

# **Komplexní www stránky pro výuku předmětu Počítačová grafika**

Bc. Jakub Hromek

---

Diplomová práce  
2016



Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky

---

# ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Jakub Hromek**  
Osobní číslo: **A14518**  
Studijní program: **N3902 Inženýrská informatika**  
Studijní obor: **Bezpečnostní technologie, systémy a management**  
Forma studia: **kombinovaná**

Téma práce: **Komplexní www stránky pro výuku předmětu Počítačová grafika**  
Téma anglicky: **Complex www Pages for the Tuiton of Computer Graphics**

## Zásady pro vypracování:

1. Navrhněte a realizujte webové stránky pro podporu výuky předmětu Počítačová grafika.
2. Koncepti těchto stránek proveďte tak, aby bylo snadné je v budoucnu rozšiřovat o další obsah.
3. Popište a zhodnoťte použité technologie pro tvorbu těchto stránek.
4. Zhodnoťte tyto stránky z pohledu bezpečnosti.
5. Seznamte se s algoritmy pro rasterizaci a kompresními algoritmy používanými v oblasti počítačové grafiky.
6. Obě témata zmíněná v předchozím bodě v práci blíže specifikujte, implementujte jejich vizualizaci pomocí webových technologií a zakomponujte je jako kurzy do vytvořených internetových stránek.

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. ŽÁRA, Jiří, Bedřich BENEŠ, Petr FELKEL a Jiří SOCHOR. Moderní počítačová grafika. Vyd 2. Brno: Computer Press, 2005, 608 s. EAN 9788025104545.
2. MARTIŠEK, Dalibor. Matematické principy grafických systémů. Vyd. 1. Brno: Litera, 2002, 278 s. ISBN 80-857-6319-2.
3. FOLEY, James. Computer Graphics. Boston, 1997, 1175 s. ISBN 02-018-4840-6.
4. JURAJ, Štugel. Netgraphics [online]. [cit. 2016-01-11]. Dostupné z: <http://netgraphics.sk/>
5. Nette Foundation. Nette Framework. [online]. [cit. 2016-01-11]. Dostupné z: <https://doc.nette.org/>
6. Mozilla Developer Network. JavaScript [online]. [cit. 2016-01-11]. Dostupné z: <https://developer.mozilla.org/cs/docs/Web/JavaScript>
7. Mozilla Developer Network. Canvas API [online]. [cit. 2016-01-11]. Dostupné z: [https://developer.mozilla.org/en-US/docs/Web/API/Canvas\\_API](https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API)
8. OWASP [online]. [cit. 2016-01-11]. Dostupné z: <https://www.owasp.org>

Vedoucí diplomové práce:

**Ing. Pavel Pokorný, Ph.D.**

Ústav počítačových a komunikačních systémů

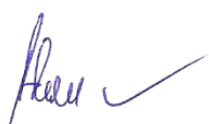
Datum zadání diplomové práce:

**5. února 2016**

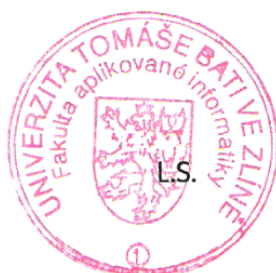
Termín odevzdání diplomové práce:

**16. května 2016**

Ve Zlíně dne 5. února 2016



doc. Mgr. Milan Adámek, Ph.D.  
*děkan*



doc. RNDr. Vojtěch Křesálek, CSc.  
*ředitel ústavu*


### Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl jsem seznámen s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

### Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne 11.5.2016

  
.....  
podpis diplomanta

## **ABSTRAKT**

Hlavním cílem této diplomové práce bylo vytvořit webovou stránku, která bude v budoucnu studentům fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně sloužit jako pomůcka pro hlubší pochopení témat předmětu počítačová grafika a bude poskytovat prostor pro umístování kurzů vytvořených pomocí webových technologií. V rámci práce jsou vytvořeny kurzy pro rasterizační algoritmy, kompresní algoritmy a prokládání. Těmto tématům se podrobněji věnuje teoretická část práce. Dále práce popisuje použité technologie pro tvorbu stránek a kurzů, kde se zaměřuje i na bezpečnostní stránku celé aplikace.

Klíčová slova: počítačová grafika, rasterizační algoritmy, kompresní algoritmy, prokládání, JavaScript, Canvas

## **ABSTRACT**

The main aim of this diploma thesis was to develop a web site, which will become a new study platform for students of the Faculty of Applied Informatics UTB who attend courses of Computer graphics. Pilot courses of raster algorithms, compression algorithms and interleaving were developed and added to the web site. Additionally, diploma thesis describes technology used for courses design and web development. Furthermore, the work focused also on the security site of the application.

Keywords: computer graphic, raster algorithms, compression algorithms, interleaving, JavaScript, Canvas

Tímto bych chtěl poděkovat vedoucímu bakalářské práce Ing. Pavlu Pokornému, Ph.D za odborný dohled, ochotu, spolupráci a čas, který mi po dobu realizace této práce věnoval.

Děkuji rodině za podporu a motivaci po celou dobu tvorby této práce.

# OBSAH

<b>ÚVOD</b> .....	<b>8</b>
<b>I TEORETICKÁ ČÁST</b> .....	<b>9</b>
<b>1 POČÍTAČOVÁ GRAFIKA</b> .....	<b>10</b>
1.1 VYUŽITÍ POČÍTAČOVÉ GRAFIKY .....	10
<b>2 RASTERIZACE</b> .....	<b>12</b>
2.1 RASTROVÁ A VEKTOROVÁ GRAFIKA .....	12
2.1.1 Vektorové obrázky .....	12
2.1.2 Rastrové obrázky .....	13
2.2 RASTERIZACE OBJEKTŮ .....	14
2.2.1 Rasterizace úsečky .....	14
2.2.1.1 DDA algoritmus.....	15
2.2.1.2 Bresenhamův algoritmus pro úsečku.....	16
2.2.2 Rasterizace kružnice.....	18
2.2.2.1 Bresenhamův algoritmus pro kružnici.....	19
2.2.3 Rasterizace elipsy.....	20
2.2.3.1 Bresenhamův algoritmus pro elipsu .....	21
2.2.4 Rasterizace křivky.....	22
2.2.4.1 Interpoláčn� křivky.....	23
2.2.4.2 Aproximačn� křivky.....	26
<b>3 KOMPRESSE RASTROVÉHO OBRAZU</b> .....	<b>30</b>
3.1 ZTRÁTOVÁ .....	30
3.1.1 Diskr�tn� kosinov� transformace.....	30
3.1.2 Frakt�ln� komprese.....	31
3.2 BEZZTR�TOV� .....	32
3.2.1 Lempel – Ziv – Welch.....	32
3.2.1.1 Komprese .....	32
3.2.1.2 Dekomprese .....	33
3.2.1.3 LZW p�i kompresi GIF .....	34
3.2.2 Run lenght encoding .....	37
3.2.3 Huffmanovo k�dov�n� .....	38
<b>4 PROKL�D�N� OBR�ZKŮ</b> .....	<b>39</b>
4.1 JEDNOROZM�RN� .....	39
4.2 DVOUROZM�RN�.....	40
<b>5 WEBOV� BEZPEČNOST</b> .....	<b>41</b>
5.1 HTTP/HTTPS.....	41
5.1.1 Metody HTTPS .....	41
5.2 TYPY ŰTOKŮ .....	42
5.2.1 SQL Injection .....	42
5.2.2 Űnos spojení.....	43
5.2.2.1 Session fixation.....	43
5.2.2.2 Session sidejacking .....	43
5.2.2.3 Cross-site scripting .....	43
5.2.2.4 Fyzick� zkop�rov�n� SID .....	44

5.2.3	Cross Site Scripting .....	44
5.2.4	Cross Site Request Forgery .....	44
<b>6</b>	<b>POUŽITÉ TECHNOLOGIE PRO IMPLEMENTACI STRÁNEK .....</b>	<b>46</b>
6.1	WORDPRESS .....	46
6.1.1	Pluginy .....	47
6.1.2	Témata vzhledu .....	47
<b>II</b>	<b>PRAKTICKÁ ČÁST .....</b>	<b>48</b>
<b>7</b>	<b>METODIKA .....</b>	<b>49</b>
<b>8</b>	<b>WEBOVÁ STRÁNKA .....</b>	<b>50</b>
8.1.1	Implementace stránek.....	50
8.2	DESIGN WEBU.....	51
8.3	SERVER SITE.....	52
8.3.1	Databáze .....	53
8.3.2	Autentizace a autorizace .....	53
8.4	POPIS UŽIVATELSKÉHO ROZHŘANÍ.....	54
8.5	BEZPEČNOST WEBU .....	56
<b>9</b>	<b>KURZY.....</b>	<b>57</b>
9.1	POUŽITÉ TECHNOLOGIE .....	57
9.1.1	Dojo Toolkit .....	57
9.1.2	Canvas .....	58
9.1.2.1	EaselJS .....	59
9.2	KURZ RASTERIZACE .....	59
9.2.1	Uživatelské rozhraní.....	59
9.2.2	Ovládání .....	60
9.3	KURZ KOMPRESY OBRAZOVÝCH DAT .....	60
9.3.1	Uživatelské rozhraní.....	61
9.4	KURZ PROKLÁDÁNÍ .....	61
9.4.1	Uživatelské rozhraní.....	62
9.4.2	Ovládání .....	62
	<b>ZÁVĚR .....</b>	<b>63</b>
	<b>SEZNAM POUŽITÉ LITERATURY.....</b>	<b>64</b>
	<b>SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....</b>	<b>68</b>
	<b>SEZNAM OBRÁZKŮ .....</b>	<b>70</b>
	<b>SEZNAM TABULEK.....</b>	<b>72</b>
	<b>SEZNAM PŘÍLOH.....</b>	<b>73</b>

## ÚVOD

Internet od svého vzniku prošel značnou proměnou, původně sloužil jen k přenášení statických dokumentů, především vědeckých prací. Největší rozmach zažil při rozšíření do veřejné sféry a od té doby se stále rozvíjí. S vývojem a celosvětovým rozšířením mobilních zařízení, jako jsou notebooky, tablety a mobilní telefony, dostal internet nový rozměr, s nímž se ruku v ruce vyvíjely i požadavky na techniku zpracování. Technologie pro tvorbu webových stránek nabízí stále kvalitnější prostředky na vývoj složitějších aplikací, které v mnoha případech dokážou plnohodnotně zastoupit spustitelné aplikace, jež vyžadují instalaci přímo v zařízení. Webové technologie jsou výhodné především jejich nenáročností na software, stačí pouze webový prohlížeč a dostatečné připojení k síti internet. Nové webové technologie jsou multiplatformní, nemusí se tedy řešit, jaký operační systém je využíván a všechny data mohou být uložena na vzdáleném serveru a přístupná odkudkoliv. Dostupnosti webových aplikací se dá také využít při studiu. Jako příklad poslouží studenti cestující do školy hromadnou dopravou, kteří čas na cestě mohou využít k sebevzdělávání a ke každodenní školní přípravě.

Cílem této práce bylo vytvoření webové stránky na podporu výuky Počítačové grafiky na FAI UTB. Stránka slouží jako portál pro nahrávání modulů vytvořených pomocí webových technologií. Koncept stránek byl nutno vytvořit tak, aby v budoucnu bylo jejich rozšíření po obsahové i funkční stránce co nejjednodušší. Dále bylo nutné popsat použité technologie a zhodnotit bezpečnostní prvky vytvořené aplikace. V teoretické části byla podrobně popsána problematika rasterizačních algoritmů u základních grafických objektů a nejpoužívanější kompresní algoritmy užívané při kompresi rastrového obrazu. Pro tyto témata byla v praktické části vytvořena jejich vizualizace pomocí webových technologií a následně byla zakomponována do vytvořené aplikace jako kurzy.

## **I. TEORETICKÁ ČÁST**

## 1 POČÍTAČOVÁ GRAFIKA

Pod pojmem počítačová grafika dnes můžeme zahrnovat cokoliv vytvořeného na počítači, mimo psaného textu a zvuku. V dnešní době je téměř každý počítač schopen vytvářet grafické výstupy a uživatelé doslova očekávají ovládnutí počítače prostřednictvím grafických obrazů a ikon, namísto zadávání textových řetězců do příkazové řádky. Do počítačové grafiky beze sporu patří obrázky, filmy, kresby, 3D modely, simulace a mnoho dalšího obsahu tvořeného prostřednictvím počítačového hardwaru a softwaru. [1]

Počítačová grafika je obor s velmi dynamickým rozvojem. Není to zase tak dávno, kdy grafické výstupy na počítači byly jen výsadou špičkových superpočítačů v grafických, reklamních a televizních studiích. V dnešní době má většina uživatelů dostatečný výpočetní výkon na tvorbu počítačové grafiky v domácích podmínkách. To také otevřelo cestu vývojářů grafických editorů k běžným uživatelům. [2]

### 1.1 Využití počítačové grafiky

Počítačová grafika se využívá v mnoha oborech a dotýká se našeho každodenního života, ať už si to uvědomujeme nebo ne.

- Tiskoviny – prakticky veškeré tiskoviny, které se vám dnes dostanou do rukou, tj. časopisy, noviny, knihy, letáky apod., jsou dílem grafiků, kteří je zpracovávali na počítači. [2]
- Reklama - obrovský obor, který počítačovou grafiku využívá na každém kroku. Ať už se podíváte na billboard, propagační materiály či reklamní televizní spot, to vše velmi pravděpodobně prošlo rukama specializovaného grafika. [2]
- Média, televize, multimédia – multimediální CD, televizní efekty, titulky a zajímavé grafické obrázky a schémata ve večerních zprávách, to je další z příkladů použití počítačové grafiky, která nemusí vždy ústít do tištěné podoby. [2]
- Bezpečnostní systémy – jednou z ne úplně známých oblastí využívající počítačové grafiky je také oblast bezpečnosti, jež např. využívá grafiky pro dnes populární znázornění dat kriminality pomocí tzv. map kriminality. Využití je široké všude tam, kde grafické znázornění dat je pro člověka přívětivější než orientace v tabulkách s čísly.
- Hry – počítačová grafika hraje významnou roli v zábavním průmyslu. Vždyť současné hry jsou kvalitní grafikou doslova nabyté a v podstatě se pokoušejí kopírovat

zobrazení reálného světa. Hry jako taktické simulátory se využívají například v armádě. [2]

- Tvorba www stránek – webové stránky jsou vykreslovány prohlížečem, který na základě kódu HTML, CSS, JavaScriptu a obrázků uživateli nabízí kompletní grafický vzhled stránek.

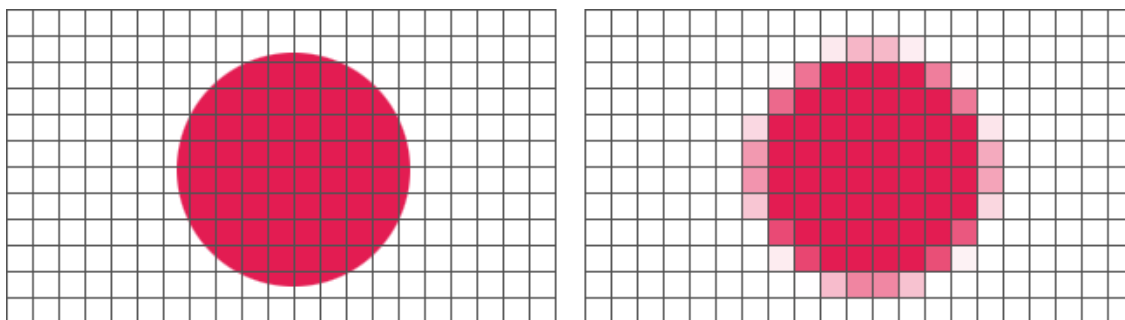
V předchozích několika bodech byly osvětleny jen některé z mnoha oborů využívajících počítačové grafiky.

## 2 RASTERIZACE

Rasterizace je proces, při kterém se vektorově definovaná grafika konvertuje na rastrově definované obrazy. Při zobrazení reálného modelu pomocí souřadnicového systému na výstupní zařízení, potřebujeme zajistit co nejvěrnější podobnost reálného a zobrazovaného modelu. Nejjednodušším grafickým prvkem většiny výstupních (zobrazovacích) zařízení je bod. Protože složitější objekty jsou jen skládkou jednodušších objektů, potřebujeme mít k dispozici algoritmy na výpočet polohy bodů jednoduchých objektů, jako jsou úsečky, kružnice, elipsy, oblouky, atd. [3]

### 2.1 Rastrová a vektorová grafika

V dnešní době se pro ukládání grafických informací v počítačích používají dva způsoby, pomocí tzv. rastrové grafiky a vektorové grafiky. Není možné jednoduše o jedné z nich říci, že je lepší než druhá, každá totiž slouží k jinému účelu. Pro rastrovou grafiku se užívá výrazu malba (paint), pro vektorovou kresba (draw). [4]



Obr. 1 Způsob zpracování obrázku v bitmapové grafice (vpravo rastrová grafika, vlevo vektorová) [5]

Jak je vidět na obrázku 1, rastrová grafika při přiblížení ztrácí kvalitu, zatímco vektorová si ji uchovává při jakémkoliv přiblížení. Důvodem, proč tomu tak je, se budou zabývat následující dvě kapitoly. [4]

#### 2.1.1 Vektorové obrázky

Vektorové obrázky jsou tvořeny základními geometrickými tvary, jako jsou body, čáry a křivky. Vlastnosti tvarů jsou vyjádřeny jako matematické rovnice, které umožňují, aby se obraz dal škálovat nebo přibližovat bez ztráty kvality. Proto se vektor většinou používá pro tvorbu jednodušších tvarů, piktogramů, logotypů a tiskařských prací, jako jsou brožury a

plakáty. Ty by měly být navrženy jako vektorové obrázky pomocí softwaru pro tvorbu vektorových obrazů, jako jsou Adobe Illustrator, Corel Draw nebo Inkscape. [6]

Výhody vektorové grafiky spočívají především v tom, že je

- lze libovolně zvětšovat a zmenšovat bez ztráty kvality,
- objekty ve vektorovém obrázku jsou od ostatních objektů oddělené, dá se s nimi tudíž manipulovat odděleně,
- na rozdíl od rastru se jejich velikost na paměťovém zařízení několikanásobně zmenší.

Naproti tomu mezi nevýhody můžeme řadit například to, že

- pro většinu zobrazovacích zařízení je nutno ji převést na rastrový obrázek,
- neexistuje jednotný formát, z čehož vyplývají problémy s otevíráním a přenosem souborů,
- složitější pořízení obrázku, v rastrové grafice lze obrázek snadno pořídit pomocí fotografie. [7]



Obr. 2 Tvorba logotypu v editoru [8]

### 2.1.2 Rastrové obrázky

Rastrové obrázky jsou tvořeny mřížkou obrazových bodů nazývanými pixely, kde je každému pixelu přiřazena barevná hodnota. Na rozdíl od vektorových obrázků jsou ty rastrové závislé na rozlišení. Pokud změníme velikost rastrového obrázku, může dojít k roztažení nebo zmenšení obrazových bodů, což může vést k výrazné ztrátě ostrosti a obrázek je poté rozmazaný. Pro editaci rastrových obrazů se nejčastěji používají editory jako GIMP nebo Adobe Photoshop, které se hodí pro úpravu fotografií, manipulaci a přidávání různých efektů rastru. [6]

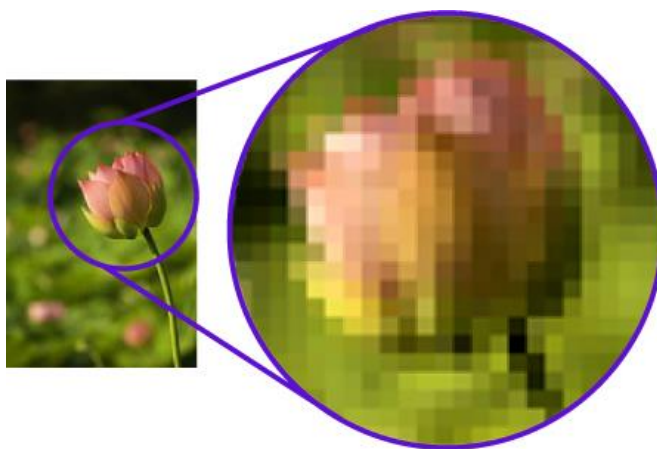
Výhody rastru se projevují například při

- jeho pořizování, které je jednoduché, provádí se například pomocí fotoaparátu a scanneru,
- další výhodou je jednoduché zobrazení a programová podpora.

Nevýhody rastru spočívají

- v problematické změně měřítka,
- velké nároky na zdroje (při velkém rozlišení a barevné hloubce může velikost obrázku dosáhnout několika megabytů, to neplatí při užití komprimovaných formátů).

[7]



Obr. 3 Přiblížení rastrového obrázku [9]

## 2.2 Rasterizace objektů

V následujících kapitolách se budeme věnovat rasterizačním algoritmům čtyř základních grafických objektů – úsečce, kružnici, elipse a křivce. Pro tyto objekty je v praktické části této práce vytvořen ukázkový modul, kde si uživatel může v praxi prohlédnout praktické použití získaných teoretických znalostí.

### 2.2.1 Rasterizace úsečky

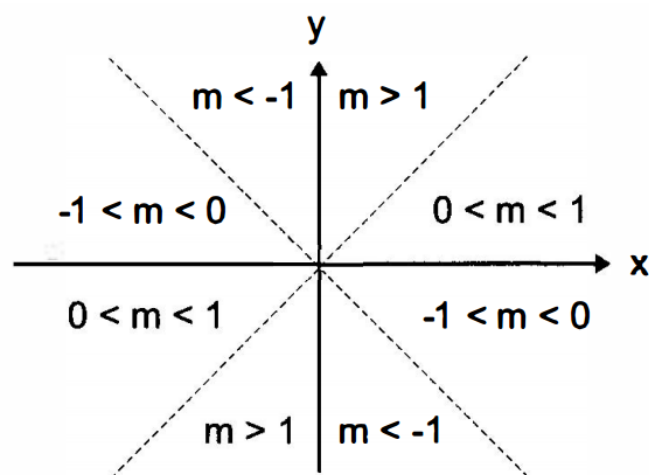
Úsečka patří mezi nejjednodušší grafické prvky. Z úseček lze skládat lomené čáry (polyline), jimiž lze nahradit složitější křivočaré obrazce. Protože úsečka náleží k základním stavebním prvkům počítačové grafiky, je vhodné, aby vykreslení úsečky bylo prováděno úsporně. Lomenou čáru pak tvoří posloupnosti úseček, kdy koncový bod jedné úsečky je počátečním

bodem úsečky navazující. Lomenou čáru lze popsat posloupností bodů nebo využít následnosti úseček, nebo lomenou čáru definovat vždy počátečním bodem a posloupností relativních přírůstků.

Úsečka je vzorkována s konstantním krokem vždy v jedné z os  $x$  a  $y$ , a to podle svého sklonu vyjádřeného směrnici  $m$ . Tato charakteristická hodnota je dána podílem rozdílů souřadnic koncových bodů úsečky:

$$m = \frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_2 - x_1} \cdot [10] \quad (1)$$

Pokud je  $|m| < 1$ , úsečka se přimyká k ose  $x$  a bude proto vzorkována na ose  $x$  s krokem jeden pixel. Úsečky, jejichž absolutní hodnota směrnice je větší než jedna, jsou vzorkovány na ose  $y$ . Osa, v jejímž směru se vzorkuje, se nazývá *řídící* osa, zbylé ose se říká *vedlejší*. Diagonální úsečky ( $|m| = 1$ ) mohou mít řídící osu libovolnou. [11]



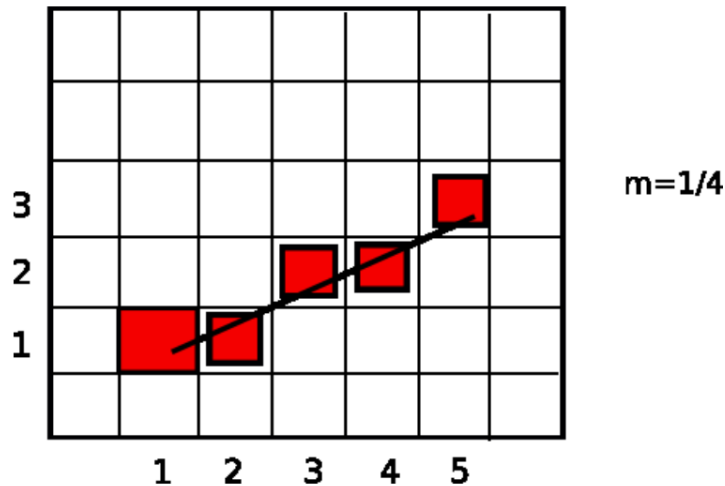
Obr. 4 Velikost směrnice  $m$  pro různé sklony úsečky [11]

### 2.2.1.1 DDA algoritmus

Jedná se o přírůstkový algoritmus pro výpočet bodů úsečky. Postup spočívá v tom, že postupně zvedáme o jeden pixel hodnoty na ose  $x$  a dopočítáváme odpovídající bod  $y$ . Jelikož jsou souřadnice pixelů vždy celá čísla, provede se zaokrouhlením hodnoty a pixel se vykreslí. Algoritmus je jednoduchý ale relativně pomalý, protože pracuje v oboru reálných čísel. [3]

Algoritmus pro rasterizaci se provádí ve směru osy  $x$ . Pro  $|m| > 1$  je prováděna rasterizace ve směru osy  $y$  a v algoritmu se zamění operace s  $x$  a  $y$ . [3]

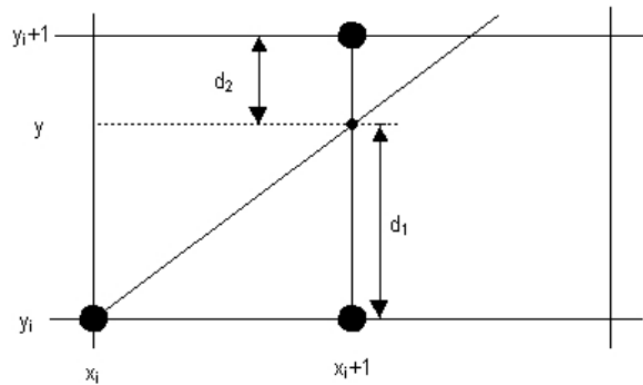
1. Z koncových bodů  $[x_1, y_1]$ ,  $[x_2, y_2]$  urči směrnici  $m$  podle vzorce výše.
2. Inicializuj bod  $[x, y]$  hodnotou  $[x_1, y_1]$ .
3. Dokud je  $x \leq x_2$  opakuj:
  - a) Vykresli bod  $[x, \text{zaokrouhlené}(y)]$
  - b)  $x = x + 1$
  - c)  $y = y + m$



Obr. 5 Vykreslení úsečky algoritmem DDA (rasterizace ve směru osy x) [3]

### 2.2.1.2 Bresenhamův algoritmus pro úsečku

Tento algoritmus byl objeven v šedesátých letech minulého století panem Bresenhamem. Jeho výhoda spočívá v tom, že pracuje pouze s celými čísly. Víme, že od daného pixelu  $[x, y]$  můžeme umístit další pixel (při rasterizaci po ose x) pouze na dvou pozicích –  $[x + 1, y]$  nebo  $[x + 1, y + 1]$ . Rozdíly mezi souřadnicí  $y$  středů uvedených pixelů a skutečnou souřadnicí  $y$  označíme  $d1$ ,  $d2$  a můžeme je celočíselně vypočítat z parametrické rovnice kreslené úsečky. Jejich rozdíl potom vyhodnotíme. Pokud je kladný, je  $d1 > d2$  a bližším pixelem je ten ve vzdálenosti  $d2$ . [3]



Obr. 6 Princip Bresenhamova algoritmu [3]

Pro skutečné souřadnice  $y$  platí:

$$y = k(x_i + 1) + q \quad (2)$$

Potom platí:

$$d_1 = y - y_i = k(x_i + 1) + q - y_i \quad (3)$$

$$d_2 = y_i + 1 - y = y_i + 1 - k(x_i + 1) - q \quad (4)$$

Rozdíl těchto dvou vzdáleností je:

$$\Delta d = d_1 - d_2 = 2k(x_i + 1) - 2y_i + 2q - 1 \quad (5)$$

Celý výpočet je vhodné převést do celočíselné aritmetiky ( $k = \Delta y / \Delta x$ , vynásobíme rovnici  $\Delta d$ ) a rozhodovat pouze podle znaménka.

$$p_i = \Delta d \Delta x = 2\Delta y x_i - 2y_i \Delta x + 2\Delta y + \Delta x(2c - 1) \quad (6)$$

Hodnota  $p_i$  je nazývána predikcí a bude určována iteračně (výpočet další hodnoty  $p_i + 1$  na základě předchozí  $p_i$ ) během výpočtů všech dalších obrazových bodů úsečky.

Hodnota  $2\Delta d + \Delta x(2c - 1)$  je reálná konstanta, které po odečtení dvou následujících rovnic z výpočtu vypadne. Při výpočtu  $p_1$  na začátku výpočtu dosadíme  $x_1, y_1$  a hodnota  $q$  ze směrnice vyjádření přímky a zaokrouhlíme.

$$p_i = 2\Delta y x_i - 2y_i \Delta x + konst \quad (7)$$

$$p_{i+1} = 2\Delta y x_{i+1} - 2y_{i+1} \Delta x + konst \quad (8)$$

Odečtením rovnice za předpokladu, že:  $x_i + 1 - x_i = 1$  dostaneme

$$p_{i+1} = p_i + 2\Delta y - 2\Delta x(y_{i+1} - y_i) \quad (9)$$

Praktický výpočet probíhá v jednom cyklu iteračním způsobem. Na počátku známe první bod úsečky  $[x_1, y_1]$  a určíme predikci  $p_1$ . Při každém průchodu cyklem na základě předchozí hodnoty  $p$  určíme novou hodnotu predikce a rozhodneme o umístění následujícího pixelu na stejné úrovni osy  $y$ , jako měl předchozí pixel, případně o jeho posunutí o jednu úroveň výše. Rozhodovací logika je zapsána pomocí následujících vztahů [12]:

$$(p_i \leq 0) \rightarrow y_{i+1} = y_i; p_{i+1} = p_i + 2\Delta y \quad (10)$$

$$(p_i > 0) \rightarrow y_{i+1} = y_i + 1; p_{i+1} = p_i + 2\Delta y - 2\Delta x \quad (11)$$

### 2.2.2 Rasterizace kružnice

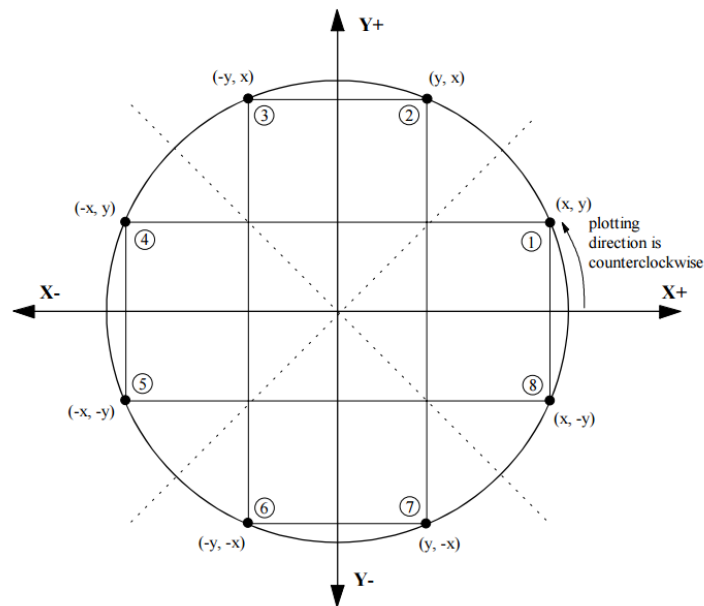
Rasterizace kružnice se většinou provádí pro kružnici se středem v počátku souřadného systému a po skončení výpočtů je posunuta na souřadnice zadaného středu. Rasterizace se provádí pouze pro jednu osminu kružnice (jeden oktan), zbytek není třeba počítat díky symetričnosti kružnice. [13] Při kresbě kružnice v rastru můžeme využít již existujících metod pro kresbu úsečky a nahradit kružnici lomenou čarou. Její vrcholy lze vypočítat iteračním způsobem pomocí transformace otáčení. Uvedený postup je nepřesný, na druhou stranu však velmi rychlý.

Další z variant rasterizace kružnice je rasterizovat na základě polárních souřadnic:

$$x = x_c + r \cdot \cos \alpha \quad (12)$$

$$y = y_c + r \cdot \sin \alpha \quad (13)$$

Úhel  $\alpha$  nabývá hodnot od 0 do  $\frac{\pi}{4}$ . Při konstantním kroku změny budou body rozloženy na kružnici rovnoměrně. Tento krok by měl být roven hodnotě  $\frac{1}{r}$ , kdy se body liší o jeden pixel. Zbýlých 7 oktanů se získá změnou znaménka nebo přehozením souřadnic získaných bodů. Tyto metody jsou však značně neefektivní a výpočetně náročné, protože používají násobení a trigonometrické výpočty. [13]



Obr. 7 Symetrie kružnice [14]

### 2.2.2.1 Bresenhamův algoritmus pro kružnici

Bresenham svůj algoritmus uplatňuje i v otázce generování bodů na kružnici, podobným způsobem jako v případě úsečky. Někdy je v literatuře označován také jako *midpoint algorithm*. [11]

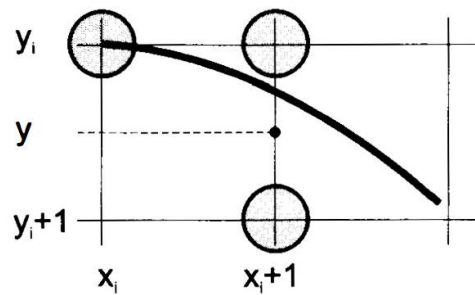
Pro body na kružnici platí implicitní rovnice  $x^2 + y^2 - r^2 = 0$ , kterou zapíšeme jako funkci:

$$F(x, y) : x^2 + y^2 - r^2 = 0 \quad (14)$$

Znaménko nám určí polohu bodů  $[x, y]$  vůči kružnici. Funkční hodnota pro vnitřní body kružnice je záporná, pro vnější body je hodnota kladná. Funkce  $F$  je vhodné kritérium při zavádění rozhodovacího členu:

$$p_i = F\left(x_i + 1, y_i - \frac{1}{2}\right) = (x_i + 1)^2 + \left(y_i - \frac{1}{2}\right)^2 - r^2 \quad (15)$$

Je-li znaménko  $p_i$  záporné, bude pro další kresbu vybrán bod se stejnou souřadnicí  $y_i$ , jinak bude nakreslen bod ležící o jeden pixel níže.



Obr. 8 Část kružnice v rastru [11]

Obdobně jako v případě úsečky hodnotu rozhodovacího členu získáme na základě předchozí hodnoty. Po úpravě získáváme výsledný vzorec:

$$p_{i+1} = p_i + 2x_i + 3 + \left(y_i - \frac{1}{2}\right)^2 + \left(y_{i+1} - \frac{1}{2}\right)^2. \quad (16)$$

Po vyčíslení zjistíme podle znaménka polohu následujícího pixelu na základě kritérií:

$$P_k \leq 0 \rightarrow P_{k+1} = P_k + 2x_k + 3, y_{k+1} = y_k, \quad (17)$$

$$P_k > 0 \rightarrow P_{k+1} = P_k + 2x_k + 5 - 2y_k, y_{k+1} = y_k + 1 \quad (18)$$

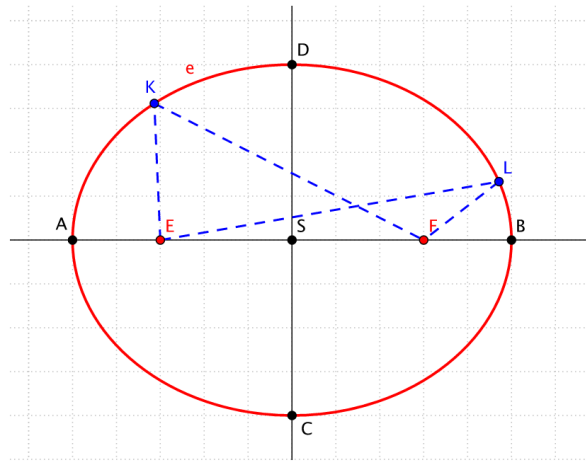
Algoritmus se také využívá pro rasterizaci kruhového oblouku. Podle koncových bodů se stanoví omezující podmínky. Kresba bodů mimo oblouk je poté podmínkami znemožněna. [11]

### 2.2.3 Rasterizace elipsy

Elipsa je kuželosečka. Základní vlastností elipsy je, že každý bod elipsy má od daných dvou bodů v rovině stejný součet vzdáleností. Těmto bodům se říká ohniska. Máme-li dva body  $E$  a  $F$ , ohniska elipsy  $e$ . Pak pro každý bod  $X$  elipsy  $e$  platí vztah:

$$|XE| + |XF| = K \quad (19)$$

Kde  $K$  je nějaké konstantní číslo. Toto číslo je tak pro všechny body elipsy stejné. V případě, že  $E = F$ , dostáváme kružnici a platí, že  $|XE| + |XF|$  se rovná průměru kružnice (neboli  $|XE|$  musí být poloměr kružnice). [15]



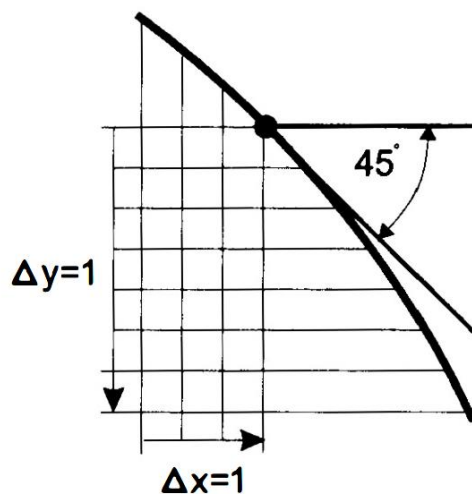
Obr. 9 Elipsa s ohnisky E a F [15]

### 2.2.3.1 Bresenhamův algoritmus pro elipsu

Pro rasterizaci elipsy stejně jako u kružnice je nejjednodušší použít iterační postup Bresenhamova algoritmu, s nímž lze nalézt body pomocí celočíselné aritmetiky. Na rozdíl od kružnice lze symetrii využívat pouze pro tři další body a algoritmus musí vygenerovat body elipsy v jednom celém kvadrantu. Zbylé body dopočítáme na základě symetrie elipsy. Rovnici elipsy se středem v počátku vyjádříme pomocí implicitní funkce:

$$F(x, y) : b^2x^2 + a^2y^2 - a^2b^2 = 0. \quad (20)$$

Z této rovnice poté za pomoci středového bodu odvodíme rozhodovací člen stejně jako u rasterizaci kružnice. [11]



Obr. 10 Změna řídicí osy při rasterizaci elipsy [11]

Rasterizace elipsy je složitější tím, že v průběhu jednoho kvadrantu se mění i řídicí a vedlejší osa. V bodě, ve kterém dojde ke změně, má tečna elipsy směrnici  $-1$ . [6] Tento bod se po vyjádření z rovnice elipsy nachází na souřadnicích, které se vypočítají podle vztahu:

$$Z = \left[ \frac{a^2}{\sqrt{a^2 + b^2}}, \frac{b^2}{\sqrt{a^2 + b^2}} \right]. \quad (21)$$

Zde  $a$  a  $b$  jsou délky poloos. V prvním kvadrantu při pohybu zleva doprava je nejprve řídicí osa  $x$ , od bodu  $Z$  dále se pak role os zamění. [10]

V části s řídicí osou  $x$  se pro výpočet rozhodovacího členu použijí tato pravidla:

$$d_i \leq 0 \rightarrow d_{i+1} = d_i + b^2(2x_i + 1) \quad (22)$$

$$d_i > 0 \rightarrow d_{i+1} = d_i + b^2(2x_i + 1) - 2a^2y_i \quad (23)$$

Analogické vztahy platí i pro druhou část algoritmu. Vhodnou volbou počáteční inicializace a vhodnou volbou proměnných a jejich významu lze opět celý výpočet řešit celočíselně. [10]

#### 2.2.4 Rasterizace křivky

Křivky patří mezi nejpoužívanější objekty počítačové grafiky. Jednoduchý případ křivky je i například kružnice nebo přímka. Křivku můžeme chápat jako dráhu pohybujícího se bodu. Používají se například pro definici fontů, modelování ve 3D a kreslení ve 2D.

Křivku v počítači můžeme vyjádřit třemi způsoby, buď implicitně, explicitně nebo parametricky:

- Implicitně:  $F(x, y, z) = 0$
- Explicitně:  $z = f(x, y)$
- Parametricky:  $x = x(t)$ ,  $y = y(t)$ ,  $z = z(t)$

Základním druhem parametrických křivek používaných v počítačové grafice jsou křivky polynomiální:

$$Q(t) = a_0 + a_1 t^2 \dots a_{n-1} t^{n-1} + a_n t^n \quad (24)$$

Stupeň polynomu se určí podle nejvyšší mocniny argumentu, zde proměnné  $t$ . Nejpoužívanější křivky jsou třetího stupně, takzvané *kubiky*. Základní vlastnosti kubik:

- jsou dostatečně tvarovatelné,
- jejich výpočet je nenáročný,

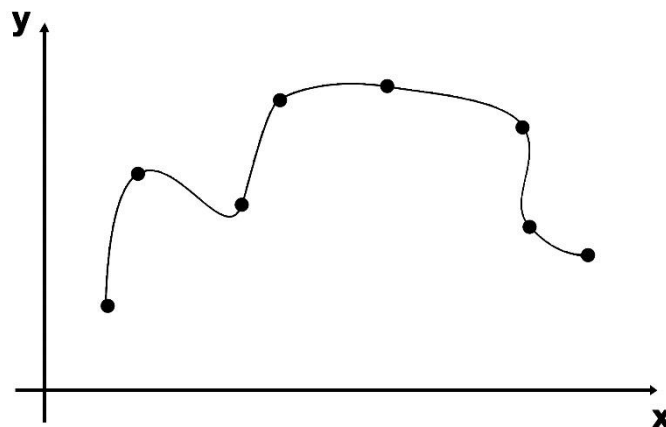
- lze s nimi snadno manipulovat,
- je u nich možné zaručit spojitost  $C^2$  (spojitost i první derivace tečného vektoru) často požadovanou při modelování v systémech CADU.

Při modelování křivky je možné zadávat koeficienty polynomických křivek číselně. Běžnější způsob je však pomocí definice řídicích bodů a na základě jejich polohy určit průběh křivky. Pokud požadujeme spojitost a hladkost navázání segmentů, je možno křivky zadat i pomocí tečných vektorů.

Existují dva základní druhy interpretace řídicích bodů – interpolace a aproximace. [10]

#### 2.2.4.1 Interpolační křivky

Křivka, která je modelována pomocí interpolace, prochází řídicími body.



Obr. 11 Interpolační křivka

V praxi se nejčastěji k interpolování využívána technika interpolace polynomem, a to buď jediným polynomem  $n-1$  řádu pro  $n$  bodů, nebo po částech.

##### 2.2.4.1.1 Interpolace jediným polynomem

Interpolace jediným polynomem  $n-1$  řádu znamená najít řešení soustavy rovnic v maticovém tvaru:

$$y_i = \begin{vmatrix} a_{11} & \cdot & \cdot & \cdot & a_{1n} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ a_{n1} & \cdot & \cdot & \cdot & a_{nn} \end{vmatrix} \begin{vmatrix} 1 \\ x_i \\ x_i^2 \\ \cdot \\ \cdot \\ x_i^{n-1} \end{vmatrix} \quad (25)$$

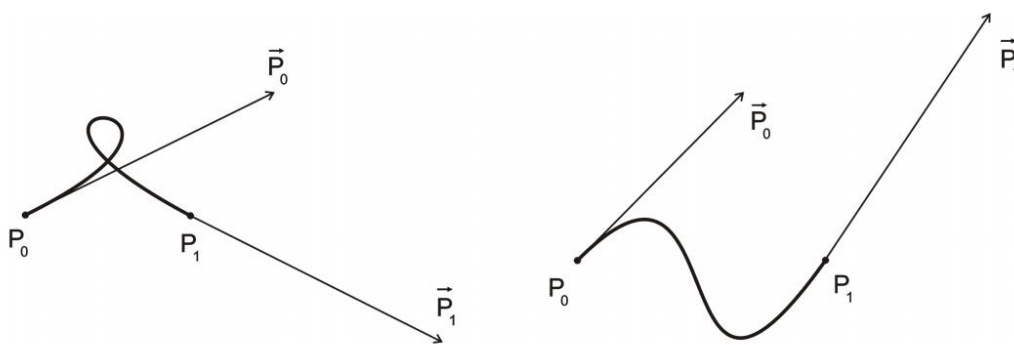
Vstupními parametry této rovnice jsou body, jimiž má křivka procházet, a jejím řešením jsou parametry matice  $a_{ij}$ . Do výrazu se postupně dosazují body, jimiž má křivka procházet, z nichž se získá soustava rovnic. Nevýhodou polynomiální interpolace je, že polynomy vyššího řádu mohou vnutit křivce nepřirozené vlnění. V počítačové praxi bývá polynomiální interpolace používána pro křivky stupně maximálně pět.

#### 2.2.4.1.2 Interpolace po částech

Častěji využívaná technika pro interpolaci křivky je interpolace po jejích částech. Uvažujeme-li  $n$ -tici bodů. Interpolací po částech polynomem třetího řádu potom rozumíme interpolaci jejích každých čtyř bodů polynomem třetího stupně. Problémem, který je potřeba poté řešit, je navazování křivek v bodech, kde jeden segment přechází v druhý. Existuje několik metod, které řeší navazování v těchto bodech automaticky. Akimova interpolace, interpolace Hermitovskými polynomy (jejímž zvláštním případem jsou Fergusonovy kubiky, které se používají i pro aproximace) a konečně splynutí křivky  $k$ -tého řádu, které zaručuje spojitost  $k-1$  řádu. [16]

#### 2.2.4.1.3 Fergusonovy kubiky

Tuto křivku zavedl v roce 1964 J. C. Ferguson. Jedná se o křivky určenými dvěma řídicími body  $P_0$  a  $P_1$  (jimiž křivka prochází) a tečnými vektory v těchto bodech  $\vec{P}_0$  a  $\vec{P}_1$ . Křivka začíná v bodě  $P_0$  a  $P_1$ . [17]



Obr. 12 Fergusonova kubika [13]

Čím větší jsou vektory  $\vec{P}_0$  a  $\vec{P}_1$ , tím více k nim křivka přimyká. Pokud je velikost obou vektorů nulová, získáme úsečku. Matice křivky je:

$$Q(t) = T \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ \vec{P}_0 \\ \vec{P}_1 \end{bmatrix}, \quad (26)$$

$$T = [t^3 t^2 t^1], t \in \langle 0, 1 \rangle. \quad (27)$$

Úpravou získáme tuto rovnici:

$$P(t) = P_0 F_1(t) + P_1 F_2(t) + P_0 F_3(t) + P_1 F_4(t), \quad (28)$$

kde  $F_1, F_2, F_3, F_4$ , jsou kubické Hermitovské polynomy varu:

$$F_1(t) = 2t^3 - 3t^2 + 1, \quad (29)$$

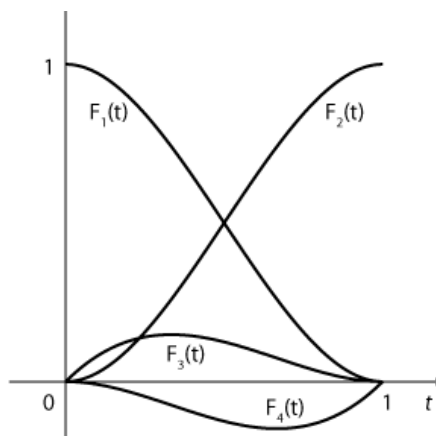
$$F_2(t) = -2t^3 + 3t^2, \quad (30)$$

$$F_3(t) = t^3 - 2t^2 + t, \quad (31)$$

$$F_4(t) = t^3 - t^2. \quad (32)$$

Dosazením hodnoty  $t = 0$ , resp.  $t = 1$  lze ukázat, že křivka bude začínat v bodu  $P_0$ , resp. končit v bodu  $P_1$ . [10]

Při navazování Fergusonových kubik do složených křivek se projeví výhody jejich zadávání dvěma body a dvěma tečnými vektory. Pokud u počátečního bodu jednoho segmentu použijeme jako tečný vektor ten, který byl přiřazen u předcházejícího segmentu ke koncovému bodu, dostaneme automaticky spojitost  $C1$ . Pokud nám stačí splnit  $G1$  spojitost, nemusí být oba vektory stejně veliké, ale stále musí být lineárně závislé.

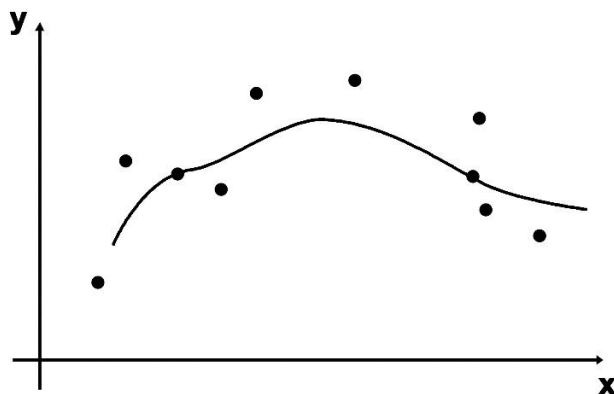


Obr. 13 Kubické Hermitovské polynomy [18]

Nevýhodou Fergusonových kubik je poměrně obtížná editace tečných vektorů v třírozměrném prostoru a určitá neintuitivnost vytváření tvaru křivky. Obecně je vhodné tyto křivky použít jen v případě, kdy jsou známy tečné vektory (jak směr, tak i velikost) v jednotlivých uzlech (body, ve kterých se napojují jednotlivé segmenty křivky) složené křivky. [18]

#### 2.2.4.2 Aproximační křivky

Při aproximaci je řídicími body určen tvar křivky, ta jimi však procházet nemusí. Způsob řízení odpovídá vytváření této křivky. Existuje v zásadě dvojí přístup k aproximacím.



Obr. 14 Aproximační křivka

Přístup první je znám z numerické matematiky a jeho cílem je smysluplná interpretace vstupních dat. Uvedme metodu nejmenších čtverců, její smyslem je nalézt křivku, jež je hladká, a čtverec vzdálenosti řídicích bodů od ní je minimální. Neměnným základem těchto postupů jsou zadané body. Generovaná křivka má jen informativní charakter.

Druhý přístup je používán v počítačové grafice. Smyslem není interpretace bodů, ale generování křivky. Křivka může být řízena body, potom hovoříme o tzv. řídicím polygonu nebo o bodech a vektorech. Metoda, která křivku vytváří, zaručuje její vlastnosti. Požadované vlastnosti jsou její hladkost, spojitost, počet inflexí aj. Tyto metody byly navrženy tak, aby vyhovovaly technické praxi, a zaručují například nízké tření navržených objektů apod.

V počítačové grafice se nejčastěji používá aproximace po částech (podobně jako interpolace), a to obvykle kubikami (tedy křivkami, které jsou generovány polynomy třetího řádu). Důvody pro to jsou dva. Kubiky jsou dostatečně "pružnými" křivkami, aby se jimi dalo vyjádřit téměř vše, co je v praxi potřeba. Dalším důležitým faktorem je, že stupeň polynomu tři umožňuje velmi rychlý výpočet výsledné křivky, což je výhodné pro její interaktivní tvorbu.

Parametricky lze danou kubiku  $Q(t)$  vyjádřit ve tvaru:

$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x \quad (33)$$

$$y(t) = a_y t^3 + b_y t^2 + c_y t + d_y \quad (34)$$

$$z(t) = a_z t^3 + b_z t^2 + c_z t + d_z \quad (35)$$

nebo také zkráceně v maticovém tvaru:

$$Q(t) = TC = [t^3, t^2, t, 1] \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix}. \quad (36)$$

Derivaci  $q'(t)$  získáme derivací vektoru  $T$

$$q'(t) = \frac{d}{dt} q(t) = \frac{d}{dt} TC = [3t^2, 2t, 1, 0]C. \quad (37)$$

Konstantní matice  $C$  můžeme rozepsat do součinu

$$C = MG, \quad (38)$$

kde matice  $M$  je typu  $4 \times 4$  a nazývá se bázová matice. Čtyř prvkový vektor se nazývá geometrický vektor. Geometrický vektor reprezentuje vliv vnějších parametrů. Obsahuje řídicí body nebo řídicí body a tečné vektory. Bázová matice, která je dána použitou metodou, poté určuje výpočet křivky podle vztahu:

$$Q(t) = [x(t), y(t), z(t)] = [t^3, t^2, t, 1] \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} \begin{bmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \end{bmatrix}. \quad [16] \quad (39)$$

#### 2.2.4.2.1 Bézierovy kubiky

Nejčastěji používanými křivkami jsou Bézierovy kubiky, která jsou určena čtyřmi body  $P_0$ ,  $P_1$ ,  $P_2$  a  $P_3$ . Vychází z prvního řídicího bodu a končí v posledním. Je určena vztahem:

$$Q(t) = \sum_{i=0}^3 P_i B_i(t). \quad (40)$$

Maticový zápis Bézierovy kubiky je:

$$B_0(t) = (1 - t)^3 \quad (41)$$

$$B_1(t) = 3t(1 - t)^2 \quad (42)$$

$$B_2(t) = 3t^2(1 - t) \quad (43)$$

$$B_3 = t^3 \quad (44)$$

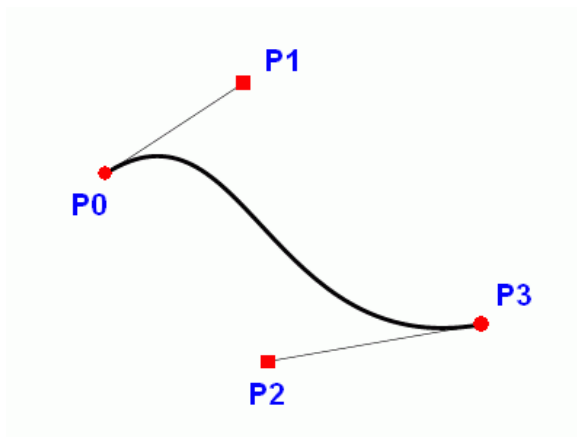
Tečné vektory v prvním a posledním bodě mají tvar:

$$\vec{p}(0) = 3(P_1 - P_0) \quad (45)$$

$$\vec{p}(1) = 3(P_3 - P_2) \quad (46)$$

Postup, jakým převedeme kubiku v Bézierově reprezentaci (čtyři řídicí body) na kubiku v Hermitovské reprezentaci (dva body a dva vektory), vychází ze vztahu pro tečné vektory.

[10]



Obr. 15 Bézierova kubická křivka [19]

### 3 KOMPRESSE RASTROVÉHO OBRAZU

Rastrová grafika je známá vysokou paměťovou náročností, která se kvadraticky navyšuje s rostoucím rozlišením obrazu, z tohoto důvodu je žádoucí rastrová data komprimovat. Komprese obrazu je efektivní způsob převodu dat do úspornější formy, která má téměř stejnou informační hodnotu jako původní obraz. Rozlišujeme zde tři základní veličiny: kompresní poměr, což je poměr mezi velikostí komprimovaného a původního obrazu, rychlost komprese a rychlost dekomprese. Existují dva základní druhy komprese: *ztrátová (lossy)* a *bez-ztrátová (lossless)*. [20] Kompresních metod existuje nepřehledné množství, v následující tabulce jsou zobrazeny ty nejpoužívanější.

Kompresní metoda	Zkratka	Ztrátová?	Formáty souborů
Run Length Encoding	RLE	ne	PCX, TGA
Huffmanovo kódování	CCITT	ne	TIFF
Lempel-Ziv-Welch	LZW	ne	GIF, PNG (ZIP)
Diskrétní kosinová transformace	DCT	ano	JPEG
Fraktální komprese	FIF	ano	FIF

Tab. 1 Přehled kompresních metod [20]

Jelikož je téma kompresních algoritmů velmi rozsáhlé, bude podrobně rozepsána pouze metoda LZW, pro kterou bude v praktické části vytvořen výukový kurz. Ostatní metody v následujících kapitolách budou popsány pouze obecně.

#### 3.1 Ztrátová

Ztrátová komprese je způsob snižování paměťové náročnosti souboru. Pomocí speciálního algoritmu se vypouští některé méně důležité informace, jež má za následek snížení výsledné velikosti souboru na jeho zlomek. Takto ztracené informace už se nedají zpětně zrekonstruovat.

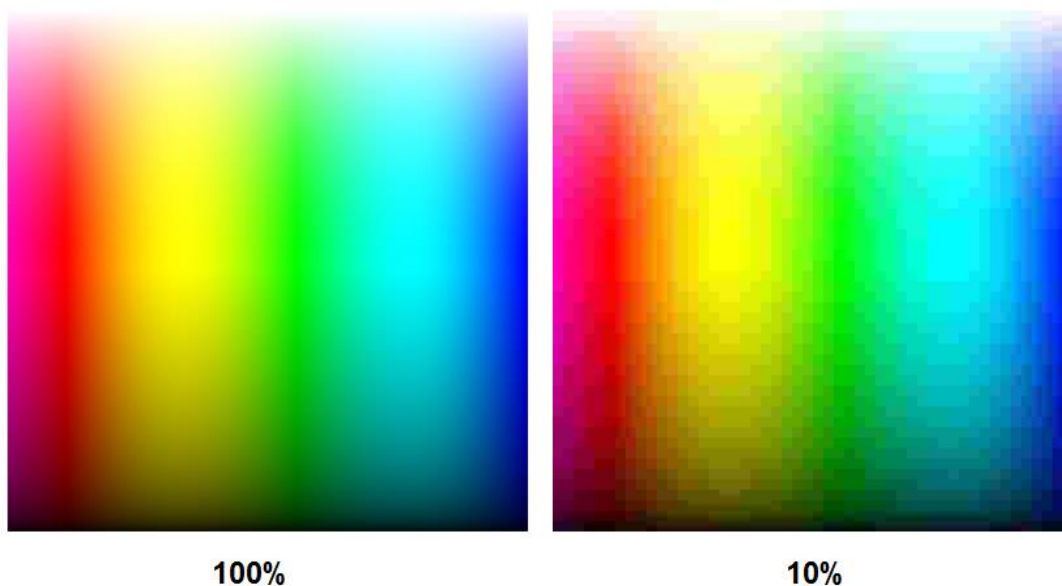
##### 3.1.1 Diskrétní kosinová transformace

Diskrétní kosinová transformace zkráceně DCT (Discrete cosine transform) patří do skupiny metod provádějících takzvané transformační kódování nad diskretním (vzorkovaným) jednorozměrným či vícerozměrným signálem. Je podobná diskretní Fourierově transformaci

(DFT) s tím rozdílem, že produkuje pouze reálné koeficienty. Transformací se získají spektrální matice. Ve frekvenční oblasti lze jednoduše rozlišit, které části grafické informace jsou pro danou oblast dominantní (především nižší frekvence) a které popisují jen jemné detaily (vyšší frekvence). Ty se potom mohou zanedbat například kvantováním. Existuje několik typů diskrétní kosinové transformace, které jsou označovány římskými čísly DCT I, DCT II až DCT VIII. Nejpoužívanějším DCT je typ II, používá se například v souborových formátech i formátech určených pro přenos videa. Například se jedná o standardy JPEG, MPEG, JVT a H.261. Vzorec jednorozměrné DCT II:

$$x_k = \sum_{n=0}^{N-1} x_n \cos \left[ \frac{\pi}{N} \left( n + \frac{1}{2} \right) k \right]. \quad [21] \quad (47)$$

*„Metoda je vhodná především pro kódování fotografií, u nichž sousední pixely mají sice odlišné, ale přesto blízké barvy. Snižování kvality se projevuje potlačováním rozdílů v blízkých barvách. U metody DCT je kompresní poměr řízen požadavkem na výši kvality dekomprimovaného obrazu. V praxi se ukazuje, že snížení kvality na 75 % je pro většinu uživatelů nepozorovatelné, přitom kompresní poměr v takovém případě může být až 25:1.“ [22]*



Obr. 16 Příklad původního a obrazu po DCT kompresi

### 3.1.2 Fraktální komprese

Moderní metoda ztrátové komprese, která se teprve rozvíjí. Patří mezi nesymetrické kompresní postupy, kde se doba komprese významně liší. V závislosti na rozlišení a obsahu obrazových dat je třeba provést miliony až miliardy iterací, komprese může trvat několik hodin. Jejím základem je vyhledávání podobností tvarových i barevných v různě velkých částech

obrazu. Algoritmus se nejprve snaží vhodně rozdělit obraz na menší nestejně velké části, takzvané domény, a poté z nich pomocí různých transformací poskládat celý obraz. Výsledná kvalita obrazu je lepší než u metody DCT. Algoritmus je schopen v obraze nalézt i základní objekty, např. úsečku, které je schopen převést do vektorové podoby a tímto je možné příslušné objekty transformovat do libovolné přesnosti a rozlišení. Jedinou nevýhodou zůstává delší doba komprese nutná pro analýzu obrazu. [22]

## 3.2 Bezztrátová

Jedná se o takový druh komprese, kdy je možná přesná zpětná rekonstrukce komprimovaných dat. Nedochází tedy ke ztrátě dat.

### 3.2.1 Lempel – Ziv – Welch

LZW je jeden z nejvýznamnějších slovníkových algoritmů. Používá se například jako základ pro program ZIP, v grafice pak pro kompresi obrázků ve formátu GIF, TIFF a také hraje důležitou roli v algoritmu PDF.

Princip algoritmu je založen na opakování frází ze slovníku. Algoritmus LZW je poměrně jednoduchý na implementaci, ale je nutné řešit velkou paměťovou náročnost slovníku. Tento nedostatek lze řešit různě, například "zmražením" slovníku při určité velikosti, jeho vymazáním, odstraněním dlouho nepoužitých frází a podobně. Algoritmus nejprve provádí inicializaci, kdy pro každý znak ve vstupním řetězci vytváří záznam ve slovníku. [23]

#### 3.2.1.1 Komprese

Pokud máme vstupní řetězec například *ABABBACAACAAB*, bude výsledný slovník po inicializaci obsahovat data, jak je zobrazeno v tabulce 2.

Index	Hodnota
0	A
1	B
2	C

Tab. 2 Slovník po inicializaci

V další fázi hledáme nejdelší řetězec na vstupu, který se již nachází ve slovníku. Pokud jej nalezneme, zapíšeme jeho index ve slovníku na výstup. Frázi ze vstupu odstraníme a její

hodnotu včetně následujícího znaku zapíšeme jako nový záznam do slovníku. Tímto způsobem pokračujeme, dokud neprojdeme řetězec na vstupu celý. Na konci komprese našeho vstupního řetězce bude situace vypadat následovně:

Index	Hodnota
0	A
1	B
2	C
3	AB
4	BA
5	ABB
6	BC
7	CA
8	AA
9	AC
10	CAA

Tab. 3 Slovník po kompresi

Výstup komprese je *013120073*.

### 3.2.1.2 Dekomprese

Při dekompresi je také nejprve potřeba inicializovat slovník viz tabulka 2. Postup dekomprese probíhá tak, že na vstupu máme posloupnost čísel *013120073*. Odebereme číslo ze vstupu, ve slovníku nalezneme frázi pod tímto indexem, kterou zapíšeme na výstup. Jako novou frázi přidáme frázi z předchozího kroku spolu s prvním znakem z aktuální fráze (ne-děje se v prvním kroku).

Index	Hodnota
0	A
1	B

2	C
3	AB
4	BA
5	ABB
6	BC
7	CA
8	AA
9	AC

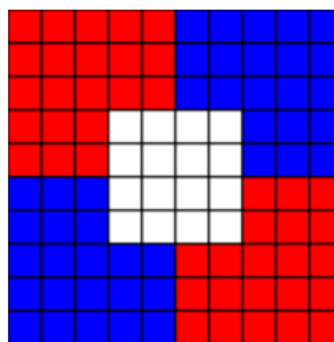
Tab. 4 Tabulka po dekompresi

Výstup po kompresi je *abacdacacadaad*.

Tento postup má však jednu výjimku. Pokud je na vstupu větší číslo, než je nejvyšší index ve slovníku, vloží se znovu na výstup poslední výstup spolu s jeho prvním znakem. To samé se přidá také do slovníku.

### 3.2.1.3 LZW při kompresi GIF

Jak už bylo zmíněno v úvodu do algoritmu LZW, využívá se tento algoritmus při kompresi obrázkových dat formátu GIF.



Obr. 17 Několikanásobně zvětšený GIF obrázek s vyznačenými pixely.

Mějme tedy například Obr. 17. Při zobrazení obrázkových dat v šestnáctkové soustavě získáme data zobrazená na Obr. 18.

```

47 49 46 38 39 61 0A 00 0A 00 91 00 00
FF FF FF FF 00 00 00 00 FF 00 00 00 21
F9 04 00 00 00 00 00 2C 00 00 00 00 0A
00 0A 00 00 02 16 8C 2D 99 87 2A 1C DC
33 A0 02 75 EC 95 FA A8 DE 60 8C 04 91
4C 01 00 3B
    
```

Obr. 18 Obrázková data formátu GIF v šestnáctkové soustavě

Červeně je podbarven blok, ve kterém je uložena barevná paleta, která je v obrázku použita.

Index	Kód barvy (RGB)	Barva
0	FF FF FF	Bílá
1	FF 00 00	Červená
2	00 00 FF	Modrá
3	00 00 00	Černá

Tab. 5 Barvy obsažené v obrázku

Modrou barvou na Obr. 18 jsou vyznačena obrázková data komprimovaná pomocí LZW. První byte tohoto bloku je LZW, minimální velikost tohoto kódu je v našem případě  $0x02$  (dekadicky 2), a hned po něm následují byte udávající celkovou velikost bloku  $0x16$  (dekadicky 22). Nakonec se umísťuje terminátor pro ukončení snímku, který má hodnotu  $0x00$ . Kromě barev zmíněných v tabulce (Tab. 5) GIF využívá ještě 2 speciální kódy, *Clear Code* (CC) a *End Of Information Code* (EOI). CC je pokyn pro znovu inicializování tabulky a EOI označuje konec snímku.

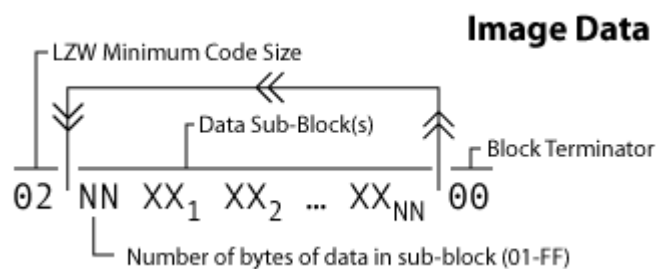
Datový blok s obrázkovými daty vzniká tak, že se nejprve vytvoří tabulka použitých barev v obrázku tak, jak je tomu u Tab.5. Dále podle ní na základě našeho obrázku vytvoříme datový vstup a očíslováme pixely podle indexu jejich barvy v tabulce.

1	1	1	1	1	2	2	2	2	2
1	1	1	1	1	2	2	2	2	2
1	1	1	1	1	2	2	2	2	2
1	1	1	0	0	0	0	2	2	2
1	1	1	0	0	0	0	2	2	2
2	2	2	0	0	0	0	1	1	1
2	2	2	0	0	0	0	1	1	1
2	2	2	2	2	1	1	1	1	1
2	2	2	2	2	1	1	1	1	1
2	2	2	2	2	1	1	1	1	1

Obr. 19 Očíslované barevné pixely podle tabulky barev.

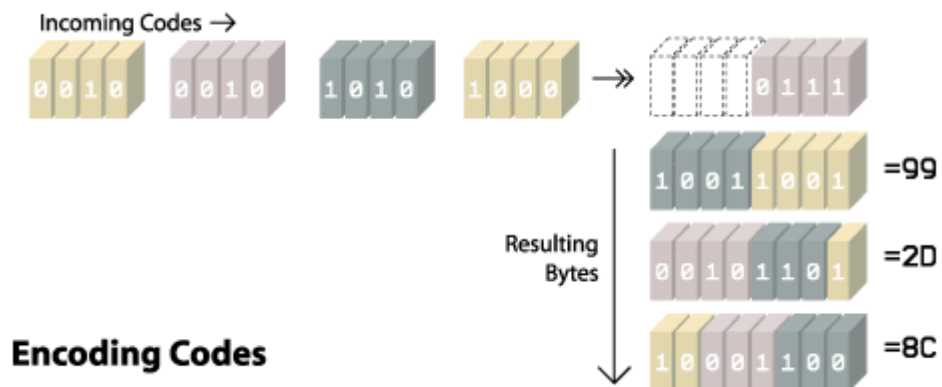
Na vstup pro kompresi LZW posíláme bity po řádcích ve směru zleva doprava, kdy začínáme levým horním pixelem. Na začátku vstupu vložíme Clear Code, který má v našem případě index 4, a na konec End Of Information Code s indexem 5.

Takový výstup však nemůže stát v obrázkových datech, je zapotřebí uložit výstup LZW komprese jako obrázková data, tedy převést do binární podoby. Kvůli úspoře místa se u formátu GIF využívá proměnlivá délka kódu. Délkou kódu je myšlen potřebný počet bitů pro uložení hodnoty kódu. Musí se tedy zjistit délka každého kódu. GIF umožňuje minimální velikost kódu 2 a maximální 12. Pokud je v obrázku 32 barev, bude potřeba 5 bitů na pixel (čísla 0 až 31, kde 31 v binárním kódu je 11111). V našem případě, jsou zde barvy 4, budou potřeba bity 2. I když se jedná o minimální délku kódu, není ve skutečnosti minimální, pouze vyjadřuje minimální počet bitů v případě vyjadřování barev. Stále jsou zde dvě speciální hodnoty CC a EOI, a proto se přidá jeden bit navíc. [24]



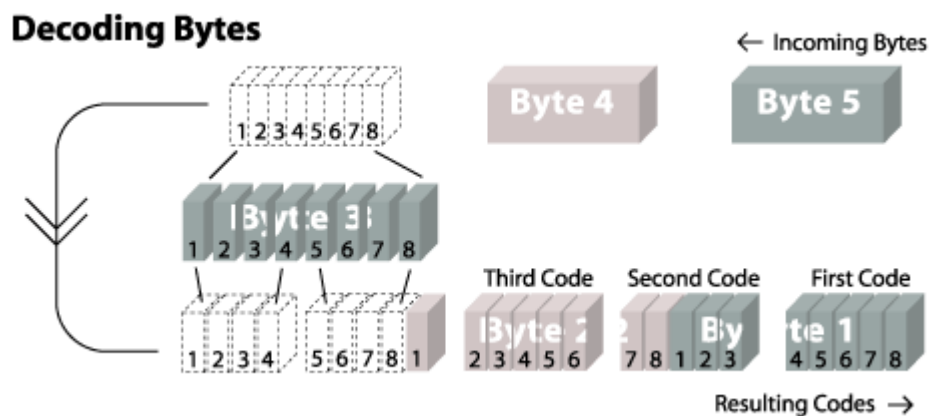
Obr. 20 Datový blok formátu GIF [24]

Pokud je známo, kolika bity se bude vyjadřovat první kód, nesmí se zapomenout, že se délka kódu bude měnit v závislosti na velikosti hodnot na vstupu. Pokud binární hodnota na vstupu přesahuje velikost minimální délky kódu, zvýší se tato hodnota o jedna. Maximální délka je 12. Pokud se tato hodnota překročí, následující kód by měl být CC, dojde tedy k reinicializaci tabulky a nastavení minimální délky kódu na počáteční hodnotu. Postup je znázorněn na Obr. 21. [24]



Obr. 21 Průběh převodu hodnot při LZW kompresi do binárního kódu [24]

Převod kódů z bytů je v podstatě stejný postup v opačném pořadí. Následující ilustrace zobrazuje, jak by vypadal postup při velikosti bloku 5 bitů.



Obr. 22 Dekomprese binárního kódu v LZW [24]

### 3.2.2 Run length encoding

Tato kompresní metoda je jednoduchá, rychlá a pro velkou třídu obrázků i velmi efektivní. Používají ji například grafické formáty TIFF, BMP a PCX. Je založena na předpokladu, že se v rastrovém obrázku hodnoty sousedících pixelů opakují. Do souboru se zapisuje počet opakujících se totožných hodnot a následně samotná hodnota.

Metoda RLE je vhodná pro barevné rozlišení 1 až 8 bitů na pixel a její hlavní uplatnění je při kompresi obrázku, kde se vyskytují plochy pixelu o stejných barvách. Nepoužívá se pro kódování pixelů definovaných přímo hodnotami RGB. V tomto případě totiž není zajištěna sousednost opakujících se pixelů a může dojít k tzv. záporné kompresi. [20]

### 3.2.3 Huffmanovo kódování

Tuto metodu lze obecně aplikovat na jakákoliv data. Princip Huffmanova kódování je založen na použití různě dlouhých bitových kódů pro symboly s různou frekvencí výskytu. Na vstupu je množina znaků s četností jejich výskytů. Výstupem je Huffmanův strom, s jehož pomocí lze znaky zakódovat do binárního kódu a také je zpětně dekodovat. Je nutné tedy spolu se zprávou přenášet i onen strom (v případě že není předem dohodnut).

Prakticky jde o to, že častěji používané znaky zakódujeme menším počtem bitů než znaky méně časté. Výsledný kód se nazývá prefixový, protože žádné kódové slovo není prefixem nějakého jiného. To zaručuje jednoznačnost kódování i dekodování. [25]

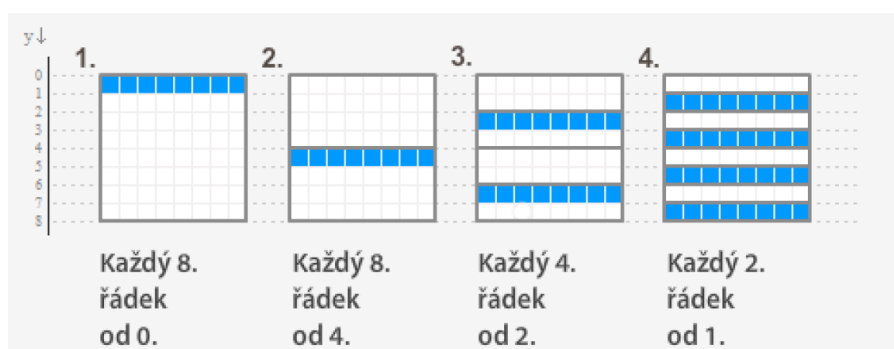
## 4 PROKLÁDÁNÍ OBRÁZKŮ

Prokládání obrazu je technika kódování, která se používá pro to, aby bylo možné rozpoznat obrázek ještě před úplným načtením všech dat. Nejprve se nám zobrazí zhoršená kopie obrazu a s postupným načítáním dat se kvalita obrázku postupně zlepšuje až do originální kvality. V některých případech není potřeba načítat ani obrázek celý. Tato technika se používala především v minulosti kvůli pomalejší komunikaci po internetu hlavně u webových prezentacích. V dnešní době již nejsou výhody prokládání tak hojně používány díky velmi rychlému připojení k internetu, které uživateli umožňuje obrázky zobrazovat takřka okamžitě. [26]

### 4.1 Jednorozměrné

Jednorozměrné prokládání se používá u grafického formátu *GIF*. Je to grafický formát určený pro rastrovou grafiku. *GIF* využívá bezztrátové komprese *LZW*. Umožňuje jednoduché animace, kdy se v určitém časovém intervalu zobrazují jednotlivé obrazy za sebou. Maximální počet současně použitých barev je 256 (8 bitů). *GIF* také umožňuje, aby jedna barva z barvové palety byla v každém rámci uvedena jako barva průhledná. [27]

Prokládání je u *GIF* formátu prováděno jednorozměrně (po řádcích). Algoritmus ukládá rastrový obraz po řádcích, které se postupným načítáním dat zjemňují. Nejprve se zobrazuje každý osmý řádek počínaje řádkem nultým. Dále každý osmý, počínaje řádkem čtvrtým. Poté každý čtvrtý řádek, počínaje řádkem druhým a nakonec každý druhý řádek, počínaje řádkem prvním. Tímto způsobem postupně dojde k vykreslení celého obrazu. Pixely na řádku, který nebyl stále vykreslen, přebírají vlastnosti od pixelů nad nimi. Kvůli tomuto postupu vykreslování nám obvykle k rozpoznání obrázku stačí již jedna čtvrtina dat. Postupné vykreslování vzbuzuje dojem zaostřování obrázku při načítání. Přítomnost prokládání navyšuje datovou velikost souboru. [26]

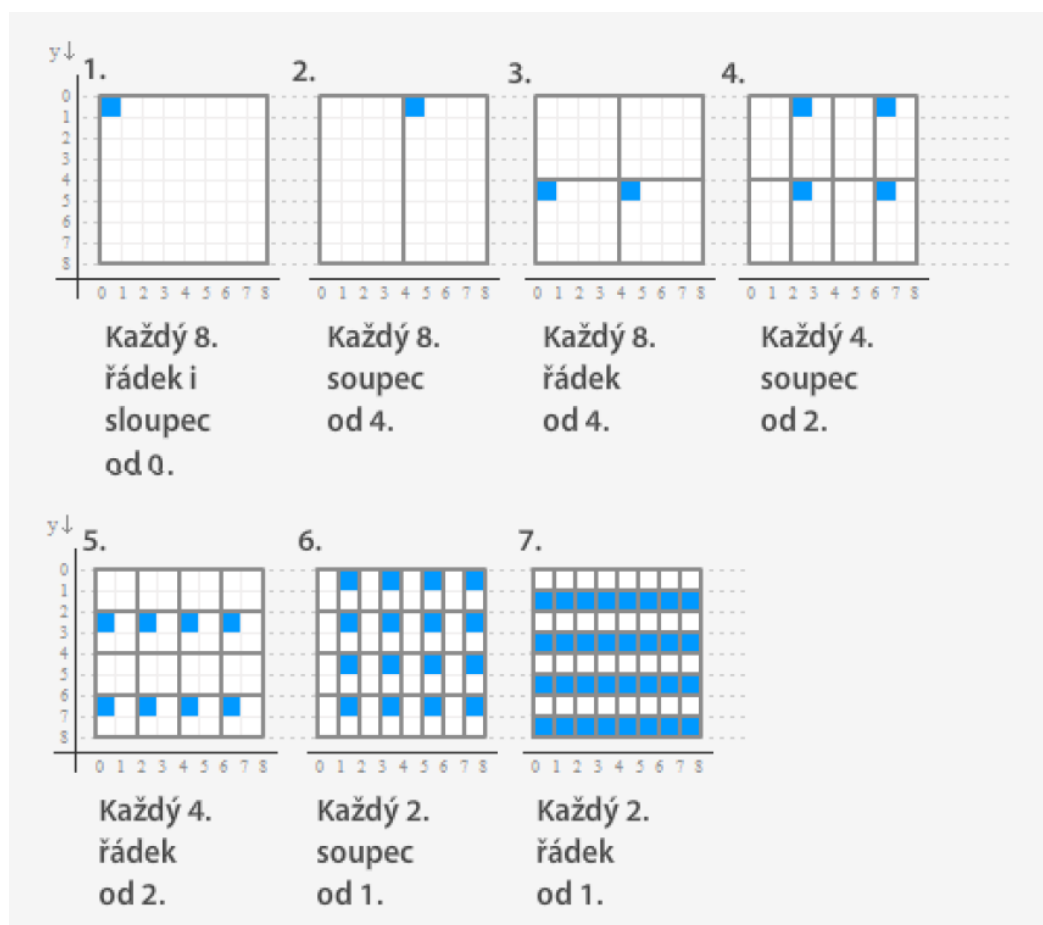


Obr. 23 Jednorozměrné prokládací schéma [26]

## 4.2 Dvourozměrné

Dvourozměrné prokládání se využívá u formátu PNG. Tento formát je často označován jako nástupce formátu GIF. Byl roku 1996 specifikován konsorciem W3C. PNG nabízí podporu 24 bitové barevné hloubky (true color), bezztrátovou kompresi. Navíc nabízí osmibitovou průhlednost (alfa kanál), ta nabízí možnost částečné průhlednosti (RGBA barevný model). PNG ovšem na rozdíl od GIF nepodporuje animace. [28]

U PNG je prokládání prováděno pomocí dvourozměrného schématu. Obraz je vykreslován v blocích, které se postupně zjemňují. Při načítání obrázku je nejprve zobrazován každý osmý pixel na řádku  $i$  ve sloupci, což rozdělí obraz do čtvercových bloků. Každý blok je vždy vyplněn dekódovanou barvou pro daný blok. Při každém kroku jsou oblasti rozděleny, čímž se počet bloků zdvojnásobí. Díky dvourozměrnému prokládání je možné rozpoznat obrázek již po načtení jedné osminy dat. Dvourozměrné prokládání je vhodnější pro zobrazování textu. Dvourozměrné prokládání má také vliv na výslednou velikost souboru. [26]



Obr. 24 Dvourozměrné prokládací schéma [26]

## 5 WEBOVÁ BEZPEČNOST

Bezpečnost webu je potřeba zajišťovat na mnoha úrovních, od operačního systému na serveru až k webovému prohlížeči uživatele. Jelikož serverovou část zajišťuje univerzita sama, bude následující kapitola věnována pouze problémům způsobeným interakcí mezi aplikací a uživatelem.

### 5.1 HTTP/HTTPS

HTTP je internetový protokol určený pro přenos dokumentů po síti internet. Tento protokol je v dnešní době nejrozšířenější. HTTP používá jednotný lokátor prostředků zvaný *URL*, který specifikuje jednoznačné umístění zdroje na internetu. Samotný protokol nijak neza bezpečuje integritu dat a neumožňuje šifrování. Pro zabezpečení se často používá *TLS* spojený nad protokolem *TCP*. Toto použití se označuje jako HTTPS. Protokol funguje způsobem dotaz-odpověď. Uživatel například pomocí internetového prohlížeče pošle serveru dotaz formou textu a server odpovídá.

Protokol HTTPS využívá asymetrické šifrování. Obě strany si před zahájením komunikace vygenerují dva klíče, jeden privátní a druhý veřejný. Při zahájení komunikace si strany vymění veřejné klíče, které je nejprve potřeba ověřit u certifikační autority. Samotné šifrování chrání komunikaci před odposloucháváním, bez ověření autenticity veřejných klíčů vzniká riziko *Man in the middle* (aktivního prostředníka).

Celá webová aplikace komunikuje prostřednictvím protokolu HTTPS. Tudiž je veškerá komunikace se serverem šifrována.

V případě HTTP se jedná o bezstavový protokol, existuje však několik způsobů, jak předávat stav aplikace. Aplikace si může například ukládat údaje do URL a ty poté přenášet ze stránky na stránku (např. *index.php?id=258*). Další možnost spočívá v použití *cookie*. Jde o krátkou textovou informaci přijatou ze serveru v HTTP hlavičce požadavku. Každá cookie je vázaná na doménu, ze které byla přijata. Při komunikaci se serverem je z této domény cookie posílána s každým HTTP požadavkem zpět na server. [29]

#### 5.1.1 Metody HTTPS

V http existuje několik metod komunikace. Těmi nejzákladnějšími jsou *GET* a *POST*. Metoda *GET* se používá pro zjištění informací ze serveru, kde může definovat například číslo

kapitoly dokumentu, na který chceme přejít. Naopak komunikace POST se používá především k modifikaci stavů a také při odesílání většího množství dat na server, například při nahrávání souborů.

## 5.2 Typy útoků

Jelikož je aplikace psána pod redakčním systémem Wordpress, většinu bezpečnostních rizik odstraňuje právě on, nebo dodává potřebné nástroje k jejich předcházení. Některá rizika však závisí pouze na obezřetnosti uživatele.

### 5.2.1 SQL Injection

Jedná se o útoky směřované na databázi webové aplikace. Smyslem útoku je databázi prostřednictvím aplikace podstrčit data, například v URL nebo skrz formulář, které následně databáze vykoná. SQL Injection využívá chyb v aplikaci, nejčastěji nekontrolovaných vstupů od uživatele.

Máme-li například v PHP následující výraz s SQL dotazem

```
$statement = "SELECT * FROM users WHERE name =" . $userName . "';"
```

stačí do proměnné *userName* zadat například

```
' or '1'='1
```

čímž vznikne dotaz, který vybere všechny záznamy z tabulky *users*. Vložit ovšem v tomto případě můžeme jakékoliv jiné SQL, které například edituje záznamy nebo celou tabulku smaže. [30]

Obrana proti těmto útokům je poměrně jednoduchá Existují různé funkce, které ošetřují vstupy například tím, že před speciální znaky jako je například středník, který v SQL ukončuje dotaz, vloží zpětné lomítko. Tím se při překladu středník nepřekládá jako řídicí znak, ale pouze jako znak řetězce. V jazyce PHP je to například funkce *mysqli\_real\_escape\_string*. Daleko elegantnější způsob obrany je *prepared statement*. Jedná se o způsob, kdy si SQL dotazy nejprve připravíme a místo hodnot vložíme zástupné znaky. Databázi dotaz a data předáváme odděleně a ona si je sama do dotazu vloží takovým způsobem, aby to bylo bezpečné. Redakční systém Wordpress využívá právě tohoto způsobu komunikace s databází, slouží k tomu knihovna *wpdb*.

## 5.2.2 Únos spojení

Protokol HTTP bezstavový, i když v některých případech je potřeba nějakým způsobem stav přenášet. Pro tyto případy se používá mechanismus sezení (session). Uživateli je při prvním přístupu vytvořeno sezení a je přidělena jeho identifikace pomocí cookie nebo pomocí předávání v URL.

Únos spojení nebo také *session hijacking* je typ útoku, který zneužívá HTTP cookie odcizené oběti útoku pro získání neoprávněného přístupu k informacím nebo službám poskytovaným webovým serverem.

Existují čtyři základní metody, kterými je možné se zmocnit sezení jiného uživatele. Všechny jsou založeny na získání identifikátoru sezení, který je uložen v cookie.

### 5.2.2.1 *Session fixation*

Metoda je založena na případě, kdy webová stránka akceptuje identifikátor sezení předávaný v URL. Útočníkovi stačí jen podstrčit oběti upravený odkaz, například ve tvaru `http://web.cz/?SID=0D6441FEA4496C2`. Potom, co oběť otevře danou stránku a přihlásí se, stačí útočníkovi zobrazit stejnou stránku, čímž získá přístup k účtu oběti.

### 5.2.2.2 *Session sidejacking*

U *session sidejacking* útočník odposlouchává nešifrovanou síťovou komunikaci mezi prohlížečem a webovým serverem pomocí analyzátoru síťového provozu. U některých stránek probíhá přenos pomocí HTTPS pouze při přihlášení, ale další komunikace již šifrována není, proto je snadné přidělené SID jednoduše získat. Útok je poměrně snadné provést například při použití veřejné wifi.

### 5.2.2.3 *Cross-site scripting*

*Cross-site scripting* je založeno na snaze útočníka podvrhnout do uživatelského počítače kód (např. JavaScript), který je považován za důvěryhodný, protože vypadá, jako by byl získán z navštíveného webového serveru. Podvržený kód může následně získat SID nebo provádět jiné nežádoucí operace. Tomuto typu útoku se bude věnovat podrobněji následující kapitola.

#### 5.2.2.4 Fyzické zkopírování SID

Útočník může získat SID prostým zkopírováním souboru s cookie z uživatelského počítače (cookie jsou typicky uložena jako běžné soubory), kopírováním části operační paměti počítače nějakým nežádoucím programem nebo zkopírováním SID přímo z napadeného webového serveru. [31]

#### 5.2.3 Cross Site Scripting

Jedná se o případ, kdy se útočník snaží do stránky podstrčit svůj vlastní JavaScriptový kód. Pomocí tohoto kódu může například poškodit vzhled stránky, znefunkčnit ji nebo získat citlivé údaje od návštěvníka. Tato bezpečnostní mezera může vzniknout například při neošetřených vstupech. Při zobrazování vstupů od uživatele je nutné, aby výstupy nebyly do stránky vkládány přímo, ale pouze jako text. Musí dojít k nahrazení klíčových znaků HTML jejich entitním vyjádřením, tj. < za &lt;, > za &gt; a & za &amp;. Wordpress v tomto případě nabízí funkci `esc_html`.

Pokud se na v souboru PHP nachází například tento kus kódu

```
echo $_GET['nadpis'];
```

stačí uživateli podstrčit adresu upravenou například takto:

```
http://URL/stranka.php?nadpis=cokoliv<script>alert("Toto je úspěšný XSS útok.");</script>
```

čímž se mu spolu se stránkou zobrazí vyskakovací okno s textem „Toto je úspěšný XSS útok.“.

Situace se komplikuje v případě použití WYSIWYG editorů. Tyto editory totiž vytváří HTML kód, který se následně nahrává na server. V případě Wordpressu je možno při vytváření stránky vkládat do editoru jakýkoliv JavaScriptový kód, není třeba vstupy nijak filtrovat, přístup do editace mají pouze uživatelé s přiřazeným oprávněním. [32]

Stojí tedy spíše za zvážení, komu danou uživatelskou roli pro editaci udělíme. V naší aplikaci nejsou implementovány žádné možnosti, kterými by běžný uživatel měl možnost cokoliv do aplikace přidávat.

#### 5.2.4 Cross Site Request Forgery

Tato metoda je založena na odeslání nezamýšleného požadavku pro vykonání určité akce ve webové aplikaci, která ovšem pochází z nelegitimního zdroje. Předpokladem pro útok je podrobná znalost aplikace, na kterou útočíme.

Průběh útoku může být následující, napadnutá stránka obsahuje formulář, který přijímá data pomocí metody *GET*. Oběti, jež je na této stránce přihlášená, podstrčíme odkaz na naši stránku, kde je umístěn například obrázek, který má v atributu *src* následující adresu *http://webmail/poslat?komu=nejakyEmail@server&predmet=test&obsah=textemailu*. Po jeho načtení se bez vědomí uživatele odešlou data do formuláře na dané stránce. Ani data přijímaná metodou *POST* nejsou bezpečná, jelikož stejným způsobem se může místo obrázku na stránce nacházet formulář, který se automaticky po načtení stránky odešle metodou *POST* na napadenou stránku.

Dobrou obranou proti tomuto druhu útoku je používání autorizačních tokenů, nutnost při vykonávání akce dodávat tajnou hodnotu, kterou generuje server, vkládání SMS kódu nebo použití CAPTCHA. Předpokladem je také opatrnost uživatel, například při používání internetového bankovníctví, nepoužívat zároveň jiné stránky. [33]

## 6 POUŽITÉ TECHNOLOGIE PRO IMPLEMENTACI STRÁNEK

Pro tvorbu internetových stránek se používají technologie jako HTML, PHP, CSS a JavaScript. Každá z technologií na webu slouží k něčemu jinému. Základem každého webu je HTML, jedná se o značkovací jazyk, který utváří strukturu celého webu pomocí párových a nepárových značek. Pro popis způsobu zobrazení elementů v HTML slouží jazyk CSS (kaskádové styly), který byl vytvořen, aby se popis vzhledu oddělil od samotné struktury dokumentu (webu). Kaskádové styly mohou ovlivnit například velikost textu, barvu, pozici zarovnání a mnoho dalšího.

Postupem času již statické stránky psané pouze pomocí HTML nebyly dostačující a bylo za potřebí dynamicky generovaného obsahu stránek. Pro tyto účely byl vyvinut skriptovací jazyk PHP, který tuto potřebu řeší. Umožňuje totiž pracovat s konstrukcemi, jako jsou podmínky, cykly, příkazy, výrazy a nabízí přístup k databázím a mnoho dalšího. Jedná se o jazyk působící na straně serveru. Nakonec je zde jazyk JavaScript, který slouží k ovládání různých interaktivních prvků, tvorbě animací a manipulací s obrázky bez nutnosti stránku znovu obnovit. Celý kód JavaScriptu je prováděn na klientské straně.

### 6.1 Wordpress

Wordpress je volně šířený open source<sup>1</sup> a redakční systém pro publikování článků napsaný v PHP a MySQL. Wordpress je šířen pod licencí GPLv2<sup>2</sup> a je vyvíjen lidmi z celého světa. Patří mezi nejrozšířenější redakční systémy na světě, je používán jako základna pro milióny webových stránek. To, že Wordpress má otevřený zdrojový kód, dává neomezené možnosti jeho modifikace. Dá se použít prakticky na cokoliv a ve většině případů úplně zdarma. [34]

Wordpress byl pro tuto aplikaci zvolen především z toho důvodu, že patří mezi nejrozšířenější a má velmi dobře zpracovanou dokumentaci plnou praktických ukázek kódu, tutoriálů a věnuje se mu široké množství jiných webů, kde může programátor čerpat inspiraci. Jelikož je tento web použit jako závěrečná práce a po ukončení studia ho dále bude zpravovat někdo ze zaměstnanců univerzity, nebylo by moudré volit nějakou nestandardní technologii, kterou zná jen původní autor.

---

<sup>1</sup> Otevřený zdrojový kód

<sup>2</sup> <http://www.gnu.org/copyleft/gpl.html>

### 6.1.1 Pluginy

Wordpress je možno jednoduše rozšiřovat o další funkcionalitu pomocí pluginů a témat vzhledu. Pluginy umožňují snadnou úpravu, přizpůsobení a zlepšování webů ve WordPressu. Namísto programování změn jádra WordPressu, můžete s jejich pomocí jednoduše přidat potřebnou funkcionalitu. Pluginy jsou psány v PHP a obsahují sadu scriptů. Na oficiálních stránkách projektu jsou zdarma ke stažení tisíce pluginů. Před vlastním programováním je proto dobré zjistit, zda stejnou problematiku nevyřešil již někdo před námi.

### 6.1.2 Témata vzhledu

Témata vzhledu jsou soubory, které společně utváří výsledný vzhled stránky a stejně jako pluginy přidávají do Wordpressu další funkcionalitu. Pro tvorbu témat se používá PHP, HTML, CSS a JavaScript. Stejně jako u pluginů existuje na stránkách projektu databáze tisíců různých témat vzhledu, kde si každý může zdarma stáhnout vhodný typ.

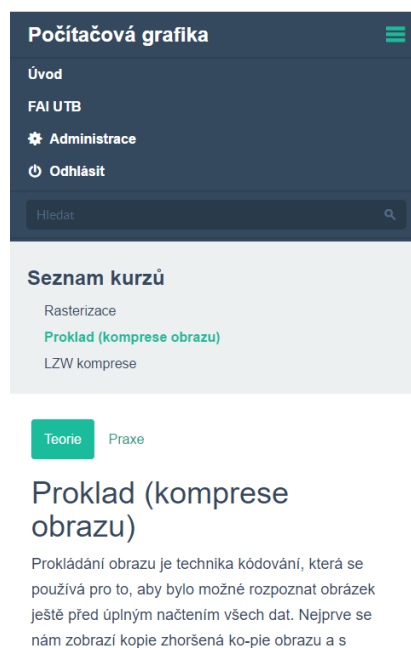
## **II. PRAKTICKÁ ČÁST**

## 7 METODIKA

Cílem této práce je vytvoření webové aplikace, která bude dostupná studentům Univerzity Tomáše Bati ve Zlíně, aby sloužila především jako podpora předmětu Počítačová grafika. Může také sloužit jako podpora u jiných předmětů, které jsou úzce na problematiku počítačové grafiky navázány. Stránky mají sloužit jako portál umožňující uchování a prezentaci různých modulů implementovaných pomocí webových technologií pro co nejsnadnější spuštění. Modul bude sloužit jako praktická ukázka teoretického základu pro lepší pochopení studované problematiky.

Před samotnou implementací bylo potřeba zmapovat a zhodnotit dostupné technologie pro implementaci s důrazem na rozšiřitelnost a jednoduchost. Tomuto problému se stručně věnuje teoretická část této práce.

Po implementaci samotné aplikace byly vytvořeny tři moduly, kde byl kladen důraz především na oddělenost od samotné aplikace, aby modul byl schopen samostatně fungovat a neovlivňoval negativně chod celého webu a obráceně. V zadání práce jsou uvedeny pouze kurzy algoritmů pro rasterizaci a kompresní algoritmy. S vedoucím práce bylo domluveno, že při dostatku času při realizaci bude vytvořen prostor pro tvorbu modulů dalších. Pro tento případ byl vytvořen modul prokládání obrázků. Tato problematika je též popsána v teoretické části práce. Kurzu pro kompresní algoritmy se z důvodu rozsáhlosti daného tématu věnuje podrobně pouze algoritmus LZW.



Obr. 25 Náhled webu na mobilním telefonu

## 8 WEBOVÁ STRÁNKA

V minulosti již pro podporu předmětu Počítačová grafika bylo v rámci závěrečných prací studentu UTB vytvořeno několik aplikací, ovšem pro jejich tvorbu nebyly použity webové technologie nebo byly použity technologie zastaralé. Neexistoval také žádný systém, který by tyto moduly udržoval na jednom místě pro lepší přehlednost a dostupnost.

V dnešní době je kladen velký důraz na to, aby byly informace přístupné z jakéhokoliv místa a v jakémkoliv čase. K tomuto účelu je internet skvělý nástroj. Většina lidí již nyní u sebe stále nosí chytrý telefon s přístupem na internet. Tyto fakta vedou ke stále vzrůstajícímu zájmu o tvorbu webových aplikací. A není se také čemu divit, výhodou webové aplikace totiž je, že uživateli k jejímu zobrazení stačí pouze webový prohlížeč, jako je například Google Chrome, Firefox nebo Safari. Není tedy potřeba nic dodatečně instalovat. I z pohledu vývoje zde máme mnoho výhod. Při implementaci stačí vytvořit pouze jednu verzi webu, není zde potřeba pro každou platformu vyvíjet speciální verzi aplikace, což vede k snížení nákladů a jednoduššímu udržování. Je zde pouze potřeba dávat pozor na podporu použitých technologií v závislosti na aktuálních verzích a na podporu funkcionality v prohlížečích, jelikož každý prohlížeč může mít například jinou verzi JavaScriptového jádra a stejná funkce se může v každém prohlížeči chovat trochu jinak, nebo nemusí být vůbec implementována.

Dále je také u vývoje webových aplikací potřeba klást důraz na responzivní design celého webu, jelikož různá zařízení mají různá rozlišení svých displejů a stránka by měla být implementována takovým způsobem, aby se sama displeji přizpůsobila a nebránila uživateli v bezproblémovém použití.

Celý web na podporu výuky počítačové grafiky je dostupný na adrese <https://pgr.fai.utb.cz>. Je implementován pomocí redakčního systému Wordpress.

### 8.1.1 Implementace stránek

Pro implementaci stránek bylo pro Wordpress vytvořeno téma vzhledu. Podrobný popis API rozhraní včetně tutoriálu jak vytvářet témata vzhledu je dostupný na stránkách projektu. Předmětem této práce není popisovat tvorbu témat vzhledu pro Wordpress, proto se mu budeme v následujícím textu věnovat pouze okrajově.

Základem každého Wordpress tématu je adresář ve složce *wp-content/themes*, která kromě obrázků a souborů JavaScript, obsahuje tři základní typy souborů.

1. Šablona stylů s názvem `style.css`, která řídí prezentaci (vizuální design a rozvržení) webové stránky.
2. Wordpress souborové šablony v PHP, které určují, jakým způsobem se budou vypisovat informace získané z databáze.
3. Volitelné funkční soubory (*function.php*) jako součásti šablony.

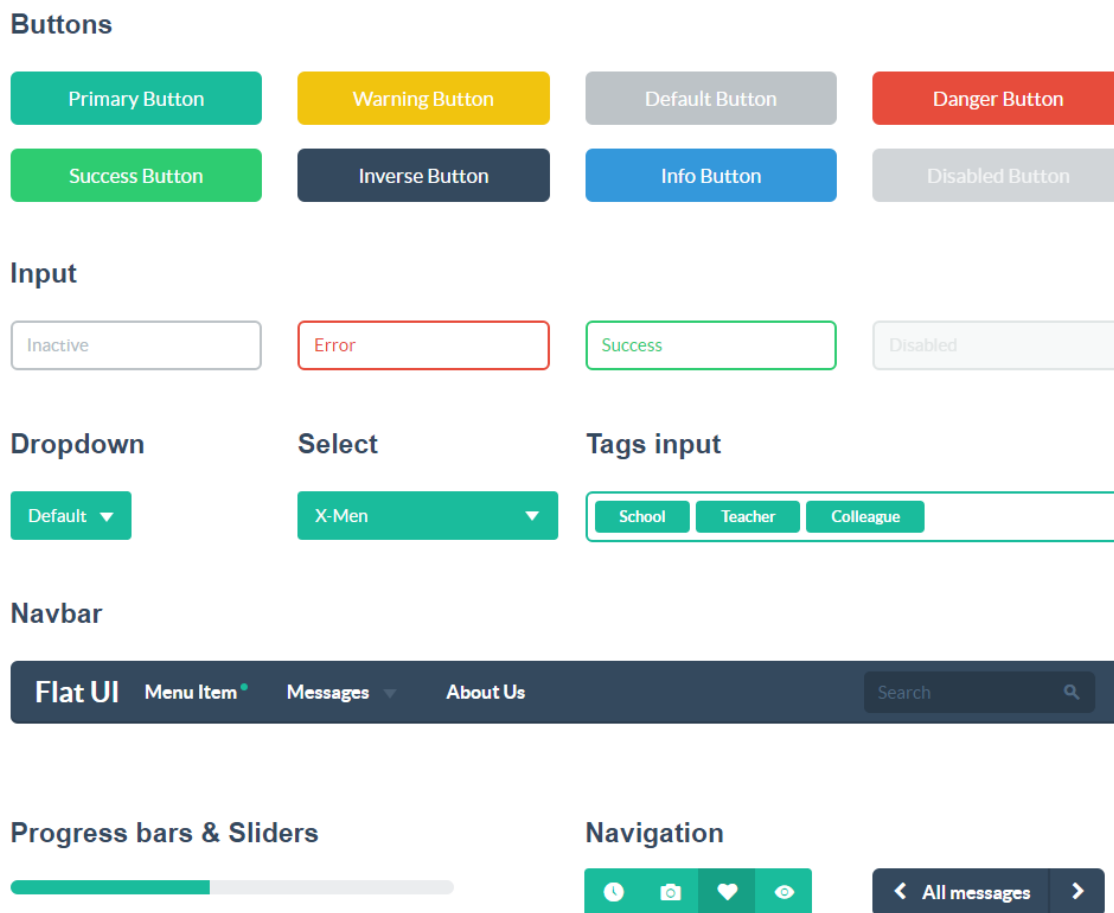
Kromě informací o CSS stylech pro zadané téma, soubor *style.css* poskytuje také podrobné informace o tématu v podobě komentářů. Šablona stylů musí tyto informace o tématu obsahovat. Žádné dva motivy nemohou mít stejné údaje uvedené ve svých komentářích.

## 8.2 Design webu

Pro designovou stránku webu byl použit framework<sup>3</sup> Bootstrap s tématem Flat UI. Bootstrap patří mezi nejpoužívanější HTML, CSS a JavaScript frameworky na světě. Jedná se o framework dodávající sadu ovládacích prvků webu. Flat UI upravuje pouze vizuální stránku prvků Bootstrapu, který je dostupný zdarma, a obsahuje sadu nástrojů pro tvorbu internetových stránek. Definiuje šablony typografie, formulářů, tlačítek, navigace a jiných komponent, včetně těch složitějších ovládaných pomocí JavaScriptu. Dále podporuje nejnovější verze prohlížečů a zároveň se dokáže elegantně přizpůsobit zobrazení na starších prohlížečích. Navíc do verze 2.0 podporuje responzivní design, což umožňuje, aby se web dokázal dynamicky přizpůsobit rozlišení displeje každého zařízení (mobilní telefon, tablet, PC) a zůstal uživatelsky příjemný. Dokumentace Bootstrapu je velmi uživatelsky přívětivá a dostupná v několika jazycích. [35] [36]

---

<sup>3</sup> Software sloužící jako podpora k programování.



Obr. 26 Vzhled základních elementů Flat UI

### 8.3 Server site

Jak už bylo zmíněno, celý web je dostupný pod doménou třetího řádu na adrese <https://pgr.fai.utb.cz>. Je umístěn na univerzitním serveru <http://mond.utb.cz/>, který běží na operačním systému Debian<sup>4</sup>. Jako webový server je zde použit Nginx, což je druhý nejpoužívanější linuxový webový server vůbec. Jedná se o webový server/reverzní proxy vydaný pod licenci BSD. Mimo protokolů http/http(s) zvládá i POP3 a IMAP poštovní proxy s podporou SSL a TLS, takže se ve skutečnosti nejedná jen o webový server. Výborně se hodí nejenom na provoz náročnějších aplikací, ale i pro provoz na menších serverech a osobních VPS<sup>5</sup>. [37]

<sup>4</sup> Distribuce GNU/Linux

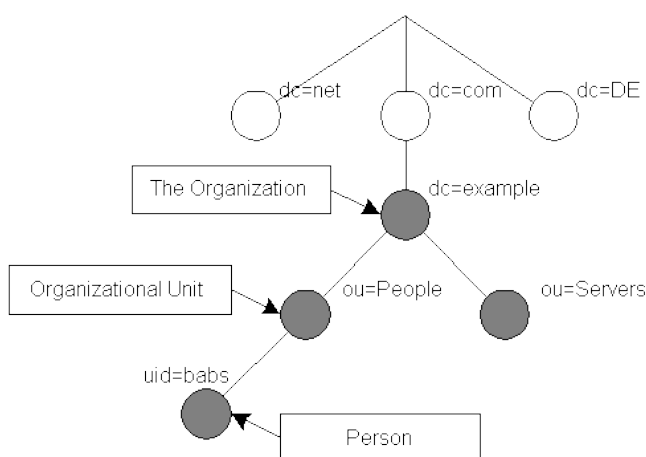
<sup>5</sup> Virtuální server

### 8.3.1 Databáze

Jako databázový systém je na serveru použit MySQL. Jedná se o relační databázi, která je poskytována pod dvojí licenci, bezplatnou GPL nebo pod komerční placenou licenci. Jedná se o multiplatformní databázový systém, komunikace s ním probíhá prostřednictvím dotazovacího jazyka SQL. Pro práci s databází je na adrese <https://db.mond.utb.cz> dostupný Adminer. Jedná se o aplikaci napsanou v PHP, jejímž prostřednictvím lze jednoduše pomocí webového rozhraní s databází pracovat. Jedná se o jednodušší alternativu proti známějšímu PhpMyAdmin.

### 8.3.2 Autentizace a autorizace

Přístup k obsahu webu je omezen přihlášením. Web je přístupný pouze studentům FAI UTB pod stejnými přihlašovacími údaji jako na adrese <https://stag.utb.cz>. Přihlašovací údaje na web se ověřují prostřednictvím protokolu LDAP vůči univerzitnímu serveru. Jedná se o standardizovaný komunikační protokol pro adresářové služby TCP/IP. Je to jednoduchý a dobře navržený protokol umožňující nejen klást poměrně složité dotazy, ale také modifikovat a mazat záznamy. Celou službu je možno si představit jako veliký strom či adresář na souborovém systému, který obsahuje celý svět. Tento strom obsahuje záznamy (entries). Každý záznam musí mít definovány atributy (attributes) – povinné a volitelné. Ty jsou definovány pomocí objektů (object class), které jsou nedefinovány na serveru. Standardně je nedefinováno jen několik základních objektů typu person nebo organization. Další objekty je možné na serveru definovat ručně. [38]



Obr. 27 Příklad LDAP adresářového stromu [39]

Pro ověření připojení na LDAP se v PHP používá funkce `ldap_connect`, která vytváří identifikátor spojení a ověřuje, zda daný hostitel a port jsou věrohodní. Tato funkce samotné


připojení k LDAP neotevívá, pouze ověřuje věrohodnost parametrů. Na dvou vstupech funkce je *hostname* a *port*. Pro samotnou autentizaci se používá funkce *ldap\_bind*, funkce má na vstupu identifikátor spojení, rozlišovací jméno záznamu (DN) a heslo. Návrátové hodnoty jsou TRUE při úspěšné autentizaci a FALSE při neúspěšné.

Po úspěšné autentizaci uživatele se při jeho prvním přihlášení vytvoří v databázi Wordpressu uživatelský účet s nastavenými právy pro prohlížení obsahu webu. Při každém dalším přihlášení se opět nejprve dotazuje na LDAP, účet v systému již existuje, proto se znovu nevytváří.

Ověření oprávnění pro přístup k obsahu webu probíhá prostřednictvím Wordpressu. Jsou zde použity výchozí uživatelské role Wordpressu<sup>6</sup>. Každý nový uživatel automaticky dostává uživatelskou roli *Návštěvník*. Tato uživatelská role má přístup pouze k obsahu webu. Vstup do administrace není umožněn. Pro získání vyššího oprávnění je potřeba, aby někdo s rolí Administrátor uživateli tuto roli nastavil, buď prostřednictvím administrace nebo úpravou v databázi.

## 8.4 Popis uživatelského rozhraní

Při prvním zadání adresy <https://pgr.fai.utb.cz/> se v prohlížeči zobrazí přihlašovací formulář pro přístup k webu. Veškerý obsah webu je totiž dostupný pouze přihlášeným uživatelům.



Obr. 28 Přihlašovací formulář

---

<sup>6</sup> [https://codex.wordpress.org/Roles\\_and\\_Capabilities](https://codex.wordpress.org/Roles_and_Capabilities)

Po zadání přihlašovacích údajů do STAGu a úspěšném přihlášení se uživateli zobrazí úvodní stránka webu. Web je rozdělen do tří částí, horní menu, seznam kurzů a obsahový rámeček.

Obr. 29 Rozložení webu

V pravém horním rohu obsahové části stránky se nachází prepínací tlačítka mezi teoretickou a praktickou částí zvoleného kurzu. Horní menu slouží jako prostor pro umístění odkazů na obsah, dále je zde vyhledávání a ovládací prvky aplikace, jako je odhlášení nebo vstup do administrace. Levý panel označený v nadpise seznam kurzu, slouží jako přechod mezi jednotlivými kurzy. Jedná se o stromovou strukturu, kterou je možno editovat prostřednictvím administrace.

Obr. 30 Administrace webu a nahrávání kurzu

Založení kurzu na webu probíhá prostřednictvím výchozího rozhraní Wordpressu pro vytváření stránek. Uživatel do pole s popisem „Zadejte název stránky“ zadá název, dále může pomocí jednoduchého WYSIWYG<sup>7</sup> editoru vytvořit teoretickou stránku kurzu. Hned pod editorem se nachází panel pro nahrání samotného kurzu. Pomocí tlačítka „Vybrat soubor“ uživatel vyvolá dialogové okno pro výběr souboru. Při nahrávání modulu jsou akceptovány pouze souboru typu ZIP, při nahrávání kurzu je třeba, aby daný archiv neobsahoval složku se zdrojovými soubory, ale přímo zdrojové soubory kurzu. Každý kurz musí obsahovat soubor *index.html*, který se volá při spouštění modulu. Včetně souboru *index.html* je potřeba, aby spolu s ním byly v archivu dostupné všechny ostatní zdroje, se kterými modul pracuje. Při komprimaci je tedy potřeba nejdříve označit všechny potřebné zdroje a ty poté pomocí dialogového okna Windows komprimovat.

Kurzy jsou do stránky vkládány jako rámce (*iframe*). Jedná se o párový tag v HTML, s jehož pomocí je možné do stránky vložit libovolný obsah stránky jiné (vnořené stránky). Výhodou je, že obsah rámce nijak svými kaskádovými styly ani JavaScriptem neovlivňuje vnější web a obráceně.

## 8.5 Bezpečnost webu

S ohledem na skutečnosti popsané v teoretické části, můžeme tedy říci, že web po stránce bezpečnosti je velmi v dobrém stavu. Veškerá komunikace s doménou, na které je stránka umístěna, probíhá šifrovaně. Přístup do aplikace je logován a přístupný pouze pod přihlášením, kde samotné ověření identity probíhá prostřednictvím univerzitního LDAP serveru. Web navíc neobsahuje žádné uživatelské formuláře a vstup do administrace je povolen pouze uživatelům pod danou uživatelskou rolí.

Jediné slabší místo může být právě v použitém redakčním systému Wordpress, který má otevřený zdrojový kód do kterého může kdokoliv nahlížet, a na rozdíl od vlastní aplikace zná dokonale strukturu celého webu. Na druhou stranu je výhodou, že do zdrojového kódu má přístup velké množství odborníků z celého světa, kteří mohou bezpečnostní problémy sami opravit nebo na ně upozornit.

---

<sup>7</sup> Česky „co vidíš, to dostaneš“

## 9 KURZY

Náplní této práce bylo také vytvoření ukázkových kurzů popisujících problematiku rasterizace a komprese obrazových dat, kterým jsou věnovány dvě kapitoly v teoretické části. Dále byl po konzultaci s vedoucím vyhrazen prostor pro vytvoření dalšího modulu, který se zabývá prokládáním, jež byl také popsán v teoretické části.

### 9.1 Použité technologie

Pro tvorbu kurzu je potřeba využívat pouze technologii, které ke svému chodu nepotřebují využívat serverovou stranu. Jak už bylo zmíněno v předchozí kapitole, kurzy jsou vkládány do stránky pomocí *iframe*, musí tedy fungovat samy o sobě. K tvorbě kurzu byly využívány pouze HTML, JavaScript a CSS, případně některé frameworky na těchto technologiích založené, kterým se budou věnovat následující kapitoly.

#### 9.1.1 Dojo Toolkit

Jedná se o open-source modulární JavaScriptovou knihovnu, která usnadňuje a zrychluje vývoj JavaScriptových aplikací. Dokáže například rozeznat rozdíly mezi jednotlivými prohlížeči a nabízí jednotné API, které bude fungovat na každém z nich. Další silnou stránkou Dojo Toolkitu je možnost definování modulů, které fungují jako třídy, umožňují dokonce mnohonásobnou dědičnost. Dále knihovna poskytuje různé nástroje na optimalizaci JavaScriptu, generování dokumentace, testování, lokalizaci textových řetězců, nabízí širokou škálu pomocných tříd a ovládacích prvků. [40]

Dojo Toolkit je rozdělen do několika částí:

- *Dojo* – Také označován jako jádro, jedná se o hlavní část Doja, ve kterém jsou obsaženy nejpoužívanější balíčky a moduly. Jádro zahrnuje širokou škálu funkcí jako AJAX, manipulaci s DOM, události, přísliby, datová úložiště, drag-and-drop a mnoho dalších.
- *Dijit* – Dijit zahrnuje rozsáhlou řadu widgetů (komponent uživatelského rozhraní) a základní systém jejich podpory. Je celý postaven nad jádrem Dojo.
- *DojoX*- Jedná se o sbírku balíčků a modulů, které poskytují velké množství funkcionality. DojoX je postaven jak na Doju, tak na Dijitu. Balíčky a moduly této části se nacházejí v různých stádiích vývoje, můžete zde nalézt jak moduly velmi vyspělé, tak i velmi experimentální.

- *Util* - Různé nástroje, které podporují zbytek sady toolkitu, jako je například tvorba dokumentace a testů. [41]

Dojo Toolkit byl nasazen především kvůli možnosti rozdělit aplikaci do jednotlivých modulů, což značně usnadňuje přehlednost, udržitelnost a rozšiřitelnost kódu. Dále sebou přinesl spoustu užitečných funkcí jako například query selektory.

```
<script>
  define([
    "dojo/_base/declare",
    "test/Predek"
  ], function(declare, Predek){
    Potomek = declare(Predek, {
      name: "Jméno potomka"
    });
    return Potomek;
  });
</script>
```

Obr. 31 Ukázka definice modulu a dědičnosti v Dojo

### 9.1.2 Canvas

Canvas je HTML prvek zahrnutý ve specifikaci HTML5. Jedná se o jakési plátno, na nějž lze dynamicky pomocí JavaScriptu vykreslovat bitmapy a grafická primitiva. Je definován párovým tagem *canvas* a zvolenou šířkou a výškou. Canvas je v dnešní době podporován všemi moderními prohlížeči. Pomocí canvas lze vykreslovat různé grafy, animace, hry a upravovat obrázky.

```
<!DOCTYPE html>
<html>
<body>

<canvas id="myCanvas" width="200" height="100" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

<script>
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.beginPath();
ctx.arc(95,50,40,0,2*Math.PI);
ctx.stroke();
</script>

</body>
</html>
```

Obr. 32 Ukázka definice černé kružnice pomocí canvas

### 9.1.2.1 EaselJS

V modulech byla dále použita JavaScriptová knihovna EaselJS, která usnadňuje a zefektivňuje práci s canvas. Nabízí možnost manipulovat s vykreslenými prvky objektově, zjednodušuje syntaxi pro práci s canvas, umožňuje navěšovat na vykreslené obrazce události, jako jsou kliknutí, posouvání a nabízí efektivní způsob, jak grafiku animovat. Je vhodný pro vytváření her, generování obrazců, reklam a jiných grafických úkonů. [42]

```
<script>
  var stage = new createjs.Stage('myCanvas');
  var shape = new createjs.Shape();
  shape.graphics.beginFill('black').drawRect(0, 0, 120, 120);
  stage.addChild(shape);
  stage.update();
</script>
```

Obr. 33 Kreslení černé kružnice pomocí EaselJS

## 9.2 Kurz rasterizace

Jako první kurz pro podporu výuky byl vytvořen kurz pro rasterizaci. Předlohou tomuto kurzu sloužila aplikace Jana Sečkaře zpracovaná v rámci bakalářské práce Algoritmy a rasterizace 2D grafických objektů (2007). Aplikace byla napsána v jazyku Java, tudíž její spouštění přes webový prohlížeč formou Appletu není v dnešní době JavaScriptu úplně optimální. Bylo tedy nutno celou aplikaci přepsat. Nová aplikace od té předchozí přebírá pouze grafický vzhled.

### 9.2.1 Uživatelské rozhraní

V horní části modulu se nachází ovládací prvky, kde si uživatel zvolí požadovaný typ objektu a k němu příslušnou rasterizační funkci. Na výběr jsou DDA Line, Bresenham Line, Circle, Bresenham Circle, Bresenham Ellipse, Ferguson Cubic a Bezier Cubic. Dále tlačítka na ovládání scény, *play*, *step*, *pause* a *reset*. Posledním ovládacím prvkem je posuvník na úpravu velikosti mřížky scény.

Největší část aplikace zabírá scéna, jež slouží jako prostor pro vykreslování zvoleného tvaru a také pro následné vykreslování po rasterizaci.

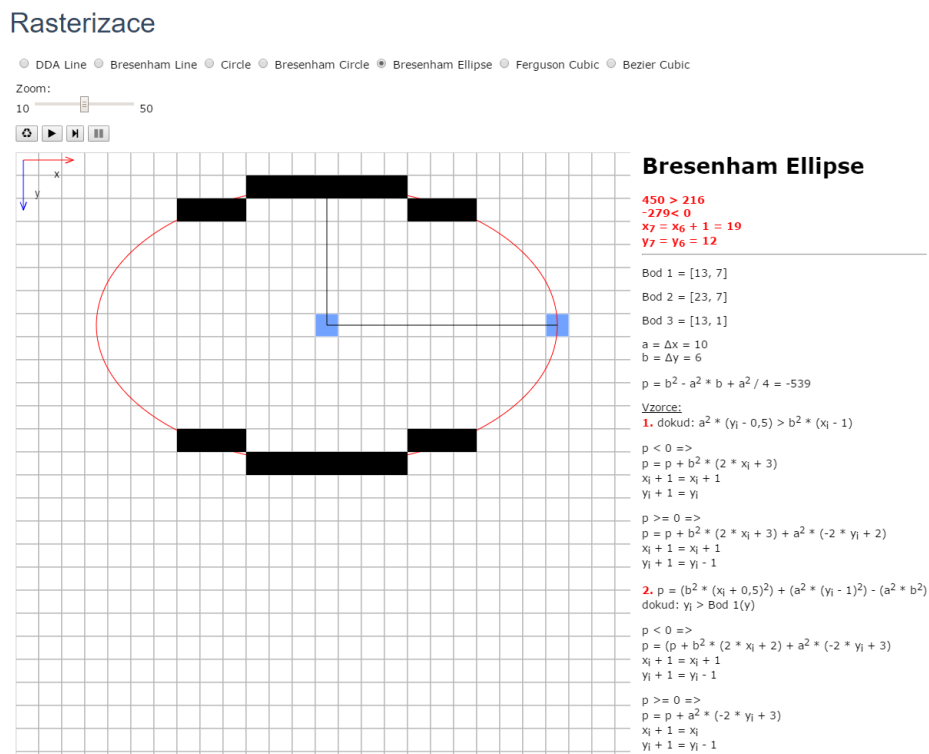
Poslední částí nacházející se na pravé straně modulu je panel pro výpis dat a rovnic rasterizačního algoritmu.

## 9.2.2 Ovládání

Celý kurz se ovládá prostřednictvím myši. Po zvolení příslušného objektu pro rasterizaci musí uživatel kliknout do mřížky scény, čímž zvolí body potřebné pro vykreslení tvaru. Při volbě bodů se v pravé části vypisují souřadnice těchto bodů. Po zvolení posledního bodu se vypíše rovnice rasterizačního algoritmu. Při spuštění rasterizace se postupně vypisují aktuální rovnice a hodnoty vypočítaných bodů.

Vlastnosti rasterizovaných objektů, jako je velikost nebo poloha, lze měnit pomocí přetažením zvolených bodů.

Spuštění rasterizace je možné provádět dvěma způsoby. Prvním z nich je použití tlačítka *play* kdy se spustí automatická animace a uživatel už nemusí do rasterizace nijak zasahovat. V případě, že by chtěl animaci pozastavit, může použít tlačítko *pause*. Další možností je postupné vykreslování pomocí tlačítka *step*, kdy se v každém kroku vypočítá a vykreslí jeden pixel.



Obr. 34 Ukázka kurzu rasterizace

## 9.3 Kurz komprese obrazových dat

Pro realizaci tohoto kurzu byla zvolena kompresní metoda LZW a její uplatnění při kompresi obrázkových dat formátu GIF. Kurz napomáhá pochopení problematiky, jakým způsobem

jsou data z obrázku popsána a jak vzniká samotný vstup pro onu kompresi. Dále vysvětluje převod komprimovaných dat do binárního kódu.

### 9.3.1 Uživatelské rozhraní

Uživatelské rozhraní tohoto kurzu se skládá ze tří stránek, mezi kterými lze přepínat pomocí tlačítek v horní části kurzu. Každá ze stránek obsahuje jednu z částí potřebných pro pochopení dané problematiky. První část nazvaná „Obrázková data“ zobrazuje, jakým způsobem se popisují pixely v obrázku a vzniká datový stream pro kompresi. V další části s názvem *Komprese LZW* se nachází tabulka, která zobrazuje průběh kompresního algoritmu, pomocí ovládacích tlačítek nad tabulkou lze postupně do tabulky přidávat řádky zobrazující jednotlivé kroky při kompresi. Na poslední stránce *Reprezentace dat v GIF* jsou zobrazeny výstupy po LZW kompresi v desítkové soustavě, následně binární stream kde se uplatňuje proměnlivá délka kódu a poté zápis v šestnáctkové soustavě s ukázkou reprezentace komprimovaných dat ve strukture obrázku GIF.

#### LZW - Komprese

Krok	Akce	Vstup	Nový záznam do slovníku	Výstup	Binární výstup	Délka kódu
0	Inicializace	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...		#4	100	3
1	Čtení	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...		#4	100	3
2	Nenalezeno	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...	#6 - 11	#4, #1	001100	3
3	Čtení	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...		#4, #1	001100	3
4	Nalezeno	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...		#4, #1	001100	3
5	Čtení	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...		#4, #1	001100	3
6	Nenalezeno	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...	#7 - 111	#4, #1, #6	1 10001100	3
7	Čtení	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...		#4, #1, #6	1 10001100	3
8	Nalezeno	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...		#4, #1, #6	1 10001100	3
9	Čtení	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...		#4, #1, #6	1 10001100	3
10	Nenalezeno	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...	#8 - 112	#4, #1, #6, #6	1101 10001100	3
11	Čtení	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...		#4, #1, #6, #6	1101 10001100	4
12	Nenalezeno	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...	#9 - 22	#4, #1, #6, #6, #2	00101101 10001100	4
13	Čtení	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...		#4, #1, #6, #6, #2	00101101 10001100	4
14	Nalezeno	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...		#4, #1, #6, #6, #2	00101101 10001100	4
15	Čtení	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...		#4, #1, #6, #6, #2	00101101 10001100	4
16	Nenalezeno	1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, ...	#10 - 222	#4, #1, #6, #6, #2, #9	1001 00101101 10001100	4

Výpis je kvůli množství záznamů omezen na zpracování čtrnácti pixelů.

Obr. 35 Ukázka kurzu komprese LZW

## 9.4 Kurz prokládání

Tento kurz byl vytvořen jako vizualizace postupného načítání obrazových dat pomocí dvou- a jednorozměrného prokládání, které jsou popsány v teoretické části této práce. Kurz byl vytvořen podle aplikace prokládání, zpracované Kamilem Stokláskou v rámci ba-

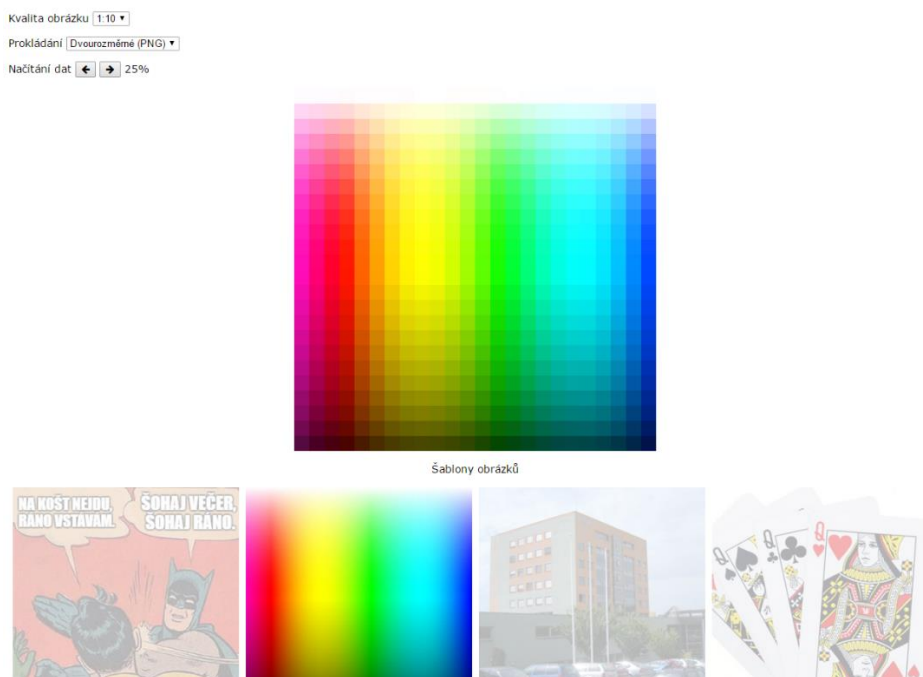
kalářské práce Programová podpora výuky předmětu Počítačová grafika (2014). Jeho aplikace byla vytvořena pomocí technologie Flash. Firma Adobe zastavila vývoj podpory Flash pro mobilní platformy, proto modul vytvořený touto cestou není na tabletu nebo mobilním telefonu zobrazitelný. Navíc je pro jeho spuštění potřeba mít v počítači nainstalovaný Flash Player. Obecně se po příchodu HTML5 od této technologie opouští. V aplikaci je předpřipraveno několik obrázků, na nichž je následně prokládání demonstrováno.

#### 9.4.1 Uživatelské rozhraní

Uživatelské rozhraní je velmi jednoduché. Skládá se z ovládacích prvků nacházejících se v levém horním rohu. Hned pod ovládacími prvky následuje část pro promítání výsledků prokládání. Na úplném konci se nachází několik volitelných obrázků pro demonstraci prokládání.

#### 9.4.2 Ovládání

Ovládání aplikace probíhá prostřednictvím ovládacích prvků v levé horní části obrazovky. Nachází se zde volba označená popiskem *kvalita obrázku*, která ovlivňuje velikost pixelů pro zřetelnější náhled při průběhu prokládání. Poté je zde rolovací nabídka pro volbu typu prokládání jednorozměrné nebo dvourozměrné. Jako poslední se v této sekci nachází tlačítka s šipkami pro změnu množství načtených dat v procentech. Na konci stránky je již zmiňovaná sada předpřipravených obrázků, které je možno kliknutím měnit.



Obr. 36 Ukázka kurzu prokládání

## ZÁVĚR

Hlavním cílem této práce bylo vytvoření webové stránky, která slouží jako sbírka programové podpory (kurzů) pro předmět Počítačová grafika splňující podmínky pro snadné rozšíření o kurzy nové. Pro vytvoření této stránky byl použit redakční systém Wordpress, pro který bylo vytvořeno téma vzhledu s potřebnou funkcionalitou k nahrávání kurzů. Vše je vytvořeno tak, aby bylo možno nové kurzy zakládat pohodlně prostřednictvím uživatelského rozhraní. V teoretické části byla zhodnocena bezpečnostní stránka této aplikace s poukázáním na nejčastější rizika u webových aplikací a byly také popsány technologie použité při implementaci této aplikace.

Práce zahrnuje seznámení se s problematikou rasterizačních algoritmu, kompresních algoritmů a prokládání obrázků. V případě rasterizačních algoritmů se práce věnuje algoritmům pro rasterizaci základních objektů počítačové grafiky, jako jsou úsečka, kružnice, elipsa a křivka. V kapitole o kompresních algoritmech jsou popsány ty nejpoužívanější algoritmy u metod ztrátových a bezztrátových. Jelikož je toto téma velmi rozsáhlé, bylo s vedoucím práce dohodnuto podrobné zpracování pouze bezztrátového algoritmu LZW se zaměřením na kompresi obrázkových dat formátu GIF. Kapitola o prokládání není součástí zadání, byla vytvořena kvůli vytvořenému kurzu prokládání, který je implementován nad rámec zadání pro rozšíření obsahu webové stránky.

Mimo samotnou webovou stránku byly vytvořeny tři plnohodnotné kurzy na témata popisovaná v předchozím odstavci. Kurz rasterizace pomáhá objasnit průběh převodu z vektorové reprezentace objektu do rastrové mřížky, kdy se při postupném vykreslování pixelu uživateli zobrazují i rovnice s výpočty. Kurz bezztrátové komprese LZW detailně popisuje celý průběh nejen samotné komprese obrázkových dat formátu GIF, ale také převod komprimovaných dat do binárního kódu a uložení do struktury tohoto formátu. Na závěr byl vytvořen kurz prokládání demonstrující průběh načítání dat formátů PNG a GIF, které používají prokládání pro zobrazení jejich obrazu ještě před kompletním načtením.

Díky zvoleným technologiím je celá aplikace dostupná online bez nutnosti přídavných technologií pro spuštění jednotlivých kurzů. Na stránce je použit responzivní design a je tedy možné využívat aplikaci i na mobilních zařízeních.

Veškeré součásti této práce, včetně zdrojových kódů aplikací a webové prezentace, jsou nahrány na přiloženém CD.

**SEZNAM POUŽITÉ LITERATURY**

- [1] „What is Computer Graphics?“, 15 4 1998. [Online]. Available: <http://www.graphics.cornell.edu/online/tutorial/>. [Přístup získán 5 4 2016].
- [2] I. M. Dosedla, „Aplikace počítačové grafiky“, 2007. [Online]. Available: [http://www.ped.muni.cz/wtech/03\\_studium/apg/apg\\_02.pdf](http://www.ped.muni.cz/wtech/03_studium/apg/apg_02.pdf). [Přístup získán 5 4 2016].
- [3] Ú. m. F. V. Brno, „Matematika online“, 2005. [Online]. Available: <http://mathonline.fme.vutbr.cz/pg/flash/TeorieGrafika/pocGrafika4.pdf>. [Přístup získán 6 4 2016].
- [4] I. Yūki, „1. díl - Úvod do počítačové grafiky - Rastr vs. vektor“, 2015. [Online]. Available: <http://www.itnetwork.cz/grafika/uvod-do-pocitacove-grafiky-rastr-vs-vektor/>. [Přístup získán 6 4 2016].
- [5] Phrood, „Rastrová grafika“, 2004. [Online]. Available: [https://cs.wikipedia.org/wiki/Rastrov%C3%A1\\_grafika](https://cs.wikipedia.org/wiki/Rastrov%C3%A1_grafika).
- [6] A. S. Gremillion, „The difference between vector and raster images“, 2 5 2011. [Online]. Available: <https://99designs.com/blog/tips/vector-vs-raster-images/>. [Přístup získán 6 4 2016].
- [7] U. P. v. Olomouci, „Počítačová grafika“, [Online]. Available: <http://www.kteiv.upol.cz/frvs/ict-kubricky/?page=pocitacova-grafika/vektorova-grafika>. [Přístup získán 6 4 2016].
- [8] V. Matula, „Grafické práce - vektorová grafika“, [Online]. Available: <http://www.vladimirmatula.zjihlavy.cz/vektorova-grafika.php>.
- [9] „Raster vs Vector“, [Online]. Available: [http://vector-conversions.com/vectorizing/raster\\_vs\\_vector.html](http://vector-conversions.com/vectorizing/raster_vs_vector.html).
- [10] R. F. Ivo Špička, Počítačová geometrie grafika, Ostrava: VŠB – Technická univerzita Ostrava, 2010.
- [11] J. Žára, Moderní počítačová grafika, Brno: Computer Press, 2004.

- [12] P. Veselý, „Blok 4 - Rasterizace objektu,“ [Online]. Available: [http://www.fe.i.doevil.cz/fei/2\\_rocnik/\[ZS\]%20IPOGR%20-%20Pocitacova%20grafika/Teorie%20LEARN/LEARN%20PG%20Blok%204%20-%20Rasterizace%20objektu.pdf](http://www.fe.i.doevil.cz/fei/2_rocnik/[ZS]%20IPOGR%20-%20Pocitacova%20grafika/Teorie%20LEARN/LEARN%20PG%20Blok%204%20-%20Rasterizace%20objektu.pdf).
- [13] J. Sečkař, „Algoritmy a rasterizace 2D grafických objektů,“ 2007. [Online]. [Přístup získán 13 4 2016].
- [14] J. Kennedy, „A Fast Bresenham Type Algorithm For Drawing Circles,“ Oregon State University, [Online]. Available: <http://web.engr.oregonstate.edu/~sllu/bcircle.pdf>.
- [15] s. r. o. Vydavatelství Nová média, „Elipsa,“ [Online]. Available: <http://www.matematika.cz/elipsa>. [Přístup získán 14 4 2016].
- [16] „Úvod do křivek,“ 10 1 2010. [Online]. Available: [https://cs.wikibooks.org/wiki/Geometrie/%C3%9Avod\\_do\\_k%C5%99ivek](https://cs.wikibooks.org/wiki/Geometrie/%C3%9Avod_do_k%C5%99ivek).
- [17] „Geometrické modelovanie kriviek,“ [Online]. Available: <http://www.netgraphics.sk/geometricke-modelovanie-kriviek>.
- [18] O. Kvasnička, „Křivky v počítačové grafice,“ [Online]. Available: [http://home.zcu.cz/~smolik/zpg/cviceni/cv\\_08.html](http://home.zcu.cz/~smolik/zpg/cviceni/cv_08.html).
- [19] „Root.cz,“ 2007. [Online]. Available: <http://www.root.cz/clanky/vytvarime-krivky-v-postscriptu/>.
- [20] „Kompresie rastrových dat,“ Mendelova univerzita v Brně, [Online]. Available: [https://is.mendelu.cz/eknihovna/opory/zobraz\\_cast.pl?cast=6330](https://is.mendelu.cz/eknihovna/opory/zobraz_cast.pl?cast=6330). [Přístup získán 2016].
- [21] „Programujeme JPEG: diskrétní kosinová transformace (DCT),“ 2007. [Online]. Available: <http://www.root.cz/clanky/programujeme-jpeg-diskretni-kosinova-transformace-dct/>.
- [22] P. I. J. P. P. Mgr. Tomáš Foltýnek, „Komprimace a šifrování,“ Mendelova Univerzita v Brně, [Online]. Available: [https://akela.mendelu.cz/~foltynek/KAS/elearning/KAS\\_PDF.pdf](https://akela.mendelu.cz/~foltynek/KAS/elearning/KAS_PDF.pdf).

- [23] I. V. Hordějčuk, „Kompresní algoritmus LZW,“ [Online]. Available: <http://voho.cz/wiki/algoritmus-lzw/>.
- [24] „What's In A GIF - LZW image data,“ 2016. [Online]. Available: [http://giflib.sourceforge.net/whatsinagif/lzw\\_image\\_data.html#color\\_table\\_size](http://giflib.sourceforge.net/whatsinagif/lzw_image_data.html#color_table_size).
- [25] I. V. Hordějčuk, „Huffmanovo kódování,“ [Online]. Available: <http://voho.cz/wiki/kodovani-huffman/>.
- [26] K. Stoklásková, „Programová podpora výuky předmětu,“ Zlín, 2014.
- [27] „Případ GIF,“ 2006. [Online]. Available: <http://www.root.cz/clanky/pripad-gif/>.
- [28] J. Skřivan, „GIF, JPEG a PNG – jak a kdy je použít?,“ 2002. [Online]. Available: <https://www.interval.cz/clanky/gif-jpeg-a-png-jak-a-kdy-je-pouzit/>.
- [29] P. Ferschmann, „Bezpečnost na webu – přehled útoků na webové aplikace,“ 2008. [Online]. Available: <https://www.zdrojak.cz/clanky/prehled-utoku-na-webove-aplikace/>.
- [30] I. Kulman, „SQL Injection pre každého,“ 2014. [Online]. Available: <https://www.zdrojak.cz/clanky/sql-injection-pre-kazdeho/>.
- [31] „Únos spojení,“ [Online]. Available: [https://cs.wikipedia.org/wiki/%C3%9Anos\\_spojen%C3%AD](https://cs.wikipedia.org/wiki/%C3%9Anos_spojen%C3%AD).
- [32] J. Pejša, „Co je Cross-site scripting jak mu předcházet,“ 2009. [Online]. Available: <https://www.zdrojak.cz/clanky/co-je-xss-jak-mu-predchazet/>.
- [33] J. Pejša, „Co je Cross-Site Request Forgery a jak se mu bránit,“ 2008. [Online]. Available: <https://www.zdrojak.cz/clanky/co-je-cross-site-request-forgery-a-jak-se-branit/>.
- [34] „About Wordpress,“ [Online]. Available: <https://wordpress.org/about/>.
- [35] J. T. a. B. c. Mark Otto, „Bootstrap,“ [Online]. Available: <http://getbootstrap.com/>.
- [36] Designmodo, „FlatUI,“ [Online]. Available: <http://designmodo.github.io/Flat-UI/>.

- [37] M. Dočekal, „Správa linuxového serveru: Webový server Nginx,“ 2011. [Online]. Available: <http://www.linuxexpres.cz/praxe/sprava-linuxoveho-serveru-webovy-server-nginx>.
- [38] „Lehký úvod do LDAP,“ 2000. [Online]. Available: <http://www.root.cz/clanky/lehky-uvod-do-ldap/>.
- [39] M. Lasoň, „Adresářová služba X.500 a LDAP,“ [Online]. Available: <http://homel.vsb.cz/~las03/ldap/>.
- [40] T. D. Foundation, „Dojo Toolkit,“ [Online]. Available: <https://dojotoolkit.org/>.
- [41] T. D. Foundation, „Dojo Toolkit Reference Guide,“ [Online]. Available: <https://dojotoolkit.org/reference-guide/1.10/>.
- [42] gskinner, „EaselJS,“ [Online]. Available: <http://www.createjs.com/easeljs>.
- [43] „Kompresa rastrového obrazu,“ [Online]. Available: <https://sites.google.com/site/xgrafika/kompresa-rastroveho-obrazu>.

**SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK**

GIF	Graphics Interchange Format
LZW	Lempel-Ziv-Welch 84
PNG	The Portable Network Graphics
W3C	World Wide Web Consortium
RGBA	red, green, blue, alpha
UTB	Univerzita Tomáše Bati ve Zlíně
PHP	Hypertext Preprocessor
CSS	Cascading Style Sheets
HTML	HyperText Markup Language
VPS	Virtuální privátní server
SSL	Secure Sockets Layer
IMAP	Internet Message Access Protocol
POP3	Post Office Protocol verze 3
TLS	Transport Layer Security
http	Hypertext Transfer Protocol
https	Hypertext Transfer Protocol Secure
FAI	Fakulta aplikované informatiky
UTB	Univerzita Tomáše Bati ve Zlíně
LDAP	Lightweight Directory Access Protocol
BSD	Berkeley Software Distribution
TCP/IP	Transmission Control Protocol/Internet Protocol
DN	Distinguished Name
DCT	Discrete Cosine Transform
WYSIWYG	What you see is what you get

API	Application Programming Interface
AJAX	Asynchronous JavaScript and XML
DOM	Document Object Model
TLS	Transport Layer Security
URL	Uniform Resource Locator
CAPTCH	completely automated public Turing test to tell computers and humans apart

**SEZNAM OBRÁZKŮ**

Obr. 1 Způsob zpracování obrázku v bitmapové grafice (vpravo rastrová grafika, vlevo vektorová) [5].....	12
Obr. 2 Tvorba logotypu v editoru [8] .....	13
Obr. 3 Přiblížení rastrového obrázku [9] .....	14
Obr. 4 Velikost směrnice $m$ pro různé sklony úsečky [11] .....	15
Obr. 5 Vykreslení úsečky algoritmem DDA (rasterizace ve směru osy $x$ ) [3].....	16
Obr. 6 Princip Bresenhamova algoritmu [3].....	17
Obr. 7 Symetrie kružnice [14] .....	19
Obr. 8 Část kružnice v rastru [11].....	20
Obr. 9 Elipsa s ohnisky $E$ a $F$ [15].....	21
Obr. 10 Změna řídicí osy při rasterizaci elipsy [11] .....	21
Obr. 11 Interpolační křivka.....	23
Obr. 12 Fergusonova kubika [13] .....	24
Obr. 13 Kubické Hermitovské polynomy [18] .....	26
Obr. 14 Aproximační křivka .....	26
Obr. 15 Bézierova kubická křivka [19] .....	29
Obr. 16 Příklad původního a obrazu po DCT kompresi .....	31
Obr. 17 Několikanásobně zvětšený GIF obrázek s vyznačenými pixely. ....	34
Obr. 18 Obrázková data formátu GIF v šestnáctkové soustavě.....	35
Obr. 19 Očíslované barevné pixely podle tabulky barev.....	35
Obr. 20 Datový blok formátu GIF [24].....	36
Obr. 21 Průběh převodu hodnot při LZW kompresi do binárního kódu [24].....	37
Obr. 22 Dekomprese binárního kódu v LZW [24] .....	37
Obr. 23 Jednorozměrné prokládací schéma [26] .....	39
Obr. 24 Dvourozměrné prokládací schéma [26].....	40
Obr. 25 Náhled webu na mobilním telefonu.....	49
Obr. 26 Vzhled základních elementů Flat UI .....	52
Obr. 27 Příklad LDAP adresářového stromu [39] .....	53
Obr. 28 Přihlašovací formulář .....	54
Obr. 29 Rozložení webu .....	55
Obr. 30 Administrace webu a nahrávání kurzu .....	55
Obr. 31 Ukázka definice modulu a dědičnosti v Dojo.....	58

---

Obr. 32 Ukázka definice černé kružnice pomocí canvas.....	58
Obr. 33 Kreslení černé kružnice pomocí EaselJS.....	59
Obr. 34 Ukázka kurzu rasterizace.....	60
Obr. 35 Ukázka kurzu komprese LZW.....	61
Obr. 36 Ukázka kurzu prokládání.....	62

**SEZNAM TABULEK**

Tab. 1 Přehled kompresních metod [20].....	30
Tab. 2 Slovník po inicializaci .....	32
Tab. 3 Slovník po kompresi.....	33
Tab. 4 Tabulka po dekompresi .....	34
Tab. 5 Barvy obsažené v obrázku.....	35

## SEZNAM PŘÍLOH

PI      Obsah vloženého CD

## **PŘÍLOHA P I: OBSAH VLOŽENÉHO CD**

- Elektronická verze této práce
- Zdrojové kódy webové stránky
- Zdrojové kódy kurzu Resterizační algoritmy
- Zdrojové kódy kurzu LZW komprese
- Zdrojové kódy kurzu prokládání obrázků