VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

NAVIGACE POMOCÍ TURISTICKÉ MAPY

NAVIGATION USING A TURISTIC MAP

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE Bc. JAKUB MIHÁL
AUTHOR

VEDOUCÍ PRÁCE Ing. KAREL HORÁK, Ph.D.
SUPERVISOR

BRNO 2012

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Diplomová práce

magisterský navazující studijní obor
Kybernetika, automatizace a měření

Student:	Bc. Jakub Mihál	***ID:***	106635
Ročník:	2	***Akademický rok:***	2011/2012

NÁZEV TÉMATU:

Navigace pomocí turistické mapy

POKYNY PRO VYPRACOVÁNÍ:

Úkolem studenta je vytvoření programu, který hodně analyzuje turistickou mapu a bude schopen nalézt optimální trasu mezi dvěma zvolenými body. Program bude vhodně volit trasu na základě profilu a druhu terénu.

DOPORUČENÁ LITERATURA:

1. HLAVÁČ, V., ŠONKA, M. Počítačové vidění. 1. vydání. Praha: Grada, 1992. 272 str. ISBN 80-85424-67-3.

Termín zadání:	6.2.2012	***Termín odevzdání:***	21.5.2012

Vedoucí práce: Ing. Karel Horák, Ph.D.
Konzultanti diplomové práce: Ing. Luděk Červinka

doc. Ing. Václav Jirsík, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

ABSTRAKT

Diplomová práce se zabývá návrhem a vytvořením programu, který je schopen nalézt nejkratší cestu po trasách na turistické mapě mezi zvolenými body. Uživatel si může zvolit typ turistické trasy. Aplikace také vytváří výškovou mapu terénu z vrstevnic a výškový profil cesty.

KLÍČOVÁ SLOVA

mapy, počítačové vidění, hledání nejkratší trasy, vrstevnice, výšková mapa, výškový profil, Matlab

ABSTRACT

The master's thesis deals with the design and realization of a program that is able to find the shortest route in the tourist map between selected points. The user can select the type of hiking trail. Application also generates terrain heightmap from contour lines and path elevation profile.

KEYWORDS

maps, computer vision, finding the shortest route, contour lines, heightmap, elevation profile, Matlab

MIHÁL, Jakub *Navigace pomocí turistické mapy*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2012. 50 s. Vedoucí práce byl Ing. Karel Horák, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Navigace pomocí turistické mapy“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

Brno ....

(podpis autora)

PODĚKOVÁNÍ

Rád bych poděkoval konzultantu diplomové práce panu Ing. Luďku Červinkovi za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....
(podpis autora)

OBSAH

1 Úvod **10**

2 Teoretický rozbor **11**

2.1 Kartografie a mapy 11

2.1.1 Jazyk mapy 11

2.1.2 Turistické mapy 15

2.2 Teorie grafů 17

2.2.1 Přehled algoritmů pro hledání nejkratší trasy 18

3 Praktická část **23**

3.1 Zdroj a formát obrázku 23

3.2 Předzpracování trasy 23

3.2.1 Segmentace trasy, [7] 23

3.2.2 Morfologické operace, [7] 24

3.2.3 Spojení přerušených tras 25

3.3 Předzpracování vrstevnic 29

3.3.1 Segmentace vrstevnic 29

3.3.2 Morfologické operace 29

3.3.3 Spojení přerušených vrstevnic 30

3.4 Vytvoření výškové mapy 33

3.4.1 Přiřazení výšky vrstevnicím 33

3.4.2 Výpočet výšky mezi vrstevnicemi 37

3.5 Nalezení nejkratší trasy 39

3.5.1 Uživatelem zadané body trasy 39

3.5.2 Vytvoření seznamu vrcholů grafu 39

3.5.3 Vytvoření seznamu hran grafu 40

3.5.4 Vyhledání nejkratší trasy v mapě 40

3.6 Uživatelské rozhraní 41

4 Závěr **43**

Literatura **45**

Seznam symbolů, veličin a zkratek **47**

A Ukázka nalezení trasy v obrazu **48**

B Grafické uživatelské rozhraní **50**

SEZNAM OBRÁZKŮ

2.1 Příklad bodových značek z turistické legendy, zdroj: [4]. 12
2.2 Příklad liniových značek z turistické legendy, zdroj: [4]. 13
2.3 Příklad areálových značek z turistické legendy, zdroj: [4]. 14
2.4 Příklad grafu vzdálenosti některých měst v ČR, vzdálenosti čerpány z http://jakus.info/skripty/mesta.htm. 17
2.5 Princip algoritmu průchodu do šířky pro neohodnocený graf. 19
2.6 Adaptace algoritmu BFS na ohodnocený graf. 19
3.1 Příklad segmentované části obrázku (cyklostezka). 24
3.2 Příklad části obrázku s aplikovanými morfologickými operacemi (vlevo dilatace, vpravo ztenčování). 25
3.3 Souřadnicová síť při určování polohy bodu v obraze. 26
3.4 Příklad špatně (vlevo) a správně (vpravo) spojené trasy. 27
3.5 Příklad distanční mapy (vpravo) pro binární obraz (vlevo) s vyznačeným výchozím bodem. 28
3.6 Příklad napojení trasy pomocí distanční mapy. 28
3.7 Příklad segmentace vrstevnic v obrázku. 29
3.8 Ukázka spojení některých vrstevnic úsečkami (vlevo) a vyznačení spojených částí (vpravo). 30
3.9 Určení směrnic α a β koncových bodů A a B v obraze. 31
3.10 Spojení některých vrstevnic polynomy (vlevo) a vyznačení spojených částí (vpravo). 32
3.11 Průsečíky úseček z vrcholu kopce a vrstevnic v okolí. 34
3.12 Distribuce výškových hodnot z výchozího vrcholu do okolních vrstevnic. 35
3.13 Určení průsečíků vrstevnic a normály bodu A. 36
3.14 Distribuce výškových hodnot z ohodnocených vrstevnic do okolních pomocí normál (první iterace). 37
3.15 Distribuce výškových hodnot z ohodnocených vrstevnic do okolních pomocí normál (druhá iterace). 37
3.16 Výpočet výškových hodnot v okolí výchozího bodu (vlevo) a mezi vrstevnicemi (vpravo). 38
3.17 Výsledek výškové mapy po aplikaci 2-D interpolace (vlevo) a Gaussova filtru (vpravo). 39
3.18 Ovládací okno grafického uživatelského prostředí. 41
A.1 Vstupní obraz turistické mapy. 48
A.2 Předzpracování obrazu. 48
A.3 Výsledek hledání trasy v grafu. 49
A.4 Vyznačená nalezená trasa délky 1230,0 obrazových elementů. 49

B.1 Ukázka zpracování obrázku mapy v grafickém uživatelské rozhraní. . . 50

SEZNAM TABULEK

2.1 Tabulka sousednosti měst zobrazených v obrázku 2.4 18
2.2 Tabulka vzdáleností měst zobrazených v obrázku 2.4 18

1 ÚVOD

Tvorba mapy Země má historickou tradici dlouhou jako samotné dějiny lidstva, první mapy vznikaly zřejmě ještě dříve než samotné písmo. Nejstarší mapy jsou datovány již v období paleolitu, tj. cca 20 tisíc let před naším letopočtem [2].

Co potřebuje k přežití takový pravěký člověk v době kamenné? Kromě dalších nezbytných vymožeností této doby (oheň, zbraň, apod.) je nutná hlavně mapa - informační záznam o bezprostředním okolí. Záznam o poloze potravy, vody, surovin, nepřátelském území. Později také osad, obchodních tras, vlastnických poměrů pozemků, apod.

Ještě dnes, v 21. století, je význam mapy nedocenitelný. Existuje velké množství různých druhů map s nejrůznějšími využitími. Mapy se liší svým rozsahem (mapa světa, zemských polokoulí, kontinenty, státy, ...), účelem (např. pro státní správu, vojenské, orientaci, geologické, ...), obsahem (zeměpisné, topografické, přehledné, tematické) a dalšími charakteristikami.

S rozvojem techniky tvorba map dospěla až k leteckému mapování a mapování pomocí satelitů z orbity Země. Téměř 99% povrchu planety je již zmapováno a dále upřesňováno [1] (http://asterweb.jpl.nasa.gov/gdem.asp).

Není od věci, že se nazývá toto období dobou informační. Lidé si žádají čím dál tím větší dostupnost informací a tedy i map. Vznikly tak tzv. interaktivní mapy, dostupné na internetu široké veřejnosti.

Cílem této diplomové práce je zaměřit se na zpracování turistické mapy. Následující kapitoly provádějí základní analýzu mapy z hlediska kartografického. Turistická mapa je reprezentována souborem rastrového obrázku, který je předzpracován algoritmy počítačového vidění. Nalezení trasy je poté provedeno pomocí grafových algoritmů. Hlavním úkolem je vytvořit aplikaci schopnou interaktivně najít ve zvolené turistické mapě cestu s definovanými vlastnostmi.

2 TEORETICKÝ ROZBOR

2.1 Kartografie a mapy

Tvorbou a zpracováním map se zabývá vědní obor kartografie [2]. Podle ČSN je kartografie definována následovně:

> „Kartografie je vědní obor zabývající se znázorněním zemského povrchu a nebeských těles a objektů, jevů na nich a jejich vztahů ve formě kartografického díla a dále soubor činností při zpracování a využívání map.“ (ČSN 73 0406 Názvosloví kartografie, 1984.)

Produkcí kartografie jsou kartografická díla, mezi které se řadí mimo jiné i mapy. Také pojem mapa má svoji danou definici podle ČSN:

> „Mapa je zmenšený generalizovaný konvenční obraz Země, nebeských těles, kosmu či jejich částí, převedený do roviny pomocí matematicky definovaných vztahů (kartografickým zobrazením), ukazující podle zvolených hledisek polohu, stav a vztahy přírodních, socioekonomických a technických objektů a jevů.“ (ČSN 73 0402, národní definice)

Tato definice v podstatě říká, že mapa je zevšeobecněným zmenšeným rovinným modelem skutečné reality povrchu zemského či jiného.

Obraz je zaznamenán na zmenšené rovině pomocí daného měřítka a za použití dohodnutých znaků konečného počtu. Díky tomu není možné zaznamenat veškeré detaily skutečnosti na mapu a většinou to není ani nutné. Obraz skutečnosti je tedy tzv. generalizován (zevšeobecněn).

Podle použití se liší smluvené znaky vyskytující se na mapách. Těmito znaky (jazyk mapy) jsou myšleny kartografické znaky a mapové značky. Tyto znaky a značky popisují polohu, popřípadě tvar či stav různých objektů a jevů skutečnosti. Na mapě se mohou vyskytovat i abstraktní objekty či jevy spojené s danou lokalitou či bodem. Je nutné, aby mapa byla přehledná, zároveň dostatečně podrobná a také estetická.

Mapovaný povrch reality není obecně rovinný útvar, ale nepravidelný zakřivený povrch. Povrch je tedy aproximován na referenční plochu (elipsoid, koule) a poté převeden do roviny pomocí daného matematického zákona (kartografické zobrazení).

2.1.1 Jazyk mapy

Základním rozlišovacím znakem map (v závislosti na použití) je jazyk mapy (kartografický jazyk) [2],[3]. Jazyk mapy je soubor obrazově znakových smluvených mapo-

vých znaků, tj. kartografických znaků. Kartografický znak je nositel určité informace vyjádřené v mapě reprezentující skutečný objekt či vlastnost.

Grafické mapové značky lze rozdělit na několik typů - bodové, liniové nebo areálové značky. Ty lze rozdělit dále podle jejich vlastností, např. tvar, velikost, orientace, struktura, výplň, atd. V závislosti na použité mapě se liší význam mapové značky. Souhrn a popis použitých mapových znaků bývá uveden v legendě konkrétní mapy.

Bodové značky

Bodové značky se používají pro objekty, které mají vzhledem k měřítku mapy bodový charakter. Ve skutečnosti se nemusí jednat jen o bod (např. vrchol hory), ale může jít i o objekty, které mají na mapě velmi malý rozměr a generalizací se nahradí bodem (např. budovy). Nemusí vyjadřovat jen polohu daného objektu, ale také i jeho kvantitu či kvalitu (např. pomocí velikosti).

Obr. 2.1: Příklad bodových značek z turistické legendy, zdroj: [4].

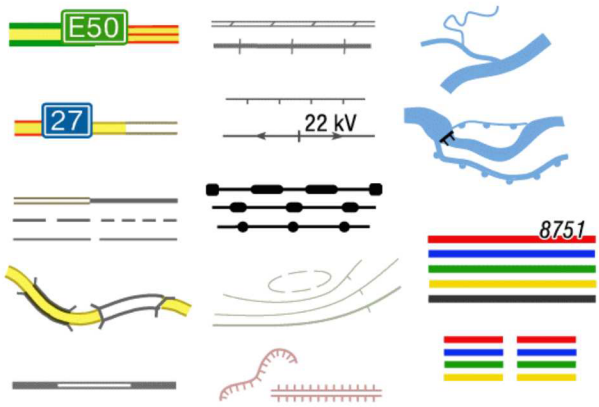
Bodové značky lze rozdělit podle jejich kvantitativního hlediska:

- geometrické - mají jednoduchý geometrický tvar (trojúhelník, čtverec, kruh, apod.), většinou to jsou skutečné bodové objekty (např. vrchol hory)
- symbolické - mají asociativní povahu, určují třídu zobrazovaného objektu, vyjadřují symbolem událost či objekty spojené s danou lokalitou (např. přístaviště, důl, pošta)
- obrázkové - zobrazují siluetou či půdorysem již konkrétní typ zobrazovaného objektu, použití pro orientačně nebo kulturně významné objekty (např. silueta konkrétního hradu), v mapě se vyskytuje pouze jednou a vzhledem ke své čitelnosti se většinou ani neuvádí v legendě

- alfanumerické - zahrnují písmena i číslice (např. významné datum či chemické značky nerostů), nejsou totožné s popisky v mapě

Liniové značky

Pro reálné i abstraktní předměty či jevy obecně křivkového charakteru se používají tzv. liniové značky. Podélná osa značek se shoduje s průběhem osy zobrazované křivky (např. vodní toky). Ve výjimečných případech pouze spojuje krajní body úsečkou (např. trasy leteckých linek). Charakteristickým rysem značky je její šířka, struktura kresby či barva, případně i směr.



Obr. 2.2: Příklad liniových značek z turistické legendy, zdroj: [4].

Liniové značky lze rozdělit podle přesnosti zobrazení daného objektu:

- geometricky přesné - vyjadřují skutečný rozměr (i šířkou) a polohu křivky zmenšeným podle měřítka mapy (např. státní hranice)
- topograficky přesné - příčný rozměr křivky je generalizován, většinou nahrazen konstantní šířkou (např. silnice, železnice)
- schematicky přesné - spojují krajní body či více bodů úsečkou, vyjadřují existující spojení (např. letecké či plavební spoje)

Speciálním typem liniových značek jsou izolinie. Tyto značky spojují body se stejnou kvantitativní hodnotou. Při zvýšení hustoty izolinií na dané ploše získávají plošný charakter a pak se někdy řadí mezi plošné kartografické značky. Příčný gradient kvantitativní hodnoty je vyjádřen pomocí vzdálenosti sousedních izolinií. Izolinie se nekříží. Mezi izolinie se řadí např. izotermy (stejná teplota), izobaty (hloubka), izohypsy (vrstevnice, výška), izobary (tlak), atd.

Areálové značky

Velká většina skutečných zobrazovaných objektů jsou plošného charakteru. Vzhledem ke generalizaci se tyto objekty zobrazují jako menší plochy nebo linie či body. Areálové značky vyjadřují v určité oblasti kvalitativní (např. klimatické oblasti, národnostně stejnorodé oblasti) či kvantitativní (např. hustota obyvatel) charakteristiku.

Obr. 2.3: Příklad areálových značek z turistické legendy, zdroj: [4].

Zobrazované oblasti mohou být ohraničené, ohraničené částečně nebo neohraničené. Dále dle prostorového uspořádání izolované, dotýkající se či překrývající se.

Areálové značky mohou mít přesnou polohu i tvar definovanou měřítkem mapy nebo mohou být pouze schématickým vyjádřením polohy či velikosti. Plochy lze potom vymezit opakovanou bodovou značkou, vykreslením čárou, šrafurou nebo vyplněním barvou.

Písmo mapy

Součástí téměř každé mapy je také popisná složka, jejíž účelem je popisovat kvantitativně (názvem) či kvalitativně (hodnotou) grafické složky. Písmo se používá v následujících prvcích mapy:

- Geografické názvosloví (popisy geografických prvků sídla, vodstvo, ohodnocení vrstevnic, atd.)
- Vysvětlivky (legenda, význam zkratek, atd.)
- Mimorámové údaje (souřadnicové sítě, atd.)
- Samostatná textová část (rejstříky, seznamy, tabulky, doprovodné texty, atd.)
- Obecná označení a zkratky

V mapovém obraze se vyskytuje pouze geografické názvosloví.

Písmo mapy se rozlišuje podle velikosti, typu nebo barvy, jejichž volba rozhoduje o další interpretaci charakteristiky daného objektu. Obecně významnější objekty se doplňují výraznějším popisem. Je potřeba, aby písmo bylo vhodně umístěno, aby bylo jednoznačně přiřazeno k popisujícímu objektu.

2.1.2 Turistické mapy

Turistická mapa je založena na topografických mapách s účelem orientace v terénu. Využívá se pro turistiku v přírodě i ve městě, tj. obecně v cestovním ruchu.

Charakteristickým rysem topografických map je podrobné zobrazení zemského povrchu - vodstva, reliéfu, sídel, komunikací, půdního povrchu a půdního pokrytu, hranic a geografického názvosloví. Zobrazení je v čase relativně statické (tj. nezobrazuje časový vývoj mapované oblasti). Vzhledem k územnímu rozsahu (o velikosti regionu či lokální oblasti) a podrobnému rozlišení je měřítko mapy střední až velké (tj. od 1 : 500 do 1 : 200 000).

Forma záznamu je analogová (na papíře či jiné rovině), digitální či reliéfní (plastické tvarované).

Topografické mapy vznikají většinou přímým mapováním terénu, mezi které se dnes z velké většiny řadí letecká fotogrammetrie. Dalším zpracováním (generalizací a zdůrazňování) vznikají odvozené topografické mapy.

Blízko k turistickým mapám mají i cyklistické mapy, mapy pro běžkaře, lyžařské, rybářské, atd. Kombinací turistické a cyklistické mapy vznikají cykloturistické mapy.

Mezi nejznámější vydavatele tištěných turistických map v ČR patří následující organizace a firmy:

- Klub českých turistů (http://www.klubturistu.cz/)
- Kartografie PRAHA, a.s. (http://www.kartografie.cz/)
- SHOCart, spol. s r.o. (http://www.shocart.cz/)
- Geodézie ČS a. s. (http://www.geodezie.cz/)
- Žaket (http://www.zaket.cz/)
- Marco Polo, s.r.o. (http://www.mairs.cz/)
- B.A.T. Program s.r.o. (http://www.batprogram.cz/)
- Kartografie HP s.r.o. (http://www.kartografiehp.cz/)

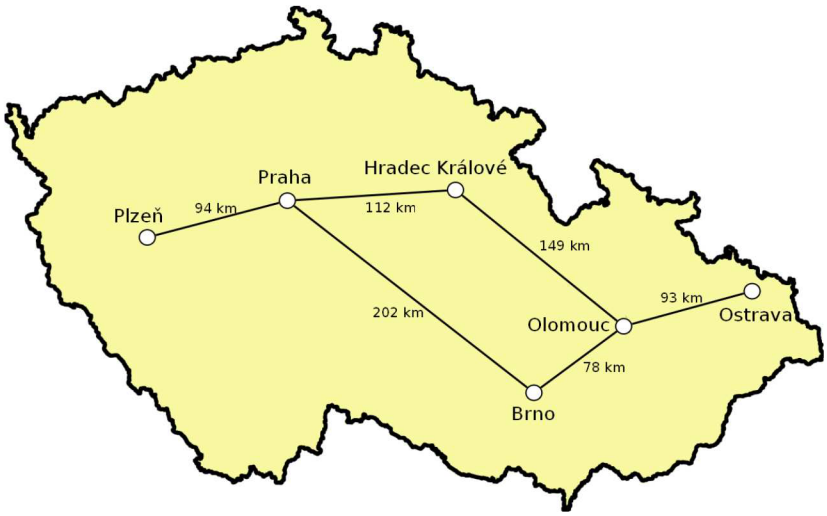
Digitální volně dostupné zdroje turistických (i jiných) map:

- Mapy.cz (http://mapy.cz/)
- Mapy Google (http://maps.google.com/)
- mapy.iDnes.cz (http://mapy.idnes.cz/)
- OpenStreetMap (http://www.openstreetmap.org/, http://openstreetmap.cz/)
- aMapy.cz (http://amapy.centrum.cz/)

- Mapy 1188 (http://mapy.1188.cz/)
- CRR - Mapový server (http://mapy.crr.cz/)
- Národní geoportál INSPIRE (http://geoportal.gov.cz/)
- Internetový zobrazovač geografických armádních dat (http://izgard.cenia.cz/)

2.2 Teorie grafů

Při hledání nejkratší trasy mezi zadanými body lze využít algoritmy hledající nejkratší cesty v grafu [9], [10]. V tomto grafu $G(V, E)$ reprezentuje množina křižovatek vrcholy V (vertex) grafu, a množina tras (spojnic) mezi nimi jejich hrany E (edge). Hrana e je tedy definována dvojicí vrcholů u a v, které jsou spojeny touto hranou.



Obr. 2.4: Příklad grafu vzdálenosti některých měst v ČR, vzdálenosti čerpány z http://jakus.info/skripty/mesta.htm.

Každá trasa má definovanou délku či obtížnost. Můžou být také vybrány preferované trasy, které jsou ve výsledku zvýhodněné. Hrany grafu jsou tedy ohodnocené (cena hrany) a ke grafu je nutné přidat funkci $h : E \to \mathbb{R}$, která přiřadí ke každé hraně e hodnotu $h(e)$. Tento graf se potom nazývá ohodnocený.

Uvažujeme, že se na mapě nevyskytují jednosměrné trasy, takže hrany grafu e mají v obou směrech totožné ohodnocení $h(e)$. Graf je potom neorientovaný.

O dvou různých vrcholech v grafu u a v je možné říci, že jsou sousední, pokud jsou přímo spojeny hranou e. Tuto vlastnost můžeme zapsat v tzv. matici sousednosti $\mathbf{A}$ podle následujícího algoritmu:

$$a_{u,v} = \begin{cases} 0 & \iff uv \notin E \\ 1 & \iff uv \in E \end{cases} \tag{2.1}$$

Pro neorientovaný graf je tato matice symetrická podle hlavní diagonály.

Vzdálenost dvou vrcholů u a v v grafové metrice je definována jako délka nejkratší cesty mezi těmito vrcholy. Metriky grafu lze zapsat (podobně jako sousednosti) do matice či tabulky vzdálenosti jednotlivých vrcholů.

Metrika grafu G je funkce $d : V \times V \to \mathbb{R}^+$, pro kterou platí následující axiomy:

	Praha	Brno	Ostrava	Plzeň	Olomouc	Hradec Králové
Praha	0	1	0	1	0	1
Brno	1	0	0	0	1	0
Ostrava	0	0	0	0	1	0
Plzeň	1	0	0	0	0	0
Olomouc	0	1	1	0	0	1
Hradec Králové	1	0	0	0	1	0

Tab. 2.1: Tabulka sousednosti měst zobrazených v obrázku 2.4

1. $d(u, v) = 0 \iff u = v$ (totožnost)
2. $d(u, v) = d(v, u)$ (symetrie)
3. $d(u, w) \leq d(u, v) + d(v, w)$ (trojúhelníková nerovnost)

Z axiomu 1 a 3 plyne nezápornost metriky $d \geq 0$.

	Praha	Brno	Ostrava	Plzeň	Olomouc	Hradec Králové
Praha	0	202	354	94	261	112
Brno	202	0	171	296	78	227
Ostrava	354	171	0	448	93	242
Plzeň	94	296	448	0	355	206
Olomouc	261	78	93	355	0	149
Hradec	112	227	242	206	149	0

Tab. 2.2: Tabulka vzdáleností měst zobrazených v obrázku 2.4

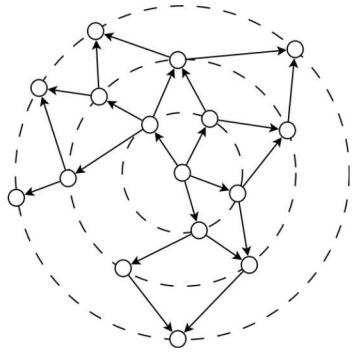
2.2.1 Přehled algoritmů pro hledání nejkratší trasy

Algoritmy pro hledání nejkratší trasy je možné rozdělit na tři různé typy:

- **Shortest path** - algoritmus nalezne nejkratší trasu mezi zadaným počátečním vrcholem s a cílovým vrcholem c, výsledkem je jediné číslo $d(s, c)$.
- **Single source shortest path** - algoritmus nalezne nejkratší trasy mezi zadaným počátečním vrcholem s a všemi ostatními zbývajícími vrcholy v, výsledky jsou zapsány jako vektor $d_v(s, v)$ s počtem prvků $|V|$.
- **All pairs shortest path** - algoritmus nalezne nejkratší trasu mezi všemi dvojicemi vrcholů grafu uv, reprezentace výsledků je pak nejlépe v matici vzdáleností $d_{u,v}(u, v)$ o velikosti $|V| \times |V|$.

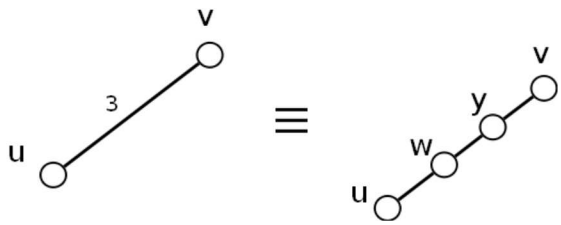
Průchod do šířky (BFS - Breadth First Search)

Algoritmus je původně vytvořen pro neohodnocené grafy nebo pro grafy, kde délka všech hran je stejná.



Obr. 2.5: Princip algoritmu průchodu do šířky pro neohodnocený graf.

Někdy je nazýván také jako algoritmus vlny: Prohledávání nejkratší trasy je realizována z počátečního bodu do všech stran (tzv. po vlnách). Pokud se narazí na vrchol s více hranami směřujícími po vlně, hledání se rozdělí (paralelizuje). Následovníci objeveného uzlu se ukládají do zásobníku. V i-té vlně (či hloubce) jsou vrcholy ve vzdálenosti i od počátečního bodu.



Obr. 2.6: Adaptace algoritmu BFS na ohodnocený graf.

Algoritmus je však možné přizpůsobit i pro ohodnocené grafy. Hrana grafu délky $d \in \mathbb{N}$ se rozdělí na d hran o jednotkové délce s $d - 1$ vrcholy.

V případě že délka hrany není celočíselně dělitelnou hodnotou, musí se všechny délky hran vhodně vynásobit transformační konstantou $r \in \mathbb{R}$, aby délky hran bylo možné rozdělit $d' = rd \in \mathbb{N}$.

Dijkstrův algoritmus

Dijkstrův algoritmus se používá pro hledání nejkratší cesty v orientovaném grafu s nezáporným ohodnocením hran. Algoritmus prochází postupně graf od počátečního vrcholu s směrem k cílovému c. Pro každý vrchol v_i je vždy počítána délka nejkratší trasy z počátečního vrcholu s a tato minimální hodnota je mu přiřazena. Navíc pro

zjištění nejkratší trasy je třeba si u každého vrcholu pamatovat předchozí vrchol, kterým se lze dostat z s do v po nejkratší trase. Zpětným vyčtením předchozích vrcholů od cílového c do počátečního s lze takto vyčíst množinu vrcholů nejkratší trasy.

Postup algoritmu lze zapsat následovně:

- inicializace
  - zadání počátečního s a cílového c vrcholu grafu
  - nastavení vzdálenosti pro počáteční vrchol $d(s, s) = 0$ a pro ostatní vrcholy $d(s, v) = \infty$
  - nastavení přiřazeného předchozího vrcholu v_{i-1} na neznámou hodnotu pro každý vrchol v_i
- vytvoření fronty a naplnění všemi vrcholy grafu
- iterace
  1. z fronty vrcholů se vybere vrchol v s nejmenší hodnotou $d(s, v)$, v prvním kroku to je vrchol s
  2. procházení všech sousedních vrcholů vrcholu v
     (a) relaxace - pokud součet hodnoty z vybraného vrcholu v a délky hrany do sousedícího vrcholu w je menší než hodnota v daném sousedícím vrcholu $d(s, w)$, pak se tato hodnota nahradí tímto součtem $d(s, w) = d(s, v) + d(v, w)$ (ověření trojúhelníkové nerovnosti)
     (b) pokud je hodnota vzdálenost $d(s, w)$ nahrazena novou, pak se k danému vrcholu w přiřadí předchozí prvek v (tj. $w_{i-1} = v$)
  3. pokud je fronta vrcholů prázdná, pak je algoritmus ukončen, jinak znovu krok 1
- výsledná nejkratší vzdálenost $d(s, c)$ je přiřazena k cílovému vrcholu c
- trasa nejkratší cesty se získá postupným zpětným čtením přiřazených vrcholů v_{i-1} od vrcholu c do s

Algoritmus dosahuje výsledku v konečném počtu kroků. Každý nedosažitelný vrchol z počátku s je po skončení iterace nastaven na hodnotu $d(s, v) = \infty$.

Při implementaci fronty vrcholů jako pole je celková časová náročnost algoritmu rovna hodnotě $O(|V|^2 + |E|)$, kde $|V|$ je počet vrcholů a $|E|$ počet hran grafu. V případě použití binární haldy je celkový čas $O((|E| + |V|) \log |V|)$.

Floyd-Warshallův algoritmus

Tento algoritmus je určen pro orientovaný graf s nezápornými cykly. Výhodou je nalezení nejkratší trasy mezi všemi dvojicemi vrcholů a v případě cest stejné délky zvolí trasu s nejmenším počtem hran.

Postupnou iterací je upravována matice vzdáleností D^i v každém kroku i. Hodnota $d^i_{u,v}$ vyjadřuje délku nejkratší trasy v podgrafu vzniklého z vrcholů $\{1, 2, \ldots, i\}$. V nulté iteraci obsahuje matice D^i délky přímých hran. Pokud taková hrana neexistuje, je hodnota nastavena ∞. Cílem je algoritmu je získat v konečném počtu kroků $n = |V|$ matici D^n, která již obsahuje vzdálenosti mezi všemi vrcholy.

V každém kroku je porovnávána hodnota vzdálenosti mezi každými dvěma vrcholy u a v a pokud zjistí, že délka cesta přes třetí vrchol w by byla kratší, použije ji.

Postup algoritmu je následující:

- inicializace
  - vytvoření počáteční matice vzdáleností D^0
- iterace od $i = 1$ do $|V|$
  1. procházení všech dvojic vrcholů u a v
     (a) pokud je $d^i_{u,v}$ větší než součet hodnoty vzdálenosti $d^{i-1}_{u,i} + d^{i-1}_{i,v}$, pak je tato hodnota nahrazena daným součtem

Po ukončení algoritmu nejkratší vzdálenost mezi vrcholy u a v je zapsána v matici jako hodnota $d^n_{u,v}$. V případě, že je potřeba vědět i trasa vedoucí mezi těmito vrcholy, je třeba si v algoritmu tyto hodnoty ukládat podobně jako u Dijkstrova algoritmu.

Výhodou je tedy velmi snadné použití. Nevýhodou je, že se výpočetní náročnost rychle zvyšuje s přibývajícím počtem vrcholů. Celková časová náročnost algoritmu je dána $O(|V|^3)$.

Bellman-Fordův algoritmus

Bellman-Fordův algoritmus nalezne nejkratší trasu v ohodnoceném grafu, který může obsahovat i záporně ohodnocené hrany.

Algoritmus je podobný Dijkstrově algoritmu: Na začátku přiřadíme všem vrcholům grafu hodnotu vzdálenosti $d(s, v) = \infty$ a počátečnímu $d(s, s) = 0$. V každém kroku iterace se provádí operace relaxace. Porovnává se součet vzdálenosti vrcholu u a cenou hrany $d(s, u) + c(u, v)$ se vzdáleností následujícího vrcholu $d(s, v)$. V případě, že je tento součet menší, pak je přiřazen tomuto vrcholu.

Počet hran z počátečního vrcholu do jakéhokoli jiného může být maximálně $|U| - 1$, algoritmus je tedy spuštěn právě $|U| - 1$ krát. Po tomto počtu iterací již musí být nalezeny všechny trasy z počátečního vrcholu.

Posledním krokem je ověření přítomnosti záporného cyklu. V případě, že po dalším spuštění algoritmu dojde k jakékoli relaxaci hran, znamená to přítomnost záporného cyklu v grafu. Pokud k žádné další relaxaci nedojde, jsou výsledky relevantní k použití.

Výsledné hodnoty nejkratších délek i tras se vyčtou podobně jako u Dijkstrově algoritmu.

Časová náročnost je $O(|U| \cdot |H|)$, protože vnější smyčka proběhne právě $|U| - 1$ krát a vnitřní smyčka probíhá přes všechny hrany grafu.

3 PRAKTICKÁ ČÁST

Celá práce je implementována v komerčním systému MATLAB vyvíjený společností MathWorks. Toto vývojové prostředí umožňuje pomocí skriptovacího jazyka s jednoduchou syntaxí rychle vyvinout a odladit program s požadovanou funkcí.

Součástí programu MATLAB je knihovna funkcí Image Processing Toolbox, která poskytuje sadu grafických nástrojů pro zpracování, analýzu, vizualizaci a algoritmizaci obrazů [11].

3.1 Zdroj a formát obrázku

Jako výchozí zdroj obrázků byl vybrán server Mapy.cz. Turistický mapový podklad pro zoom 5–13 je dodáván od vydavatelství (SHOCart, spol. s r. o. (viz nápověda na [4]). Zdroj map byl vybrán na základě velkého spektra barev (lepší rozlišení jednotlivých objektů) vyskytujících se na mapě. Pomocí ovládacích prvků na stránkách lze vypnout vybrané rušivé prvky (některé značky turistických a přírodních zajímavostí, turistické známky apod.). Také je možné vypnout či zapnout zobrazení cyklotras a turistických tras.

Export z webové stránky je možný pouze do formátu JPEG. Tento formát se ukázal jako nevhodný z důvodu velké ztrátové komprese dat. Byla tedy použita funkce sejmutí obrazu zobrazeného na obrazovce počítače (tzv. Print Screen). Obraz se načte do schránky operačního systému a poté je jej možné vložit do vybraného editoru obrázků. Při ukládání obrázků byl použit bezeztrátový formát PNG.

V programu MATLAB je soubor barevného obrázku načten funkcí `imread`. Proměnná reprezentující obraz je třírozměrná matice typu *uint8* (celočíselný formát bez znaménka, 8 bitů) a velikosti $m \times n \times 3$, kde m je počet pixelů ve sloupci a n v řádku obrazu. Každá vrstva matice odpovídá barevnému zastoupení červené, zelené a modré barvy v obrazu.

3.2 Předzpracování trasy

3.2.1 Segmentace trasy, [7]

Turistické trasy jsou v obraze segmentovány v barevném prostoru YCbCr, do kterého je možné RGB obraz převést pomocí funkce `rgb2ycbcr`. První vrstva reprezentuje jas (Y) a zbylé vrstvy chrominance modré a červené komponenty (Cb a Cr). Pro segmentaci jsou dále použity vrstvy modré a červené složky.

Fialová barva značící cyklistickou stezku je segmentována v barevném prostoru HSV. Převod z RGB do HSV je zajištěn funkcí `rgb2hsv`, Jednotlivé vrstvy reprezentují barevný tón (Hue), sytost barvy (Saturation) a hodnotu jasu (Value). Pro další zpracování je vybrána vrstva s barevným tónem.

Pro segmentaci obrazu je vhodná funkce `roicolor`. Funkce vybere oblasti ze vstupního obrazu $f(x, y)$ podle rovnice 3.1, kde f_{low} a f_{high} jsou dolní a horní meze hodnot pro výběr oblastí zájmu [6]. Jedná se tedy o prahování obrazu s dvěma prahy. Výsledný obraz $g(x, y)$ je tedy binární (černobílý) s jednou vrstvou (tj. 2D).

$$g(x, y) = \begin{cases} 1, & f(x, y) \geq f_{low} \wedge f(x, y) \leq f_{high} \\ 0, & \text{jinak} \end{cases} \tag{3.1}$$

Operaci segmentace je možné aplikovat na obraz v jakémkoli barevném prostoru (RGB, HSV, YCrCb, atd.). Zkoušením byla vybrána nejlepší metoda i koeficienty pro segmentaci jednotlivých tras (různé barvy turistických tras, cyklostezka, silnice apod.). Např. pro fialovou barvu cyklostezky se ukázala být nejvhodnější segmentace z vrstvy H barevného prostoru HSV s hodnotami $0{,}798 \pm 0{,}01$. Příklad segmentace cyklostezky je na ilustraci 3.1.

Obr. 3.1: Příklad segmentované části obrázku (cyklostezka).

3.2.2 Morfologické operace, [7]

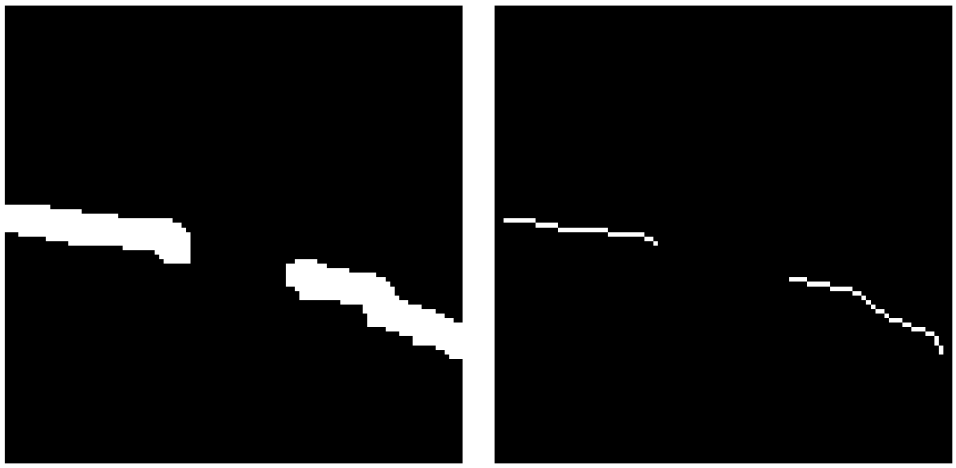
V segmentovaném obraze se vyskytují nežádoucí šumové prvky bodového charakteru. Pomocí vhodného filtrování jsou odstraněny. Morfologická operace *majority* nastaví středový pixel na 1, pokud čtyři nebo více pixelů v 8-okolí jsou 1, jinak je nastaven na 0.

Následným aplikováním operace dilatace (*dilate*) se sníží počet detailů v trase (křižovatky apod.) a spojí se některé blízko rozpojené trasy.

Dalším krokem je použití morfologické operace ztenčování (*thin*). Operace iterativně ztenčuje trasu na šířku 1 pixelu do doby ustálení výsledku. Metoda zachovává Eulerovo číslo (počet oblastí i děr objektu).

Přesný popis použitých morfologických algoritmů je popsán v nápovědě pro funkci `bwmorph`.

Příklad výsledku aplikace morfologických operací na segmentovanou část obrazu je na ilustraci 3.2.



Obr. 3.2: Příklad části obrázku s aplikovanými morfologickými operacemi (vlevo dilatace, vpravo ztenčování).

3.2.3 Spojení přerušených tras

Pro spojení přerušených tras jsou vytvořeny 2 funkce: spojení dvou bodů pomocí úsečky a spojení bodu s okolní trasou pomocí distanční mapy.

Podstatným rozdílem mezi funkcemi je ten, že první algoritmus spojuje bod s jiným koncovým bodem. Naopak druhý algoritmus napojuje koncový bod s nejbližší jinou trasou nebo bodem. Pro obě funkce lze definovat meze vzdáleností spojení bodů.

Nalezení koncových bodů trasy

Před samotným spojováním bodů je potřeba nalézt v obraze koncové body tras, které jsou potencionálně přerušené.

Aplikací morfologické operace `bwmorph` s parametrem *'endpoints'* je získán binární obraz s vyznačenými koncovými body. Protože tato operace vyhledá i body u okraje obrazu, pro odstranění těchto krajních bodů je obraz vynásoben maskou.

Souřadnice koncových bodů je možné získat pomocí `find`:

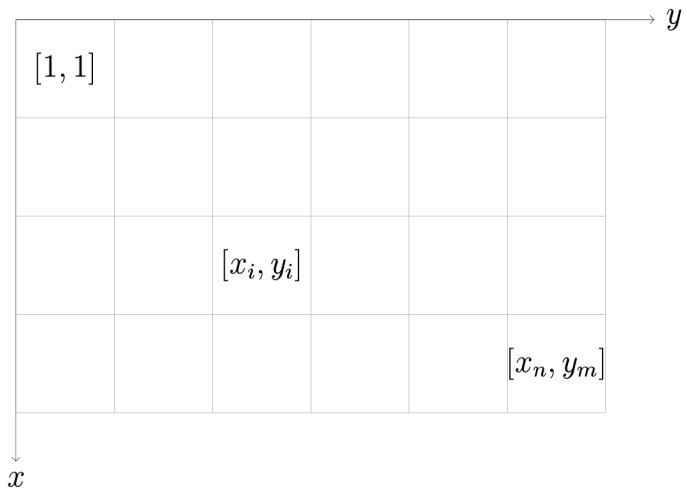
```
[x, y] = find(bw_end_points);
```

, kde *bw_end_points* je binární obraz s koncovými body.

Zjištění koncových bodů (popřípadě bodů na křížících se trasách) je implementováno ve funkci `getPoints`:

```
points = getPoints(bw,type,border);
```

, kde *bw* je binární obraz, *type* je znakový řetězec *'endpoints'* pro koncové body nebo *'branchpoints'* pro body na křížících se trasách a *border* je šířka okraje masky (potlačení okrajových bodů). Souřadnice bodů *points* jsou ve tvaru `[x,y]`, tj. číslo řádku a číslo sloupce v matici obrazu (kladná celá čísla), viz ilustrace 3.3.



Obr. 3.3: Souřadnicová síť při určování polohy bodu v obraze.

Spojení dvou bodů pomocí úsečky

Pokud je nalezeno více než jeden koncový bod v obraze, vytvoří se matice vzájemných vzdáleností všech bodů pomocí funkce `dist` s využitím Euklidovy metriky.

Z této matice jsou následně vybírány dvojice bodů, jejichž vzájemná vzdálenost je menší než definovaný koeficient. Daný práh je experimentálně určený.

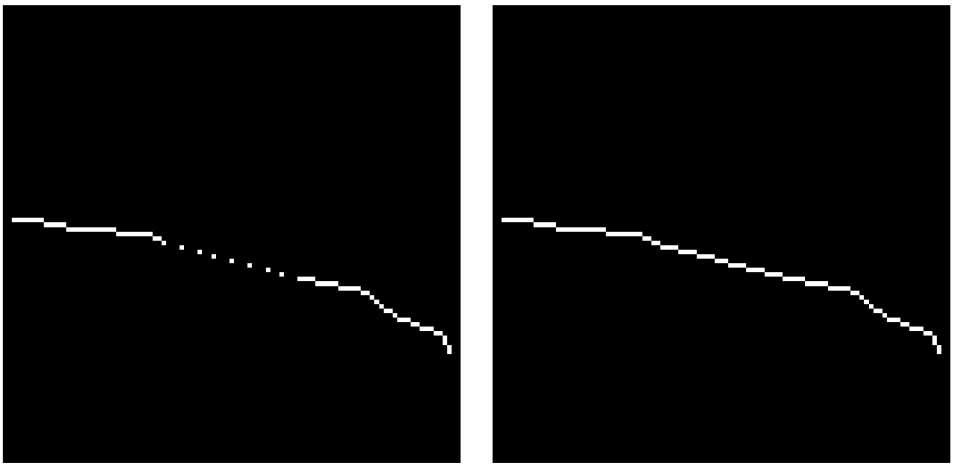
Souřadnice dvojice bodů $[x_1, y_1]$ a $[x_2, y_2]$ jsou předány funkci `makeLine`, která vypočte souřadnice bodů spojující koncové body pomocí úsečky (polynomu prvního řádu). Obecný tvar přímky je

$$y = ax + b \tag{3.2}$$

Neznámé koeficienty a a b je pak možné vypočíst úpravou následujícího vztahu

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \tag{3.3}$$

Problémem při vytvoření úsečky je diskrétní interpretace souřadnic obrazu. Body úsečky je nutné aproximovat na celá čísla, zde pomocí zaokrouhlování k nejbližšímu celému číslu `round`.



Obr. 3.4: Příklad špatně (vlevo) a správně (vpravo) spojené trasy.

Pokud je rozdíl koncových bodů v x souřadnicích menší než rozdíl v y souřadnicích, pak je možné, že vznikne chyba při zaokrouhlování výsledné úsečky (přerušovaná úsečka). Proto se v tomto případě provede záměna souřadnic a po vypočtení aproximační úsečky se souřadnice zpět zamění. Příklad nutnosti záměny souřadnicových os je na obrázku 3.4.

Spojení definovaných bodů v binárním obraze *bw* pomocí úsečky je realizováno funkcí `makeConnectionLine`:

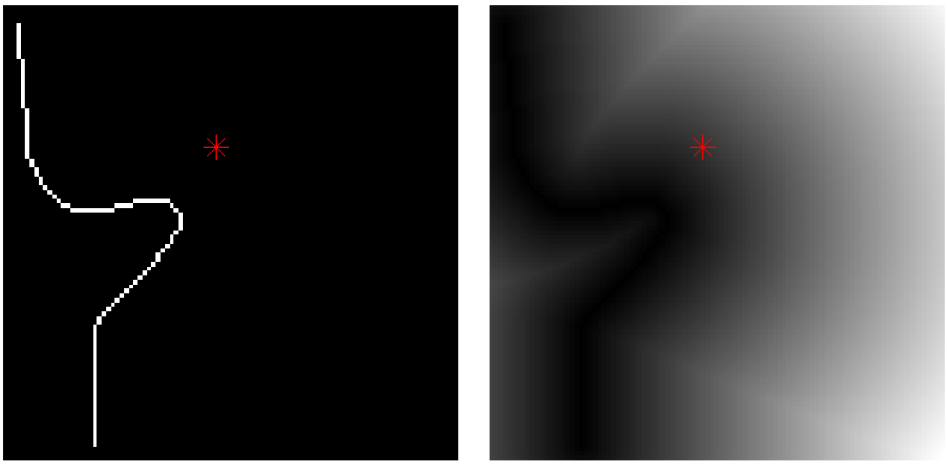
```
bw = makeConnectionLine(bw,points,thresh);
```

, kde *points* jsou souřadnice nalezených bodů v obraze a *thresh* je maximální vzdálenost dvou bodů.

Spojení přerušené trasy pomocí distanční mapy

Ze seznamu bodů se postupně vybírají jednotlivé body a definované okolí bodu v obraze se zkopíruje do pomocné matice. Pomocí funkce `bwlabel` (označení samostatných objektů) se získají souřadnice trasy náležející danému bodu. Trasa je zároveň z matice odstraněna a z výsledku je vytvořena funkcí `bwdist` distanční mapa s Euklidovou metrikou.

Pokud je v distanční mapě na souřadnicích výchozího bodu hodnota menší než definovaný práh, znamená to, že se poblíž nachází trasa. Tento bod je tedy pravděpodobně přerušen a provede se operace hledání nejkratší trasy.



Obr. 3.5: Příklad distanční mapy (vpravo) pro binární obraz (vlevo) s vyznačeným výchozím bodem.

Funkce `findTrace` postupně prohledává v distanční mapě 8-okolí výchozího bodu a hledá nejmenší hodnotu, tedy směr, kudy je nejkratší vzdálenost k trase. Bod v okolí s minimální hodnotou je následujícím výchozím bodem. Lze si to také představit jako hledání nejstrmější trasy. Postupnou iterací se nakonec dojde k bodu, kde je hodnota v distanční mapě rovna hodnotě 0.

Obr. 3.6: Příklad napojení trasy pomocí distanční mapy.

Celý algoritmus spojení koncových bodů *points* pomocí distanční mapy je implementováno ve funkci `makeConnectionDist`:

```
bw = makeConnectionDist(bw,points,thresh)
```

, kde *bw* je binární obraz a *thresh* je maximální vzdálenost jiné trasy od koncového bodu.

Na obrázku 3.6 je možné vidět příklad výsledku napojení koncového bodu slepé trasy na jinou trasu. Toto napojení by nebylo provedeno pomocí spojení úsečkou z důvodu chybějícího druhého koncového bodu.

3.3 Předzpracování vrstevnic

3.3.1 Segmentace vrstevnic

Pro segmentaci vrstevnic v obraze je použit prostor YCbCr, do kterého se výchozí RGB obraz převede pomocí funkce `rgb2ycbcr`.

Obr. 3.7: Příklad segmentace vrstevnic v obrázku.

Vrstevnice jsou segmentovány ze všech vrstev tohoto prostoru. Pro tuto příležitost byla vytvořena funkce `ycbcr`:

```
bw = ycbcr(rgbImage,YChannel,CbChannel,CrChannel)
```

, kde vstupními argumenty jsou obraz v RGB prostoru *rgbImage* a rozsahy segmentačních hodnot v jednotlivých vrstvách YCbCr prostoru. Poté je pomocí operace průniku vytvořen výsledný binární obraz ze segmentovaných jednotlivých vrstev.

Příklad segmentace vrstevnic v obrázku je na ilustraci 3.7.

3.3.2 Morfologické operace

Pro úpravu segmentovaných vrstevnic je použito morfologické operace *clean*, která odstraní izolované pixely v obraze, tj. bodový šum. Podobně jako u tras, je i zde nutné ztenčení vrstevnic na šířku jednoho pixelu pomocí *thin*.

Obě morfologické operace jsou provedeny pomocí funkce `bwmorph`.

3.3.3 Spojení přerušených vrstevnic

Při spojování přerušených vrstevnic dochází vždy k napojení koncového bodu na jiný koncový bod. Naopak při spojování tras mohlo dojít ke spojení koncového bodu s jiným bodem nebo s trasou (křižovatky). Vrstevnice se tedy nikdy nekříží.

Pro rekonstrukci vrstevnic je využito spojení pomocí úsečky či polynomu. Detekce bodů je pomocí funkce `getPoint`, viz 3.2.3.

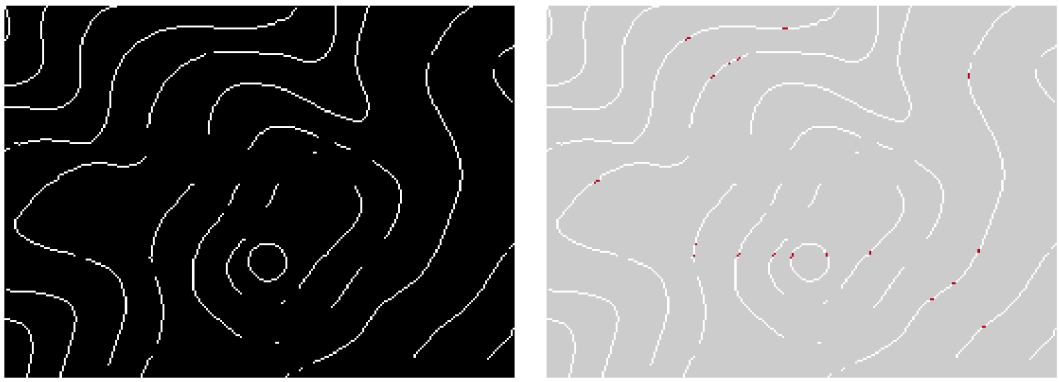
Spojení dvou bodů pomocí úsečky

Spojování dvou koncových bodů vrstevnic pomocí úsečky je realizováno (obdobně jako u tras) pomocí funkce `makeConnectionLine`, viz 3.2.3.

Vrstevnice však často mění směr a při spojování větších mezer může dojít k překřížení vrstevnic. Dále je třeba přihlédnout k jejich hustotě, kdy může být spojení vyhodnoceno špatně. Bylo by nutné zohlednit také směrnice spojovaných bodů.

Spojení bodů vrstevnic pomocí úsečky je tedy využito pro velmi malé mezery.

Ukázka výsledku rekonstrukce částí vrstevnic pomocí úsečky je na ilustraci 3.8.



Obr. 3.8: Ukázka spojení některých vrstevnic úsečkami (vlevo) a vyznačení spojených částí (vpravo).

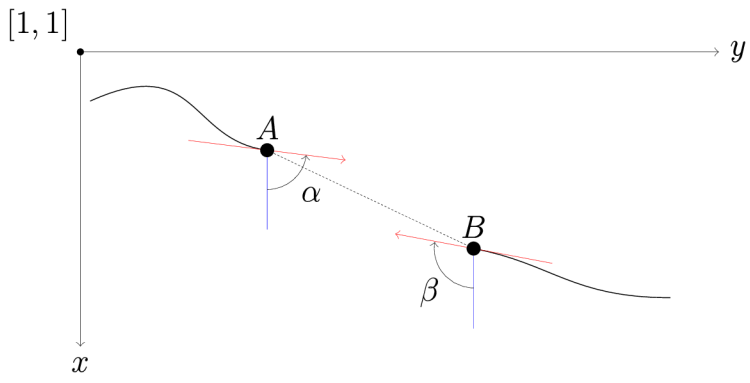
Spojení dvou bodů pomocí polynomu

Při větší vzdálenosti přerušených částí vrstevnic je spojení dvou bodů pomocí úsečky nepřesné. Lepší výsledky podává spojení pomocí polynomu. Zde je využito výpočtu polynomu 3. řádu (kubický polynom).

Před samotným spojením dvou bodů $A = [x_1, y_1]$ a $B = [x_2, y_2]$ polynomem je potřeba vyhodnotit vhodnost jejich spojení. Celkem byly vytvořeny 4 podmínky:

1. vzájemná vzdálenost
2. vzájemná směrnice
3. vzájemná poloha

4. zachování směrnice



Obr. 3.9: Určení směrnic α a β koncových bodů A a B v obraze.

První podmínka uvažuje, že vzájemná vzdálenost $|AB|$ koncových bodů musí být menší nebo rovna než definovaný práh d. Zároveň oba body spolu nesousedí, tj. vzájemná vzdálenost je větší než $\sqrt{2}$. Body musí tedy vyhovovat následující podmínce:

$$|AB| \leq d \quad \wedge \quad |AB| > \sqrt{2} \tag{3.4}$$

Výpočet vzdálenosti všech bodů Euklidovou metrikou je pomocí funkce `dist`:

$$|AB| = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{3.5}$$

Podmínka číslo dvě určuje, že směrnice α a β obou koncových bodů jsou vzájemně pootočeny o π rad s tolerancí $\pm\varepsilon$.

$$|\alpha - \beta\ | \leq \pi + \varepsilon \quad \wedge \quad |\alpha - \beta\ | \geq \pi - \varepsilon \tag{3.6}$$

Směrnice koncového bodu v radiánech se vypočte zkopírováním okolí bodu do matice a jejím předáním funkci `getTangent`.

Třetí podmínka sleduje jak vzájemnou polohu obou bodů, tak i jejich směrnice:

$$\left(|\alpha| \leq \frac{\pi}{2} \quad \wedge \quad x_1 \leq x_2\right) \quad \vee \quad \left(|\alpha| > \frac{\pi}{2} \quad \wedge \quad x_1 > x_2\right) \tag{3.7}$$

$$(\operatorname{sgn} \alpha \geq 0 \quad \wedge \quad y_1 \leq y_2) \quad \vee \quad (\operatorname{sgn} \alpha < 0 \quad \wedge \quad y_1 > y_2) \tag{3.8}$$

Ke správnému spojení dvou bodů polynomem musí být oba uvedené vztahy pravdivé.

Poslední podmínka uvažuje zachování směrnice při vytvoření polynomu. Nejdříve se vypočte vzájemná směrnice obou bodů γ

$$\gamma = \arctan\left(\frac{y_2 - y_1}{x_2 - x_1}\right) \tag{3.9}$$

, která je porovnána se směrnicí α:

$$|\alpha - \gamma| \leq \frac{\varepsilon}{2} \quad \wedge \quad |\alpha - \gamma| \geq -\frac{\varepsilon}{2} \tag{3.10}$$

Samotný výpočet bodů polynomu je realizován ve funkci `makePoly`. Obecná rovnice polynomu třetího řádu je

$$y = ax^3 + bx^2 + cx + d \tag{3.11}$$

první derivace této funkce podle x pak je

$$y' = 3ax^2 + 2bx + c \tag{3.12}$$

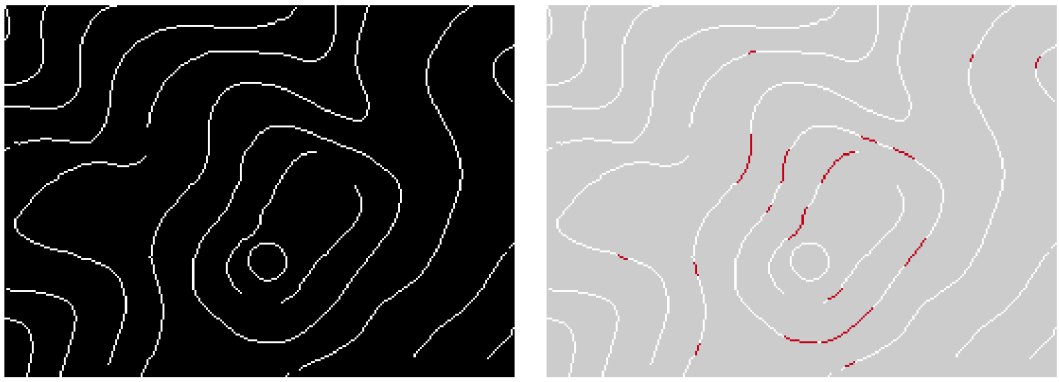
Přepočet směrnic α a β v rad na první derivace y_1' a y_2' se provede pomocí tangenty

$$\begin{aligned} y_1' &= \tan\alpha \\ y_2' &= \tan\beta \end{aligned} \tag{3.13}$$

Pomocí souřadnic a derivací koncových bodů jsou známy celkem 6 proměnných $[x_1, y_1, y_1']$ a $[x_2, y_2, y_2']$ pro výpočet koeficientů rovnice polynomu $[a, b, c, d]$:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_1' \\ y_2' \end{bmatrix} = \begin{bmatrix} {x_1}^3 & {x_1}^2 & x_1 & 1 \\ {x_2}^3 & {x_2}^2 & x_2 & 1 \\ 3{x_1}^2 & 2x_1 & 1 & 0 \\ 3{x_2}^2 & 2x_2 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \tag{3.14}$$

Podobně jako u funkce `makeLine`, i zde je potřeba záměny souřadnic, pokud je rozdíl koncových bodů v x souřadnicích menší než rozdíl v y souřadnicích. V tomto případě je zapotřebí i přepočítat derivace bodů (inverzní funkce).



Obr. 3.10: Spojení některých vrstevnic polynomy (vlevo) a vyznačení spojených částí (vpravo).

Algoritmus vyhodnocení a následného spojení bodů *points* v binárním obraze *bw* je realizován ve funkci `makeConnectionPoly`:

```
bw = makeConnectionPoly(bw, points, thresh, eps);
```

, kde *thresh* je práh d maximální vzájemné vzdálenosti dvou bodů a *eps* představuje toleranci směrnic $\pm\varepsilon$.

Ukázka spojení některých vrstevnic polynomy třetího řádu je na ilustraci 3.10. Z ilustrace je patrné, že ne všechny vrstevnice jsou spojeny, záleží totiž na nastavení vstupních argumentů *thresh* a *eps*. Ovlivňujícími koeficienty konkrétní mapy jsou hustota vrstevnic, míra změny směru vrstevnic a míra poškozených částí. Je účinnější začít spojovat vrstevnice s přísnějšími podmínkami (nízké hodnoty tolerance *thresh* a *eps*) a postupně z podmínek slevovat.

K dalšímu spojování vrstevnic dochází až při jejich výškovém ohodnocení. Viz 3.4.1.

3.4 Vytvoření výškové mapy

V následujících kapitolách jsou částečně použity postupy popsané v bakalářské práci [12], která byla realizována v programovacím jazyce C/C++ s knihovnou pro zpracování obrazu OpenCV.

K vytvoření výškové mapy je potřeba znát ve vstupním obraze referenční bod. V tomto případě se ukázala nejvhodnější identifikace vrcholu hory či kopce, kde je uvedena výšková hodnota v metrech nad mořem (m n. m.). Od tohoto bodu jsou postupně distribuovány výškové hodnoty do okolních vrstevnic.

V programu není řešena identifikace tohoto výchozího bodu a musí ji provést uživatel, tj. zadat souřadnice a výšku výchozího bodu $P[x, y, z]$.

3.4.1 Přiřazení výšky vrstevnicím

Distribuce výšek v okolí výchozího bodu

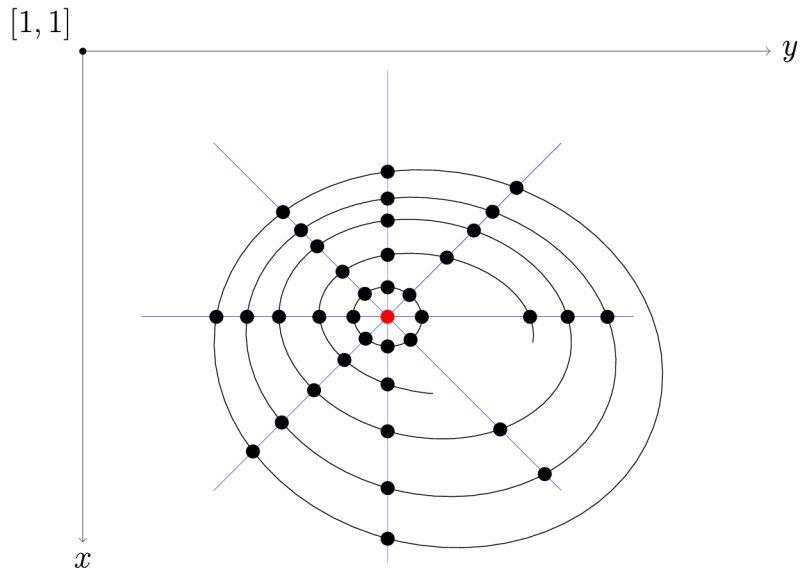
Předpokladem je, že výchozí bod $P[x, y, z]$ je vrchol kopce či hory, od kterého výšková hodnota do okolí monotónně klesá. Z tohoto bodu jsou sestrojeny úsečky délky d v určitém úhlovém rozestupu φ, viz ilustrace 3.11.

Pro každý úhel φ je vypočten druhý koncový bod *pt2* úsečky pomocí funkce `getSecondPoint`:

```
pt2 = getSecondPoint(pt1,phi,length_line,size_matrix)
```

, kde *pt1* jsou souřadnice výchozího bodu, *phi* je úhlové otočení φ, *length_line* je délka úsečky. Kvůli omezené velikosti obrazu je potřeba i vstupní argument *size_matrix*.

Vytvořené úsečky se postupně protínají v bodech okolních vrstevnic. Souřadnice těchto bodů jsou získány funkcí `intersectWithLine`:



Obr. 3.11: Průsečíky úseček z vrcholu kopce a vrstevnic v okolí.

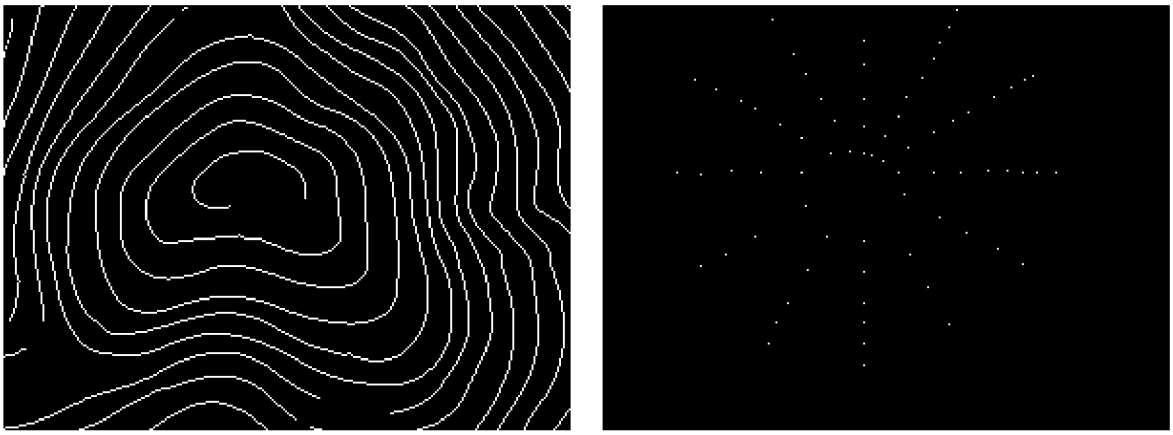
```
points = intersectWithLine(pt1,pt2,bw_in);
```

, kde *pt1* a *pt2* jsou souřadnice koncových bodů úsečky a *bw_in* je vstupní binární obraz předzpracovaných vrstevnic. Návratová hodnota *points* jsou souřadnice všech bodů vrstevnic protínajících se s úsečkou.

Vlastní funkce `intersectWithLine` vytvoří do pomocné matice stejné velikosti jako výchozí obraz úsečku z bodu *pt1* do bodu *pt2* pomocí funkce `makeLine`. Jelikož v počítačové grafice průnik dvou diskrétních přímek může být nulový, je třeba vytvořenou úsečku dilatovat. Následně je proveden průnik pomocné matice s dilatovanou úsečkou a vstupním obrazem vrstevnic. Tyto průsečíky mohou být z důvodu dilatace úsečky částí křivky. Samostatné body z nich vytvoříme aplikací morfologické operace ztenčování *shrink*. Body jsou poté seřazeny podle vzdálenosti od výchozího bodu *pt1*.

Získaným bodům jsou postupně vypočteny výškové hodnoty. První nejbližší bod na vrstevnici od vrcholu je nejbližší nižší hodnota násobku 10, např. pro vrchol 562 m n. m. je nejbližší nižší hodnota 560 m n. m. Výškové rozdíly všech vrstevnic jsou po 10-ti metrech (dáno zdrojem konkrétní použité mapy). Následující hodnoty průsečíků jsou tedy vždy o číslo 10 nižší.

Tato distribuce výškových hodnot do okolí vrcholu je implementována ve funkci `distributePeaks`. Ukázka přiřazení výškových hodnot bodům průsečíku úseček z vrcholu a okolních vrstevnic je na ilustraci 3.12, kde jednotlivé barvy odpovídají výškovým hodnotám.



Obr. 3.12: Distribuce výškových hodnot z výchozího vrcholu do okolních vrstevnic.

Tímto postupem jsou ohodnoceny pouze některé body nejbližších vrstevnic. Dalším krokem je vyhodnocení výšky jednotlivých vrstevnic. Tento postup zajišťuje funkce `evaluateHeight`, kde jsou postupně vybírány vrstevnice, které obsahují určitý počet nenulových výškových hodnot. Pro tyto hodnoty je vypočten modus (tj. nejčetnější hodnota) a pokud se ten vyskytuje v hodnotách více než n-krát, pak je celé vrstevnici přiřazena tato hodnota.

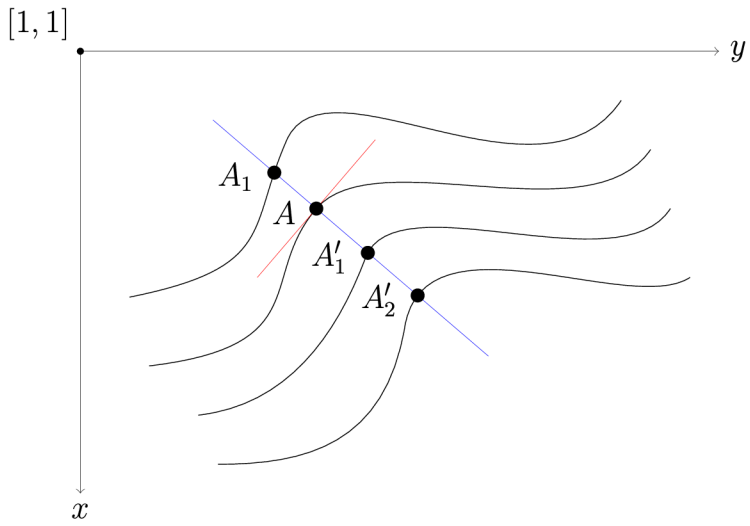
Nyní se algoritmus pokusí o spojení některých vrstevnic na základě výškových hodnot. Tyto vrstevnice nemusely být správně identifikovány v některém z kroků v předzpracování. Algoritmus je totožný jako při předzpracování vrstevnic a využívá funkce `getPoints`, `makeConnectionPoly` a `makeConnectionLine`, avšak vyhodnocuje pouze vrstevnice se stejnou výškovou hodnotou.

Distribuce výšek do ostatních vrstevnic

V určité vzdálenosti od výchozího bodu již předchozí metoda distribuce výšek může selhat. Může se totiž změnit směr vrstevnic a tyto by byly nesprávně vyhodnoceny. Proto je potřeba v dalším ohodnocování vrstevnic změnit taktiku.

Pro distribuci výšek mezi vrstevnicemi je dále využito úseček kolmých k jednotlivým bodům již ohodnocených vrstevnic. Tyto normály vrstevnic protínají některé nejbližší vrstevnice a jednotlivým průsečíkům jsou poté přiřazovány výškové hodnoty. Principiální schéma metody je na ilustraci 3.13.

Prvním krokem je zjištění směrnice bodu, ke kterému se dále vytváří normála. Pomocí funkce `getTangent` je vypočtena hodnota směrnice a k ní je přičtena konstanta $\pi/2$ z důvodu pootočení tečny na normálu. Nyní je možné vypočíst dva koncové body úsečky ve vzdálenosti d pomocí dvojího volání funkce `getSecondPoint` s opačnými úhly φ a $\varphi - \pi$. Souřadnice průsečíků normály a okolních vrstevnic jsou získány z funkce `intersectWithLine`, podobně jako v distribuci výšek z vrcholu.



Obr. 3.13: Určení průsečíků vrstevnic a normály bodu A.

Z výškové mapy je nyní možné vyčíst vektor hodnot $V = [v_1, ..., v_i, ..., v_n]$ vyskytujících se na souřadnicích průsečíků. Tento vektor je složen z nul a násobků 10-ti. Některé hodnoty se mohou opakovat nebo celý vektor může být nulový. Je potřeba odhadnout trend klesání či stoupání výšky a doplnit do vektoru nenulové hodnoty namísto nulových. Ke správnému odhadu trendu je nutné znát minimálně dvě nenulové hodnoty.

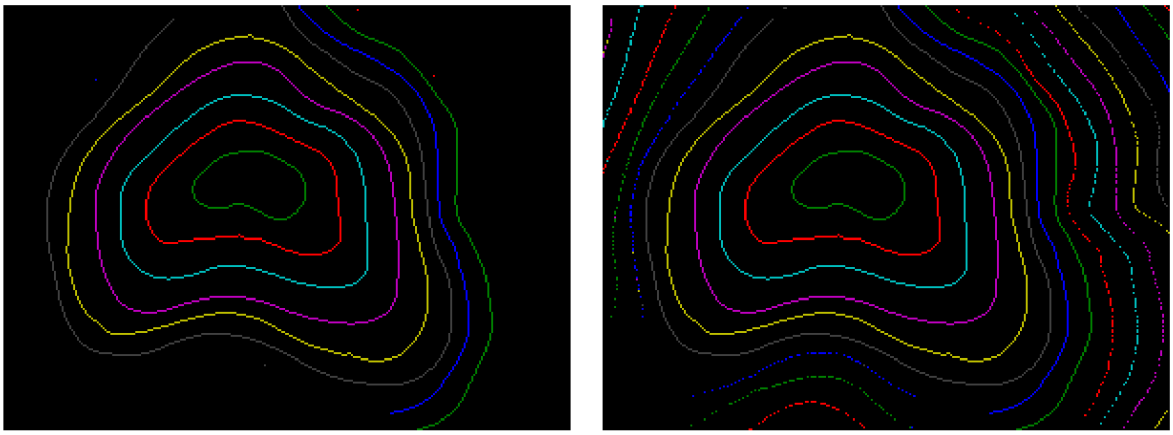
Hodnotu v_i je možné doplnit na základě nenulových dvou horních sousedních hodnot v_{i+1}, v_{i+2} či dvou dolních sousedních hodnot v_{i-1}, v_{i-2}. Druhou možností je doplnit v_i podle sousedních nenulových hodnot z obou stran v_{i-1}, v_{i+1}. Algoritmus se může vícekrát opakovat do té doby, dokud není jasné, jaké hodnoty by se měly doplnit. Je nutné však splnit, aby rozdíl dvou sousedních nenulových hodnot byl 10, -10 či 0 (při opakování hodnoty).

Algoritmus výpočtu výškových hodnot ve vektoru je implementován ve funkci `fillValues`. Vypočtené hodnoty jsou zpět doplněny na původní souřadnice do výškové mapy.

Je zde nutné říci, že se zvyšujícím se počtem bodů vrstevnice je třeba přímo úměrně delší čas k ohodnocení okolních vrstevnic. Proto se v tomto algoritmu pro urychlení výpočtu neprocházejí všechny body vrstevnice, ale vypočte se přiměřený krok n a poté se procházejí jen n-té body.

Na ilustraci 3.14 lze vlevo vidět vyhodnocené a pospojované vrstevnice v okolí výchozího bodu (kopce) a vpravo distribuce výškových hodnot pomocí normál v první iteraci.

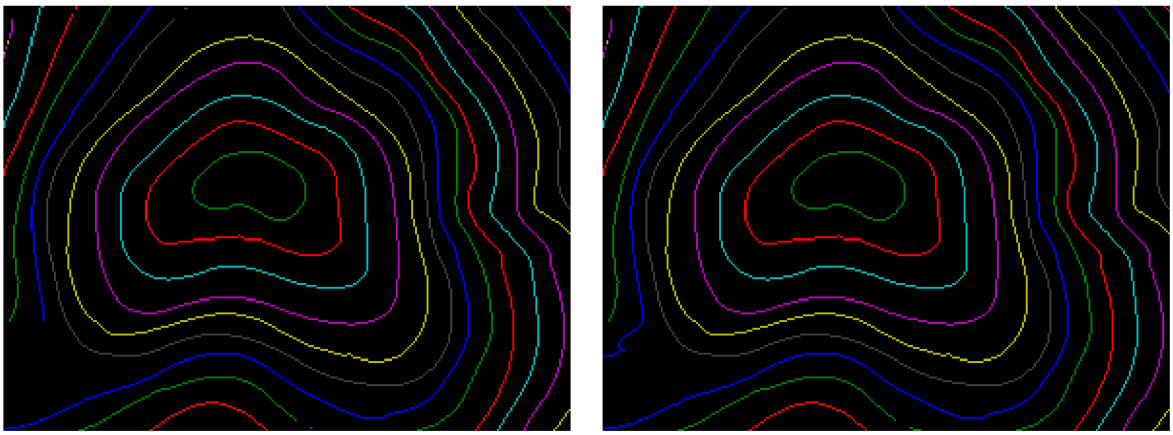
Po distribuci výškových bodů do okolí je znovu volána funkce `evaluateHeight`,



Obr. 3.14: Distribuce výškových hodnot z ohodnocených vrstevnic do okolních pomocí normál (první iterace).

kde se vyhodnotí výšky některých vrstevnic pomocí operace modus. Dále je zde znovu pokus o spojení vrstevnic se stejnou výškovou hodnotou pomocí `connectHeight`.

Ukázka výsledku vyhodnocení a spojení vrstevnic po první iteraci je na ilustraci 3.15 vlevo a výsledek distribuce výšek pomocí normál po druhé iteraci je na stejné ilustraci vpravo.



Obr. 3.15: Distribuce výškových hodnot z ohodnocených vrstevnic do okolních pomocí normál (druhá iterace).

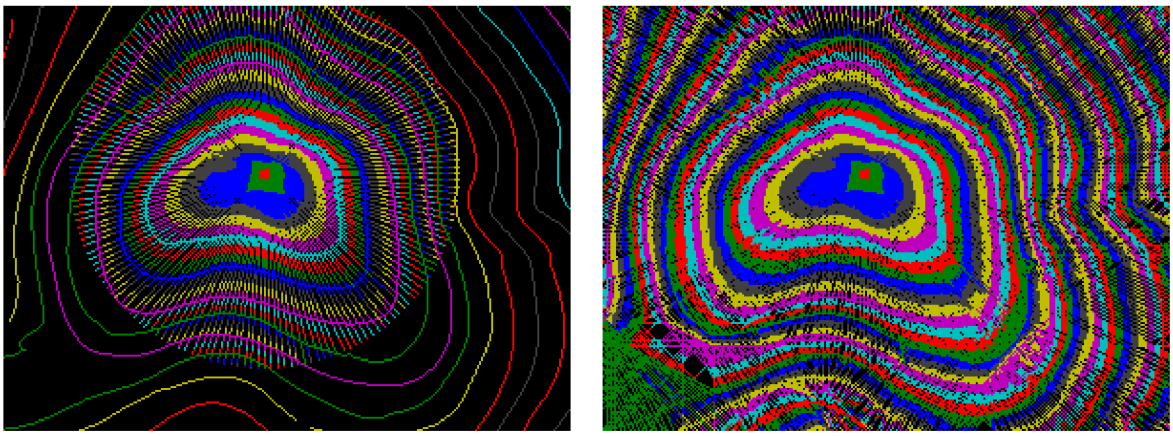
3.4.2 Výpočet výšky mezi vrstevnicemi

Po vyhodnocení výškových hodnot všech vrstevnic, přichází nyní na řadu výpočet výšek mezi vrstevnicemi. Postup je principiálně podobný, jako v předchozím kroku.

Předně se kolem výchozího vrcholu (kopec, hora) vysílají do různých malých úhlů φ úsečky definované délky d. Na rozdíl od předchozí metody, kdy se vyčítaly

a zpětně dosazovaly hodnoty na průsečících úsečky a vrstevnic, zde se vyhodnocuje celá délka úsečky. Pomocí již zmíněných funkcí `getSecondPoint` a `makeLine` se získají souřadnice úsečky a z výškové mapy se vyčtou hodnoty *values* na celé délce úsečky.

Pokud jsou nejméně dva body úsečky ohodnoceny, lze odhadnout trend klesání či stoupání terénu. V tomto případě se výborně hodí interpolace 1-D dat, která je implementována ve funkci `interp1`. Je zde použita lineární metoda interpolace i extrapolace dat. Vypočtené hodnoty se zpět dosadí do výškové mapy. Výsledek distribuce výškových hodnot do okolí vrcholu lze vidět na ilustraci 3.16 vlevo.

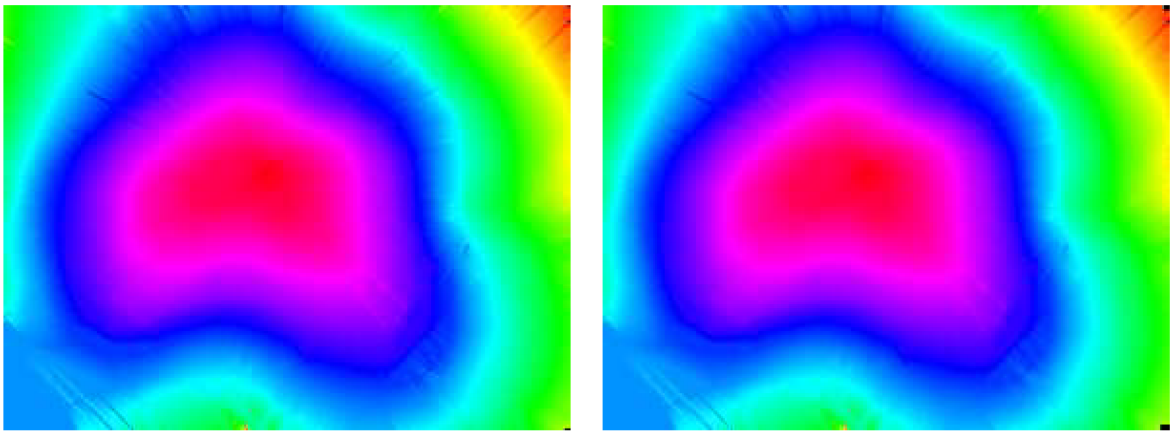


Obr. 3.16: Výpočet výškových hodnot v okolí výchozího bodu (vlevo) a mezi vrstevnicemi (vpravo).

Dále se pokračuje metodou úseček kolmých k jednotlivým bodům vrstevnic. Tentokrát se však již procházejí všechny body z důvodu co největšího pokrytí výškové mapy. Pro výpočet hodnot na normále bodu vrstevnice se zde využívá také interpolace výše zmíněnou funkcí `interp1`.

Na ilustraci 3.16 vpravo je vidět, že obě předchozí metody nemusí pokrýt celý prostor mapy. Je tedy nutné vypočíst zbývající neohodnocené body. Tyto body jsou 2-D interpolovány pomocí funkce `TriScatteredInterp` s parametrem *natural* pro přirozenou interpolaci podle sousedních bodů (angl. Natural neighbor interpolation). Této funkci jsou předány vektory souřadnic X, Y a odpovídající nenulová výšková hodnota Z. Výstupem je interpolant celé výškové mapy. Výsledek výpočtu v nepravých barvách je na ilustraci 3.17 vlevo.

Posledním krokem při vytvoření výškové mapy je filtrace pomocí 2-D Gaussova filtru, který vyhladí ostré hrany, tj. potlačuje vysoké frekvence v obraze. Nejdříve se vytvoří jádro filtru funkcí `fspecial` s argumenty *'gaussian'*, požadovanou velikostí jádra a hodnotou σ proměnné. Filtr se aplikuje pomocí `imfilter`. Ukázka výsledné výškové mapy po aplikaci Gaussova filtru je na ilustraci 3.17 vpravo.



Obr. 3.17: Výsledek výškové mapy po aplikaci 2-D interpolace (vlevo) a Gaussova filtru (vpravo).

3.5 Nalezení nejkratší trasy

3.5.1 Uživatelem zadané body trasy

Souřadnice požadovaných bodů trasy jsou získány pomocí grafické funkce `ginput`. Výsledku je nutné zaměnit souřadnice a zaokrouhlit je na celá čísla pomocí `round`.

Jelikož není zajištěné, že dané body budou přesně na existující trase, je v tom případě nutné vytvořit trasu k nejbližší existující trase. Algoritmus výpočtu této trasy je založen na vytvoření distanční mapy a hledání nejkratší trasy pomocí funkce `findTrace`, obdobně jako v kapitole o spojení přerušených tras pomocí distanční mapy, viz kapitola 3.2.3.

Získané souřadnice bodů zadaných uživatelem jsou uloženy do seznamu vrcholů grafu. Případná vypočtená část trasy je uložena do binární mapy tras. Výše popsané postupy jsou implementovány ve funkci `addVertexes`.

3.5.2 Vytvoření seznamu vrcholů grafu

Vrcholy grafu představují v předzpracovaném obraze koncové body tras a jejich křižovatky. Požadované souřadnice bodů *points* jsou získány sjednocením výsledků funkce `getPoints` s parametry *'endpoints'* a *'branchpoints'*. Podrobnější popis této funkce viz kapitola 3.2.3.

Následnou aplikací funkce `deleteDuplicates` na seznam výsledných souřadnic bodů jsou odstraněny opakující se body se zachováním pořadí.

Získání seznamu vrcholů grafu a následná kontrola dat je vytvořeno ve funkci `findVertexes`.

3.5.3 Vytvoření seznamu hran grafu

Dále je nutné vytvořit vzdálenostní matici představující vzdálenosti mezi jednotlivými sousedními vrcholy grafu.

Pro každý vrchol se provádí operace trasování. Přesněji řečeno, referenční bod se přesouvá ze startovního bodu (vrchol grafu) na jiný nenulový prvek v 8-okolí a zároveň se předchozí bod vynuluje (aby ze algoritmus nevracel zpět). Jsou zaznamenávány souřadnice prošlé trasy a inkrementována délka s Euklidovou metrikou.

Algoritmus se ukončuje v případě nalezení jiného vrcholu ze seznamu vrcholů.

Je tak získána jak matice vzdáleností mezi dvojicí sousedních vrcholů, tak i souřadnice této trasy. Souřadnice i délka trasy jsou uloženy v matici datového typu *cell* na pozici $\{x, y\}$, kde x je číslo výchozího vrcholu a y je číslo koncového vrcholu hrany grafu. Jelikož jsou trasy neomezené ve směru cestování (neorientovaný graf), je výsledná matice symetrická a tedy hodnoty na pozici $\{x, y\}$ odpovídají i hodnotám na pozici $\{y, x\}$.

Je nutné počítat i s tzv. paralelními hranami, tj. hranami, které mají stejný výchozí i cílový bod. Ve vzdálenostní matici se poté mohou na pozici $\{x, y\}$ vyskytnout více různých hodnot, tento případ je vyřešen pomocí matice typu *cell*, kdy se na danou pozici mohou zapisovat vektorové hodnoty $[d_1, d_2, ...]$.

Výpočet vzdálenostní matice a získání souřadnic hran grafu z předzpracovaného binárního obrazu tras je implementováno ve funkci `findEdges`.

3.5.4 Vyhledání nejkratší trasy v mapě

Jednou z úplné reprezentace grafu je např. matice vzdálenosti. Proto pokud je již nalezena matice vzdáleností sousedních vrcholů (viz předchozí), je možné již použít algoritmus hledání nejkratší trasy v grafu. V tomto případě byl implementován vlastní návrh Dijkstrova algoritmu ve funkci `dijkstra`.

Algoritmus postupně prochází všechny kombinace vrcholů $\{v, w\}$ a hledá nejkratší vzdálenost ze startovního vrcholu s ke zbývajícím w, kterou pak zapisuje do vektoru $d(w)$. Zároveň je vytvořen vektor předchůdců vrcholů $p(w)$ vrcholu w přes které vede nejkratší trasa ze startovního s do jiného vrcholu w.

Po ukončení iterace se z vektoru vzdáleností $d(w)$ vyčte výsledná nejkratší vzdálenost k požadovanému cílovému vrcholu c na pozici $d(c)$.

Požadovaná trasa je zpětně odvozena z předchůdců cílového vrcholu $p(c)$ dokud není předchůdce výchozí vrchol s.

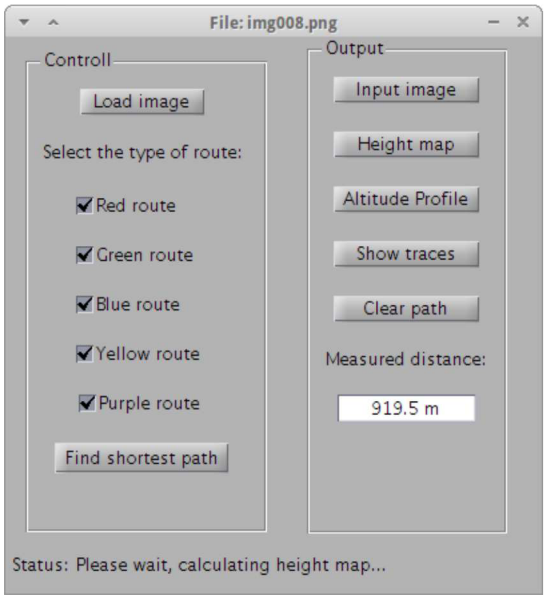
Podrobnější rozbor a popis Dijkstrova algoritmu viz 2.2.1.

3.6 Uživatelské rozhraní

Závěrečnou částí práce je vytvoření grafického uživatelského rozhraní (GUI) pro manipulaci rastrových obrazů map.

Uživatelské rozhraní je možné vytvořit v Matlabu pomocí aplikace `guide` (z angl. zkratky Graphical User Interface Development Environment), kde je možné navrhnout a vytvořit interaktivně design žádané aplikace. Výsledkem návrhu jsou soubory *CV_Maps_GUI.fig* obsahující informace o grafické stránce a *CV_Maps_GUI.m* implementující chování aplikace. Výsledné okno uživatelského rozhraní je na ilustraci 3.18.

Chování aplikace bylo uzpůsobeno jednoduchému stavovému automatu, který přechází mezi jednotlivými stavy na základě událostí. Při spuštění programu se nachází v počátečním stavu 0, kdy zatím uživatel nezvolil požadovaný zpracovávaný obraz turistické mapy. Není možné spustit např. zobrazení mapy, výpočet trasy apod., dokud nenastane načtení obrazu.



Obr. 3.18: Ovládací okno grafického uživatelského prostředí.

Po korektním načtení obrazu se program přesune do stavu 1. Nyní je povoleno zobrazit načtený obraz a je dále možné spustit algoritmus nalezení nejkratší trasy. Před spuštěním algoritmu je uživateli umožněno pomocí zaškrtávacích políček vybrat typ zpracovávané trasy. Na výběr je červená, zelená, modrá a žlutá turistická trasa a fialová cyklistická trasa.

Před spuštěním algoritmu vyhledání nejkratší trasy je nejprve zkontrolováno, zda je vybrána minimálně jedna trasa. Pokud ano, je dále zobrazen originální obrázek mapy a uživatel je vyzván k zadání požadovaných bodů trasy a k potvrzení klávesou

Enter. Program je ošetřen na možnost zadání bodů mimo mapu či zadání pouze jednoho bodu. Poté je již spuštěno samotné zpracování mapy a hledání nejkratší trasy přes zadané body.

Souřadnice tras mezi jednotlivými sousedními vrcholy se získají trasováním. Pomocí Dijkstrova algoritmu je získán seznam vrcholů na trase mezi zadanými dvojicemi uživatelských bodů. Nyní tedy stačí z matice souřadnic tras vyčítat jednotlivé trasy mezi těmito vrcholy a postupně vykreslovat výslednou trasu od startu do cíle. Výsledná celková délka trasy je získána součtem dílčích tras. Pro získání skutečné délky je však nutné číslo vynásobit koeficientem rozlišení 1 pixelu a měřítkem mapy. Po úspěšném nalezení trasy je vypočtená trasa zobrazena do originální mapy i do segmentovaného obrázku. Automat tak přejde do stavu 2.

Pokud je program ve stavu 1 nebo 2 (tj. je načten obrázek mapy), je povoleno spustit výpočet výškové mapy. Uživatel je vyzván k zadání bodu vrcholu kopce a následně jeho výškové hodnoty. Výpočet výškové mapy trvá obvykle poměrně dlouho, proto je oddělen od algoritmu hledání trasy. Po ukončení výpočtu je zobrazena výšková mapa. V případě, že byla výšková mapa daného obrázku již vypočtena dříve, pak se znovu nepočítá.

Získání výškového profilu nalezené trasy je povoleno jen ve stavu 2, kdy jsou souřadnice hledané trasy v mapě již známé. Pokud ještě nebyla vypočtena výšková mapa, algoritmus se spustí nyní. Hodnoty výškového profilu trasy odpovídají výškovým hodnotám na souřadnicích trasy ve výškové mapě.

4 ZÁVĚR

Tato diplomová práce se zabývá grafickým zpracováním turistické mapy pomocí počítačových nástrojů.

První část práce je zaměřena na zpracování mapy jako grafu. Je provedena segmentace vybraných turistických tras či cyklotras a případné opravy přerušených částí. Dalším zpracováním mapy je vytvořena reprezentace tras jako neorientovaný graf s nezáporně ohodnocenými hranami. Ohodnocení hran představuje délku trasy. V programu je použit Dijkstrův algoritmus pro hledání nejkratší trasy mezi uživatelem definovanými body.

Druhá část práce se zabývá výškovým zpracováním mapy. Po segmentaci a restauraci vrstevnic jsou postupně vypočítávány z výchozího bodu (vrchol) výškové hodnoty zbývající části mapy. K tomuto postupu bylo využito distribuce výšek z referenčního bodu do okolních vrstevnic a poté postupná interpolace ostatních bodů.

Celý program je implementován ve vývojovém prostředí MATLAB a z výsledných algoritmů je vytvořena třída *@Map*. Je vytvořeno také grafické uživatelské rozhraní (GUI), které využívá tuto třídu pro zpracování turistické mapy.

Důležitým krokem při vývoji programu je správný výběr zdroje obrázků. Zdroj by měl mít co největší paletu barev, ale zároveň co nejméně nadbytečných informací. Čím více objektů se vyskytuje v obraze, tím obtížnější je segmentace důležitých částí. Např. ve výchozím zdroji obrázků se vyskytují v obraze vrstevnice s výškovým rozdílem po 10 m n. m., ale mohou zde být i přerušované vrstevnice s rozdílem 5 m n. m. indikující mírnou změnu terénu. Dalším příkladem je výskyt bodových značek skalních útvarů, které mají totožnou barvu s vrstevnicemi a nelze je tedy barevnou segmentací rozlišit.

Algoritmy programu jsou navrženy pro konkrétní typ map z předem definovaného zdroje. Obraz se předpokládá s pevným rozlišením, formátem a měřítkem mapy, se stejnou barevnou paletou, atd. Není proto nutné řešit osvětlení, geometrické zkreslení (natočení a deformace obrazu, sever-jih) apod. Nevýhodou je naopak závislost na tomto zdroji, kdy předzpracování obrazu nemusí správně pracovat na obrazu mapy z jiného než uvedeného zdroje. Program obsahuje mnoho heuristicky zjištěných koeficientů, které by bylo nutné adaptovat.

Nejobtížnější částí práce byla rekonstrukce vrstevnic, které jsou často poškozeny překrývajícími objekty. Dalším negativním vlivem je hustota vrstevnic, kdy selhávají algoritmy spojení a výpočty směrnic koncových bodů.

Výpočet výškové mapy je celkem náročný na procesorový čas, kdy je nutné procházet velké množství bodů vrstevnic. Tento výpočet je navíc uskutečňován ve smyčce iteračních kroků do doby ustálení výsledků. I přesto může být výsledná výšková mapa nepřesná a zašuměná. Je to dáno nepřesným výpočtem směrnic (diskrétní

hodnoty), zaokrouhlováním a použitím interpolačních metod.

Při výpočtu výškové mapy není řešena změna gradientu terénu, tj. stoupání či klesání. Předpokládá se, že výchozí bod je nejvýše položený bod v celé mapě a od tohoto se terén svažuje. Možným řešením by se naskýtala identifikace vodních toků a jejich směru toku. Předpokladem by mohla být zvětšující se šířka směrem od pramenu. Další možností by mohlo být optické rozpoznání čísel výškových hodnot vyskytujících se u vrstevnic. Obě metody jsou však již náročnější na realizaci.

Při realizaci diplomové práce se ukázalo, že je nutné neustále počítat s diskrétní interpretací obrazu, např. při výpočtech úsečky či průniku dvou úseček. Dále je obraz omezený okrajově, tedy přímka či polopřímka musí nutně končit na kraji obrazu a tento bod je nutné vypočíst či jinak odvodit.

LITERATURA

[1] *NASA s Japonci představila dosud nejucelenější mapu světa.* Týden.cz [online]. 01. 07. 2009, [cit. 15. 1. 2011]. Dostupné z URL: <http://www.tyden.cz/rubriky/veda/veda-a-my/nasa-s-japonci-predstavila-dosud-nejucelenejsi-mapu-sveta_127226.html>.

[2] VEVERKA, Bohuslav - ZIMOVÁ, Růžena. *Topografická a tematická kartografie.* 1. vyd. Praha: České vysoké učení technické, 2008. 198 s. ISBN: 978-80-01-04157-4 (brož.). Sign: 3-1211.933.

[3] ČERBA, Otakar. *Tematická kartografie(KMA/TKA).* Geomatika na ZČU v Plzni [online]. 17. 3. 2011 [cit. 7. 5. 2011]. Dostupné z URL: <http://gis.zcu.cz/?page=tka>.

[4] *Mapy.cz* [online]. [cit. 8. 5. 2011]. Dostupné z URl: <http://mapy.cz>.

[5] HLAVÁČ, Václav - ŠONKA, Milan. *Počítačové vidění : první česká kniha o zpracování digitalizovaných obrazů* Praha: Grada, 1992. 252 s., 32 fot. na příl. Edice: Nestůjte za dveřmi. ISBN: 80-85424-67-3.

[6] JAN, Jiří. *Číslicová filtrace, analýza a restaurace signálů.* 2. upr. a rozš. vyd. Brno : VUTIUM, 2002. 427 s. ISBN 80-214-2911-9.

[7] HORÁK, Karel a kol. *Počítačové vidění*, Přednášky MAPV [online]. [cit. 16. 5. 2012], UAMT FEKT VUT Brno, 2012. Dostupné z URL: <http://midas.uamt.feec.vutbr.cz/POV/Overview/overview_cz.php>.

[8] HORÁK, Karel a kol. *Aplikace počítačového vidění*, Přednášky MPOV [online]. [cit. 16. 5. 2012], UAMT FEKT VUT Brno, 2012. Dostupné z URL: <http://midas.uamt.feec.vutbr.cz/APV/Overview/overview_cz.php>.

[9] ČERNÝ, Jakub. *Základní grafové algoritmy.* KAM, MFF UK, 24. listopadu 2010, verze 0.95, 204 s. Dostupné z URL: <http://kam.mff.cuni.cz/~kuba/ka/ka.pdf>.

[10] *Algoritmy.net*, příručka vývojáře [online]. [cit. 10. 5. 2011]. Dostupné z URL: <http://www.algoritmy.net/>.

[11] *MATLAB - The Language Of Technical Computing* [počítačový program]. Version 7.13.0.564 (R2011b). Pro Windows 32-bit (win32). September 1, 2011. Dostupné z URL: <http://www.mathworks.com/products/matlab/>

[12] TUŠKA, David. *Rekonstrukce výškového terénu z turistické mapy.* Bakalářská práce, Brno, FIT VUT v Brně, 2008, vedoucí práce Ing. Michal Španěl.

[13] BOLDIŠ, Petr. *Bibliografické citace dokumentů podle ČSN ISO 690 a ČSN ISO 690-2* [online]. 2001, poslední aktualizace 11. 11. 2004 [cit. 15. 4. 2011]. Dostupné z URL: <http://www.boldis.cz/citace/citace.html>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

ČSN Česká státní norma

RGB Red, Green, Blue (červená, zelená, modrá), barevný prostor obrazu

HSV Hue, Saturation, Value (odstín, sytost, jas), barevný prostor obrazu

YCbCr (jas, chrominance modré a červené komponenty) barevný prostor obrazu

JPEG Joint Photographic Experts Group, formát ztrátové komprese digitálních obrazů

PNG Portable Network Graphics, formát bezeztrátové komprese digitálních obrazů

GUI Graphical User Interface, grafické uživatelské rozhraní

A UKÁZKA NALEZENÍ TRASY V OBRAZU

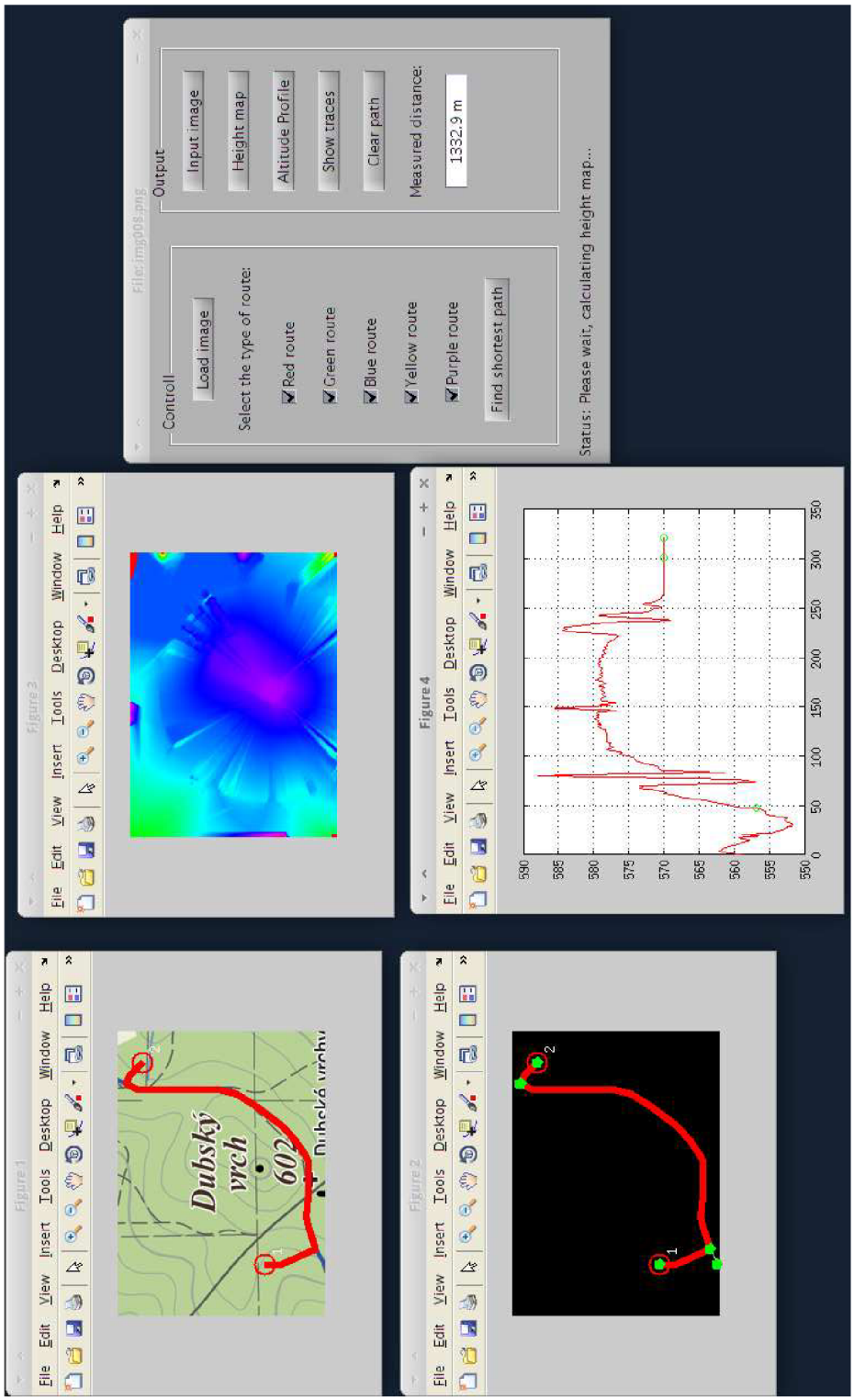
Obr. A.1: Vstupní obraz turistické mapy.

Obr. A.2: Předzpracování obrazu.

Obr. A.3: Výsledek hledání trasy v grafu.

Obr. A.4: Vyznačená nalezená trasa délky 1230,0 obrazových elementů.

B GRAFICKÉ UŽIVATELSKÉ ROZHRANÍ



Obr. B.1: Ukázka zpracování obrázku mapy v grafickém uživatelské rozhraní.