VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
FACULTY OF MECHANICAL ENGINEERING

ÚSTAV AUTOMATIZACE A INFORMATIKY
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

VEKTOROVÝ GRAFICKÝ EDITOR

VECTOR GRAPHIC EDITOR

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. Jan Bartoněk
AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. Jan Roupec, Ph.D.
SUPERVISOR

BRNO 2022

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání diplomové práce

Ústav:	Ústav automatizace a informatiky
Student:	**Bc. Jan Bartoněk**
Studijní program:	Aplikovaná informatika a řízení
Studijní obor:	bez specializace
Vedoucí práce:	**doc. Ing. Jan Roupec, Ph.D.**
Akademický rok:	2021/22

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Vektorový grafický editor

Stručná charakteristika problematiky úkolu:

Grafický vektorový editor je velmi užitečným programem pro celou řadu činností. Existuje více sw titulů, které tuto problematiky pokrývají. Smyslem zadání je zmapovat situaci a pro konkrétní požadavky vytvořit jednoduchý a přitom výkonný a spolehlivý editor.

Cíle diplomové práce:

Autor prostuduje problematiku počítačové geometrie se zvláštním zřetelem na vektorovou grafiku. Bude provedena analýza vhodných funkcí grafického editoru. Výstupem realizační části bude vektorový editor vytvořený jako Windows aplikace. Editor bude schopen kreslit základní vektorové objekty včetně vkládání textu. Předpokládá se práce slespoň s grafickým formátem svg a export do vybraného rastrového formátu.

Seznam doporučené literatury:

MARTIŠEK, D., Matematické principy grafických systémů, Littera, Brno, 2002, ISBN 80-85763-19-2.

PETZOLD, C. Programming Windows: Writing Windows 8 Apps With C# and XAML (Developer Reference) 6th Edition. Microsoft Press. 2013.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2021/22

V Brně, dne

L. S.

doc. Ing. Radomil Matoušek, Ph.D. ředitel ústavu	doc. Ing. Jaroslav Katolický, Ph.D. děkan fakulty

ABSTRAKT

Cílem této diplomové práce je navrhnout jednoduchý, ale přitom výkonný vektorový grafický editor. Rešeršní část práce se věnuje problematice počítačové grafiky, její historii, dělení a jejím základním pojmům. Dále je v této části rozebrána rastrová a vektorová grafika a v neposlední řadě jsou zde uvedeny a popsány nejpoužívanější vektorové grafické editory. Samotný vektorový grafický editor je naprogramován v jazyce Python s využitím knihovny PyQt5. Zhodnocení výsledku práce je uvedeno v závěru.

ABSTRACT

The aim of this diploma thesis is to design a simple, yet powerful vector graphics editor. The research part of the thesis deals with the issue of computer graphics, its history, the division, and its basics concepts. Furthermore, this section discusses raster and vector graphics and last but not least, it lists and describes the most popular vector graphics editors. The vector graphics editor itself is programmed in Python using the PyQt5 library. Finally, the conclusion reviews and evaluates the work and its results.

KLÍČOVÁ SLOVA

počítačová grafika, vektorové grafické editory, programovací jazyk Python, knihovna PyQt5

KEYWORDS

computer graphics, graphics vector editors, programming language Python, PyQt5 library

ÚSTAV AUTOMATIZACE A INFORMATIKY

2022

BIBLIOGRAFICKÁ CITACE

BARTONĚK, Jan. *Vektorový grafický editor,* Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav automatizace a informatiky, 2022, 65 s. Diplomová práce. Vedoucí práce: doc. Ing. Jan Roupec, Ph.D.

PODĚKOVÁNÍ

Rád bych poděkoval mé rodině a přátelům za podporu a trpělivost, kterou se mnou při psaní této práce měli. Dále bych chtěl poděkovat panu doc. Ing. Janu Roupcovi, Ph.D. za odborné vedení mé práce a paní Ing. Blance Rybkové za přínosné konzultace.

ČESTNÉ PROHLÁŠENÍ

Prohlašuji, že tato práce je mým původním dílem, vypracoval jsem ji samostatně pod vedením vedoucího práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury.

Jako autor uvedené práce dále prohlašuji, že v souvislosti s vytvořením této práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následku porušení ustanovení § 11 a následujících autorského zákona c. 121/2000 Sb., včetně možných trestně právních důsledků.

V Brně dne 20. 5. 2022 ……

Bc. Jan Bartoněk

OBSAH

1 ÚVOD

Tato diplomová práce byla vytvořena s cílem seznámit čtenáře s problematikou počítačové grafiky a především vektorových grafických editorů. Obsahem praktické části této diplomové práce je navrhnout a naprogramovat vlastní vektorový grafický editor, který bude svými funkcionalitami odpovídat zadání.

Vektorové grafické editory jsou velmi užitečné programy, které se využívají pro celou řadu činností. Mezi tyto činnosti patří například výroba různých tiskovin, návrhy výkresů pro rozličné stroje či budovy, nebo také tvorba grafiky pro počítačové hry. S produkty počítačové grafiky, a tedy i grafických editorů, se člověk setkává denně téměř na každém kroku, aniž by si to možná uvědomoval.

Tématu počítačové grafiky se věnuje první kapitola rešeršní části této diplomové práce. Je v ní uvedena historie tohoto oboru informačních technologií, dále její rozdělení a základní pojmy. Druhá kapitola rešeršní části je věnována rastrové grafice a jejím formátům. Ve třetí kapitole se autor věnuje tématu vektorové grafiky. Detailní popis vybraných vektorových editorů je uveden v kapitole čtvrté.

Praktická část diplomové práce popisuje návrh vlastního grafického editoru a jeho implementaci. Dále popisuje vybrané funkcionality tohoto editoru.

Výsledky a další možné využití vektorového grafického editoru jsou diskutovány v závěru.

2 POČÍTAČOVÁ GRAFIKA

Díky počítačové grafice lze v dnešní době vytvářet, upravovat a sdělovat grafickou informaci. V běžném životě se s ní člověk setkává na denní bázi – od etiket různých výrobků, přes dopravní značky, až po velkolepé celovečerní filmy. Z technického hlediska se jedná o obor informatiky, ovšem lze na ni nahlížet i z uměleckého pohledu, kde tvoří samostatnou kategorii.

Následující kapitoly se věnují stručné historii počítačové grafiky, základním pojmům počítačové grafiky a jejímu rozdělení.

2.1 Historie počítačové grafiky

Historie počítačové grafiky je velice obsáhlá, a proto budou v této podkapitole popsány pouze ty nejdůležitější body, které ovlivnily vývoj tohoto vědního oboru.

Počítačová grafika se vyvíjela souběžně s rozvojem počítačů a s jejich využitím pro širší veřejnost. Ovšem její historie sahá i do dob, kdy počítače ještě nebyly běžně využívané. Již ve 40. a 50. letech 20. století se siru Johnu Whitneymu a jeho bratru Jamesovi podařilo vytvořit animaci pomocí analogového počítače, jenž byl inspirován protiletadlovým počítačem, který se využíval za II. světové války [1].

Za zmínku jistě stojí i pan Benjamin Laposky, s jehož jménem je spojen první grafický obrázek. Ten byl vytvořen pomocí osciloskopu, který umožňoval vytvářet elektronické abstrakce díky úpravě elektronových paprsků. [2]

Dalším významným projektem byl sálový počítač Whirlwind vytvořený v roce 1951 Jayem Forresterem a Robertem Everettem z MIT (Massachussetts Institute of Technology). Tento počítač dokázal zobrazovat data z reálného času na televizní obrazovce. V roce 1955 byl vyvinut počítač SAGE (Semi-Automatic Ground Equipment), který vycházel přímo z počítače Whirlwind a umožňoval vykreslovat radarové snímky pomocí vektorové grafiky. Ve stejném roce vzniklo také světelné pero. Jednalo se o vstupní zařízení, díky němuž mohl uživatel kreslit přímo na CRT (Cathod Ray Tube) monitor. Roku 1959 byl společnostmi General Motors a IBM vyvinut DAC-1 (Design Augmented by Computers-1). Tento program se využíval pro vytváření návrhů automobilů a jejich součástí. Lze jej považovat za první CAD (Computer-Aided Design) systém. [1] [3]

Termín *počítačová grafika* poprvé použil v roce 1960 ve své práci William Fetter [2]. Vytvořil první 3D animovaný drátový model lidského těla. Model se později stal jedním z nejvíce ikonických děl rané historie počítačové grafiky a je označován jako „Boeing Man“ (z důvodu, že Fetter v té době pracoval pro firmu Boeing). Technika drátového modelu byla později použita v prvním díle filmové série Hvězdné války. [1]

V roce 1962 byl vytvořen studentem MIT Ivanem Sutherlandem program s názvem Sketchpad. Jednalo se o revoluci jak z hlediska počítačové grafiky, tak z hlediska programování. Uživatel mohl kreslit pomocí zmíněného světelného pera

různé jednoduché tvary, svou práci si uložit a následně poté i znovu otevřít. Sketchapad byl první program s GUI (Graphical User Interface), a mimo jiné položil základy objektově orientovanému programování. [1]

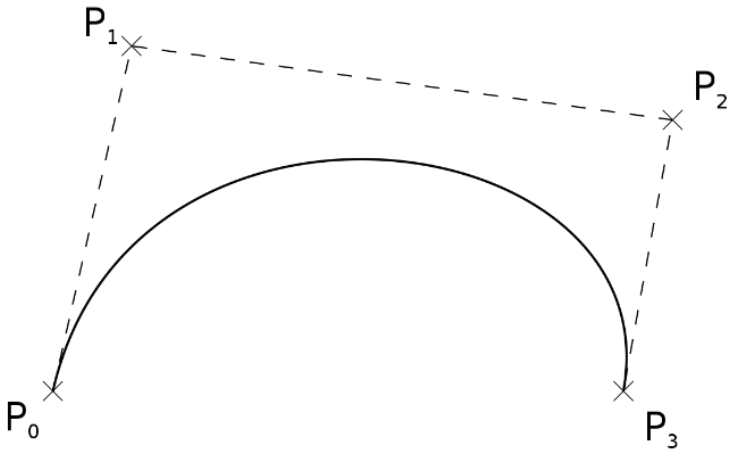
Dalším významným momentem počítačové grafiky bylo, když v období mezi lety 1959 a 1962 navrhli nezávisle na sobě Pierre E. Bézier a Paul de Casteljau parametrickou křivku, která je určena čtyřmi body P_0, P_1, P_2, P_3 a je definována následujícími parametrickými rovnicemi:

$$Q(t) = \sum_{i=0}^{3} P_i B_i; (t) \in \langle 0; 1 \rangle \tag{1}$$

$$B_0(t) = (1-t)^2;\ B_1(t) = 3t(1-t)^2;$$
$$B_2(t) = 3t^2(1-t)^2;\ B_3(t) = t^3 \tag{2}$$

Jde o kubickou parabolu, procházející body P_0 a P_3. Úsečky P_0P_1 a P_3P_2 určují tečny v krajních bodech a jejich směrnice jsou číselně rovny třetině délky těchto úseček. Bézierovy křivky, jak se jim začalo říkat, jsou v technické praxi oblíbené hlavně z důvodu jejich snadného napojování. Ve spojovacím bodě lze totiž snadno docílit spojitosti křivky i její derivace tím, že v bodě spojení zajistíme společnou tečnu. [4]

V dnešní době je kreslení Bézierových křivek obsaženo v každém grafickém programu. Příklad Bézierovy křivky lze vidět na obr. 1.

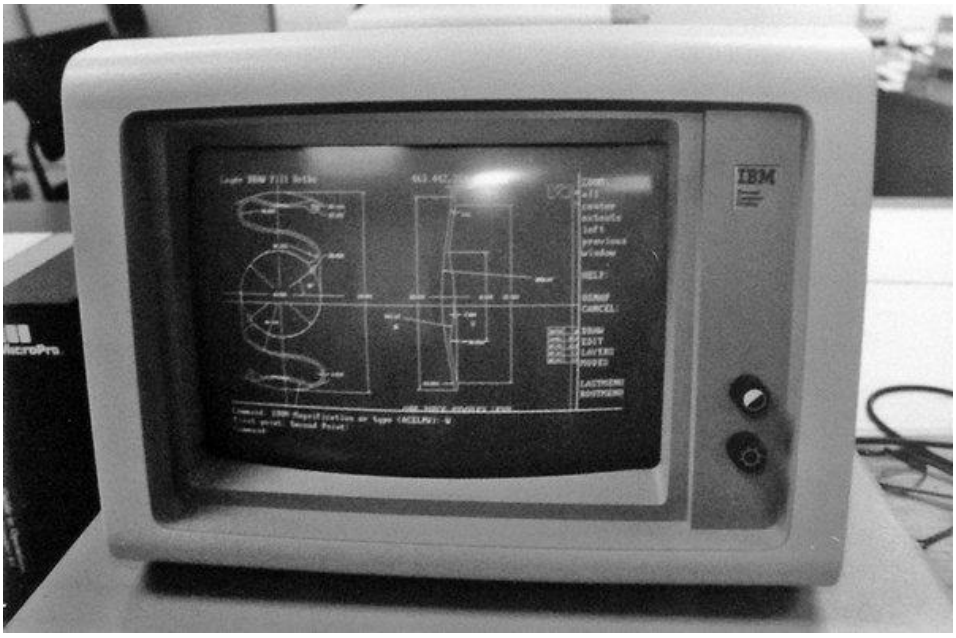


Obr. 1: Bézierova křivka

Na konci sedmdesátých a počátkem osmdesátých let dvacátého století dochází k postupnému rozšiřování osobních počítačů mezi běžnou veřejnost. Příkladem je rok 1977, kdy byl na trh uveden Apple II, což byl první osobní počítač s grafickým výstupem. Díky této skutečnosti vzrostl zájem o počítačovou grafiku v zábavním průmyslu, konkrétně u filmů a u videoher. Hlavní firmy zabývající se odvětvím videoher byly Atari, Sega a Nintendo. Kinematografií v kombinaci s počítačovou grafikou, tedy animovanými filmy, se zabývala a dodnes zabývá společnost Pixar. V této oblasti se začíná využívat

tzv. CGI (Computer Generated Graphic), tedy počítačem generovaná grafika. Společně s tím bylo zavedeno barevné klíčování, díky němuž mohlo být nahrazeno jednobarevné pozadí libovolnou scénou. [3]

V období osmdesátých let, konktrétně v roce 1981, byl představen produkt CATIA – první 3D CAD systém, který existuje dodnes. V následujícím roce byl vyvinut dnes velmi známý program AutoCAD, který je zobrazen níže na obr. 2. Ve svých začátcích však umožnoval práci pouze s drátovými modely. Na konci osmdesátých let se již začínala 3D grafika používat ve zřetelněji globálnějším měřítku. [5]



Obr. 2: AutoCAD verze 1.0 [5]

Výraznější využití 3D grafiky bylo však až v devadesátých letech dvacátého století. V tomto období začaly vznikat první 3D videohry a také byl v roce 1995 natočen první celovečerní film animovaný počítačovou grafikou – Toy Story (Příběh hraček). Film vydala již zmíněná společnost Pixar. [1]

V období po přelomu tisíciletí nastal obrovský rozmach počítačové grafiky. Začala být využívána ve více a více oblastech lidské činnosti. Televizní reklamy byly z velké části tvořeny právě počítačovou grafikou, stejně tak jako noviny nebo časopisy. Rychlým tempem se začínala objevovat i na webových stránkách.

V dnešní době je počítačová grafika na takové úrovni, že již nelze lidským okem poznat, zda se, například ve filmovém průmyslu, jedná o skutečný záběr, či nikoliv.

2.2 Základní pojmy počítačové grafiky

V následující podkapitole budou zmíněny základní pojmy, které se budou dále v textu vyskytovat nebo byly autorem této práce považovány za podstatné z hlediska pochopení tématiky počítačové grafiky.

Pixel

Pixel je zkratka z anglického pojmu „Picture Element“, tedy obrazový bod. Jedná se o nejmenší, bezrozměrnou jednotku rastrové grafiky. Představuje jeden bod obrázku, charakterizovaný jasem a barvou. Tyto body tvoří obdélníkovou síť, ve které je možné jednoznačně každý pixel identifikovat podle jeho souřadnic. [6]

DPI / PPI

DPI (Dots Per Inch) nebo PPI (Pixel Per Inch) je údaj určující, kolik bodů či pixelů se vejde do délky jednoho palce. [6]

Rozlišení

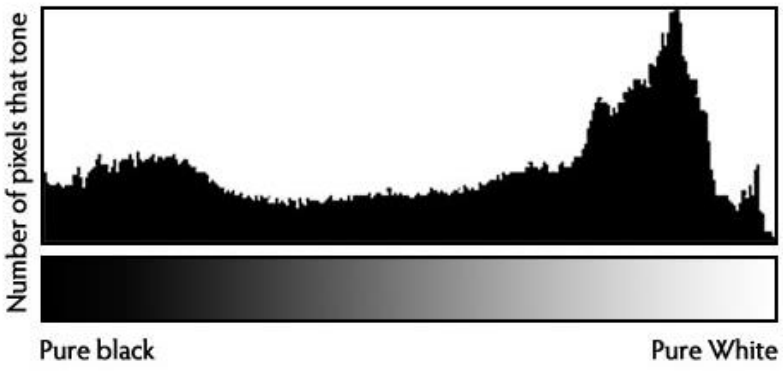
Hodnota rozlišení je udávána ve dvou hlavních jednotkách – v pixelech a v DPI. Rozlišení monitoru je počet pixelů, které mohou být zobrazeny na obrazovce, udává se tedy v pixelech. Ovšem rozlišení tiskáren se udává v DPI, popřípadě v PPI. [6]

Odstín

Odstín neboli tón je základní vlastností barvy. Podle odstínu jsou rovněž barvy ve většině pojmenovány. [4]

Jas

Jas je v počítačové grafice vnímán jako intenzita světla vycházejícího z obrazovky. Se stoupající intenzitou světla je osvětlený objekt jasnější, světlejší, a tudíž má větší jas. Tuto veličinu lze znázornit pomocí histogramu. Histogram je graf zobrazující četnost pixelů v obraze na škále od černé po bílou barvu. Příklad histogramu je zobrazen na obr. 3. [7]



Obr. 3: Histogram jasu [8]

Sytost

Sytost je definována jako čistota a intenzita barvy. Čím je barva sytější, tím méně v sobě obsahuje bílé a černé příměsi. [7]

Kontrast

Termín kontrast označuje rozdíl mezi nejtmavším a nejsvětlejším bodem v černobílém obraze. Pro barevný obraz je zaveden pojem barevný kontrast, jenž je určen jako rozdíl mezi vzájemně doplňkovými barvami. Kontrast lze graficky zobrazit pomocí histogramu, podobně jako jas. [7]

Barevná hloubka

Barevná hloubka určuje, kolik bitů je potřeba k popisu konkrétní barvy v obrázku. Čím větší je tedy barevná hloubka, tím více může výsledný obraz nabývat barev. [6] Tab. 1 zobrazuje několik barevných hloubek a k nim příslušný počet barev.

Tab. 1: Příklad barevné hloubky

Barevná hloubka	Počet barev
4 bity	16
8 bitů	256
16 bitů	65 536

2.3 Dělení počítačové grafiky

Základní rozdělení počítačové grafiky je dle počtu zobrazených dimenzí. Jedná se tedy o 2D (rovinnou) a 3D (prostorovou) grafiku.

2.3.1 2D grafika

2D grafika se zabývá grafickou informací popsanou v rovině. Dále se dělí na grafiku rastrovou a vektorovou.

Rastrová grafika popisuje obraz pomocí pravidelné sítě pixelů, což je dvourozměrná matice bodů. Každý pixel matice obsahuje informaci o jeho vlastnostech, jako jsou barva, jas, průhlednost bodu, či kombinace uvedených. [2]

Vektorová grafika naopak neukládá informace o jednotlivých bodech, ale informace o základních geometrických tvarech, jako jsou přímky, kružnice, polygony a další. [2]

Oba typy zmíněných druhů 2D grafiky jsou důkladně rozebrány v samostatných kapitolách níže, proto se zde nachází pouze jejich stručný popis.

2.3.2 3D grafika

3D grafika zobrazuje obrazovou informaci v trojrozměrném prostoru a je velice podobná 2D vektorové grafice. Zásadní rozdíl je v tom, že se geometrická data neukládají pouze v rovině, ale v prostorové souřadné soustavě.

Základním geometrickým útvarem 3D grafiky jsou polygony. Z nich se dají vytvářet libovolné trojrozměrné objekty a scény. Tyto objekty v dnešní době vypadají velice realisticky, a to z důvodu věrné simulace světelných a optických jevů, kterými jsou stíny, odrazy či lom světla. Z výsledného objektu či scény se následně renderuje 2D obrázek.

Je důležité zmínit, že vektorová grafika i 3D grafika se uživateli ve finále prezentuje jako grafika rastrová. Tento fakt je dán použitým zobrazovacím zařízením, tedy rastrovým displejem. [2]

2.4 Barevné modely

Barevné modely slouží k popisování způsobů míchání základních barev pro dosažení co nejvěrnějšího napodobení reality. Jedná se o kombinace základních barev a vlastností, které jsou popsány výše – tj. odstín, jas a sytost.

V současné praxi jsou využívány tyto barevné modely: RGB(A), CMYK, HSV (HSB) a HSL. Použití jednotlivého barevného modelu je dáno zařízením. Například barevný model RGB využívají zařízení, která vyzařují světla, tedy monitory, dataprojektory a jiné. Naopak barevný model CMYK je použit u zařízení, která využívají odrazu světla, například tiskárny.

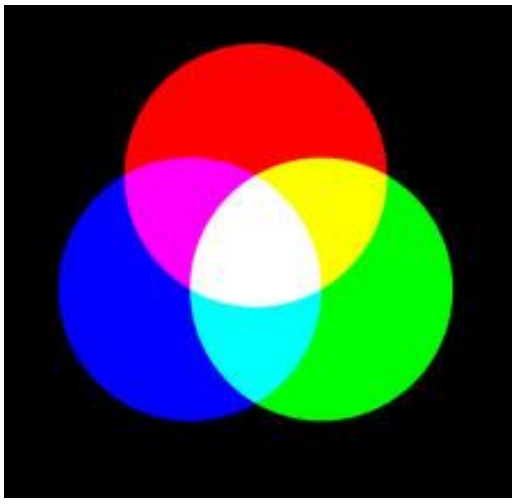
Každé zařízení má jinou dosažitelnou oblast barev v určitém barevném prostoru. Tato oblast se nazývá gamut. Při překročení gamutu je požadovanou barvu možné zobrazit pouze přibližně. [7]

V následujících podkapitolách jsou detailně rozebrány všechny výše uvedené barevné modely.

2.4.1 RGB(A)

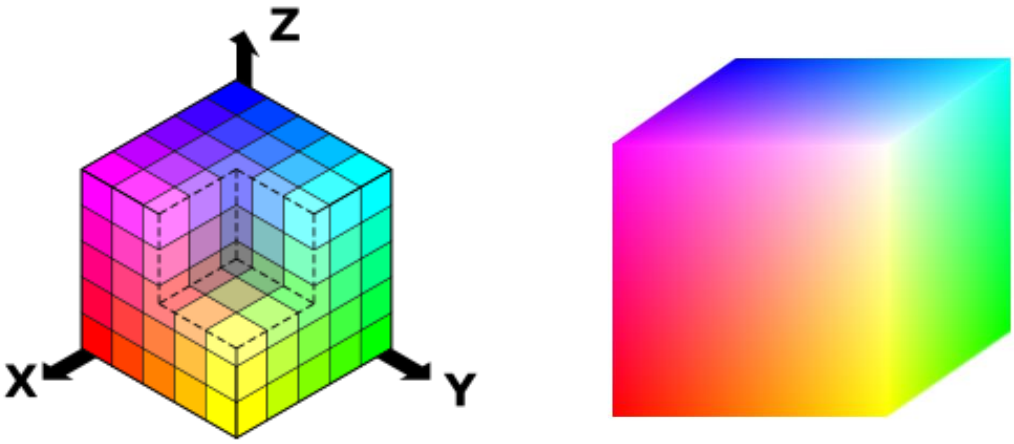
Nejrozšířenějším barevným modelem v počítačové grafice je RGB. Tento model je pojmenován podle tří základních barev, kterými je tvořen. Těmito barvami jsou červená (Red), zelená (Green) a modrá (Blue).

Barevný model RGB využívá aditivního způsobu míchání barev. To, jak název napovídá, znamená, že se jednotlivé složky barev sčítají. Aditivní způsob míchání barev si lze představit jako míchání světel barevných reflektorů, které svítí na jedno místo. Ve stručnosti se jedná o to, že v místě, kde se dvě ze základních barev překrývají, tam vzniká barva doplňková k barvě třetí. Doplňková barva modré je žlutá, pro zelenou je to purpurová a doplňková barva červené je azurová. V části, kde se překrývají všechny tři základní barvy, vzniká barva bílá. Bílá barva rovněž vzniká smícháním barvy základní a barvy k ní doplňkové. Pro lepší představu je tento způsob míchání barev znázorněn na obr. 4. [4]



Obr. 4: Aditivní míchání barev [7]

RGB model lze znázornit pomocí krychle zobrazené na obr. 5. V šesti vrcholech této krychle se nacházejí základní a doplňkové barvy. Ve zbývajících dvou vrcholech pak barvy černá a bílá. Na osách souřadného systému se nalézají barvy základní. V počátku souřadného systému je barva černá a v protilehlém vrcholu leží barva bílá. Základní barvy a jejich barvy doplňkové jsou vždy v protilehlých vrcholech krychle. Dále platí, že směrem k černému vrcholu klesá jas barev a s blížícím se bílým vrcholem klesá jejich sytost. [4]



Obr. 5: RGB model znázorněný na krychli [4]

Následující tabulka tab. 2 zobrazuje kódování RGB modelu při 24 bitové hloubce. Existují dva způsoby, jak kódovat barvu u RGB modelu – pomocí dekadické a pomocí hexadecimální soustavy.

V prvním případě se kód zapíše jako tři čísla v rozsahu 0-255, která představují podíl barevné složky ve výsledné barvě. Čísla jsou v pořadí červená, zelená modrá.

Výslednou barvu je možné také zapsat prostřednictvím tří hexadecimálních čísel, která jsou v rozsahu od 0 do FF. Tento způsob kódování se využívá častěji.

Tab. 2: Kódování barev RGB modelu

Barva	Dekadický kód	Hexadecimální kód
červená	255,0,0	FF0000
zelená	0,255,0	00FF00
modrá	0,0,255	0000FF
azurová	0,255,255	00FFFF
purpurová	255,0,255	FF00FF
žlutá	255,255,0	FFFF00
bílá	255,255,255	FFFFFF
černá	0,0,0	000000

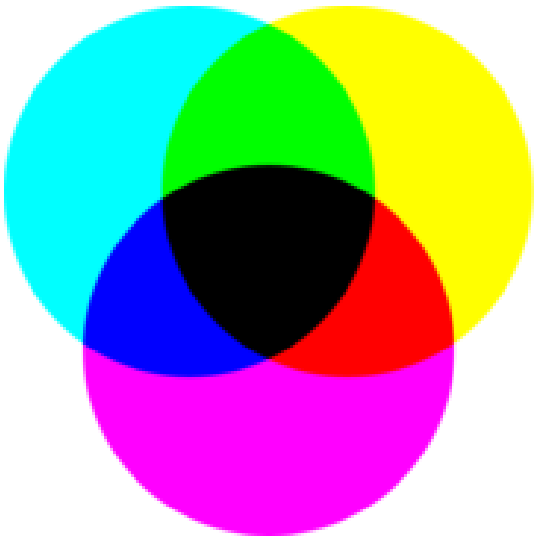
Písmeno *A*, které je v názvu této podkapitoly uvedeno v závorce, je čtvrtý parametr značící alfa-kanál. Ten představuje průhlednost a obvykle je taktéž v rozsahu hodnot 0–255. [6]

2.4.2 CMY(K)

Druhým nejrozšířenějším barevným modelem je CMY. Stejně jako model RGB je pojmenován podle základních barev, jež ho tvoří. Jedná se o barvu azurovou (Cyan), purpurovou (Magenta) a žlutou (Yellow).

Zásadní rozdíl mezi tímto barevným modelem a výše popsaným je to, že zde je využíváno subtraktivního míchání barev. To znamená, že základní plocha je bílá, nanesená barva určuje, která část barevného spektra se pohltí a která odrazí. Složka, která je pohlcena, se odčítá od bílé. Tento způsob míchání barev lze považovat za opak aditivního. Základní barvy u modelu CMY jsou totiž doplňkovými barvami modelu RGB a opačně. [4]

Pro názornost je na obr. 6 zobrazeno schéma subtraktivního míchání barev. Podobně jako u aditivního míchání barev platí, že při překrytí dvou základních barev vzniká barva doplňková ke zbylé základní barvě. Ovšem při překrytí všech tří základních barev dostáváme barvu černou. Tato barva vzniká taktéž kombinací základní barvy a barvy k ní doplňkové. Bílou barvu logicky dostaneme při absenci všech ostatních barev. Při tisku je bílá zhotovena jako nepotištěná plocha papíru. Jednotlivé barevné složky se udávají v rozsahu od 0 do 100 %.



Obr. 6: Subtraktivního míchání barev [7]

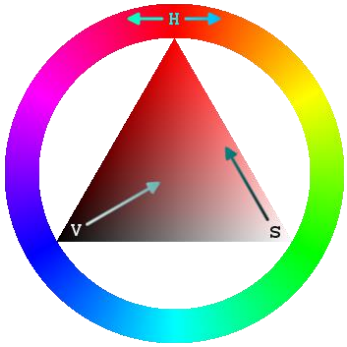
Jak již bylo zmíněno v úvodu této podkapitoly, barevný model CMY se využívá obzvláště v tiskárnách. Přesněji řečeno, využívá se především jeho rozšířená verze, tj. model CMYK. Tento model je doplněn o barevný kanál s černou barvou, označený písmenem *K* (Key). Díky tomuto modelu lze černou barvu zobrazit mnohem lépe a zároveň dochází k úspoře toneru v tiskárně. Kombinací použitých barev v modelu CMY totiž ve skutečnosti nevznikne sytě černá barva. [6]

2.4.3 HSV (HSB)

Název barevného modelu HSV je rovněž složen z počátečních písmen slov, jež jej charakterizují. Tato písmena znamenají odstín (Hue), sytost (Saturation) a hodnotu (Value). Hodnotou je myšlen jas. Proto se někdy název tohoto barevného modelu uvádí jako HSB, kde písmeno *B* značí právě zmíněný jas (Brightness).

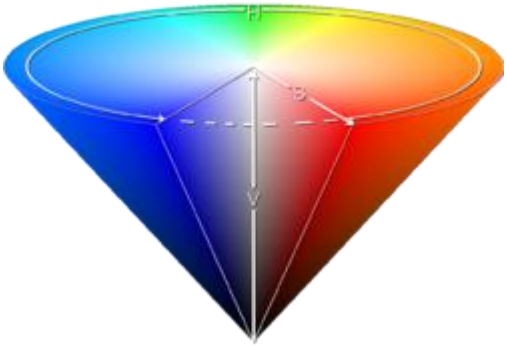
Díky jmenovaným vlastnostem je tento model nejblíže intuitivnímu vnímání člověkem. Tím získal velkou popularitu převážně mezi umělci, při úpravě fotografií a v tisku. [6]

Existují dva způsoby jak HSV (HSB) model graficky znázornit. První z nich je zobrazen na obr. 7. Jedná se o HSV (HSB) kruh, který se převážně využívá v grafických aplikacích pro výběr barvy. Výsledná barva vznikne poté, co si uživatel vybere odstín z kruhové oblasti a následně si zvolí nasycení a hodnotu (jas) z oblasti trojúhelníka. [4]



Obr. 7: HSV (HSB) kruh

Dalším způsobem vizualizace HSV (HSB) barevného modelu je pomocí kužele (obr. 8). Je tak možné znázornit celý barevný prostor modelu v jediném objektu. Protože se jedná o trojrozměrné prostředí, není tento způsob zobrazení vhodný pro výběr barev v softwarových aplikacích. Byl zde zařazen hlavně kvůli názornosti a pochopení dané problematiky. Pro úplnost je v následujícím odstavci stručně popsán.

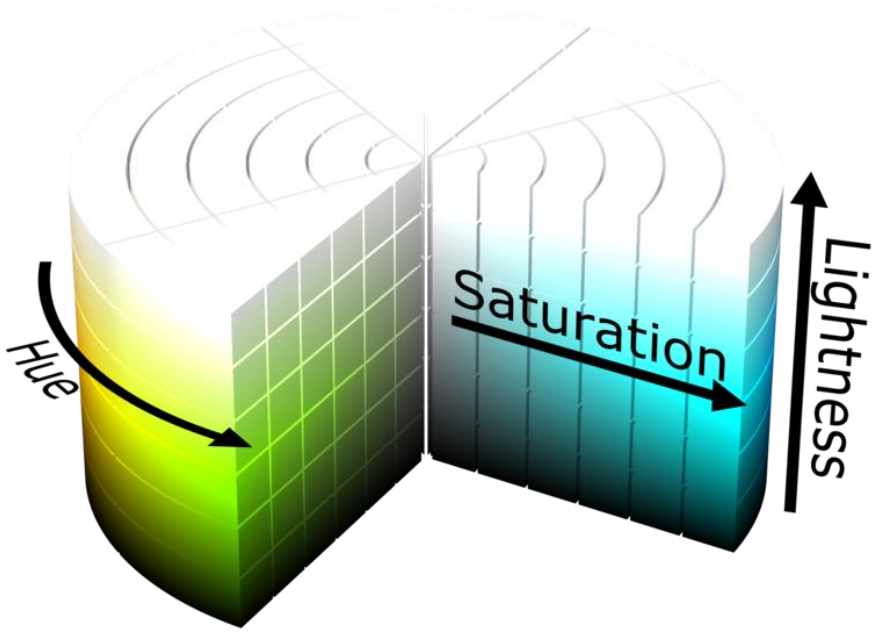


Obr. 8: HSV (HSB) kužel [9]

Na obvodu podstavy kuželu se nachází čisté, syté barvy. Čím blíže jsme středu podstavy, tím jsou barvy méně syté. Směrem k vrcholu kuželu dochází k poklesu jasu. Při průřezu tělesa nalezneme trojúhelník, jehož vrcholy jsou střed podstavy, vrchol a bod na obvodu jehlanu. V tomto geometrickém útvaru se nachází barvy jednoho barevného odstínu. [9]

2.4.4 HSL

Barevný model HSL je předchozímu modelu velmi podobný. Zásadní rozdíl je v tom, že místo jasu se zde využívá tzv. světelnost (Lightness). Tato veličina může nabývat hodnot od 0 do 100 %. Jak je možné z jejího názvu usoudit, vyjadřuje, jak bude obraz světelný. Přesněji řečeno, pokud je hodnota světelnosti menší než 50 %, je do výsledného obrazu přidávána černá barva. Naopak, pokud hodnota zmíněné veličiny přesáhne hranici 50 %, je obraz postupně zesvětlován. [10] Grafická reprezentace modelu je zobrazena na obr. 9.



Obr. 9: Grafická reprezentace HSL modelu [10]

2.4.5 Převod z RGB do HSV a naopak

Hodnoty jednotlivých barevných modelů lze mezi sebou samozřejmě převádět. Příklad převodu z RGB do HSV modelu a naopak byl vybrán z toho důvodu, že je použit i v praktické části této diplomové práce, která se věnuje vytvoření vektorového grafického editoru.

Následující rovnice ukazují, jak převést barevný model RGB na HSV. Písmena *R*, *G* a *B* značí základní barevné složky modelu RGB. Složky modelu HSV jsou příznačně pojmenovány písmeny *H*, *S*, a *V*. [11]

$$I_{max} = \max(R, G, B)\,, I_{min} = \min\,(R, G, B) \tag{3}$$

$$\mathrm{H} = \begin{cases} \text{není definováno}, & \rightarrow I_{max} = I_{min} \\ 60° \times \dfrac{\mathrm{G} - \mathrm{B}}{I_{max} - I_{min}} + 0°, & \rightarrow I_{max} = R \text{ a } G \geq B \\ 60° \times \dfrac{\mathrm{G} - \mathrm{B}}{I_{max} - I_{min}} + 360°, & \rightarrow I_{max} = R \text{ a } G < B \\ 60° \times \dfrac{\mathrm{B} - \mathrm{R}}{I_{max} - I_{min}} + 120°, & \rightarrow I_{max} = G \\ 60° \times \dfrac{\mathrm{R} - \mathrm{G}}{I_{max} - I_{min}} + 240°, & \rightarrow I_{max} = B \end{cases} \tag{4}$$

$$S = \begin{cases} 0, & \rightarrow I_{max} = 0 \\ \dfrac{I_{max} - I_{min}}{I_{max}} = 1 - \dfrac{I_{min}}{I_{max}}, & \rightarrow v\ ostatních\ případech \end{cases} \tag{5}$$

$$V = I_{max} \tag{6}$$

Rovnice níže popisují inverzní operaci, tedy převod z HSV do RGB. V případě, že proměnná *S* je rovna nule, pak se do proměnných *R*, *G*, *B* dosadí hodnota *V*. Jestliže je *S* větší než nula, potom je postup algoritmu převodu následující. [11]

$$\mathrm{H} = 6 \cdot \mathrm{H} \tag{7}$$

$$\mathrm{I} = [H] = \mathrm{floor}(\mathrm{H}) \tag{8}$$

$$\mathrm{F} = H - I \tag{9}$$

$$M = V \cdot (1 - S) \tag{10}$$

$$N = V \cdot (1 - S \cdot F) \tag{11}$$

$$K = \mathrm{V} \cdot (1 - \mathrm{S} \cdot (1 - F)) \tag{12}$$

$$(R, G, B) = \begin{cases} (V, K, M), & \rightarrow I = 0 \\ (N, V, M), & \rightarrow I = 1 \\ (M, V, K), & \rightarrow I = 2 \\ (M, N, V), & \rightarrow I = 3 \\ (K, M, V), & \rightarrow I = 4 \\ (V, M, N), & \rightarrow I = 5 \end{cases} \tag{13}$$

V prvním kroku je hodnota *H* vynásobena šesti. Dále se vypočítá index sektoru odstínu *I*. Ten je vypočten jako dolní celá část hodnoty *H*. Následně je vypočítán barevný zlomek *F*. Poté jsou pomocí získaných hodnot vypočítány pomocné proměnné *M*, *N*, *K*. Ty se v posledním kroku algoritmu využívají k získání požadovaných hodnot *R*, *G*, *B* dle rovnice 13. [11]

3 RASTROVÁ GRAFIKA

Rastrová, jinými slovy bitmapová, grafika je jedním ze dvou způsobů, jak lze ukládat dvourozměrné obrazy v elektronické podobě. Druhým způsobem je využití grafiky vektorové, kterému se podrobněji věnuje následující kapitola.

Bitmapový obraz je tvořen pravidelnou mřížkou pixelů. Příklad tohoto zobrazení je vidět na obr. 10. Každý pixel v mřížce má přiřazenou určitou barvu a odpovídá jednomu či více bitům v paměti počítače. Teprve při pohledu na celý obraz může uživatel pozorovat, jak jednotlivé body splývají a tvoří barevné plochy, přechody a tak dále. [1]

Celkový rastrový obraz určují jeho parametry, mezi ty nejdůležitější patří velikost obrázku (jeho šířka a výška), rozlišení a barevná hloubka.

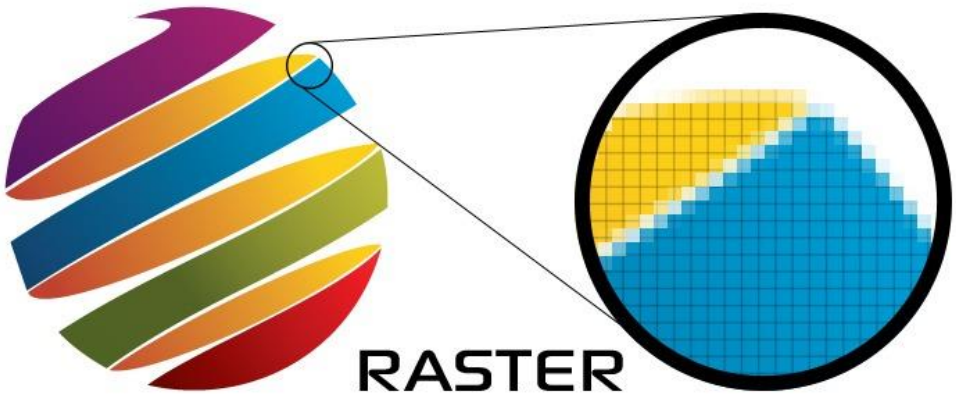
Mezi nevýhody rastrové grafiky patří především to, že je poměrně náročná na paměť počítače. Řešením tohoto problému je využití různých kompresních algoritmů, díky nimž je možné paměťovou náročnost poměrně výrazně snížit. Tyto formáty fungují zjednodušeně řečeno tak, že se stejné nebo velmi podobné pixely spojí v jeden celek. Mezi nejpoužívanější kompresní formáty bitmapové grafiky patří JPG, GIF a PNG. [7] Podrobněji jsou popsány v následující podkapitole.

Další specifickou nevýhodou bitmapové grafiky je to, že není možné měnit velikost obrázku bez toho, aniž by došlo ke ztrátě kvality. Při výrazném zvětšení může uživatel zpozorovat zmíněnou mřížku s pixely.

Výraznou výhodou rastrové grafiky je snadné pořízení obrazu například pomocí fotoaparátu nebo skeneru.

S výsledným rastrovým obrazem lze pomocí grafických programů provádět různé efekty, mezi které patří změna jasu, kontrastu apod. Dále je možná vytvářet fotomontáže, koláže a další. Tyto vlastnosti se využívají například při editaci fotek, tvorbě reklam, nebo vývoji počítačových her. [7]

Mezi nejpoužívanější software pro tvorbu a editaci rastrové grafiky patří Adobe Photoshop, Gimp, či MS Paint, které je součástí operačního systému Windows již od roku 1985. [1]



Obr. 10: Příklad rastrové grafiky [12]

3.1 Komprese rastrového obrazu

Jak již bylo zmíněno výše, největší nevýhoda rastrové grafiky spočívá ve velké paměťové náročnosti pro uložení výsledného obrázku. Způsob řešení tohoto problému je pomocí komprimačních algoritmů.

Tyto algoritmy se dělí na bezztrátové a ztrátové. Bezztrátové algoritmy jsou ty, při kterých dochází ke zmenšení objemu dat, a přitom nedochází ke ztrátě původní informace. U ztrátových algoritmů naopak dochází ke ztrátě informace, a díky tomu je jejich komprimační poměr zpravidla lepší než u algoritmů bezdrátových. [13]

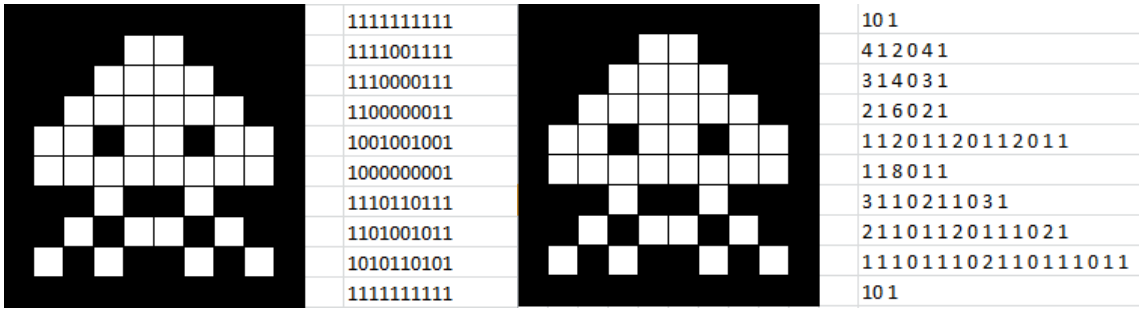
Kompresní poměr je, stručně řečeno, podíl mezi velikostí nekomprimovaných dat a velikostí dat komprimovaných. Tato hodnota se může využívat při srovnání různých komprimačních algoritmů. [13]

Níže jsou stručně popsány komprimační algoritmy, které jsou využívány rastrovými formáty, o nichž se píše dále v této kapitole.

3.1.1 RLE

Bezztrátový algoritmus RLE (Run Length Encoding), též známý jako RLC (Run Length Compression), je nejjednodušší ze zde popsaných. Pracuje na principu komprese opakujících se symbolů, v případě rastrové grafiky se pak jedná o hodnoty pixelů. [14]

Názorný příklad tohoto algoritmu je zobrazen na obr. 11. Vlevo se nachází nekomprimovaný obrázek s jeho reprezentací hodnot pixelů pomocí jedniček a nul. Vpravo je pak zobrazen ten samý obrázek za použití algoritmu RLE.



Obr. 11: Příklad algoritmu RLE – Vlevo nekomprimovaný obraz, Vpravo komprimovaný [14]

Z obrázku je zřejmé, že metoda RLE pracuje s každým řádkem samostatně. Nejdříve se definuje takzvané počítadlo, které nese údaj o počtu opakovaných znaků. Následuje samotný znak, který má být komprimován. Tímto způsobem zkomprimuje algoritmus RLE všechny řádky daného objektu. [13]

Existuje varianta RLE, kde je použita takzvaná zarážka, která slouží k identifikaci komprimované sekvence. Jako zarážku je nutné použít znak, který se nevyskytuje ve vstupním proudu dat. [13]

Algoritmus RLE je vhodné využívat na obrázky, ve kterých je více stejných pixelů, v opačném případě může dojít k záporné kompresi, tedy ke zvětšení výsledného souboru. Tento případ je zobrazen na devátém řádku obr. 11. [1]

3.1.2 LZ77 a LZW

Tato metoda byla navržena Abrahamem Lempelem v roce 1977. V následujícím roce ji vylepšil pan Jacob Ziv. Z tohoto důvodu nese tento algoritmus název LZ77, případně LZ78. Za dalším vývojem popisovaného algoritmu stojí Terry Welsch, který jej upravil pro potřeby hardwarových řadičů. Jeho verze se nazývá LZW. Oba algoritmy se řadí mezi takzvané slovníkové metody. [13]

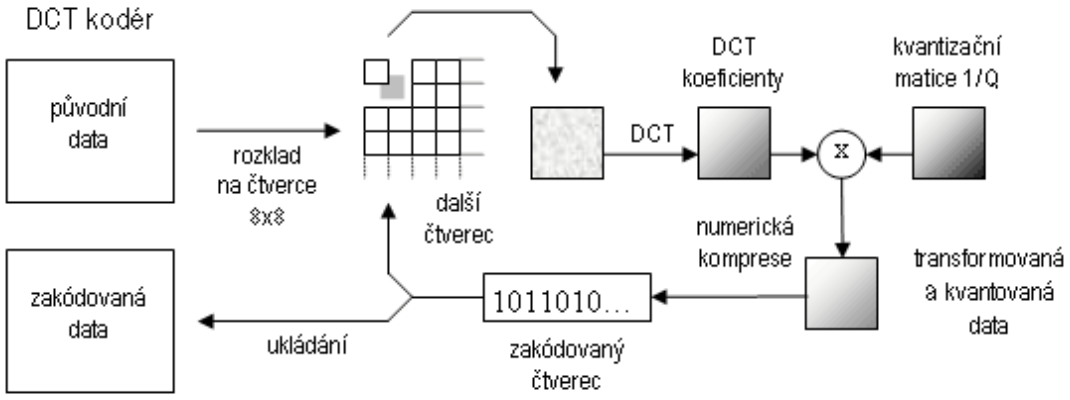
Princip algoritmu LZ77 je následující. Data jsou procházena od začátku do konce a v případě, že je nalezena skupina znaků, která se již v předchozích datech objevila, je tato skupina nahrazena odkazem na předchozí výskyt. Odkaz obsahuje vzdálenost od současné pozice spolu s délkou skupiny znaků. Jestliže je délka odkazu kratší než reprezentace skupiny znaků, logicky dochází k úspoře místa. [15]

Hlavní výhodou verze LZW je to, že kódové slovo, u LZ77 to byl odkaz, je kratší. Jedná se pouze o jedno číslo, kterým je index slova ve slovníku. Při každé iteraci tohoto algoritmu se hledá nejdelší fráze ve slovníku, která odpovídá doposud nezpracované části vstupu. Výstupem je poté kódové slovo a fráze ve slovníku je o tento znak rozšířena a označena nejmenším možným číslem. [13] [15]

3.1.3 Diskrétní kosinová transformace

Diskrétní kosinová transformace je odvozena od diskrétní Fourierovy transformace s tím rozdílem, že vrací pouze reálnou složku. Vstupní obrazová data se považují za vzorky spojitých funkcí naměřených v diskrétní pixelové mřížce. Výstupem kosinové transformace je sada parametrů kosinových funkcí, díky nimž lze zrekonstruovat původní obraz. [1]

Na obr. 12 je zobrazen princip diskrétní kosinové transformace při kompresi obrazu. Vstupní obraz je nejdříve rozdělen na 8x8 pixelové segmenty. Následně proběhne samotná DCT a kvantování, při kterém se koeficienty DCT dělí a následně zaokrouhlují. Koeficienty se dále načtou do zásobníku dle své důležitosti. Poté probíhá jejich kódování a nakonec jsou data uložena. [16]



Obr. 12: Princip DCT komprese obrazu [16]

3.2 Formáty rastrové grafiky

Rastrové formáty se zpravidla liší způsobem reprezentace obrazových dat a také metodou použité komprese. V dnešní době existuje přes 50 různých rastrových formátů [1]. Níže uvedené a stručně popsané formáty patří mezi ty nejvyužívanější a nejznámější v oboru počítačové grafiky.

3.2.1 GIF

Za vznikem formátu GIF (Graphic Interchange Format) stojí společnost CIS (CompuServe Information Service), která jej představila v roce 1987. Původně byl tento formát určen pro přenos obrázků po telefonních linkách. Své uplatnění ovšem našel především na internetu. Je zde využito kompresní metody LZW. [1]

GIF je schopen uložit více obrázků v jednom souboru, kde každý z obrázků může mít vlastní barevnou paletu. Další vlastností tohoto formátu je takzvaný *interlacing* neboli prokládání řádků. Díky této funkcionalitě může uživatel již po získání ¼ objemu obrazových dat rozpoznat vzhled obrázku. Tento bitmapový formát také umožňuje jednoduché animace. [17]

Nevýhodou formátu GIF je to, že dokáže pracovat pouze s 8 bitovou barevnou hloubkou, tj. s 256 barvami. Také z tohoto důvodu je dnes tento formát nahrazován modernějším formátem, jehož zkratka je PNG. [18]

3.2.2 BMP

Druhým nejstarším zde zmíněným rastrovým formátem je BMP (Bit Mapped Picture). Poprvé se s ním uživatel mohl setkat v roce 1988, kdy byl představen jako součást operačního systému OS/2 verze 1.10 SE. [17]

Tento formát byl dříve označován zkratkou DIB (Device Independent Bitmap), není totiž závislý na zařízení, na kterém se dá zobrazit. Společně s tímto faktem bylo jeho další výhodou to, že nepodléhal patentové ochraně. Díky těmto skutečnostem byl ve své době velmi populární. [1]

Většina variant formátu BMP nevyužívá žádnou kompresi. Existuje varianta, která využívá kompresní algoritmus RLE, ovšem i při tomto kódování je ve většině případů velikost souboru příliš velká ve srovnání s ostatními rastrovými formáty. Z tohoto důvodu je tento rastrový formát v dnešní době spíše na ústupu. [18]

3.2.3 JPEG

Další rastrový formát nese název podle výzkumné skupiny vědců, která jej v roce 1991 vytvořila. Skupina JPEG (Joint Photographic Experts Group) pracovala v rámci mezinárodní standardizační organizace ISO (International Organization for Standardization). [1]

Existuje varianta tohoto formátu, která využívá bezztrátovou kompresi, ovšem v praxi se používají hlavně varianty formátu JPEG, které jsou založeny na kompresi ztrátové, konkrétně na diskrétní kosinové transformaci. V těchto variantách dochází

ke snížení kvality typicky na 75 %, což je pro mnoho lidí nerozpoznatelné, a přitom se může kompresní poměr pohybovat v rozmezí 20 : 1 až 25 : 1. Některé programy umožnují uživateli míru komprese při ukládání souboru zvolit. [1]

Rastrový formát JPEG podporuje maximální velikost obrázku 65 535 x 65 535 pixelů. Jeho přípona může být *.jpg* nebo *.jpeg*. Je vhodný především pro obrázky, které mají plynulé přechody mezi barvami. Na obrázcích, ve kterých se nacházejí ostré hrany nebo například text, zanechává takzvané pixelové artefakty. Proto je pro tento typ obrázků lepší volbou využití jiného rastrového formátu, kterým může být PNG nebo GIF. Jako příklad srovnání uvedených bitmapových formátů je přiložen obr. 13. [17] [19]

Formát JPG by neměl být využíván k několikanásobným úpravám obrázku. Důvodem je, že při každém uložení dochází k další kompresi a tím i k dalšímu zhoršení kvality. [1]

Obr. 13: Rozdíl mezi PNG (vlevo) a JPEG (vpravo) [19]

3.2.4 PNG

Vznik tohoto bitmapového formátu byl motivován především tím, že se společnost CIS rozhodla patentovat kompresní algoritmus LZW. Kvůli tomu došlo ke zpoplatnění formátu GIF, čímž se spustila velká vlna nevole tehdejších uživatelů tohoto formátu. V roce 1997 proto přišlo mezinárodní konsorcium W3C (World Wide Web Consortium) starající se o vývoj webových standardů s novým grafickým formátem, který nesl název PNG (Portable Network Graphics). [1]

Stejně jako JPEG je i tento rastrový formát podporován mezinárodní standardizační organizací ISO.

Zásadní změna oproti formátu GIF spočívá ve vylepšení prokládání řádků, kdy je nyní uživatel schopen rozpoznat obraz již po přenesení jedné osminy celkového množství dat. [17]

Formát PNG je založen na bezztrátovém komprimačním algoritmu LZ77. Výsledný kompresní poměr proto není tak značný jako u formátu JPEG, ale například oproti metodě RLE dosahuje výrazně lepších výsledků. [18]

Původně byl formát PNG vyvinut pouze pro používání na internetu, z toho důvodu podporuje pouze barevný model RGB. Společně s modelem RGB je ovšem formát PNG schopen pracovat s rozšířeným modelem RGBA, tedy modelem, který dokáže uchovat informace o průhlednosti. [1]

Je velmi vhodné využívat tento formát především pro obrázky, ve kterých se vyskytují ostré hrany, výrazné barevné přechody nebo velké množství textu. Právě v těchto případech je formát PNG lepší volbou než zmíněný JPEG. [19]

3.2.5 TIFF

Posledním zde zmíněným rastrovým formátem je TIFF (Tag Image File Format). Tento formát vyvinula společnost Aldus Corporation v polovině osmdesátých let minulého století. Tuto firmu později odkoupila společnost Adobe, která od té doby vlastní autorská práva na formát TIFF. [1]

Popisovaný bitmapový formát využívá bezztrátovou kompresi, díky čemuž zachovává barevnou hloubku původního obrázku. Je proto využíván především pro vysoce kvalitní profesionální fotografie, nebo například při skenování ve vysokém rozlišení. [18]

Důsledkem použití bezztrátové komprese je poměrně velká velikost výsledného souboru. Z tohoto důvodu není vhodné formát TIFF používat na webu, protože by mohlo dojít k zpomalení rychlosti načítání internetových stránek.

4 VEKTOROVÁ GRAFIKA

Druhým způsobem uložení dvourozměrného obrazu v elektronické podobě je uložení pomocí vektorové grafiky. V tomto případě je obraz reprezentován pomocí geometrických objektů.

Z biologického hlediska vnímá lidské oko vektorový obraz jako rastrový, protože sítnice představuje bitmapový rastr. Lidský mozek ovšem zpracuje obraz jako vektorovou grafiku. [20]

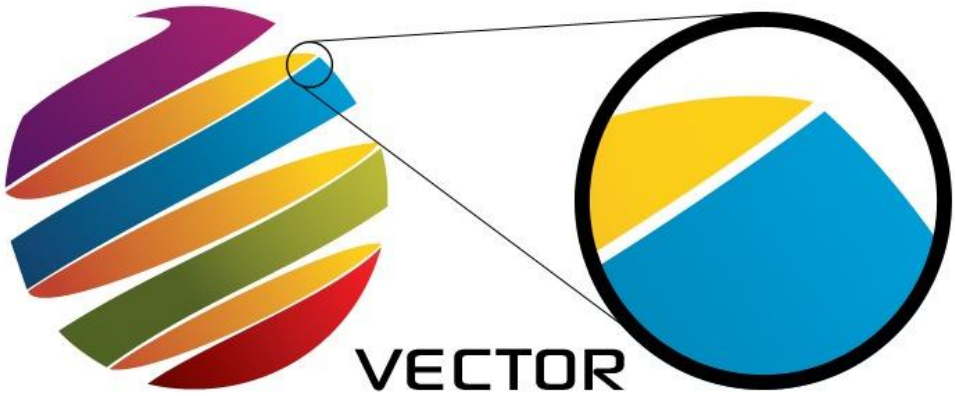
Teoretickým základem vektorové grafiky je analytická geometrie. Obraz vektorové grafiky je totiž složen z jednotlivých geometrických útvarů, kterými jsou přímky, křivky či mnohoúhelníky. [6]

Mezi výhody tohoto způsobu zobrazení grafické informace oproti grafice rastrové patří zejména to, že obraz lze libovolně zvětšovat či zmenšovat, a to bez ztráty kvality. Další výhodou je možnost pracovat s každým objektem v obrázku samostatně. [1]

Paměťová náročnost u jednoduchých obrázků je výrazně menší než u rastrové grafiky. Ovšem při překročení určité meze začne být vektorová grafika náročnější na operační paměť a procesor než grafika rastrová. [17]

Nevýhodou vektorové grafiky je náročnost jejího pořízení. Zatímco u bitmapové grafiky stačí použít fotoaparát či skener, vektorová grafika vzniká pomocí specializovaných grafických programů. Mezi software pro tvorbu a editaci vektorové grafiky patří například Inkscape, Adobe Illustrator, Zoner Callisto, různé CAD systémy a jiné. [17]

Každý z výše zmíněných programů má svůj vlastní nativní formát pro ukládání vektorové grafiky. To někdy může způsobovat problémy v kompatibilitě, ovšem v dnešní době existuje mnoho způsobů, jak tyto formáty mezi sebou převádět.



Obr. 14: Příklad vektorové grafiky [12]

4.1 Formáty vektorové grafiky

Stejně jako formátů rastrové grafiky, tak i formátů grafiky vektorové existuje spousta druhů. V této práci budou ovšem uvedeny pouze dva z nich, které jsou považovány za nejznámější a nejvyužívanější.

4.1.1 SVG

Vektorový formát SVG (Scalable Vector Graphics) byl vytvořen už na konci devadesátých let. Za jeho vznikem stojí již dříve zmíněné mezinárodní konsorcium W3C, které jej učinilo otevřeným standardem, jenž měl být využíván především na webových stránkách. [1]

Tento formát je založen na značkovacím jazyce XML (Extensible Markup Language) a umožňuje zobrazovat dvourozměrnou grafiku, interaktivní prvky a animace. Ukázka XML zápisu SVG formátu je zobrazena na obr. 15. Výhody využití jazyka XML spočívají především v tom, že je v souborech možné vyhledávat, indexovat je, komprimovat je, nebo rozšiřovat dalšími skripty. Výsledný XML dokument je textový soubor, SVG grafiku lze tedy vytvářet a editovat pouze za pomoci jednoduchého textového editoru. [17]

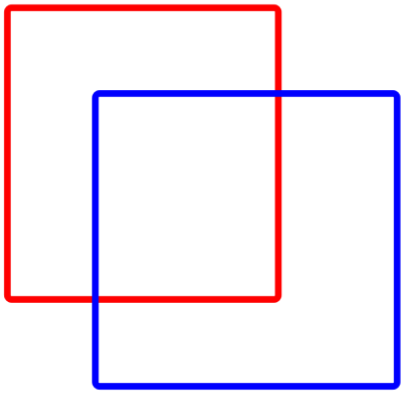
```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<svg
   width="210mm"
   height="297mm"
   viewBox="0 0 210 297"
   version="1.1"
   id="svg5">

    <rect
       style="fill:#0000ff;fill-opacity:0;stroke:#ff0000;stroke-width:2"
       id="rect244"
       width="50.849609"
       height="43.408203"
       x="34.313152"
       y="54.983723"
       ry="1.2509023" />
    <rect
       style="fill:#0000ff;fill-opacity:0;stroke:#0000ff;stroke-width:2"
       id="rect356"
       width="52.916664"
       height="46.302082"
       x="54.983723"
       y="70.693359"
       ry="1.2509023" />
</svg>
```

Obr. 15: Příklad XML zápisu SVG formátu

V roce 1999 se vektorový formát SVG dočkal podpory všech hlavních webových prohlížečů. Lze na něj tedy uplatnit CSS (Cascading Style Sheets) pravidla, která definují vzhled internetových stránek. S tím je spojená i možnost využití skriptovacích jazyků, jako je JavaScript, díky nimž se může SVG obraz stát interaktivním. [1]

Kromě využití formátu SVG na internetových stránkách s ním lze samozřejmě pracovat i ve většině grafických editorů. V praktické části této diplomové práce, která se zabývá právě vytvořením vektorového grafického editoru, je tento formát využit pro ukládání a následné načítání výsledného obrazu.

Pro úplnost je na obr. 16 uveden i samotný obrázek, který se skrývá za XML zápisem z minulé stránky.



Obr. 16: Výsledný SVG obraz

4.1.2 DWG

Druhým zde zmíněným vektorovým formátem je DWG. Jeho název je odvozen od anglického slovíčka *drawing*, tedy malovat. Jedná se o nativní formát softwaru AutoCAD.

V dnešní době jej lze považovat za nejrozšířenější formát pro vytváření a úpravu odborných výkresů. Formát DWG je využíván v různých průmyslových odvětvích, kde s ním pracují inženýři navrhující rozličné strojní součástky, nebo například architekti vytvářející plány budov. V tomto formátu je možné ukládat jak 2D tak 3D vektorová data.

Některé programy, které jsou konkurencí pro software AutoCAD, umožňují čtení i zápis ve formátu DWG. Ovšem zpětná kompatibilita takto vytvořených souborů je neúplná a není zcela zaručena. Důvodem je fakt, že vektorový formát DWG je proprietárním vlastnictvím společnosti Autodesk, která stojí za jeho vývojem. Jedná se tedy o neveřejný formát a pro jeho použití je potřeba vlastnit licenci. [17]

4.2 Metaformáty

Existuje speciální kategorie grafických formátů, kterými jsou metaformáty. Ty jsou schopny sloučit vektorový popis grafické informace společně s popisem rastrovým. Mezi tyto formáty se řadí například PostScript, PDF, AI, CDR nebo WMF. Zmíněné metaformáty jsou detailněji rozebrány v dalších odstavcích.

4.2.1 PostScript – PS a EPS

Grafické formáty PS (PostScript) a EPS (Encapsuled PostScript) jsou založeny, jak je z jejich názvu zřejmé, na jazyku PostScript. Tento jazyk je stejně jako XML značkovací. Za jeho vznikem stojí firma Adobe Systems Incorporated, která jej vyvinula v roce 1985. PostScript je určen ke grafickému popisu tisknutelných dokumentů. Je výjimečný tím, že je podporován dražšími tiskárnami, kde je považován za standard. Jeho značná výhoda spočívá v tom, že je nezávislý na zařízení, na kterém se má dokument tisknout. Po určité době se z PostScriptu stal také formát pro ukládání obrázků. [1]

EPS je jasně definovaná podmnožina jazyka PostScript. Jedná se o metaformát a je tedy schopen ukládat jak vektorové, tak rastrové obrázky. Jedním z rozdílů oproti formátu PS je to, že EPS má vždy pouze jednu stránku, zatímco formát PS jich může obsahovat více. [17]

V dnešní době je PostScript postupně vytlačován formátem PDF, který z něj vychází.

4.2.2 PDF

Metaformát PDF (Portable Document Format) je založen, jak již bylo zmíněno, na jazyce PostScript. Nejedná se však o jeho kopii, některé funkcionality PostScriptu jsou v PDF implementovány odlišně, jiné naopak nebyly využity vůbec. Nejvýraznější změna spočívá ve výsledné velikosti souboru, kdy PDF zabírá podstatně méně místa. Tento fakt je logicky zapříčiněn tím, že formát PDF využívá slovníkového komprimačního algoritmu LZW. [1]

Za vznikem formátu PDF stojí společnost Adobe, jež jej představila v roce 1993. Později, roku 2008 byl tento formát uveden jako standard pro elektronickou výměnu dokumentů podle normy ISO 32000. Tato norma specifikuje i různé speciální standardy, kterými jsou například PDF/A pro účely archivování, PDF/E pro technické obory, PDF/X pro tisk nebo PDF/UA pro usnadnění přístupu. [21]

Cílem firmy Adobe při tvorbě PDF bylo především vyvinout formát, který bude nezávislý na softwaru i hardwaru. Tedy aby se libovolný dokument zobrazoval na všech zařízeních stejným způsobem. Tento cíl byl naplněn a PDF je v dnešní době nejvyužívanějším formátem pro uchovávání dokumentů.

PDF dokumenty je možné vytvářet v softwaru Adobe Acrobat, který je ovšem chráněn licencí, kterou je nutné si zakoupit. Export do formátu PDF umožnuje celá řada jak komerčních, tak volně dostupných programů. Prohlížení PDF dokumentů je možné

za použití například volně dostupného programu Adobe Reader. Funkce pro prohlížení PDF je v současné době implementována rovněž ve většině webových prohlížečů. [21]

4.2.3 AI

Formát AI (Adobe Illustrator) se taktéž řadí mezi metaformáty. Jedná se o nativní formát grafického editoru Adobe Illustrator, který je detailněji popsán v následující kapitole.

AI je do jisté míry podobný formátu PDF. Jedním z rozdílů je to, že PDF podporuje více stran dokumentu, AI ovšem podporuje pouze jednu. Což vzhledem k určení programu Adobe Illustrator nepředstavuje velký problém. [22]

Metaformát AI využívá jak ztrátovou, tak bezztrátovou kompresi. Umožnuje využití různých barevných hloubek a různých barevných modelů. Mezi jeho další vlastnosti patří podpora vrstev či průhlednosti. [22]

4.2.4 CDR

Nativním formátem softwaru CorelDRAW, jenž je stejně jako Adobe Illustrator popsán v další kapitole, je metaformát CDR.

Oproti formátu AI podporuje CDR více než jednu stranu dokumentu. Zpočátku byl tento formát zcela proprietární. V dnešní době je však možné tento formát otevřít i v jiných grafických editorech než je CorelDRAW. Je ovšem nutné zkontrolovat správnost grafiky, protože se některé nepodporované grafické prvky mohou ztratit. [17]

4.2.5 WMF

Posledním zde zmíněným metaformátem je WMF (Windows Metafile Format). Tento formát byl představen společností Microsoft v roce 1988 při příležitosti vydání operačního systému Windows 2.0. Přestože byl tento formát vyvíjen primárně pro platformu Windows, rozšířil se i na další operační systémy, včetně Linuxu. [1]

Formát WMF je vhodné využívat především pro černobílou grafiku, při práci s grafikou barevnou nevykazuje uspokojivé výsledky. Nevýhodou tohoto formátu je zejména to, že není schopen akceptovat rastrové ani vektorové výplně. [1]

4.3 Prvky vektorové grafiky

Již v úvodu této kapitoly bylo zmíněno, že se vektorová grafika skládá z různých objektů, se kterými lze samostatně manipulovat, upravovat je nebo je mazat. Pro úplnost zde budou jednotlivé prvky vektorové grafiky stručně popsány.

Bod

Základním stavebním kamenem objektů vektorové grafiky je bod. Ten je určen souřadnicemi, které definují jeho pozici. Samostatně však nemá nárok na existenci a není možné jej tedy nakreslit.

Křivka

Nejnižším nakreslitelným objektem vektorové grafiky je až křivka. Tou se rozumí jakákoliv čára, tedy rovná, lomená, či zakřivená. Zakřivenou je myšlena křivka Bézierova, jež je velmi důležitým prvkem vektorové grafiky a již byla popsána v druhé kapitole této práce.

Geometrický tvar

Uzavřený útvar, jenž je ohraničený křivkou, je nazýván geometrickým útvarem. V matematice je jím například čtverec, kruh, trojúhelník a tak dále. Počítačová geometrie tento útvar definuje poněkud obecněji. Kromě matematických útvarů se jedná o jakýkoliv uzavřený útvar, který je ohraničený křivkou. Tento útvar tak nemusí být pravidelný a dokonce je možné, aby se křivky, které jej ohraničují, překrývaly.

Skupina

Jako skupina je ve vektorové grafice definováno seskupení několika objektů, které nadále vystupují jako samostatný objekt. To v mnoha případech umožňuje uživateli, aby si ulehčil práci.

Text

Velmi důležitým prvkem vektorové grafiky je text. Vkládání textu do dokumentu je jednou ze základních funkcí každého vektorového editoru. Většina z nich umožnuje i jeho úpravy, kterými mohou být například změna použitého fontu, změna velikosti textu a jiné.

5 VEKTOROVÉ GRAFICKÉ EDITORY

V dnešní době existuje celá řada různých vektorových grafických editorů, které se mohou lišit svými nástroji, podporou formátů, nebo například tím, jestli se jedná o open source, nebo je nutné si pro využívání editoru zakoupit licenci. V nadcházejících kapitolách budou rozebrány funkce vektorových grafických editorů a následně budou popsány ty editory, jež jsou považovány za nejrozšířenější.

5.1 Funkce vektorových grafických editorů

Vektorové grafické editory mají poměrně značné množství funkcí. Níže jsou vyjmenovány především ty nejdůležitější. Jejich výčet je taktéž účelně sestaven s ohledem na předpokládané funkce, které by měl obsahovat vektorový grafický editor, dále jen VGE_B (Vector Graphic Editor Bartoněk), jenž je obsahem praktické části této diplomové práce.

5.1.1 Základní funkce

Mezi základní funkce každého vektorového editoru patří především export výsledného obrazu do specifických vektorových formátů, případně možnost uložit obraz jako bitmapu. Profesionální vektorové editory většinou umožňují export do více vektorových i rastrových formátů.

5.1.2 Vytváření grafických objektů

Hlavním účelem vektorových grafických editorů je právě možnost vytvářet grafické objekty. Těmito objekty jsou myšleny grafické prvky popsané v kapitole 4.3.

5.1.3 Práce s grafickými objekty

Práce s grafickými objekty je vcelku široké téma. Může se jednat o mazání objektů, změnu jejich polohy či velikosti, jejich otáčení či kopírování a následné vkládání. Dalšími možnostmi je sloučení objektů do skupin nebo například různé vzájemné uspořádání. Uspořádáním je především myšleno zarovnání objektů do specifického směru. Pokud se objekty překrývají, je změnou upořádání i změna pořadí, ve kterém jsou vykresleny.

5.1.4 Práce s textem

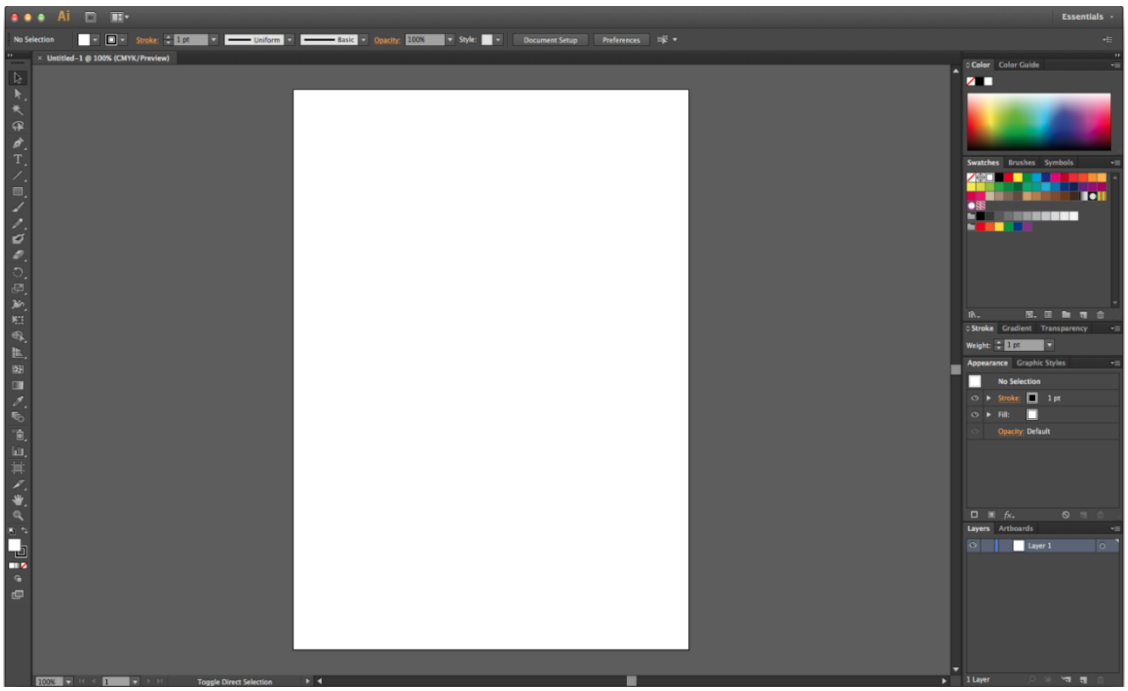
Dalšími neméně důležitými funkcemi grafických editorů jsou ty, které se věnují práci s textem. Mezi tyto funkce patří například změna velikosti textu, jeho barvy či fontu. Mezi pokročilejší funkce pro formátování textu lze zařadit jeho zarovnání, či editace velikosti mezer mezi slovy a tak dále.

5.2 Adobe Illustrator

Velmi oblíbenou volbou uživatelů pro vytváření vektorové grafiky je Adobe Illustrator. Za vznikem tohoto softwaru stojí již dříve zmíněná společnost Adobe Systems, jež jej vyvíjí již od roku 1986. Aktuální verze popisovaného vektorového editoru byla představena v dubnu roku 2021 a její název je Adobe Illustrator CC 2021- 25.2.3. [23]

Nejedná se o open source, tudíž je nutné mít pro používání tohoto produktu zakoupenou licenci. Pro úplnost je níže na obr. 17 zobrazeno GUI tohoto vektorového grafického editoru.

Z obrázku je patrné, že v horní části editoru je uživateli k dispozici malé menu, v němž může měnit velikost čáry, jíž se budou kreslit obrazce. Je možné také zvolit druh čáry či její průhlednost. V levé části vývojového prostředí se nachází lišta s nástroji, pomocí nichž lze vytvářet různé grafické objekty, případně je mazat, označovat a tak dále. Napravo je poté možné zvolit barvu čáry. Celé GUI je modulární a uživatel si jej proto může nastavit zcela podle své libosti.



Obr. 17: Adobe Illustrator GUI

Nativním formátem pro ukládání výsledného obrazu je AI, jenž již byl popsán výše. Samozřejmostí je však export i do jiných jak vektorových tak rastrových formátů. Adobe Illustartor umožnuje nastavit barevné modely RGB i CMYK, dále je schopen pracovat s výplněmi a přechody. Mimo to je taktéž účinným nástrojem zaměřeným na typografii. Další oblíbenou funkcí je možnost převádění rastrové grafiky na vektorovou. [23]

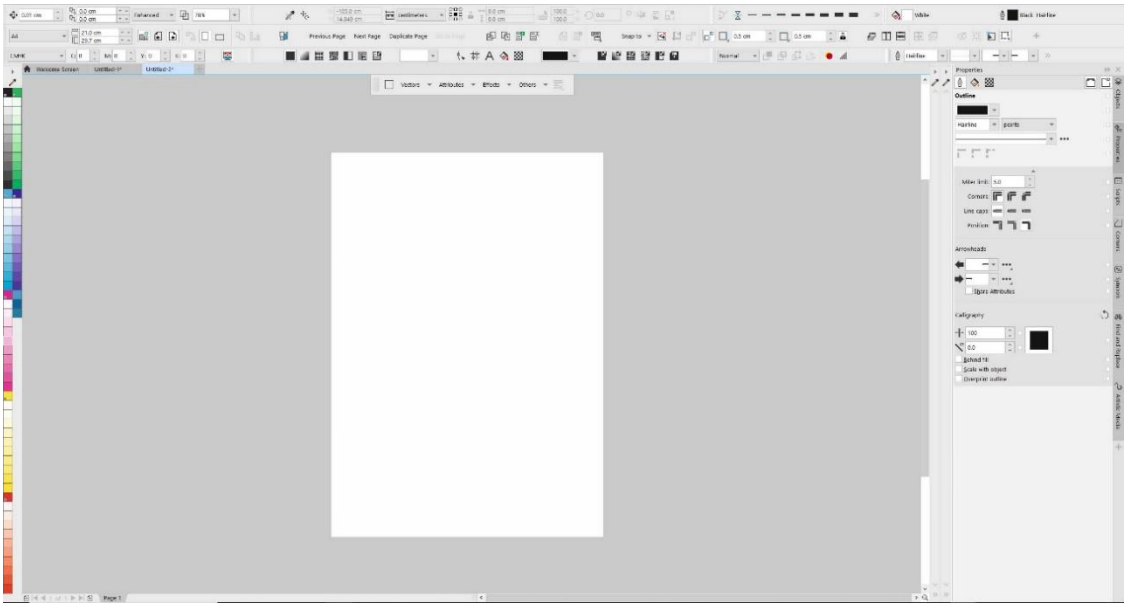
Program Adobe Illustrator je součástí grafické sady Creative Cloud. Ta dále obsahuje software Adobe Photoshop pro tvorbu rastrové grafiky, programy pro tvorbu animací, webových stránek či textových dokumentů. Všechny aplikace v této sadě mají

velmi podobný vzhled uživatelského prostředí a používají stejné klávesové zkratky, díky čemuž nemá uživatel problém přejít z jednoho programu ke druhému. Jejich ovládání se tak stává velmi intuitivním. [23]

5.3 CorelDRAW

Dalším velmi rozšířeným nástrojem pro tvorbu vektorové grafiky je software CorelDRAW. Byl vyvinut kanadskou společností Corel v roce 1989. Jeho vývoj, stejně jako u Adobe Illustratoru, přetrvává do dnešních dní. Aktuální verze tohoto programu je CorelDRAW v24, která byla představena devátého března roku 2022. Nativní formát tohoto programu je CDR, jenž je uveden v minulé kapitole. Podoba GUI aplikace CorelDRAW je pro názornost zobrazena na obr. 18. [24]

Uživatelské prostředí je do jisté míry obdobné jako u Adobe Illustratoru. Existuje více verzí, přičemž si uživatel může zvolit tu, která odpovídá jeho schopnostem. Například pro nováčky existuje nastavení *Lite*.



Obr. 18: CorelDRAW GUI

Podobně jako u výše popisovaného produktu od společnosti Adobe Systems je i tento software součástí větší grafické sady. Jedná se o sadu s názvem CorelDRAW Graphics Suite. I zde se nachází programy zabývající se vytvářením rastrové grafiky, tvorbou animací, typografií nebo tiskem. [24]

Co se týče funkcí, jsou si programy Adobe Illustrator a CorelDraw velmi podobné. Liší se zpracováním svých uživatelských rozhraní a také tím, že druhý zmíněný software má více propracovanou funkcionalitu pro správu barevných profilů a vyniká svými možnostmi v oblasti tisku.

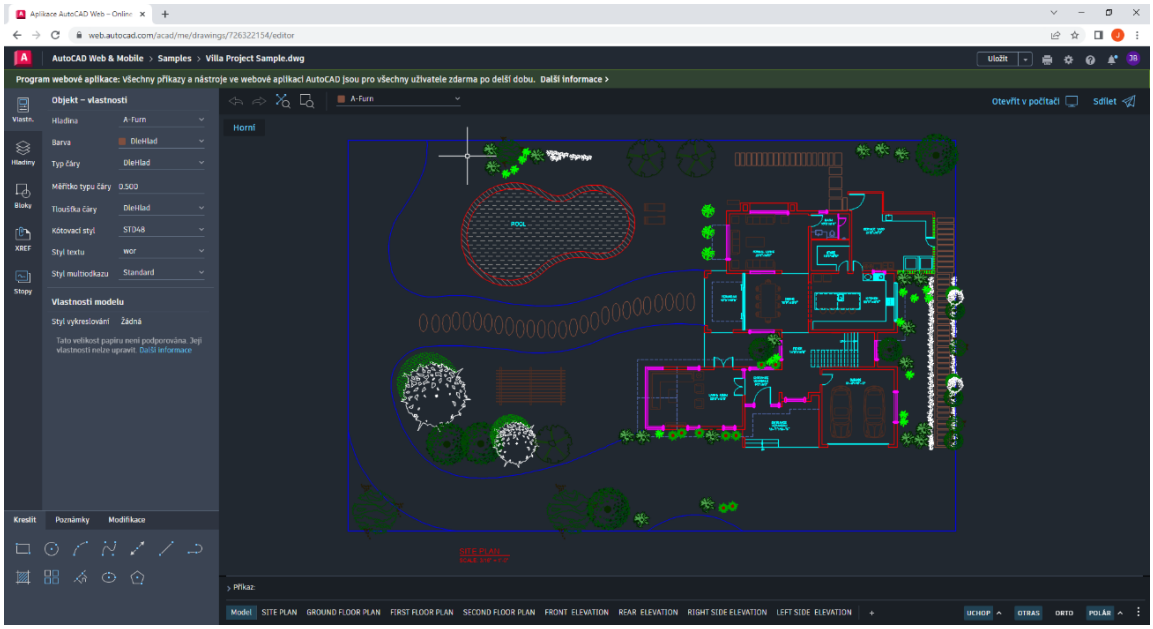
5.4 AutoCAD

Software AutoCAD je primární volbou pro většinu inženýrů, jejichž práce spočívá v projektování a konstruování různých strojních součástí. Využívají jej také architekti při tvorbě plánů budov či návrhu územních plánů.

AutoCAD byl poprvé představen již v roce 1982 firmou Autodesk. Původně byl vyvíjen jako multiplatformní aplikace, ale od roku 1994 jeho vývoj pokračoval pouze pro operační systém Microsoft Windows. Ovšem v roce 2010 vyšla opět verze AutoCADu pro Mac OS. V dnešní době je tedy tento software spustitelný na obou nejvyužívanějších operačních systémech, kterými jsou Microsoft Windows a Mac OS. [25]

Společnost Autodesk nabízí AutoCAD ve více verzích. Jedná se o desktop verzi, verzi pro mobilní zařízení, jenž nese název AutoCAD Mobile, a verzi, kterou je možné spustit ve webovém prohlížeči – AutoCAD web. Uživatelské rozhraní poslední zmíněné verze je zobrazeno na obr. 19.

V porovnání s verzí pro počítače není AutoCAD web zcela stoprocentní kopií tohoto softwaru. Většina složitějších funkcí pro kreslení a editaci zde totiž chybí. Pro jednoduché úpravy a zobrazení výsledné práce je tato verze ovšem velmi vhodná. Je proto na místě o ní vědět. [25]



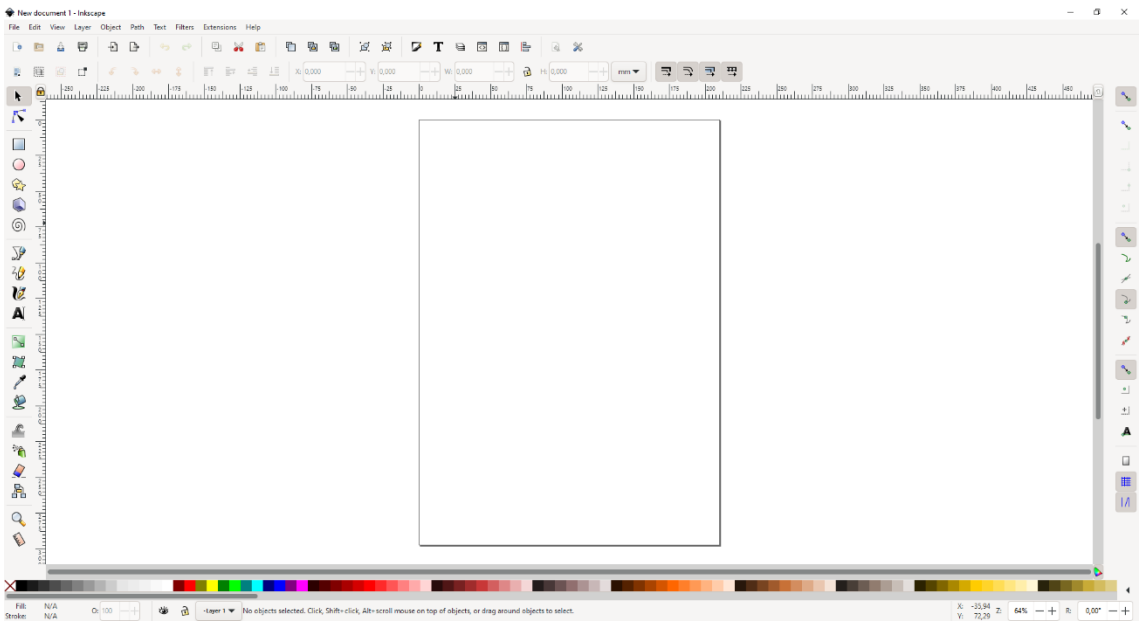
Obr. 19: AutoCAD web GUI

Stejně jako oba dříve popisované softwary je i AutoCAD součástí speciální sady aplikací. Mezi ty patří například Inventor, který se specializuje na tvorbu 3D modelů.

5.5 Inkscape

Posledním zde zmíněným vektorovým editorem je Inkscape. Tento software je jako jediný ze všech popisovaných open source. Není proto třeba si pro jeho používání kupovat licenci a lze si ho zdarma legálně stáhnout. Podporuje operační systémy Microsoft Windows, Mac OS i Linux.

Inkscape obsahuje základní funkce, kterými jsou například kreslení křivek a geometrických útvarů. Objekty je možné následně editovat, změnit jejich ohraničení, výplň, či průhlednost. Objekty lze seskupovat a různě překrývat. Inkscape je taktéž schopen převádět rastrovou grafiku na grafiku vektorovou. Pro kompletnost je přiložen obr. 20, na němž je zobrazeno uživatelské prostředí popisovaného softwaru.



Obr. 20: Inkscape GUI

6 NÁVRH VEKTOROVÉHO GRAFICKÉHO EDITORU

Praktická část této diplomové práce spočívá v návrhu a naprogramování vlastního grafického editoru. Cílem je vytvořit jednoduchý, ale přitom výkonný a spolehlivý nástroj pro tvorbu vektorové grafiky.

Podle zadání by měl být výsledný software schopný kreslit základní geometrické objekty, umožňovat vkládání textu, pracovat s vektorovým formátem SVG, a také by měl dovolovat export do vybraného rastrového formátu. Společně s těmito funkcemi bylo zapotřebí, aby bylo možné nahradit oblouky lomenými čarami. Důvodem této funkce je plánované použití VGE_B jako podpůrného sw při výzkumné práci pro rozpoznávání textů, kde je právě tato náhrada vyžadována. Objekty, uložené jako oblouky, zpomalují výpočty v algoritmu, který je vyvíjen.

Společně s těmito funkcemi byly do programu přidány i další, jejichž cílem je zjednodušení práce v softwaru a učinit ho vhodným i pro běžnou práci. Jsou jimi například možnost zvětšování či zmenšování scény, otáčení objektů či jejich přesouvání. Detailněji jsou funkce popsány na konci této kapitoly. Níže jsou rozebrány nástroje, které byly využity pro tvorbu VGE_B.

6.1 Programovací jazyk a nástroje využité při vývoji

Jako programovací jazyk pro tento úkol byl vybrán Python. Jedná se o jazyk, se kterým má autor nejvíc zkušeností, což bylo jedno z primárních kritérií pro jeho volbu. Dalším kritériem bylo to, že jde o velmi komplexní jazyk, který nabízí spoustu možností. Také je jeho uživateli k dispozici mnoho knihoven, které usnadňují práci při programování.

Python byl vytvořen v roce 1991 panem Guidem van Rossumem. Je to vysokoúrovňový programovací jazyk, který nabízí dynamickou kontrolu datových typů a podporuje různá programovací paradigmata. Mezi ty patří objektově orientované, procedurální či funkcionální programování. [26]

Důvodem, proč je tento jazyk tak populární, je především jeho univerzálnost. Je možné pomocí něj programovat internetové stránky prostřednictvím frameworků, jako jsou Django nebo Flask. Pojem *framework* je chápán jako softwarová struktura, která slouží jako podpora při programování [28].

V dnešní době je Python primární volbou při vývoji aplikací z oblasti umělé inteligence a neuronových sítí. Lze pomocí něj programovat i různé mikropočítače, příkladem může být Rapsberry Pi. V neposlední řadě je velkou výhodou tohoto programovacího jazyka početná komunita uživatelů. [27]

6.1.1 PyQt5

Hlavní knihovna, která byla při tvorbě vektorového editoru použita, nese název PyQt5. Tato robustní knihovna je verzí frameworku Qt, jenž je naprogramovaný v jazyce

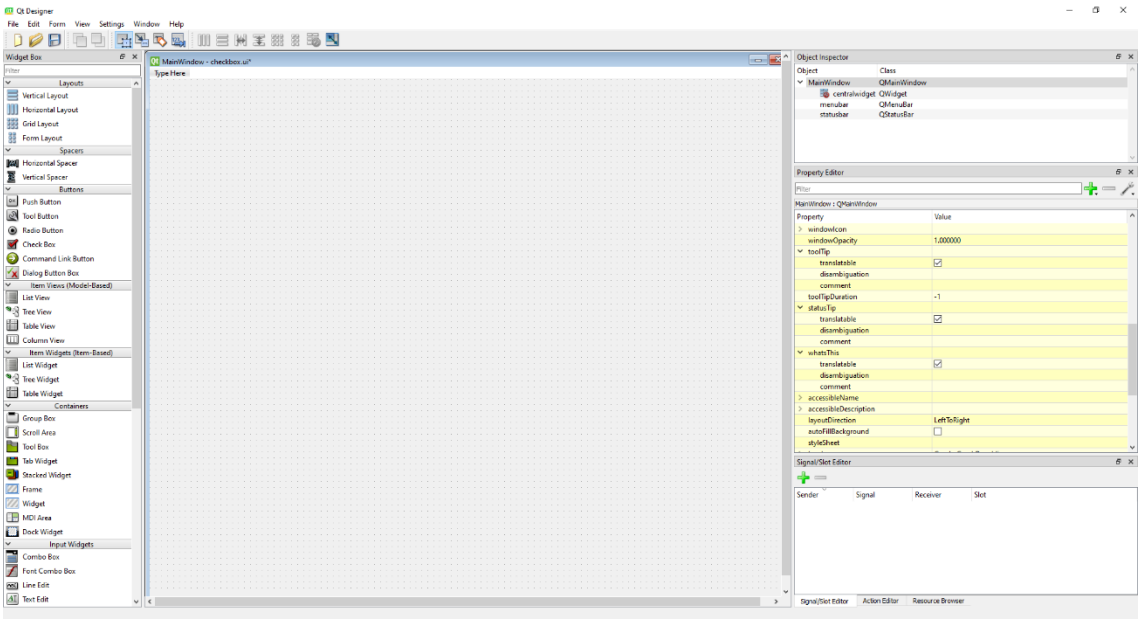
C++. Umožňuje vývoj multiplatformních aplikací a jejich grafických uživatelských rozhraní. Za jejím vznikem stojí firma Riverbank Computing Limited, a aktuální verze této knihovny, které byla představena v prosinci roku 2021, je 6.2.2 [29]. Popisovaná knihovna se skládá ze značného množství komponent. Mezi ty patří například Qt Designer, který umožnuje tvorbu GUI bez nutnosti programování. Tento nástroj byl využit pro návrh uživatelského rozhraní VGE_B a je detailněji popsán níže.

6.1.2 Qt Designer

Aplikace Qt Designer je velice užitečným nástrojem pro návrh grafického uživatelského rozhraní. Umožňuje totiž, jak již bylo řečeno výše, vytvářet GUI bez nutnosti programování.

Tato komponenta knihovny PyQt5 samozřejmě není jediným způsobem, jak tvořit GUI. Toho lze docílit i obvyklou metodou, tedy psaním kódu. Je ovšem velmi vhodné Qt Designer využívat, a to z docela prostých důvodů. Práce s ním je mnohem názornější a jednodušší. Pro větší projekty je tedy téměř nutností. Na druhou stranu, pokud programátor potřebuje vytvořit pouze malé GUI, vystačí si s psaním kódu.

Pracovní plocha Qt Designeru je zobrazena na obr. 21. Pro představu zde bude ve stručnosti uvedeno, jak tento nástroj funguje.



Obr. 21: Qt Designer GUI

V levé části si uživatel vybere objekt, který chce vložit do své aplikace, tím může být tlačítko, text, horizontální či vertikální posuvník a jiné. Následně si v pravém menu může změnit vlastnosti daného objektu. Tímto způsobem pokračuje, dokud není uživatelské rozhraní hotové. Poté výsledek uloží. Uložený soubor má koncovku *.ui*. Aby bylo GUI kompatibilní s jazykem Python, je nutné uložený soubor převést na formát *.py*. To je možné provést pomocí speciálního nástroje *pyuic5*, jenž je součástí knihovny PyQt5.

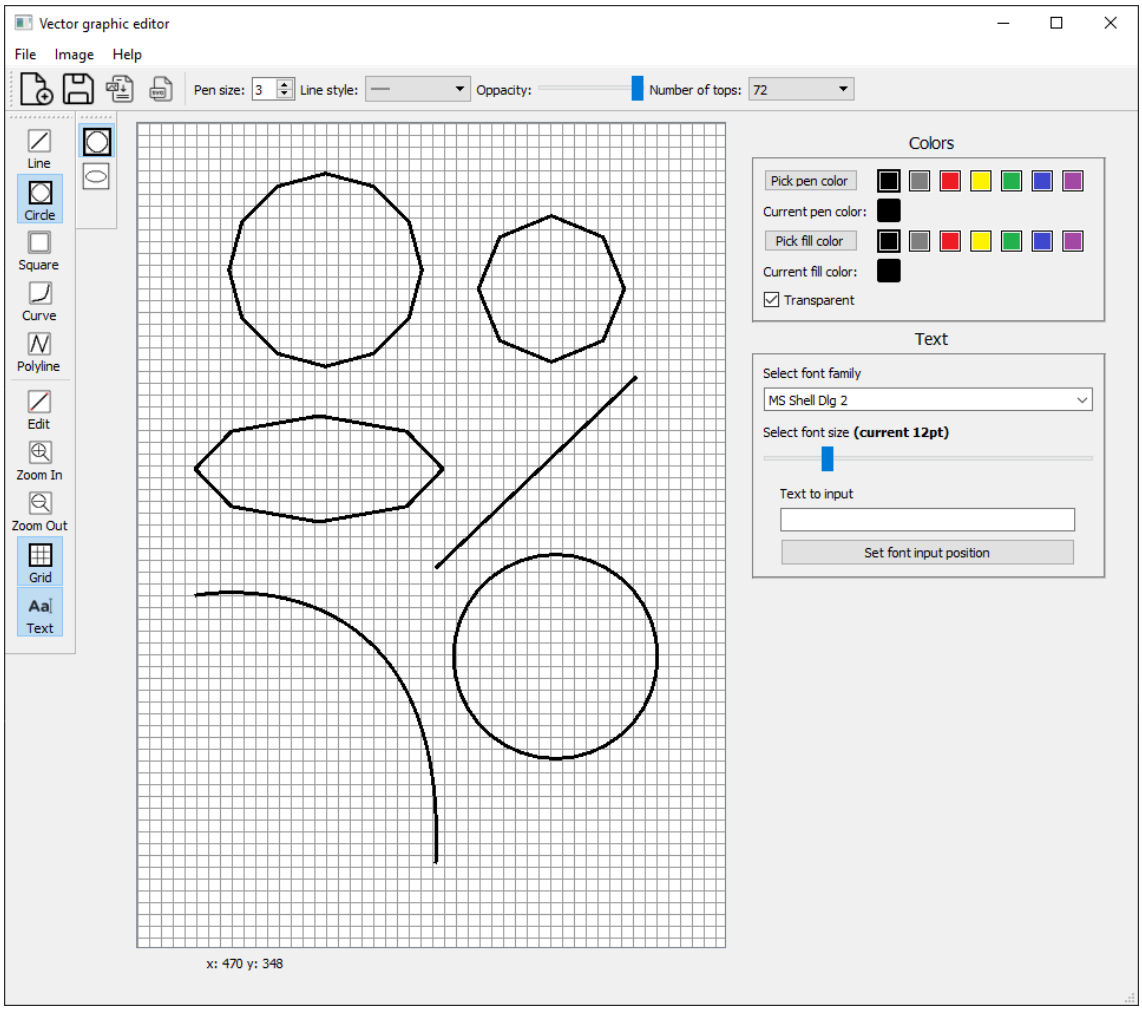
Konverze z formátu *.ui* na formát *.py* je ovšem jednosměrná. Knihovna PyQt5 bohužel touto funkcionalitou nedisponuje. Nedoporučuje se tedy provádět změny v konvertovaném souboru, protože by poté nemusel být kompatibilní se souborem, z něhož byl vytvořen. [29]

6.1.3 SVG Elements

Knihovna PyQ5 umožňuje pouze ukládání objektů do formátu SVG, neumožňuje však jejich zpětné načítání. Kvůli tomu byla zvolena pro tuto funkcionalitu knihovna SVG Elements [30], která umožňuje takzvané parsování SVG souboru. Tím je myšleno úplné a přesné zpracování SVG souboru, kdy z něj tato knihovna zvládne extrahovat všechna geometrická data, která jsou zapotřebí.

6.2 Implementace

Uživatelské prostředí výsledné aplikace je zobrazeno na obr. 22. Na následujících řádcích je aplikace stručně okomentována a v dalších kapitolách budou probrány detailněji všechny její funkcionality.



Obr. 22: Výsledný vektorový grafický editor

V horní části okna se nachází hlavní menu, které obsahuje položky určené ke správě souboru, možnosti zobrazení kreslící plochy, nástroje pro editaci a uživatelský manuál.

Pod hlavním menu je panel nástrojů, který umožňuje rychlejší přístup k některým položkám menu a dále obsahuje nastavení velikosti použitého pera, jeho styl a průhlednost objektů.

V levé části aplikace je umístěn další panel nástrojů. Ten umožňuje uživateli zvolit geometrický tvar, jejž chce zrovna nakreslit. Jedná se o přímku, kruh, čtverec, Bézierovu křivku určenou třemi body a lomenou čáru. Součástí tohoto panelu nástrojů je i tlačítko, které spouští editační režim.

Oba zmíněné panely nástrojů jsou pohyblivé. Uživatel si je tak může nastavit podle libosti tak, jak mu to bude nejvíce vyhovovat.

Největší prostor v GUI aplikace zabírá plátno, na které je možné kreslit již zmíněné geometrické tvary.

Na pravé straně okna se nachází nastavení barev. Je zde zobrazeno několik základních barev, jež jsou rozděleny do dvou kategorií. První z nich je barva čáry a druhá je barva výplně objektů. Výplň objektů lze samozřejmě nastavit i jako transparentní a to pomocí zaškrtávacího políčka umístěného u výběru barevné výplně.

6.3 Funkce VGE_B

V následujících podkapitolách jsou popsány vybrané vlastnosti a funkce VGE_B. Ve vhodných případech jsou některé z nich doplněny ilustračními obrázky, či částmi kódu. Veškeré zdrojové soubory jsou k dispozici v příloze 1.

6.3.1 Kreslící nástroje

Aplikace VGE_B podporuje kreslení jednoduché čáry, lomené čáry, čtyřúhelníku, kruhu, elipsy a také umožňuje vkládání textu.

Pro vkládání textu je nutné kliknout na tlačítko k tomu určené, které se nachází v levém panelu nástrojů. Po stisknutí tohoto tlačítka se zobrazí možnosti textu, kde lze vybrat požadovaný font a velikost písma. Následně do vstupního pole napíše uživatel svůj text a po potvrzení vybere pozici na plátně, kam se má text vykreslit.

Postup při kreslení základních grafických objektů je poměrně přímočarý. Po zvolení příslušného geometrického útvaru stačí levým tlačítkem myši zvolit počáteční bod na plátně, tahem kurzoru nastavit velikost útvaru a po uvolnění tlačítka se daný objekt vykreslí na obrazovku.

Rozdílný je způsob kreslení pouze při volbě lomené čáry a křivky. Křivka je definována třemi body, a proto je nejdříve nutné pomocí levého tlačítka myši zvolit počáteční a koncový bod dané křivky. Třetím kliknutím na plátno se zvolí kontrolní bod křivky a ta je následně vykreslena. Lomená čára se kreslí velmi podobným způsobem jako čára klasická. Rozdíl je v tom, že při uvolnění levého tlačítka myši se dále pokračuje

v kreslení. Pro ukončení módu lomené čáry je nutné stlačit pravé tlačítko myši, či vybrat jiný grafický objekt.

Veškeré oblouky se z důvodu, který byl vysvětlen výše, vykreslují jako lomené čáry. U kružnice a elipsy lze dokonce zvolit, kolik vrcholů má výsledná lomená čára mít. Algoritmy pro tyto dva geometrické útvary využívají parametrických rovnic kružnice a elipsy. Pro vykreslení křivky jako lomené čáry je využito rovnice kvadratické Bézierovy křivky [4], jež je zobrazena níže.

$$\mathrm{C(t)} = (1 - \mathrm{t})^2 P_0 + 2t(1-t)P_1 + t^2 P_2; t \in \langle 0,1 \rangle \tag{14}$$

Proměnné P_0, P_1 a P_2 znázorňují počáteční, kontrolní a koncový bod křivky. Pomocný parametr t může nabývat reálných hodnot od nuly do jedné včetně. Výsledný bod na křivce je pak označen $C(t)$.

Níže na obr. 23 je zobrazena implementace popsané rovnice do VGE_B. Jedná se o funkci *create_curve(points)*. Ta jako vstup přijímá tři body Bézierovy křivky, jež jsou zadány uživatelem. Nejdříve je do proměnné *path* uložena instance třídy *QPainterPath()*. Dále je vytvořen prázdný list *bezier_poinst*, jenž bude sloužit pro ukládání vypočítaných bodů křivky. Poté následuje *for* cyklus, který postupně vypočítává souřadnice všech bodů křivky a zapisuje je do zmíněného seznamu. Poté jsou použity metody *moveTo()* a *lineTo()* třídy *QPainterPath()* pro nastavení startovního bodu lomené čáry a následně i všech ostatních. Funkce vrací proměnnou *path*, která se poté dále využije v programu pro vykreslení lomené čáry.

```
def create_curve(points):
    path = QtGui.QPainterPath()
    bezier_points = []
    for i in range(0, 100, 1):
        i = i / 100
        x = math.pow((1 - i), 2) * points[0].x() + 2 * i * (1 - i) * points[2].x() + \
            math.pow(i, 2) * points[1].x()
        y = math.pow((1 - i), 2) * points[0].y() + 2 * i * (1 - i) * points[2].y() + \
            math.pow(i, 2) * points[1].y()
        i = i * 100
        bezier_points.insert(int(i), (x, y))
    path.moveTo(points[0])
    for i in range(len(bezier_points)):
        path.lineTo(bezier_points[i][0], bezier_points[i][1])
    return path
```

Obr. 23: Implementace kvadratické Bézierovy křivky

6.3.2 Možnosti grafických objektů

U všech grafických objektů lze nastavit typ použité čáry (plná, čárkovaná, tečkovaná) a její šířku.

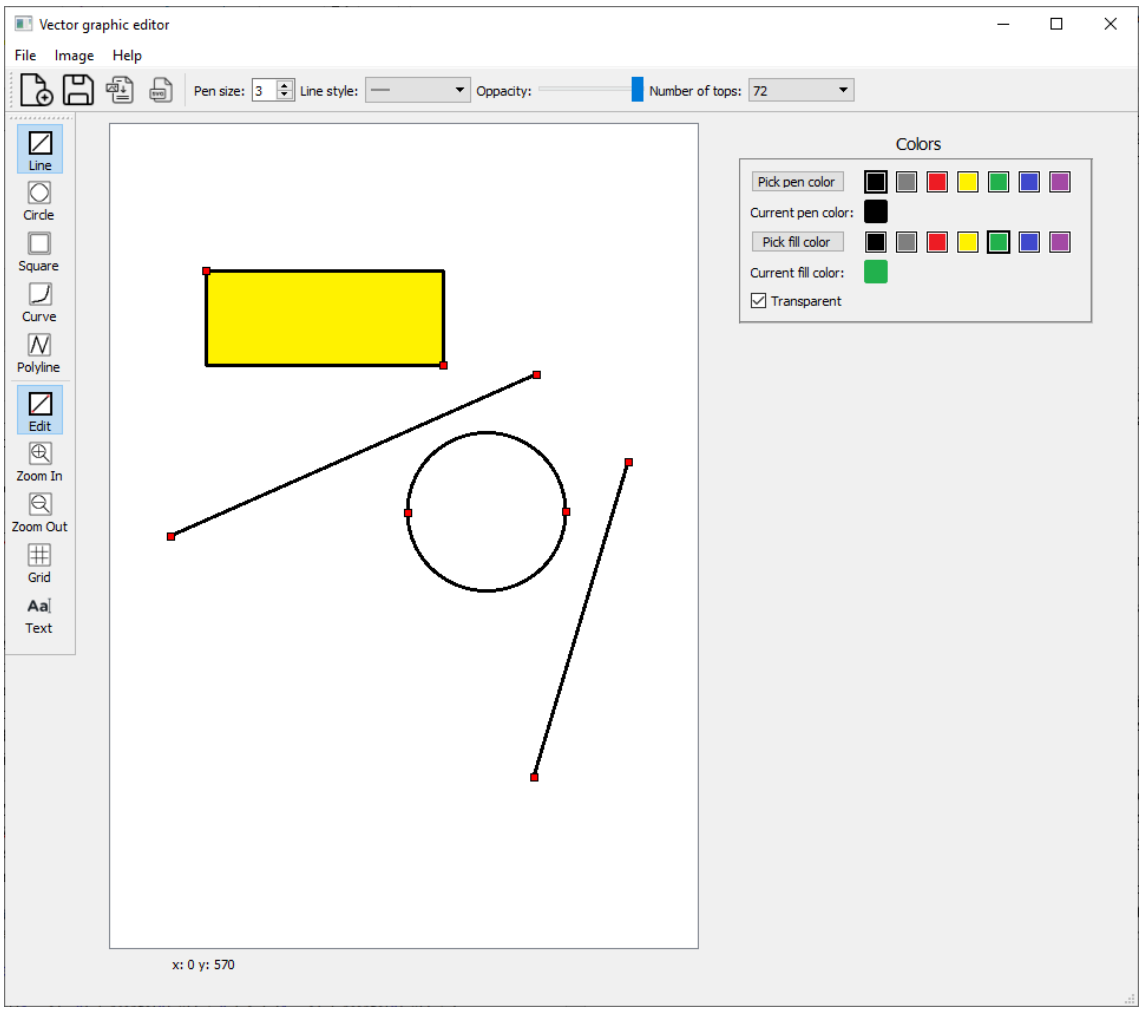
Je také možné měnit barvu čáry pomocí panelu nástrojů v pravé části aplikace. V této časti se nachází přednastavené barvy, ale je rovněž možné zvolit si barvu vlastní

pomocí tlačítka *Pick pen color*. Podobně jako volba barvy čáry funguje i barva výplně objektů. Zde je rozdíl v tom, že uživatel volí míru průhlednosti objektu.

6.3.3 Manipulační a editační nástroje

Polohu vykreslených objektů lze měnit pomocí pravého tlačítka myši. Společně s tím lze objekty otáčet o devadesát stupňů doleva či doprava. To je možné vybráním objektu stiskem pravého tlačítka myši a následným stiskem šipky doprava nebo doleva na klávesnici.

Do aplikace byl přidán i editační mód, při jehož zapnutí se zobrazí řídící body vykreslených objektů a jejich popotažením lze měnit velikost daného objektu. Podoba tohoto módu je zobrazena na obr. 24.



Obr. 24: VGE_B – editační mód

Další funkcionalitou v tomto směru je možnost kopírování a vkládání objektů. Toho lze docílit vybráním objektu a následným stiskem známých klávesových zkratek *Ctrl* + *C* a *Ctrl* + *V*.

Objekty lze i mazat. To je možné dvěma způsoby. Prvním z nich je vybrání objektu a následný stisk tlačítka *delete* na klávesnici a druhým způsobem je využití klávesové zkratky *Ctrl + Z*, jež funguje na principu zásobníku LIFO (Last In First Out). Objekt, který je nakreslen jako poslední, se po stisknutí zmíněné klávesové zkratky vymaže jako první.

Kreslící plátno je možné přibližovat a oddalovat pomocí kolečka myši. Druhou možností, jak toho docílit, je pomocí příslušných tlačítek v levé liště nástrojů. Jestliže je scéna přiblížená, automaticky se v ní objeví horizontální a vertikální posuvníky, jimiž se po ní dá pohybovat.

Pomůckou pro uživatele je dozajista funkce mřížky. Jestliže je tato funkce zapnutá, objekty se při vytváření a při změně jejich polohy k mřížce přichycují, což může v některých případech značně zjednodušit práci.

6.3.4 Ukládání, import a export

Výsledná aplikace umožňuje ukládání práce do vektorového formátu SVG a rovněž i import SVG souborů. Funkce načítání je však omezena jen na SVG soubory, které byly vytvořené pomocí aplikace VGE_B.

Je umožněn taktéž export do rastrových formátů, kterými jsou PNG, JPEG, BMP a GIF.

Jako ukázka možností vytvořeného vektorového grafického editoru bylo nakresleno logo VUT.

Obr. 25: Logo VUT vytvořené ve VGE_B

7 ZÁVĚR

Rešeršní část této diplomové práce se věnovala počítačové grafice, především pak grafice vektorové a vektorovým grafickým editorům. Na základě informací získaných z této rešerše byl poté navržen a naprogramován vlastní vektorový editor.

Vytvořený editor bude dále využíván zejména jako pomocný sw pro generování testovacích dat pro výzkum v oblasti rozpoznávání vektorově zadaných objektů v rastrovém obrázku. Právě pro tento účel bylo potřeba vytvořit vektorový editor, který bude implicitně ukládat oblouky jako lomené čáry z důvodu zjednodušení dalších výpočtů, které využívá vyvíjený algoritmus. Tento cíl byl splněn, výsledný editor dokáže ukládat kružnice a elipsy jako lomené čáry. Dokonce je možné nastavit, kolik čar má být pro uložení použito. Společně s touto funkcionalitou editor obsahuje obecné funkce běžných vektorových editorů, které jsou detailně popsány v kapitole 6.

Kromě shromáždění údajů a vytvoření použitelného a snad i užitečného programu lze konstatovat, že práce byla pro autora velice zajímavá, protože jej obohatila o hlubší znalosti počítačové grafiky a grafických editorů. Hlavním přínosem byla možnost vyzkoušet si práci na větším projektu, který autor plánuje do budoucna dále rozvíjet. Existuje totiž řada funkcionalit, kterými by se dal vektorový editor rozšířit. Příkladem může být podpora více než jednoho vektorového formátu, jímž je v tomto případě SVG, popřípadě vytvoření formátu vlastního.

8 SEZNAM POUŽITÉ LITERATURY

[1] PÁSZTO, Vít a Zdeňka Krišová. *Počítačová grafika*. Olomouc: Moravská vysoká škola Olomouc, o.p.s., 2018. ISBN 978-80-7455-089-8.

[2] ZÍDEK, Karel. Vývoj počítačové grafiky. In: *fi.muni.cz* [online]. 2006-12-06 [cit. 2022-02-26]. Dostupné z: https://www.fi.muni.cz/usr/jkucera/pv109/2006/xzidek2.htm.

[3] Computer History Museum. Timeline of Computer History. In: *ComputerHistory.org* [online]. Computer History Museum ©2022 [cit. 2022-02-27]. Dostupné z: https://www.computerhistory.org/timeline/graphics-games/.

[4] MARTÍŠEK, Dalibor. *Matematické principy grafických systémů*. Brno: Littera, 2002. ISBN 80-85763-19-2.

[5] CORRERIA, Paulo. AutoCAD History. In: *2d3danima.com* [online]. 2020-04-01 [cit. 2022-02-27]. Dostupné z: https://2d3danima.com/autocad-history.

[6] ITnetwork. Úvod do počítačové grafiky. In: *ITnetwork.cz* [online]. ITnetwork ©2022 [cit. 2022-03-03]. Dostupné z: https://www.itnetwork.cz/grafika/uvod.

[7] MYŠKA, Karel a Michal Munzar. *Počítačová grafika a základy videotvorby*. Hradec králové: Univerzita Hradec Králové, Ústav sociální práce, 2014. ISBN 978-80-7435-457-1.

[8] HILDEBRANDT, Darlene. How to Read (and Use) Histograms for Beautiful Exposures. In: *digital-photography-school.com* [online]. Digital Photography School ©2022 [cit. 2022-03-05]. Dostupné z: https://digital-photography-school.com/how-to-read-and-use-histograms/.

[9] Psychology Wiki. HSL and HSV. In: *Psychology.fandom.com* [online]. Psychology Wiki ©2022 [cit. 2022-03-10]. Dostupné z: https://psychology.fandom.com/wiki/HSL_and_HSV.

[10] AUGUSTA, Lukáš. Co jsou to barevné modely RGB, HSL a HSB a který je lepší?. In: *designui.cz* [online]. ©2022 [cit. 2022-03-10]. Dostupné z: https://www.designui.cz/lekce/co-jsou-to-barevne-modely-rgb-hsl-a-hsb-a-ktery-je-lepsi.

[11] CHERNOV, Vladimir; ALANDER, Jarmo; BOCHKO, Vladimir. Integer-based accurate conversion between RGB and HSV color spaces. *Computers & Electrical Engineering*, 2015, 46: 328-337.

[12] The Printing Connection. Raster Images vs. Vector Graphics. In: *printcnx.com* [online]. The Printing Connection ©2022 [cit. 2022-03-20]. Dostupné z: https://www.printcnx.com/resources-and-support/addiational-resources/raster-images-vs-vector-graphics/.

[13] MATOUŠEK, Radomil. Metody kódování. Brno: Vysoké učení technické. [online] Dostupné z: http://www.uai.fme.vutbr. cz/~matousek/TIK/TIKv19.pdf, 2006.

[14] MrDoranComputing. RLE. In: *mrdorancomputing.com* [online]. MrDoranComputing ©2022 [cit. 2022-03-25]. Dostupné z: https://mrdorancomputing.com/data-representation/rle/.

[15] HORDĚJČUK, Vojtěch. Kompresní algoritmus LZW. In: *voho.eu* [online]. ©2022 [cit. 2022-03-25]. Dostupné z: https://www. http://voho.eu/wiki/algoritmus-lzw/.

[16] MALÝ, Jan. Srovnání metod pro ztrátovou kompresi obrazu. In: *elektrorevue.cz* [online]. ©2022 [cit. 2022-03-25]. Dostupné z: https:// http://www.elektrorevue.cz/clanky/06042/index.html.

[17] HORČIČKO, Vladimír. *Rozbor možností počítačové grafiky s důrazem na grafiku vektorovou*. Praha: Bankovní institut vysoká škola Praha, Katedra informatiky a kvantitativních metod, 2015. 60 s. Vedoucí bakalářské práce Ing. Bohuslav Růžička, CSc.

[18] Stránky k výuce informatiky. Grafické formáty. In: *ivt.mzf.cz* [online]. ©2022 [cit. 2022-03-27]. Dostupné z: http://www.ivt.mzf.cz/grafika/graficke-formaty/.

[19] ReadyTechGo. What´s the difference between a PND and a JPEG. In: *readytechgo.com* [online]. ©2022 [cit. 2022-03-28]. Dostupné z: https://www.readytechgo.com.au/whats-the-difference-between-a-png-and-a-jpeg/.

[20] HOLANEC, Michal. *Srovnání programů pro editace 2D grafiky*. Praha: České vysoké učení technické v Praze, Fakulta elektrotechnická, 2009. 71 s. Vedoucí bakalářské práce Mgr. Jiří Danihelka.

[21] Adobe. Co je PDF. In: *adobe.com* [online]. ©2022 [cit. 2022-04-05]. Dostupné z: https://www.adobe.com/cz/acrobat/about-adobe-pdf.html.

[22] Adobe. Soubory AI. In: *adobe.com* [online]. ©2022 [cit. 2022-04-07]. Dostupné z: https://www.adobe.com/cz/creativecloud/file-types/image/vector/ai-file.html.

[23] *Adobe* [online]. ©2020 [cit. 2022-04-10]. Dostupné z: https://www.adobe.com/.

[24] *Coreldraw* [online]. ©2020 [cit. 2022-04-12]. Dostupné z: https://www.coreldraw.com/.

[25] *Autodesk* [online]. ©2020 [cit. 2022-04-15]. Dostupné z: https://www.autodesk.com/.

[26] ROOT, Dan. A brief history of the Python programming language. In: *python.plainenglish.io* [online]. Python in Plain English ©2022 2020-08-16 [cit. 2022-04-20]. Dostupné z: https://python.plainenglish.io/a-brief-history-of-the-python-programming-language-4661fcd48a04.

[27] BLAŽKOVÁ, Tereza. Umí Python opravdu všechno, aby se ti jej vyplatilo umět?. In: *ITnetwork.cz* [online]. ITnetwork ©2022 2021-08-09 [cit. 2022-04-20]. Dostupné z: https://www.itnetwork.cz/blog/umi-python-opravdu-vsechno-aby-se-ti-jej-vyplatilo-umet.

[28] Codeacademy. What Is a Framework?. In: *codeacademy.com* [online]. ITnetwork ©2022 [cit. 2022-04-20]. Dostupné z: https://www.codecademy.com/resources/blog/what-is-a-framework/.

[29] *Qt for Python* [online]. ©2020 [cit. 2022-04-20]. Dostupné z: https://doc.qt.io/qtforpython/.

[30] *SVG elements* [online]. ©2020 [cit. 2022-04-20]. Dostupné z: https://github.com/meerk40t/svgelements.

9 SEZNAM OBRÁZKŮ

10 SEZNAM TABULEK

11 SEZNAM ZKRATEK

CAD (Computer-Aided Design) – Počítačem podporované projektování

CGI (Computer Generated Graphic) – Počítačem generovaná grafika

CRT (Cathod Ray Tube) – Katodová trubice

DAC-1 (Design Augmented by Computers-1) – Design rozšířený počítačem

DPI (Dots Per Inch) – Body na na jeden palec

DPP (Dots Per Pixel) – Pixely na jeden palec

GUI (Graphical User Interface) – Grafické uživatelské rozhraní

MIT (Massachussetts Institute of Technology) – Massachusettský technologický institut

SAGE (Semi-Automatic Ground Equipment) – Poloautomatické pozemní prostředí

12 SEZNAM PŘÍLOH

1. VGE_B.zip – Archiv obsahující všechny zdrojové kódy k výsledné aplikaci