \tag{2.9}$$

Táto rovnica sa využíva hlavne pri naviazaní a skladaní kriviek do rôznych (zložitejších) tvarov z jednoduchých segmentov [1, 2].

Na obrázku 2.1 je nakreslená krivka $V(t)$, ktorá sa skladá z dvoch častí. Tieto časti sú označené $V_1(t)$ a $V_2(t)$ a môžu byť definované buď polynomiálne alebo inými spôsobmi, ktoré sú popísané v kapitole 2.1.7. Sú spojené v bode $V_1(1) = V_2(0)$, ktorý sa v odbornej literatúre nazýva uzol [1, 2].



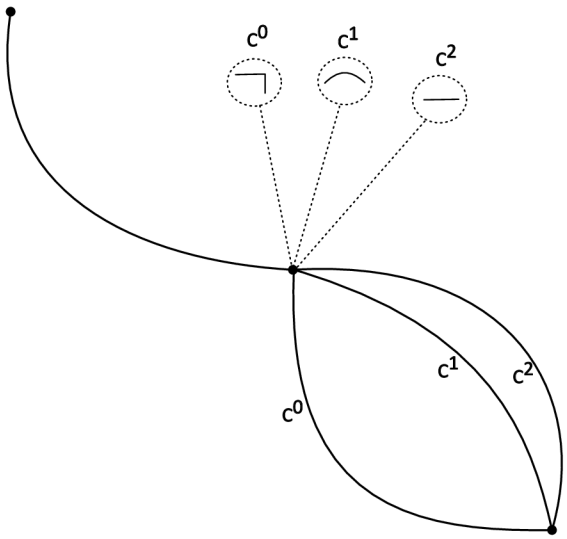
Obr. 2.1: Vytvorená krivka napojením dvoch segmentov

2.1.5 Parametrická spojitosť kriviek

Krivky môžu byť spojité v rôznych stupňoch. Označujú sa triedami C^n, toto označenie sa volá parametrická spojitosť stupňa n, kde n naberá hodnoty $n = 0, 1, 2, \ldots, \infty$. Znamená to, že majú spojité derivácie podľa parametru času t do rádu n vo všetkých bodoch a krivka sa vtedy volá spojitá [1, 2].

Krivka je hladká v prípade, že sú spojité aj jej prvé derivácie vo všetkých bodoch. Pri vyšších deriváciách sa krivka označuje spojitosťou druhého, tretieho a vyššieho rádu, prípadne obecne pomocou n-tého rádu. V obrázku 2.2 sú zobrazené spojitosti C^0, C^1, C^2. Ak je koncový bod prvej časti počiatočným bodom druhej časti, má krivka napojenie triedy C^0, hodnoty prvých derivácií môžu byť rôzne. Ak je dotyčnicový vektor v koncovom bode časti V_1 rovný v počiatočnom bode dotyčnicovému vektoru V_2, majú segmenty napojenie C^1, znamená to, že krivka musí mať prvú

deriváciu spojitú v tomto bode. Z toho vyplýva, že pre napojenie C^2 musí byť splnená rovnosť vektorov prvej aj druhej derivácie (musí byť spojitá prvá aj druhá derivácia) [1, 2].



Obr. 2.2: Krivka a jej spojitosti C^0, C^1, C^2

2.1.6 Geometrická spojitosť

Geometrická spojitosť sa označuje ako G^n, kde n určuje zase triedu spojitosti. Touto spojitosťou sa určuje hladkosť napojenia kriviek. Ak sú dve časti krivky $V(t)$ spojité pomocou spojitosti G^0, tak to znamená že majú spoločný počiatočný bod V_1 s koncovým bodom V_2. Spojitosť G^0 vyjadruje, že krivky nemajú ostré hrany a nemusia byť spojité v prvej derivácií. Ak sú krivky G^0 spojité a zároveň ich dotyčnicové vektory $\vec{v_1'}(0)$ a $\vec{v_2'}(1)$ súhlasne kolineárne, tzn. že sú minimálne rovnobežné alebo súhlasné, kde nezáleží na smere, tak majú spojitosť G^1. Označuje to, že majú spojitú prvú deriváciu v spojovacom mieste, čo vytvára plynulý prechod. Spojitosť G^2, musí mať nie len spojitosť prvej derivácie, ale aj druhej. Krivka má vtedy konzistentné (plynulé) zakrivenie na oboch stranách spojovacieho miesta [1, 2].

Geometrická spojitosť nevyjadruje spojitosť dotyčnicových vektorov, ale totožnosť dotyčníc. Zároveň geometrická spojitosť nezávisí na parametri t, z čoho vyplýva, že bod v uzle môže meniť rýchlosť ale nemôže rázne meniť smer [1, 2].

Krivky, ktoré majú spojitosť C, tak majú aj spojitosť G. Neplatí to ale opačne, krivky so spojitosťou G nemusia mať spojitosť C [1, 2].

2.1.7 Polynomiálne krivky

Polynomiálne krivky patria medzi najpoužívanejšie druhy vyjadrenia parametrických kriviek práve v počítačovej grafike. Tieto krivky majú tvar polynómu vzhľadom k parametru alebo premennej. Sú definované ako:

$$\mathbf{V}_n(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix} = \begin{pmatrix} a_{0,x} + a_{1,x}t + a_{2,x}t^2 + \cdots + a_{n,x}t^n \\ a_{0,y} + a_{1,y}t + a_{2,y}t^2 + \cdots + a_{n,y}t^n \\ a_{0,z} + a_{1,z}t + a_{2,z}t^2 + \cdots + a_{n,z}t^n \end{pmatrix}, \tag{2.10}$$

kde t je parameter a $a_0, a_1, a_2, ..., a_n$ sú koeficienty. Medzi hlavné výhody patrí, že s týmito krivkami sa dá veľmi ľahko manipulovať, pretože sú diferencovateľné a dajú sa veľmi ľahko vyčísliť. Môžu sa tiež skladať do zložitejších kriviek pomocou tzv. po častiach polynomiálnych [1, 3].

Jedným z bežných typov týchto kriviek sú kubické krivky, čo sú polynomiálne krivky tretieho stupňa. Sú veľmi využívané, pretože umožňujú modelovať rôzne tvary a ľahko sa s nimi dosiahne požadovaná spojitosť včetne spojitosti C^2 [1, 3].

Pri modelovaní kriviek sa často zadáva len niekoľko riadiacich bodov (riadiacich polygónov) a matematický aparát na základe ich polohy určí daný priebeh krivky. Niektoré programy umožňujú zadávať krivky pomocou dotyčnicových vektorov, čo uľahčuje manipuláciu s krivkami. Tiež to zaistí spojitosť a hladkosť pri napojení kriviek [1, 3].

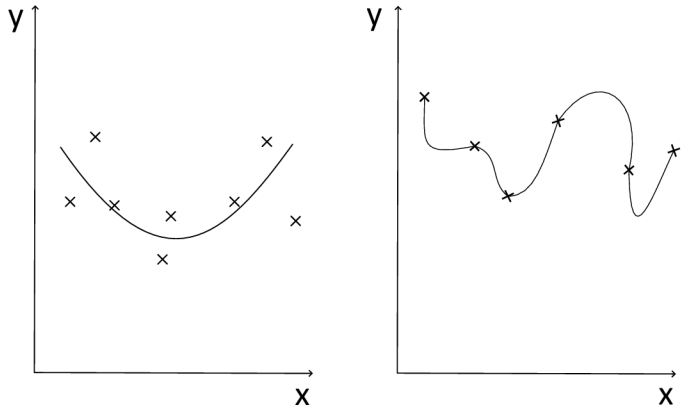
Na obrázku 2.3 je zobrazený rozdiel určovania riadiacich bodov pomocou aproximácie a interpolácie. Z obrázku vyplýva, keď vytvárame krivku aproximačne, je síce bodmi určený tvar krivky, ale krivka nemusí prechádzať týmito bodmi. Pri generovaní krivky pomocou interpolácie musí krivka prechádzať týmito bodmi. Medzi najznámejšie interpolačné krivky patria Hermitovské krivky alebo Hermitovské kubiky. V tejto práci interpolačné krivky rozoberané nebudú [1, 3].

2.1.8 Bézierove krivky

Bézierove krivky sú matematickými krivkami a sú často používané v počítačovej grafike, designu alebo pri animáciách (modelovaní pohybu). Tiež sa využívajú aj pri modelovaní písma (fontov). Prvý krát ich použili Pierre Bézier a Paul de Casteljau, v automobilovom priemysle. Po Bézierovy sú pomenované, pretože ako prvý publikoval výsledky svojej práce. Patria medzi najpoužívanejšie aproximačné krivky a pomocou nich sa modeluje ako v 2D rozmeroch, tak aj v 3D [1, 4].

Bézierová krivka stupňa n je definovaná matematicky vzorcom:

$$B(t) = \sum_{i=0}^{n} P_i \cdot B_i^n(t), \tag{2.11}$$



Obr. 2.3: Rozdiel medzi aproximačne (vľavo) a interpolačne (vpravo) určenou krivkou

kde je krivka určená $n+1$ bodmi riadiaceho polygónu P_i a $B_i^n(t)$ sú Bézierove bázické funkcie (Bernsteinove polynómy) stupňa n a parametru t. Medzi základné vlastnosti týchto kriviek patrí, že ak sa zmení jediný riadiaci bod P_i, tak sa zmení celý tvar krivky. Z tohto dôvodu sa krivky rozdeľujú na menšie časti (segmenty) nižšieho stupňa, a potom sa napájajú. Bernsteinové polynómy slúžia ako základné polynómy, ktoré vážia riadiace body, a tým definujú tvar Bézierovej krivky. Bernsteinove polynómy sú definované ako:

$$B_i^n(t) = C_n^i \cdot (1-t)^{n-i} \cdot t^i; t \in \langle 0,1 \rangle; i = 0,1,...,n. \tag{2.12}$$

Binomický koeficient C_n^i je určený vzorcom:

$$C_n^i = \binom{n}{i}, \tag{2.13}$$

v ktorom je $\binom{n}{0} = 1$ a $0^0 = 1$ [1, 4].

Dotyčnicové vektory v počiatočnom a konečnom bode sa dajú ľahko vypočítať zo vzorca 2.11 deriváciou a dosadením $t = 0$ a $t = 1$. Vzťah pre hraničné (krajné) dotyčnicové vektory bude vypadať následovne [1, 4]:

$$\vec{q}'(0) = n(P_1 - P_0), \tag{2.14}$$

$$\vec{q}'(1) = n(P_n - P_{n-1}). \tag{2.15}$$

2.1.9 Algoritmus de Casteljau

Tento algoritmus je obecne numerická metóda, ktorá slúži na vykresľovanie a vyhodnocovanie Bézierových kriviek. Umožňuje krivku rozdeliť na časti a pre daný parameter t sa priblížiť k hodnote krivky. Funguje to tak, že pomocou algoritmu sa

postupne interpoluje pomocou lineárnych kombinácií medzi bodmi a vytvárajú sa tak nové body až k dosiahnutiu konečného bodu na krivke [4].

Ak sa hľadá bod na Bézierovej kubike a je daný predpoklad, že parameter t je z intervalu $0 \leq t \leq 1$, je možné previesť lineárnu interpoláciu pomocou obecných vzorcov:

$$P_0^1(t) = (1-t)P_0 + tP_1, \tag{2.16}$$

$$P_1^1(t) = (1-t)P_1 + tP_2, \tag{2.17}$$

$$P_2^1(t) = (1-t)P_2 + tP_3. \tag{2.18}$$

Tieto vzorce udávajú prvý krok interpolácií. V ďalšom kroku sa pokračuje interpoláciou s rovnakým parametrom t pomocou vzorcov:

$$P_0^2(t) = (1-t)P_0^1 + tP_1^1, \tag{2.19}$$

$$P_1^2(t) = (1-t)P_1^1 + tP_1^2. \tag{2.20}$$

V poslednom kroku sa určí posledná interpolácia taktiež s rovnakým parametrom t,

$$P(t) = P_0^3(t) = (1-t)P_0^2 + tP_1^2, \tag{2.21}$$

kde už je výsledkom hľadaný bod na Bézierovej kubike. Najvyšší stupeň parametru t je v tomto prípade 3, tak sa jedná o kubickú Bézierovu krivku, ktorá je rozobratá v podkapitole 2.1.10. Z opakovaného výpočtu interpolácií vyplýva, že sa vytvára trojuholníková schéma výpočtov [4].

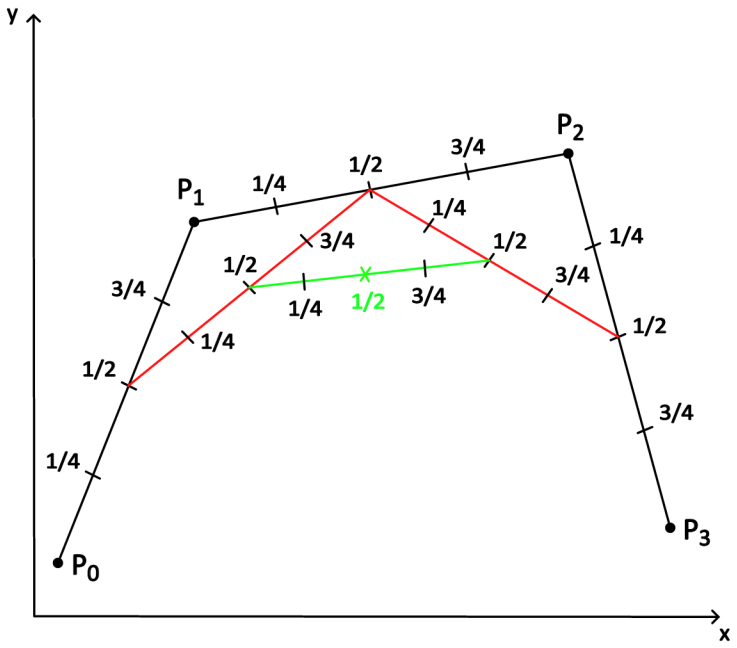
Na obrázku 2.4 je zobrazený grafický výpočet hľadaného bodu na krivke podľa hodnoty parametru $t = 1/2$. V prvom kroku sa rozdelia úsečky na štyri časti. V druhom kroku sa spoja body na úsečkách v čase $t = 1/2$ (červené úsečky). Následne sa opakuje prvý a druhý krok znova, pokiaľ nebude len jedna úsečka, na ktorej sa nachádza hľadaný bod (zelená farba).

V applete pre ukážku parametrickej definície krivky v 3D priestore je použitá trieda `THREE.CubicBezierCurve3` z knižnice `Three.js` na vytvorenie a vykreslenie kubických Bézierových kriviek. Táto trieda vypočítava body na krivke pomocou ekvivalentného algoritmu. Tento algoritmus počíta ale s tretími mocninami. Algoritmus de Casteljau počíta s prvými mocninami.

2.1.10 Bézierove kubiky

Beziérová kubika v 3D priestore je definovaná nasledujúcim vzorcom, ktorý plynie z Bernsteinových polynómov 2.12:

$$B(t) = (1-t)^3P_0 + 3(1-t)^2tP_1 + 3(1-t)t^2P_2 + t^3P_3, \tag{2.22}$$



Obr. 2.4: Ukážka výpočtu bodu pomocou Algoritmu de Casteljau

kde P_0, P_1, P_2, P_3 sú riadiace body v 3D priestore a `t` je parameter v intervale od 0 do 1.

Každý riadiaci bod P_i má tri súradnice (x_i, y_i, z_i). Pre jednotlivé súradnice platia nasledovné vzorce:

$$x(t) = (1-t)^3 x_0 + 3(1-t)^2 t x_1 + 3(1-t)t^2 x_2 + t^3 x_3, \tag{2.23}$$

$$y(t) = (1-t)^3 y_0 + 3(1-t)^2 t y_1 + 3(1-t)t^2 y_2 + t^3 y_3, \tag{2.24}$$

$$z(t) = (1-t)^3 z_0 + 3(1-t)^2 t z_1 + 3(1-t)t^2 z_2 + t^3 z_3. \tag{2.25}$$

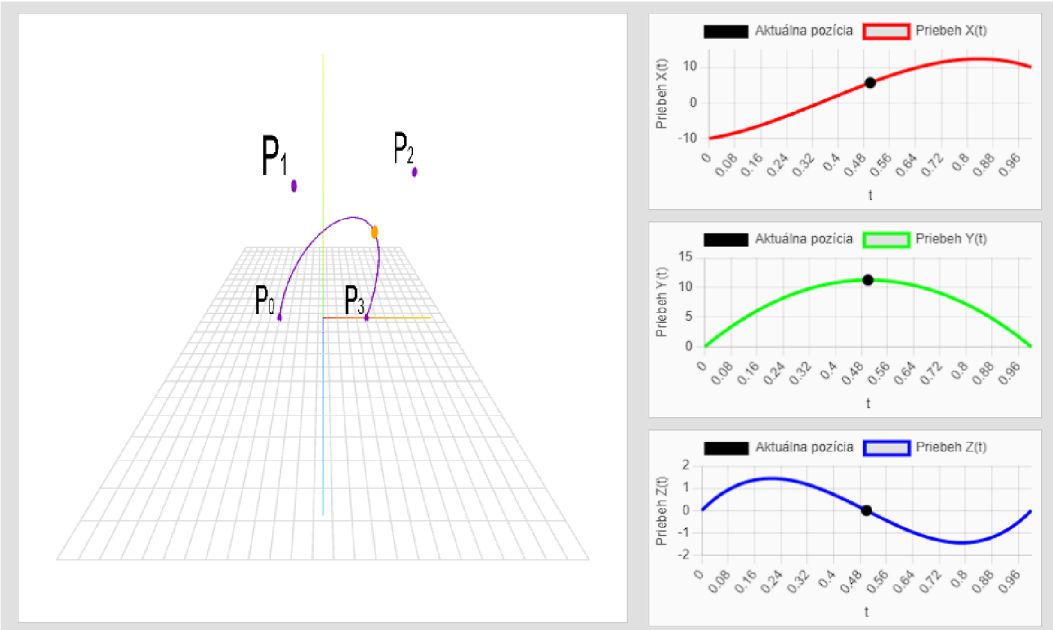
Každá súradnica sa dá vypočítať samostatne podľa vyššie uvedených vzorcov.

2.2 Applet - parametrická definícia krivky v 3D priestore

V tejto podkapitole je postupne rozobratá štruktúra appletu. Nižšie sú opísané jednotlivé časti aplikácie z hľadiska programovacích jazykov HTML, CSS a Javascript. Tiež je tu zdôvodnený výber Javascriptových knižníc a ich použitie v applete na ukážku parametrickej definície krivky v 3D priestore.

Východzie zobrazenie aplikácie je ukázané v obrázku 2.5, kde je vidieť vľavo 3D graf s pevne zadanou krivkou a jej riadiacimi bodmi P_0 až P_3. Na krivke sa nachádza aj oranžový bod, ktorým sa dá hýbať pomocou posuvníku, ktorý označuje parameter

`t` zobrazený v obrázku 2.6. 3D graf je možné otáčať myšou a približovať, prípadne oddiaľovať pomocou kolieska myši.



Obr. 2.5: Applet pre ukážku definície parametrickej krivky v 3D priestore

Vedľa 3D grafu sa v strede nachádzajú 2D grafy, ktoré zobrazujú jednotlivé priebehy `x(t)`, `y(t)` a `z(t)`, ktoré sú popísané v podkapitole 2.1.10. Pohybom posuvníka sa zároveň pohybujú aj body čiernej farby v týchto grafoch.

V prípade potreby sú pod posuvníkom dve tlačidlá. Tlačidlo „Náhodná krivka“ slúži na vytvorenie novej náhodnej krivky, ktorá sa nakreslí v 3D grafe a tiež sa obnovia priebehy v 2D grafoch. Tlačidlo „Riadiaci polygon“ zobrazí riadiace polygony v 3D grafe, čo určí ako riadiace body po sebe nasledujú.

2.2.1 Štruktúra súboru HTML

Súbor `index.html` má základnú štruktúru, ktorá je rozdelená na hlavičku a telo dokumentu.

V hlavičke je definované nastavenie kódovania znakov a responzívne zobrazenie stránky na rôznych zariadeniach. Značka `title` určuje názov stránky v prehliadači. V hlavičke je pripojený súbor `style.ccs`, v ktorom je nastavená štylizácia stránky. Takisto je tam pripojená aj knižnica `Chart.js` z CDN.

V tele dokumentu sa už nachádza samotná štruktúra stránky. Je tam definovaný nadpis úrovne `h1`. Následne sú definované kontajnery a prvky:

- `content-container` ako hlavný kontajner, v ktorom sa nachádzajú ďalšie kontajnery,

Obr. 2.6: Posuvník pre parameter `t`

- `curve-container` pre vizualizáciu 3D krivky,
- posuvník, ktorý umožňuje meniť parameter `t`,
- tlačidlá, pre generovanie náhodnej krivky a pre zobrazenie riadiaceho polygonu,
- textový blok pre vysvetlenie funkcie a ovládania appletu.

Pod týmito prvkami sa nachádza textová sekcia, kde je definovaná teória k pochopeniu významu appletu. Pod textovou sekciu je pripojený súbor `script.js`, ktorý obsahuje jednotlivé funkcie pre interaktivitu stránky.

2.2.2 Programovanie CSS

Celému telu stránky je pridelená rodina písma `Arial` a `sans-serif`. Rodina písma `sans-serif` sú bezpätkové písma a je to definované z dôvodu, ak prvé písmo by nebolo dostupné. Nastavené je nulové odsadenie od okrajov, šedé pozadie a `flexbox` rozloženie stránky do stĺpcov.

Jednotlivé kontajnery používajú taktiež `flexbox` pre ľahkú manipuláciu a pre responzívne rozloženie stránky. Každý segment stránky ako tlačidlo, posuvník, kontajnery, majú definované štýly zamerané na zvýraznenie funkčných oblastí.

2.2.3 Použité knižnice Javascript

Prvou použitou knižnicou je `Chart.js` na vykreslenie grafov pre jednotlivé zložky parametricky definovanej krivky `x(t)`, `y(t)` a `z(t)`. Grafy pomáhajú vizualizovať, ako sa menia hodnoty `x`, `y` a `z` v závislosti na parametri `t`. Práve táto knižnica je použitá z dôvodu, že umožňuje veľmi jednoduchú integráciu a manipuláciu s prvkami. Pri týchto grafoch mohli byť využité aj iné knižnice ako `D3.js, Plotly.js`, s rovnakými funkciami ale z hľadiska grafickej stránky je najlepšia pre manipuláciu s grafmi. Medzi ďalšiu výhodu patrí, že je to open-source knižnica [29].

Druhou požitou knižnicou je `Three.js`. Táto knižnica sa v applete využíva na manipuláciu s 3D grafickým prostredím, kde sa vizualizuje parametrická krivka pomocou riadiacich bodov. Knižnica umožňuje ovládanie prvkov v scéne napríklad pomocou modulu `OrbitControls`, ktorý slúži pre otáčanie kamery okolo scény a približovanie či oddiaľovanie scény. `Three.js` knižnica je použitá z dôvodu, že využíva `WebGL`. Práca so samotnou knižnicou `WebGL` je zložitá a obsahuje dlhšiu syntax. Naopak `Three.js` knižnica sa oveľa jednoduchšie aplikuje pri vytváraní 3D scén a má kratšiu a prehľadnejšiu syntax. Medzi ďalšie výhody patrí aj kompatibilita s rôznymi prehliadačmi a má veľmi prehľadnú a dobre spísanú dokumentáciu a následné príklady použitia [30].

2.2.4 Programovanie v jazyku Javascript

Na začiatku javascriptového súboru `script.js` je importovaná knižnica `three.js` s modulom `OrbitControls` pre pohyb v 3D grafe. Pod importom sú zadefinované pomocné premenné, ktoré sa využívajú v programe ďalej.

Najprv je zadefinovaná funkcia `initScene`, ktorá vytvára scénu, kameru a render. Pridáva pomocnú mriežku v 3D grafe a tiež aj vykresľuje osy pre lepšiu orientáciu. Následne sa spustí animačná slučka na vykresľovanie scény.

Funkcia `createTextSprite` vytvára 2D text v 3D grafe. To znamená, že vytvára popisky riadiciach bodov v 3D grafe. Využíva sa na vykreslenie textu 2D canvas, ktorý sa prevedie na textúru a aplikuje sa na objekty v 3D. Je možné s touto textúrou voľne pohybovať a otáčať ju v 3D priestore pomocou úpravy kódu.

Pri prvotnom zobrazení stránky funkcia `initDefaultCurve` inicializuje predvolenú krivku s konkrétnymi riadiacimi bodmi. Najprv sa inicializujú body, potom sa vytvoria 3D objekty, pridá sa popis riadiacich bodov, vytvorí sa krivka pomocou riadiacich bodov a nakoniec sa umožní pohybovať bodom po krivke pomocou posuvníku umiestneného na stránke. Nakoniec je zadefinované čistenie scény, ktoré odstraňuje predchádzajúce body a ďalšie objekty z 3D scény.

Úlohou funkcie `createRandomCurve` je odstránenie predchádzajúcej krivky, riadiacich bodov a ich popisiek a vytvára nové riadiace body, novú krivku a aktualizuje

všetky grafy. Nové riadiace body sa v každom smere generujú z rozsahu od -20 do 20, aby bola krivka vždy celá viditeľná.

Funkcia `updateSpherePosition` má za úlohu počítať polohu na krivke v závislosti od parametra `t` a aktualizuje polohu objektu (bodu) na krivke. Poloha bodu závisí na polohe posuvníku.

Funkcia `createPlot` vytvára za pomoci knižnice `chart.js` 2D grafy, ktoré zobrazujú priebehy jednotlivých zložiek krivky. Aj v týchto grafoch sa mení poloha bodu na základe polohy posuvníku.

Aby bolo umožnené vykresliť riadiace polygony (úsečky) medzi bodmi, tak ako po sebe nasledujú, tak na to je definovaná funkcia `drawControlPolygon`, ktorá vytvára tieto úsečky v 3D grafe. Túto vyššie spomenutú funkciu doplňuje ďalšia funkcia `hideControlPolygon`, ktorá odstraňuje riadiace polygony zo scény. Je definovaná preto, lebo pomocou tlačidla môžme tieto polygony v 3D grafe zapínať a vypínať.

Ďalšia časť programu `window.onload` sa stará o prvotnú inicializáciu stránky, kde sa zavolajú funkcie `initScene` a `initDefaultCurve`, ďalej sa zavolá funkcia na vykreslenie 2D grafov a tiež aj sa nakonfiguruje posuvník na predvolenú hodnotu 0,5 a s tým aj body na krivkách grafov. Obsahom tejto funkcie je aj definovanie tlačidiel, aby boli funkčné.

Posledné funkcie `updateCharts` a `updatePlot` aktualizujú 2D grafy. Slúžia na to, aby bolo možné pohybovať pomocou posuvníku bodmi po krivkách, aby sa vždy aktualizovala nová hodnota.

3 Barycentrické súradnice trojuholníka v 2D priestore

Barycentrické súradnice trojuholníka alebo súradnice oblasti trojuholníka vynašiel August Ferdinand Möbius už v 19. storočí. Týmito súradnicami je možné nájsť napríklad stred trojuholníka I, ťažisko trojuholníka T a podobne. Taktiež sa ich pomocou riešia integrály alebo sa používajú v Gaussových tabuľkách, ktoré sa prezentujú vo forme súradníc na ploche [18].

3.1 Základné vzťahy

Nech je daný trojuholník ABC a čísla $\lambda_1, \lambda_2, \lambda_3$. Tieto čísla sa po sčítaní rovnajú 1. Následne je označený pod P tak, že [19]:

$$P(\lambda_1, \lambda_2, \lambda_3) = \lambda_1 A + \lambda_2 B + \lambda_3 C = (\lambda_1 + \lambda_2 + \lambda_3)A + \lambda_2(BA) + \lambda_3(C - A) \quad (3.1)$$

$$P(\lambda_1, \lambda_2, \lambda_3) = A + \lambda_2(B - A) + \lambda_3(C - A). \quad (3.2)$$

Čísla $(\lambda_1, \lambda_2, \lambda_3)$ je možné označiť za barycentrické súradnice bodu P vzhľadom na trojuholník ABC. Miesto označenia bodov $\lambda_1, \lambda_2, \lambda_3$ sa zvykne používať aj označenie α, β, γ [18, 19].

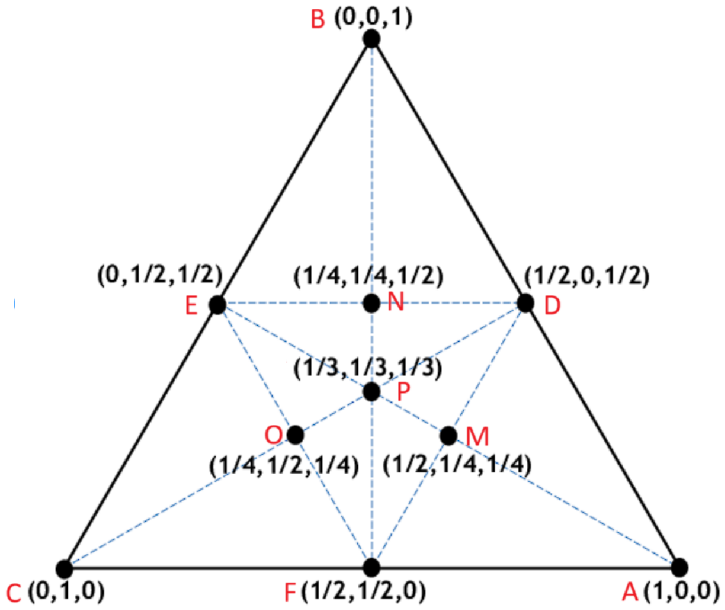
Je veľmi dôležité si uvedomiť, že súradnice ležiace vo vnútri trojuholníka sú kladné. Súradnice záporné ležia určite mimo trojuholníka [19].

3.1.1 Príklad určenia barycentrických súradníc

Na obrázku 3.1 je vyobrazený výpočet barycentrických súradníc rovnostranného trojuholníka ABC so súradnicami $A(1,0,0)$, $B(0,0,1)$, $C(0,1,0)$. Ťažisko trojuholníka je bod P so súradnicami $P(1/3,1/3,1/3)$. Body D, E, H majú jednu súradnicu nulovú. Znamená to, že ležia na priamkách. Ako bolo spomenuté vyššie, súradnice ležiace v trojuholníku musia byť kladné. To dokazujú body M, N, Q, ktoré majú kladné všetky súradnice [18, 19].

Okrem určenia barycentrických súradníc podľa vzorca 3.1 a 3.2 sa zvyknú tieto súradnice určovať aj podľa vypočítania obsahov určitých čiastočných trojuholníkov. Súradnice bodu $P(\lambda_1, \lambda_2, \lambda_3)$ v trojuholníku ABC sa dajú určiť pomocou troch výpočtov, pre ktoré platí [18, 19]:

$$\lambda_1 = \frac{S_{\Delta_{\mathrm{BCP}}}}{S_{\Delta_{\mathrm{ABC}}}},\ \lambda_2 = \frac{S_{\Delta_{\mathrm{ACP}}}}{S_{\Delta_{\mathrm{ABC}}}},\ \lambda_3 = \frac{S_{\Delta_{\mathrm{BBP}}}}{S_{\Delta_{\mathrm{ABC}}}}. \quad (3.3)$$



Obr. 3.1: Barycentrické súradnice trojuholníka. Prevzaté a upravené z [20].

Vzhľadom na túto vlastnosť sa dá konštatovať, že v každom obecnom trojuholníku ťažisko T rozdeľuje trojuholník na tri menšie trojuholníky s rovnakým obsahom ako je zobrazené na obrázku 3.2. Tento spôsob výpočtu má viac využití. Barycentrické súradnice v jednorozmernom priestore vyjadrujú pomer vzdialenosti. V trojrozmernom priestore zase pomer objemov štvorstenov alebo tetraédrov (je to mnohosten, ktorý má 4 steny tvorené z trojuholníkov). Práve z tohto dôvodu sa barycentrickým súradniciam hovorí aj, že sú to plošné súradnice [19].

3.1.2 Barycentrická sústava versus karteziánska sústava

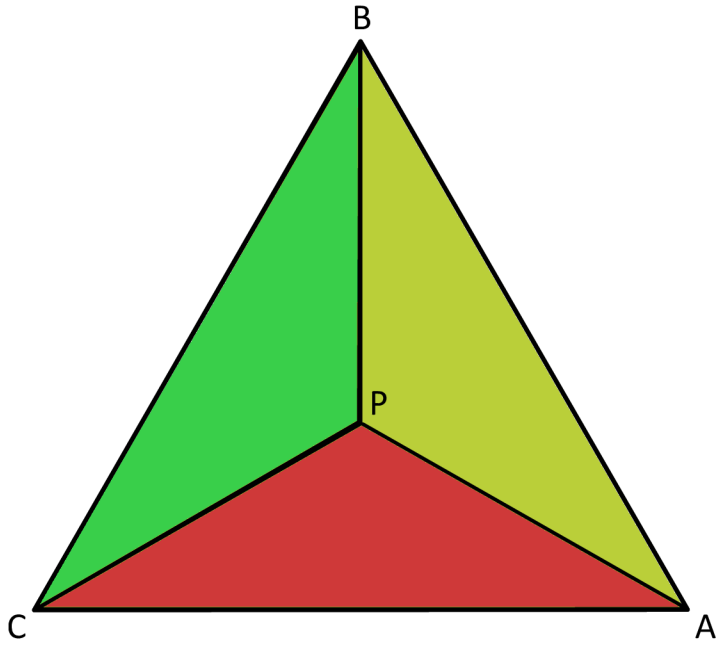
Pre ľubovoľný bod $R(r_1, r_2, r_3)$ v rovine trojuholníku sa dajú vypočítať karteziánske súradnice z barycentrických alebo naopak. Ak sú dané karteziánske súradnice bodov $A[a_x, a_y], B[b_x, b_y], C[c_x, c_y], R[r_x, r_y]$ v trojuholníku ABC, tak platí [18, 19]:

$$r_x = r_1 a_x + r_2 b_x + r_3 c_x, \tag{3.4}$$

$$r_y = r_1 a_y + r_2 b_y + r_3 c_y. \tag{3.5}$$

Z vyššie uvedených vzorcov plynie, že karteziánske súradnice ľubovoľného bodu vzhľadom na karteziánske súradnice vrcholov trojuholníkov sú váženým priemerom. V tomto prípade sú barycentrické súradnice váhy bodov sčítané do jedného celku [19] .

Pri prevode karteziánskych súradníc na barycentrické je nutné nájsť spätnú transformáciu. To pozostáva z niekoľkých krokov. V prvom kroku je potrebné dosadiť



Obr. 3.2: Rozdelenie trojuholníka na časti vzhľadom na ťažisko

do vzorcov 3.4 a 3.5 daný výraz: $r_3 = 1 - r_2 - r_1$ a to tak, že [18]:

$$r_x = r_1 a_x + r_2 b_x + (1 - r_2 - r_1) c_x, \tag{3.6}$$

$$r_y = r_1 a_y + r_2 b_y + (1 - r_2 - r_1) c_y. \tag{3.7}$$

V druhom kroku sa výrazy 3.6 a 3.7 môžu upraviť a usporiadať následovne [18]:

$$r_1(a_x - c_x) + r_2(b_x - c_x) + c_x - x = 0, \tag{3.8}$$

$$r_1(a_y - c_y) + r_2(b_y - c_y) + c_y - y = 0. \tag{3.9}$$

Na pohľad je zrejmé, že vznikla lineárna transformácia. Tá sa dá zjednodušiť na maticu s rozmermi 2x2 a následne ľahko vypočítať, ako je ukázané vo vzorci 3.10 nižšie [19]:

$$T \cdot \begin{pmatrix} r_1 \\ r_2 \end{pmatrix} = R - C, \; kde \; T = \begin{pmatrix} a_x - c_x & b_x - c_x \\ a_y - c_y & b_y - c_y \end{pmatrix}. \tag{3.10}$$

Posledným krokom je finálna úprava a vyjadrenie súradníc r_1 *a* r_2, kde vznikne vzorec [19]:

$$\begin{pmatrix} r_1 \\ r_2 \end{pmatrix} = T^{-1} \cdot \begin{pmatrix} r_x - c_x \\ r_y - c_y \end{pmatrix}. \tag{3.11}$$

Tretia súradnica r_3 sa dá ľahko dopočítať pomocou zvyšných dvoch pomocou vzorca $r_3 = 1 - r_2 - r_1$, ktorý bol uvedený aj na začiatku postupu prevodu súradníc. Z vyššie uvedeného odvodenia je možné vypočítať barycentrické súradnice z karteziánskych a pri malej úprave aj karteziánske súradnice z barycentrických.

3.2 Využitie barycentrických súradníc v praxi

Barycentrické súradnice sa využívajú hlavne v počítačovej grafike a hernom dizajne. Používajú sa pri transformáciách geometrických tvarov a pri mapovaní textúr. Používajú sa na umiestňovanie bodov vo vnútri trojuholníkov, ktoré sú základnými blokmi pri 3D modelovania. Je to dôležité hlavne pri použitých technikách ako interpolácia textúr a pri animáciách [18] [19].

Tiež sa dajú používať pri obrazovej analýze. Presnejšie sa využívajú pri transformácií 3D scén z 2D obrázkov, kde umožňujú manipulovať efektívne s geometrickými tvarmi a modelmi [18] [19].

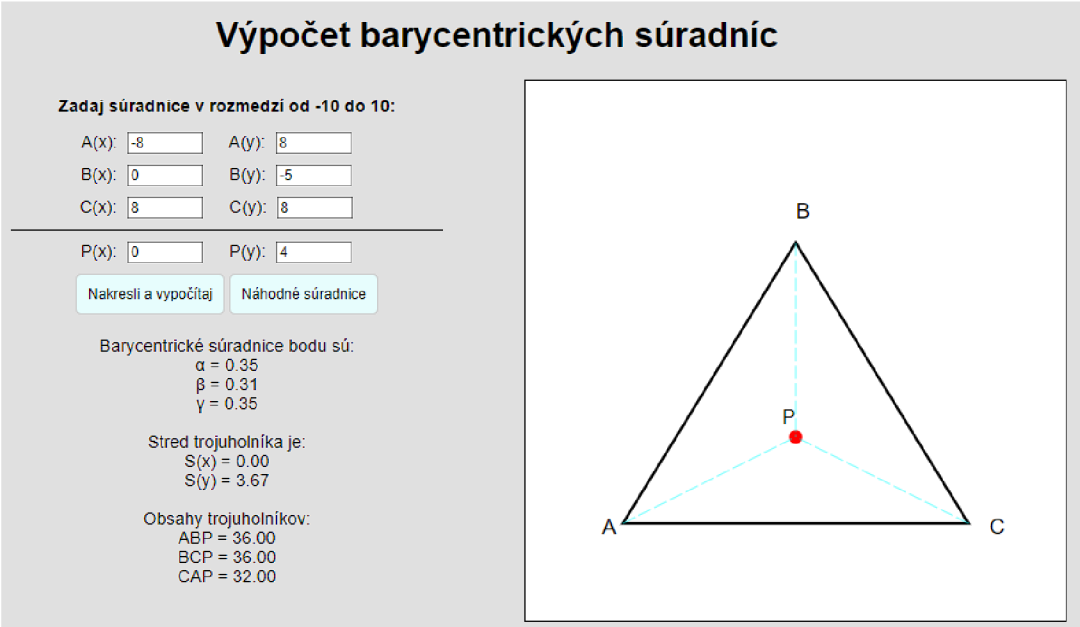
3.3 Programovanie aplikácie na demonštráciu barycentrických súradníc trojuholníka

Súčasťou tejto podkapitoly je vysvetlenie funkcie webového appletu, ktorý sa zameriava na výpočet barycentrických súradníc bodu vzhľadom na trojuholník. Taktiež ukazuje, ako sa tieto súradnice menia, keď sa mení poloha bodu.

Na obrázku 3.3 je zobrazená vytvorená webová aplikácia. Jej obsahom sú vstupné polia na určenie súradníc trojuholníku a bodu. Pri spustení aplikácie sú nastavené východzie hodnoty, ktoré sú zobrazené vo vyššie spomínanom obrázku. Tieto hodnoty sa dajú upravovať v rozsahu od -10 do 10 v karteziánskej sústave. Tento rozsah je určený z dôvodu, aby trojuholník a bod boli jasne viditeľné aj s označením. Pri zadaní súradníc je nutné voľbu potvrdiť tlačidlom: „Nakresli a vypočítaj“. Po zadaní súradníc a následne ich potvrdení, sa vypočítajú nové barycentrické súradnice a nakreslí sa nový trojuholník a bod. Súčasťou appletu je aj výpočet ťažiska trojuholníka a takisto obsahy trojuholníkov, pomocou ktorých sa dajú taktiež vypočítať barycentrické súradnice pomocou vzorca 3.3. Tieto trojuholníky sú z časti označené prerušovanou modrou čiarou, ktorá spája vrcholy trojuholníku a bod `P`. Aplikácia ponúka aj zadanie náhodných súradníc pomocou tlačidla. Tieto súradnice sa generujú vždy tak, aby vznikol viditeľný trojuholník a tiež aj bod, ktorý sa nachádza vo vnútri trojuholníku. Bodom je možné tiež pohybovať pomocou myši a zároveň sa hneď prepočítavajú barycentrické súradnice a obsahy trojuholníkov. Pod appletom sa nachádza samotný úvod do teórie o barycentrických súradniciach a tiež aj návod na ovládanie appletu.

3.3.1 Použitá knižnica vo webovej aplikácií

V tejto aplikácií je použitá knižnica D3. Je aplikovaná z dôvodu, že je flexibilná pri vytváraní vizualizácie a interakcie na rozdiel od iných knižníc s rovnakými funk-



Obr. 3.3: Ukážka appletu pre význam barycentrických súradníc

ciami. Tiež má prehľadnejšiu syntax. V applete je konkrétne použitá pre manipuláciu so SVG elementom. SVG element umožňuje vektorovú grafiku. Odstraňuje z neho prvky, dynamicky pridáva nové prvky. Interaktívne vizualizuje bod, s ktorým sa dá pohybovať pomocou myši. Taktiež sa pomocou knižnice vyobrazuje aj trojuholník [27].

3.3.2 Štruktúra HTML

V súbore `index.html` je nastavená základná štruktúra webovej aplikácie. V hlavičke je použitá znaková sada `UTF-8` a nastavený je aj viewport pre lepšiu responzivitu stránky. Taktiež je tu pripojený externý CSS súbor pre štýlové formátovanie a skript `D3.js` pre manipuláciu s knižnicou D3.

V tele dokumentu je umiestnený formulár so vstupnými poľami pre zadávanie súradnic trojuholníka, bodu a tiež aj dve tlačidlá. Obsah tiež zahŕňa SVG element pre vizualizáciu. Na konci dokumentu je pripojený súbor `script.js`, v ktorom je nastavená logika fungovania webovej aplikácie.

3.3.3 Grafická úprava pomocou CSS

Súbor `style.css` obsahuje nastavenie štýlov pre stránku ako písmo, ktoré je nastavené na Arial, sivé pozadie a zarovnanie všetkých blokov stránky na stred s flexibilným rozložením. Tlačidlá majú nastavené hodnoty tak, aby pri pohybe myši sa

menila ich farba pozadia a okraja. SVG kontajner má pevné rozmery a je vizuálne odlíšený od pozadia stránky. Horizontálne čiary `hr` sú upravené tak, aby nemali typické ohraničenie, ale len vrchnú čiaru pre vizuálne oddelenie zadávania súradníc trojuholníka a bodu.

3.3.4 Použité metódy a funkcie v jazyku Javascript

Na začiatku dokumentu `script.js` je definovaná funkcia `throttle(func, limit)`. Táto funkcia je užitočná pri optimalizácií výkonu pri častých udalostiach, napríklad posun myšou alebo zmeny veľkosti okna. V tejto aplikácií optimalizuje pohyb bodu myšou a taktiež aby sa správne pohybovali prerušované čiary, ktoré spájajú bod `P` a vrcholy trojuholníku.

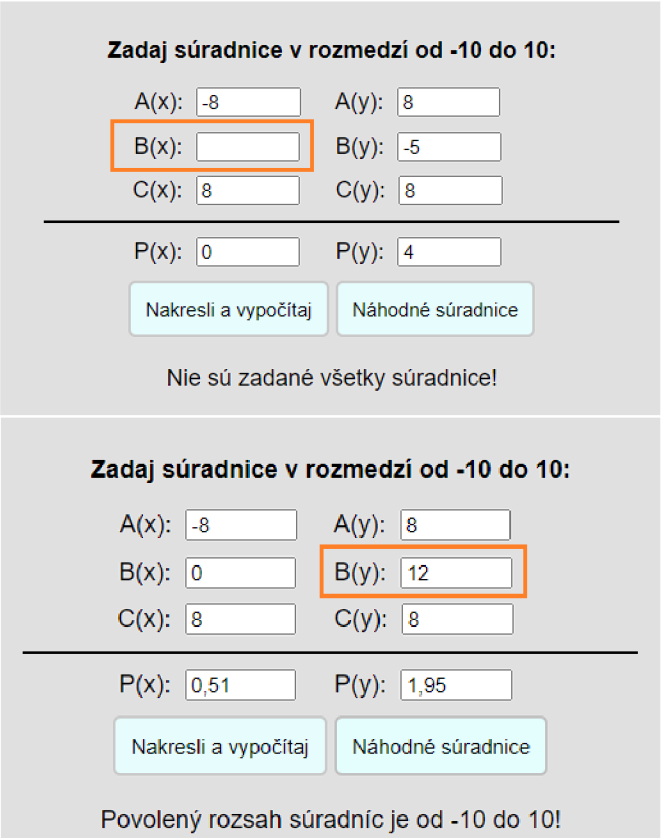
Funkcia `calculateBarycentric` ako hlavná časť aplikácie počíta barycentrické súradnice bodu `P` v závislosti na trojuholníku a počítajú sa pomocou nasledujúcich vzorcov:

$$\begin{aligned} det &= (b_y - c_y) \cdot (a_x - c_x) + (c_x - b_x) \cdot (a_y - c_y), \\ \alpha &= \frac{(b_y - c_y) \cdot (p_x - c_x) + (c_x - b_x) \cdot (p_y - c_y)}{det}, \\ \beta &= \frac{(c_y - a_y) \cdot (p_x - c_x) + (a_x - c_x) \cdot (p_y - c_y)}{det}, \\ \gamma &= 1 - \alpha - \beta. \end{aligned} \tag{3.12}$$

Súradnice sú načítané z HTML formulára. Ak sú súradnice zadané v rámci povoleného rozsahu, t.j. od -10 do 10, funkcia vypočíta barycentrické súradnice. V prípade, že nejaká súradnica nie je zadaná, vypíše chybovú hlášku, ktorá je zobrazená na obrázku 3.4 vo vrchnej polovici a je nutné ju doplniť. Ak je hodnota súradnice mimo definovaného rozsahu, taktiež je táto chyba ošetrená a vypíše sa chybová hláška, ktorá je takisto vyobrazená v obrázku 3.4 v spodnej polovici obrázku.

Medzi ďalšiu súčasť funkcie `calculateBarycentric` patrí aj výpočet ťažiska trojuholníka, a tiež aj výpočet obsahu trojuholníkov `ABP`, `BCP` a `CAP`. Obsahy týchto trojuholníkov sa počítajú z dôvodu, že aj podľa nich sa dajú vypočítať barycentrické súradnice. Táto funkcia taktiež vizualizuje výsledky (nakreslí trojuholník a bod podľa zadaných parametrov). Nakoniec je tu definované aj pomenovanie rohov trojuholníka a bodu vo vhodnej vzdialenosti od objektov, aby sa objekty a pomenovanie navzájom neprekrývali.

Ďalšou funkciou je `drawLine`, ktorá vykresľuje čiary medzi dvomi bodmi v SVG elemente. Táto funkcia sa využíva na grafické znázornenie medzi bodmi trojuholníka a bodom `P`. Jedná sa o slabo-modrú prerušovanú čiaru.



Obr. 3.4: Chybové hlášky pre ošetrenie zadávania súradníc

Funkcia `transformCoord` slúži na transformáciu súradníc bodov pre správnu vizualizáciu (umiestnenie) v elemente SVG. Inak povedané, táto funkcia škáluje a posúva súradnice, aby sa vizuálne zmestili do rozsahu a polohy, ktoré je možné zobraziť v rámci definovaného SVG elementu. V rámci funkcie je škálovateľnosť nastavená na hodnotu 16. To znamená, že sa objekt zväčší 16×, aby bol viditeľný. Posunutie je posunuté o hodnotu `+ 200`, čo značí že objekt je posunutý o 200 jednotiek hore a doprava.

O pohyb a interaktivitu myši sa stará funkcia `addMouseListeners`. Pomocou nej je umožnené chytiť bod ľavým tlačidlom myši a presúvať ho po plátne. Funkcia spracováva udalosti `mousedown`, `mousemove` a `mouseup` pre správu drag-and-drop interakcie.

Aby sa súradnice nemuseli zadávať, je možné pomocou tlačidla vygenerovať náhodné súradnice. To zaisťuje funkcia `generateRandomCoordinates`. Táto funkcia generuje náhodné súradnice tak, aby tvorili platný trojuholník, ktorý nie je nejako degenerovaný. Uhly v trojuholníku musia mať minimálny uhol 10°. Pre obsah trojuholníka je stanovená podmienka, že musí byť väčší ako 12. To zabezpečí, aby bol trojuholník jasne priestorovo viditeľný pre účely pochopenia barycentrických súrad-

níc. Vo funkcií je tiež zaistené, že bod `P` sa vždy umiestňuje vo vnútri trojuholníku. Nikdy sa pri vygenerovaní náhodných súradníc nezobrazí mimo neho.

4 Bayerova schéma v digitálnej fotografií

Bayerov filter vynašiel v roku 1974 Bryce Bayer, ktorý bol vtedy zamestnancom firmy Eastman Kodak. Bayerovou schémou alebo filtrom sa nazýva pole farebných filtrov (CFA). Existuje viacej druhov farebných filtrov, no najpoužívanejší je práve Bayerov. Používa sa vo veľkej časti digitálnych obrazových snímačov. To zahrňuje videokamery, skenery a digitálne fotoaparáty [21, 22].

Aby fotoaparáty a kamery vedeli získať RGB farby obrazu inou cestou ako použitím CFA, potrebujú 3 farebné senzory. Každý senzor spracováva jednu farbu a každý musí obsahovať svoju riadiacu elektroniku. To znamená, že výrobky na spracovanie digitálnych snímok musia byť väčšie a zložitejšie na skompletizovanie. To sa určite odzrkadlí aj na cene výrobku. Práve to je dôvod, prečo sa používa snímač CFA. Existujú aj odlišné metódy, napríklad snímač Foveon X3, použitie dichroických zrkadiel alebo difrakčné filtre s priehľadným poľom [21, 22].

4.1 Usporiadanie farebných filtrov

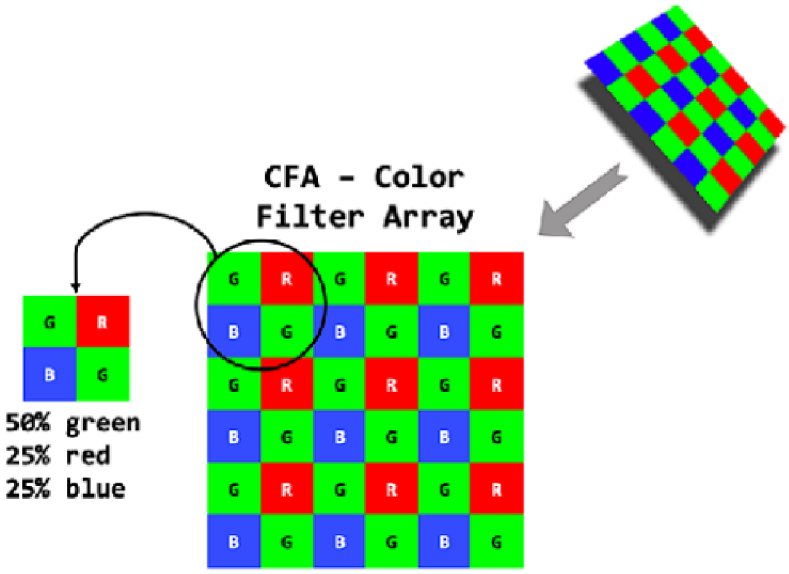
Pôvodne pri patentovaní filtra, Bayer navrhol použiť kombináciu farieb: azúrová, purpurová a žltá označením CMY. V tom čase také farbivá neboli dostupné a odvtedy sa používa kombinácia farieb RGB. Modernejšie (drahšie) snímače používajú práve CMY, pretože majú lepšiu charakteristiku absorpcie svetla [22].

Bayerová schéma má na každom pixely obrazového snímača farebné filtre. Filtre obsahujú základné farby RGB a prepúšťajú len jednu danú farbu. Až 50 % filtra tvorí farba zelená. Ďalších 25 % tvorí farba modrá a posledných 25 % tvorí farba červená. Práve z dôvodu najväčšieho zastúpenia zelenej farby sa niekedy usporiadanie farieb označuje aj RGGB [22].

Filtre sú usporiadané pravidelne a v skupinách po štyroch, ako je vyobrazené na obrázku 4.1. V skupine štyroch farieb môžu byť rôzne kombinácie ako sú usporiadané. Podmienka je, že 2 zelené filtre nemôžu byť vedľa seba, to znamená, že tam môžme mať 4 kombinácie po stĺpcoch GR-BG, GB-RG, RG-GB, BG-GR. Práve takéto usporiadanie farieb je použité preto, pretože ľudské oko je najcitlivejšie na zelenú farbu a tá reprezentuje v tomto prípade citlivosť na jas. Červená a modrá reprezentujú chrominanciu [21, 22].

4.1.1 Debayerizácia

Debayerizácia alebo demosaicing je proces, ktorý sa používa pri digitálnom spracovaní obrazu kvôli rekonštrukcií plnofarebného obrazu z dát, ktoré sú zachytené pomocou snímača s CFA (Color Filter Array) [21].



Obr. 4.1: Bayerov filter a jeho usporiadanie. Prevzaté z [23].

Pri debayerizácií fotografia obsahuje len informácie o jednej farbe z dôvodu použitia farebného filtra, tzn. z jedného farebného kanála v každom pixele a je spracovaná tak, aby sa získala plnofarebná informácia pre každý pixel fotografie. Informácie o ďalších farbách sa získajú interpoláciou chýbajúcich farieb na základe okolitých pixelov s rozličnými farbami [21].

4.1.2 Metódy debayerizácie

Existuje viacej metód alebo algoritmov, ktorými sa dajú dopočítať farby. Rozlišujú sa vo výpočetnej náročnosti a v presnosti. Jedná sa o metódy bilineárna interpolácia, interpolácia splajnom, metóda superpixelu alebo Lanczové prevzorkovanie. Nižšie sú popísané tri základné metódy debayerizácie, ktoré sa veľmi často využívajú v praxi [21, 24].

Bilineárna interpolácia

Najľahšia metóda využíva bilineárnu interpoláciu. Napríklad pri počítaní hodnoty červenej farby nie červeného pixelu sa vypočíta vážený priemer z dvoch alebo štyroch susedných červených pixelov. Podobne je to pre modrý a zelený pixel. Táto metóda sa používa vtedy, keď veľmi nezáleží na kvalite. Pri použití tejto metódy môžu nastať chyby, napríklad aliasing pri nie jasne ohraničených okrajoch scenérie. Táto metóda má výhodu v tom, že je veľmi rýchla a jednoduchá [24].

Metóda superpixelu

Táto metóda je pokročilejšia a zložitejšia, ale snaží sa zlepšiť kvalitu výsledného obrazu na rozdiel od bilineárnej interpolácie. Spočíva v tom, že snímka sa rozdelí na malé regióny, ktoré sa nazývajú superpixely. Obsahujú viacero pixelov. Pre každý superpixel sa použije sofistikovanejší spôsob interpolácie ako napríklad adaptívna interpolácia. Ta zohľadňuje farby a štruktúru v okolí superpixelu. Medzi hlavnú výhodu metódy patrí, že je veľmi rýchla a neobsahuje skoro žiadne chyby (artefakty). Je veľmi dobrá pri prevzorkovaní obrázkov, kde rozlíšenie snímača je oveľa vyššie ako rozlíšenie optiky. Taktiež sa využíva pri komplexných scénach s veľkými rozdielmi v osvetlení a farbách [25].

Metóda Malvar-He-Cutler

Metóda MHC kombinuje metódu bilineárnej interpolácie s ďalšími technikami z dôvodu dosiahnutia lepších výsledkov v častiach, kde je vysoký kontrast [26].

MHC najprv analyzuje gradienty v obraze. Gradienty slúžia na identifikáciu oblastí s vysokým kontrastom, kde sa nemôže použiť bilineárna interpolácia, pretože by sa mohli vyskytnúť silné artefakty. Následne sa snímok rozdelí na malé oblasti alebo bloky, tak aby mohli byť spracované nezávisle od seba. Pre každý blok sa vypočítajú váhové faktory na základe gradientov a textúry v určitej oblasti. Váhy sa v tomto prípade využijú na určenie hodnôt susedných pixelov pri interpolácií. Taktiež sa využívajú na interpoláciu chýbajúcich farieb v každom bloku. V častiach, kde je vysoký kontrast a jasné textúry sa viac zohľadňujú okolité hodnoty, aby sa minimalizovali artefakty [26].

4.1.3 Artefakty pri debayerizácií

Hlavný cieľ debayerizácie je dosiahnuť čo najpresnejšiu rekonštrukciu farebného obrázku bez výrazných artefaktov alebo straty detailov. Úspešnosť funkcie metód je často ovplyvňovaná rôznymi faktormi, ako je napríklad kvalita snímača, expozícia, prítomnosť šumu v obraze a podobne. V praxi sa používajú rôzne kombinácie algoritmov a metód, aby sa eliminovalo čo najviac artefaktov. Najčastejšie sa vyskytujú artefakty [21, 24]:

- chybné farby a aliasing - vznikajú kvôli interpolácie farieb najmä pri ostrých hranách a detailoch,
- šum - spôsobené hlavne z dôsledku interpolácie farieb,
- mozaika - niektoré metódy debayerizácie spôsobujú mozaikové vzory, ktoré je vidieť hlavne pri veľkých zväčšeniach.

4.2 Programovanie webovej aplikácie na ukážku funkcie Bayerovho filtra

V tejto podkapitole bude postupne vysvetlené fungovanie webového appletu, ktorý ukazuje ako funguje Bayerov filter za pomoci programovacích jazykov HTML, CSS a Javascript s využitím knižnice jQuery.

Pri otvorení webovej stránky sa zobrazí vopred vybratý obrázok v originálnej podobe na ľavej strane obrazovky. Na pravej strane sa zobrazí obrázok po aplikácií Bayerovho filtra. Veľkosť obrázku je upravená nastaveným rozmerom v projekte tak, aby sa zmestil do stanoveného rámu. Rám je nastavený tak, aby bol viditeľný v celej svojej veľkosti bez nutného posúvania sa na stránke. Súčasťou appletu je aj voľba veľkosti Bayerovho filtra pre lepšie pochopenie funkčnosti. applet umožňuje zvoliť pár hodnôt od 1 do 20 čo ukazuje rozmer $1 = 1\,\text{px} \times 1\,\text{px}, 25 = 1\,\text{px} \times 1\,\text{px}$. V ľavom hornom rohu je vytvorené tlačidlo na vloženie vlastného obrázku z počítača. Pod zobrazovaným plátnom (canvasom) sú umiestnené všeobecné informácie o Bayerovom filtri a taktiež popis funkcie stránky. Ukážka naprogramovaného appletu je v obrázku 4.2.

Obr. 4.2: Ukážka webového appletu funkcie Bayerovho filtra

4.2.1 Použitá knižnica v applete

V aplikácií pre ukážku funkcie Bayerovho filtra je použitá knižnica jQuery. Bola zvolená z dôvodu, že sa s ňou ľahko pracuje, má prehľadnú syntax a je voľne dostupná. Poskytuje konzistentnú funkčnosť naprieč všetkými prehliadačmi a jej kódy sú kratšie ako pri použití iných knižníc. V aplikácií je použitá na zjednodušenie syntaxe [28].

4.2.2 Štruktúra appletu

Štruktúra webového appletu je definovaná v súbore `index.html`. Na začiatku je definovaný dokument ako HTML5 a tiež definovaný jazyk obsahu stránky `lang="sk"`.

V hlavičke súboru je nastavené použité kódovanie znakov na `charset="UTF-8"`. Nasleduje optimalizovanie stránky na rôznych zariadeniach tým, že kontroluje šírku zobrazenia, čo určuje, že je stránka responzívna. V hlavičke je tiež pripojený CSS súbor pre určenie štýlov stránky a vložená knižnica jQuery.

V tele dokumentu sú definované kontajnery, v ktorých sa nachádzajú jednotlivé elementy ako nadpis, tlačidlo, rozbaľovacie menu, text a canvas. Canvas je plátno, v ktorom sa zobrazujú oba obrázky, ako pôvodný, tak aj upravený.

Pätička stránky je oddelená čiarou a obsahuje informácie o autorských právach.

4.2.3 Kaskádové štýly

Kaskádové štýly sú definované v súbore `style.css`. Nastavená je veľkosť stránky, odstránenie okrajov, centrovanie elementov. Tiež zahrňuje ďalšie grafické úpravy všetkých súčastí na stránke ako nadpisy, tlačidlá, orámovanie a podobne.

4.2.4 Použité metódy a funkcie

Metóda `getColorIndicesForCoord` má vstupné argumenty `x, y, width` a slúži k výpočtu pozície farieb v poli dát obrázku na základe daných súradníc. Táto metóda obsahuje vzorec, ktorý vyčíta indexy pre červenú farbu na základe súradníc `x, y`. Pozície pre jednotlivé kanály spočítané násobením súradníc a násobením argumentu `width`, čo označuje šírku obrazu. Metóda vracia pole, ktoré obsahuje vypočítané indexy pre všetky farebné zložky.

Ďalšia metóda `getRGBFromCoord` slúži k získaniu hodnôt RGB pre súradnice `x, y` v rámci obrázku. Vracia zase pole, ktoré obsahuje hodnoty farieb na daných súradniciach. Táto metóda využíva metódu `getColorIndicesForCoord` pre získanie potrebných indexov farieb.

Dôležitou súčasťou je aj metóda `setRGB`, ktorá slúži k nastaveniu hodnôt farieb pre dané súradnice. Funguje tak, že získava indexy pozícií farieb a nastavuje nové

hodnoty farieb na daných súradniciach osobitne pre každú farbu. Navráti aktualizované hodnoty.

Metóda `drawSquare` stanovuje štvorec, kde sa začína z ľavého horného rohu a dvomi cyklami `for` sa prechádzajú všetky pixely štvorca. Parametrami tejto metódy sú súradnice x, y, šírka štvorca, index farby obsahujúci hodnoty 0 až 2, hodnotu farby a dáta obrázku, aby sa vykreslil štvorec. Pomocou tohto sa nakreslí štvorec na danej pozícií.

V projekte je stanovený tvar na vykreslenie ako pôvodného obrazu tak aj upraveného cez Bayerov filter. Metóda `drawInitialImage` slúži k načítaniu obrázku a k jeho vykresleniu na určenú polohu (`canvas`). Takisto je dôležitá pre zistenie dát z obrázku, s ktorými sa pracuje, pomocou funkcie `callback`.

Metóda `getBayerColorAtLocation` je vlastne metóda zmenšenia počtu farieb, ktorá sa často používa pri grafickom spracovaní obrázkov.

Poslednou metódou je `calculateBayerPixelRepresentation`, ktorá je určená na výpočet a reprezentáciu pixelov obrázku pomocou Bayerovho filtra. Najprv zisťuje, či sú k dispozícií dáta z obrázku. Ak áno, následne získa hodnotu veľkosti rozmeru štvorca pre Bayerov filter z HTML prvku a nastaví túto hodnotu pre ďalšie výpočty. Metóda zase obsahuje cykly na prechádzanie všetkých pixelov v obrázku ako pre riadky, tak aj pre stĺpce. Následne sa získa hodnota farby pre aktuálny blok pixelov pomocou funkcie `getBayerColorAtLocation` a vypočíta sa priemerná hodnota pre štvorec (blok pixelov z html). V ďalšom kroku sa nakreslia štvorce zodpovedajúce zadaným rozmerom a Bayerovmu vzoru na danú pozíciu s vypočítanou priemernou hodnotou. Poslednými krokmi sú už len aktualizácia canvasu a vrátenie poľa obsahujúceho priemerné hodnoty pixelov pre každý blok v obraze.

Funkcia `initCanvas` je vytvorená pre inicializáciu a nastavenie pre HTML element `<canvas>`. Funkcia získava rozmery obrazovky (viewportu), ako výšku, tak aj šírku. Pomocou nich vypočítava veľkosť canvasu. Veľkosť je v tomto prípade nastavená na 75 % šírky a 75 % výšky obrazovky, pri načítaní stránky. Následne nastaví nové rozmery pre canvas.

V ďalšej funkcií sa inicializuje canvas a je vytvorená inštancia triedy `BayerImage`. To načíta obrázok, nastaví rozmery canvasu a získa dáta obrazu a aplikujú sa v nej vyššie uvedené metódy. V konzole, pre kontrolu sa vypíšu dáta načítaného obrázku a tiež hodnoty farieb pre pozíciu (0,0).

Funkcia `changePixelSize` prevádza zmeny veľkostí pixelov pre Bayerov vzor a vykresľuje obrázok na canvas s novým vzorom. V rozbaľovacom menu v applete je možnosť nastavenia rozmeru Bayerovho vzoru na 1×1px, 2×2px a podobne.

Posledná súčasť tohto appletu slúži na vloženie obrázku pomocou tlačidla. Po kliknutí na tlačidlo sa otvorí dialógové okno pre výber súboru. Je tam nastavené ošetrenie, že v prípade výberu viacerých obrázkov sa vyberie len prvý a ten sa vloží

na stránku. Po nahratí obrázku sú definované funkcie, ktoré sa hneď majú vykonať ako zobrazenie na plátne a následne aj zobrazenie po úpravách, aby bolo vidieť, ako funguje Bayerov filter.

Tento kód využíva knižnicu jQuery na výber elementov DOM pomocou selektorov (reťazcov) `($('#element_id'))`. Takisto sa používa jQuery pre pridanie event listenerov (poslúchačov udalostí) `($(selector).event(function()))`. Je to z dôvodu, že kód je ľahšie čitateľný a prehľadnejší.

Záver

Táto diplomová práca predstavuje projekt zameraný na vytvorenie troch webových aplikácií v jazyku JavaScript, ktoré demonštrujú aspekty počítačovej grafiky: parametrické krivky v 3D priestore, barycentrické súradnice a Bayerov filter. Každá z aplikácií bola navrhnutá tak, aby študentom uľahčila pochopenie zložitých konceptov a poskytla im interaktívne prostredie pre ľahšie pochopenie preberaných učív.

V rámci práce je vysvetlená problematika potrebná pre pochopenie tém, na ktoré sú vytvorené webové aplikácie. Ďalšia časť obsahuje úvod do programovania webových stránok a appletov s využitím jazykov HTML, CSS a Javascript. Hlavnou časťou je výber vhodných knižníc pre programovanie v Javascripte a samotné vytvorenie troch funkčných aplikácií.

Prvý applet umožňuje vygenerovať náhodnú krivku v 3D priestore s riadiacimi bodmi. V prípade potreby je možné zobraziť riadiace polygóny alebo aj vygenerovať novú náhodnú krivku. Bod na krivke a priebehy jednotlivých zložiek je možné ovládať pomocou posuvníku. V applete sa nachádza aj manuál k ovládaniu a tiež teória k pochopeniu významu. V diplomovej práci sa nachádza opis funkcií a metód, ktoré boli pri programovaní appletu použité.

Ďalšia webová aplikácia poskytuje vizualizáciu a výpočet barycentrických súradníc bodu vo vzťahu k trojuholníku. Umožňuje do vstupných polí zadať vlastné súradnice v definovanom rozsahu. Je možné meniť polohu bodu myšou v plátne a zároveň dynamicky počíta nové súradnice a obsahy trojuholníkov medzi vrcholmi trojuholníka a meniacim sa bodom. V práci je tiež opis štruktúry stránky a grafického rozloženia. Jednotlivé metódy a funkcie sú presne popísané a vysvetlené.

Posledný aplet je vytvorený pre ukážku funkcie (výzoru) Bayerovho filtra. Vopred stanový obrázok sa zobrazí ako originálny obrázok na ľavej strane obrazovky, obrázok po aplikácií Bayerovho filtra sa zobrazí na pravej strane. Obrázky sú veľkostne spracované na štvorce, aby to zodpovedalo aj rozumnému umiestneniu na stránke bez nutnosti posúvania sa v rámci stránky. Veľkosť Bayerovho filtra sa dá meniť aby bolo pre užívateľa pochopenie fungovania filtra ľahšie. V práci sú vysvetlené aj funkcie a metódy, ktoré boli pri programovaní apletu použité a tiež aj celá štruktúra a formátovanie stránky v jazykoch HTML a CSS.

Literatúra

[1] ŽÁRA, Jiří. Moderní počítačová grafika. Online. 2., přeprac. a rozš. vyd. Brno: Computer Press, 2004. ISBN 80-251-0454-0. Dostupné z: https://dcgi.fel.cvut.cz/ModerniPocitacovaGrafika/. [cit. 2023-11-13].

[2] *WIKIBOOKS. Geometrie/Úvod do křivek.* Online. 2023, 12.04.2023. Dostupné z: https://cs.wikibooks.org/wiki/Geometrie/%C3%9Avod_do_k%C5%99ivek. [cit. 2023-10-24].

[3] TIŠNOVSKÝ, PAVEL. *Křivky určené polynomem – nejpoužívanější křivky v současnosti.* Online. ROOT.cz. 2021, s. 1-15. Dostupné z: https://www.root.cz/clanky/krivky-urcene-polynomem-nejpouzivanejsi-krivky-v-soucasnosti/. [cit. 2023-11-17].

[4] *Bézierovy krivky.* Online. In: *ZÁPADOČESKÁ UNIVERZITA V PLZNI. Katedra Matematiky.* 2009, 2023. Dostupné z: https://home.zcu.cz/~bastl/GM1/GM1_lecture04.pdf. [cit. 2023-11-17].

[5] *Hypertext Markup Language.* Online. In: *Wikipedia: the free encyclopedia.* San Francisco (CA): Wikimedia Foundation, 2023. Dostupné z: https://cs.wikipedia.org/wiki/Hypertext_Markup_Language. [cit. 2023-11-21].

[6] *Značkovací jazyk (HTML).* Online. VEV-VA VYŠKOV. Tvorba webu - HTML, CSS, JS. Dostupné z: https://web.vavyskov.cz/znackovaci-jazyk.html. [cit. 2023-11-21].

[7] *Verzie HTML.* Online. WEB Tutorial. 2012, 2017. Dostupné z: http://www.webtutorial.cekuj.net/html/verziehtml.html. [cit. 2023-11-21].

[8] JANOVSKÝ, Dušan. *HTML příručka.* Online. Jak psát web. 2015. Dostupné z: https://www.jakpsatweb.cz/html/. [cit. 2023-11-21].

[9] JANOVSKÝ, Dušan. *CSS styly - úvod.* Online. Jak psát web. 2015. Dostupné z: https://www.jakpsatweb.cz/css/css-uvod.html. [cit. 2023-11-21].

[10] *Kaskádové štýly.* Online. In: *Wikipedia: the free encyclopedia.* San Francisco (CA): Wikimedia Foundation, 2016. Dostupné z: https://cs.wikipedia.org/wiki/Kask%C3%A1dov%C3%A9_styly. [cit. 2023-11-21].

[11] *JavaScript.* Online. In: *Wikipedia: the free encyclopedia.* San Francisco (CA): Wikimedia Foundation, 2018, 2023. Dostupné z: https://cs.wikipedia.org/wiki/JavaScript. [cit. 2023-11-24].

[12] JANOVSKÝ, Dušan. *Javascript.* Online. Jak psát web. 2015. Dostupné z: https://www.jakpsatweb.cz/javascript/javascript-uvod.html. [cit. 2023-11-24].

[13] *JavaScript library.* Online. In: *Wikipedia: the free encyclopedia.* San Francisco (CA): Wikimedia Foundation, 2016, 2023. Dostupné z: https://en.wikipedia.org/wiki/JavaScript_library. [cit. 2023-12-02].

[14] *Three.js.* Online. In: *Wikipedia: the free encyclopedia.* San Francisco (CA): Wikimedia Foundation, 2020, 2023. Dostupné z: https://en.wikipedia.org/wiki/Three.js. [cit. 2023-12-02].

[15] *JQuery.* Online. In: *Wikipedia: the free encyclopedia.* San Francisco (CA): Wikimedia Foundation, 2023. Dostupné z: https://en.wikipedia.org/wiki/JQuery. [cit. 2023-12-02].

[16] *Verge3D.* Online. In: *Wikipedia: the free encyclopedia.* San Francisco (CA): Wikimedia Foundation, 2019, 2023. Dostupné z: https://en.wikipedia.org/wiki/Verge3D. [cit. 2023-12-02].

[17] *D3.js* Online. In: *Wikipedia: the free encyclopedia.* San Francisco (CA): Wikimedia Foundation, 2019, 2023. Dostupné z: https://en.wikipedia.org/wiki/D3.js. [cit. 2023-12-02].

[18] *Barycentric coordinate system.* Online. In: *Wikipedia: the free encyclopedia.* San Francisco (CA): Wikimedia Foundation, 2023. Dostupné z: https://en.m.wikipedia.org/wiki/Barycentric_coordinate_system. [cit. 2023-10-24].

[19] MAŠATOVÁ, Zora. *Barycentrické souřadnice.* Bakalárska práca. *Univerzita Karlova v Praze. Pedagogická fakulta..* Praha, 2014. strany 33–44. [cit. 2023-10-24].

[20] *Barycentric coordinates on an equilateral triangle.* Online. In: *ResearchGate.* Dostupné z: https://www.researchgate.net/figure/Barycentric-coordinates-on-an-equilateral-triangle_fig5_264825595. [cit. 2023-10-24]

[21] *Bayer filter.* Online. In: *Wikipedia: the free encyclopedia.* San Francisco (CA): Wikimedia Foundation, 2023. Dostupné z: https://en.wikipedia.org/wiki/Bayer_filter. [cit. 2023-10-21].

[22] BALUŠÍK, Peter. *RAW image debayerization using deep neural network.* Bakalárska práca. *Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií.* Brno, 2023. strany 30–32. [cit. 2023-10-21].

[23] *Scheme of a Bayer filter.* Online. In: *ResearchGate.* 2022. Dostupné z: https://www.researchgate.net/figure/Scheme-of-a-Bayer-filter-On-the-left-a-sample-cell-of-2x2-photosites-BGGR-pattern-In_fig2_359076583. [cit. 2023-10-22].

[24] *Demosaicing.* Online. In: *Wikipedia: the free encyclopedia.* San Francisco (CA): Wikimedia Foundation, 2023. Dostupné z: https://en.wikipedia.org/wiki/Demosaicing. [cit. 2023-10-22].

[25] VRASTIL, Zbynek a POOL, Sander. *Pixinsight.* Online. Debayer. 2014. Dostupné z: https://pixinsight.com/. [cit. 2024-02-12].

[26] PASCAL, Getreuer. *Malvar-He-Cutler Linear Image Demosaicking.* In: IPOL - Image Processing On Line. 2011, s. 84-86. ISSN 2105–1232. Dostupné z: https://www.researchgate.net/publication/270045976_Malvar-He-Cutler_Linear_Image_Demosaicking. [cit. 2024-02-15].

[27] D3.js. *D3 by Observable.* Online. 2013, 2024. Dostupné z: https://d3js.org/. [cit. 2024-04-24].

[28] JQuery. *JQuery - write less, do more.* Online. 2007, 2024. Dostupné z: https://jquery.com/download/. [cit. 2024-04-24].

[29] *Chart.js.* Online. 2016, 2024. Dostupné z: https://www.chartjs.org/. [cit. 2024-04-26].

[30] *Three.js.* Online. 2010, 2024. Dostupné z: https://threejs.org/. [cit. 2024-04-26].

Zoznam symbolov a skratiek

HTML	značkovací jazyk – HyperText Markup Language
CSS	Cascading Style Sheets
WWW	World Wide Web
SGML	značkovací jazyk – Standard Generalized Markup Language
utf-8	spôsob kódovania znakov – Unicode Transformation Format, 8-bit
URL	Uniform Resource Locator
WebGL	Web Graphics Library
DOM	reprezentácia štruktúry dokumentu – Document Object Model
SVG	vektorová grafika – Scalable Vector Graphics
CFA	Color Filter Array
CMY	azurová, ružová, žltá – Cyan, Magenta, Yellow
URL	adresa používaná na prístup k webstránkam - Uniform Resource Locator
SVG	formát založený ny XML na popis vektorovej grafiky – Scalable Vector Graphics
HTML5	nová verzia značkovacieho jazyka HTML

Zoznam príloh

A Obsah elektronickej prílohy

```
Webové aplikácie
├── Bayerov filter
│   ├── index.html
│   ├── script.js
│   ├── style.css
│   └── test.jpg
├── Barycentrické súradnice
│   ├── index.html
│   ├── script.js
│   └── style.css
└── Parametrické vyjadrenie krivky v 3D priestore
    ├── bezier.png
    ├── index.html
    ├── OrbitControls.js
    ├── script.js
    ├── style.css
    └── three.module.js
```