VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV MECHANIKY TĚLES, MECHATRONIKY
A BIOMECHANIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

DETEKCE CESTY PRO MOBILNÍ ROBOT ANALÝZOU OBRAZU

ROAD DETECTION FOR MOBILE ROBOT USING IMAGE PROCESSING

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — ING. JAN COUFAL
AUTHOR

VEDOUCÍ PRÁCE — ING. JIŘÍ KREJSA, PHD.
SUPERVISOR

BRNO 2010

ZADÁNÍ

Cíle, kterých má být dosaženo:

1. Analyzujte metody zpracování obrazu použitelné pro danou úlohu
2. Vyberte některou z perspektivních metod a implementujte ji
3. Metodu otestujte na reálných datech

Charakteristika problematiky úkolu:

Navrhněte robustní metodu pro detekci cesty z obrazu pořízeného kamerou mobilního robotu pomocí metod segmentace obrazu. Pomocí parametrů segmentovaného objektu extrahujte potřebné rysy prostředí (cesta / křižovatka, …)

ABSTRAKT

Diplomová práce se zabývá zpracováním obrazu pro mobilní robot pohybující se ve venkovním prostředí. V první části je analyzován problém, navrženo řešení a prozkoumány vhodné metody zpracování obrazu. V druhé části jsou jednotlivé metody porovnány mezi sebou a na základě výsledků je sestaveno konkrétní řešení. Třetí část obsahuje výsledy navrženého řešení na reálných datech.

ABSTRACT

Diploma thesis deals with image processing for outdoor environment mobile robot. In first part, the problem is analyzed, general solution is proposed and suitable image processing methods are presented. In second part presented methods are tested and methods with best results are proposed. In third part is particular solution tested on real data.

KLÍČOVÁ SLOVA

Segmentace obrazu, zpracování obrazu, navigace robota

KEYWORDS

Image segmentation, image processing, robot navigation, vision

BIBLIOGRAFICKÁ CITACE

COUFAL, J. *Detekce cesty pro mobilní robot analýzou obrazu.* Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2010. 49 s. Vedoucí diplomové práce Ing. Jiří Krejsa, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že jsem diplomovou práci vypracoval samostatně pod vedením svého vedoucího práce Ing. Jiří Krejsy, PhD., a pouze s použitím uvedené literatury.

Jan Coufal

PODĚKOVÁNÍ

Upřímně děkuji všem, kteří mě kdy směrovali správným směrem. Zejména chci poděkovat svému otci Janu Coufalovi za veškerou podporu a svému vedoucímu práce, Ing. Jiřímu Krejsovi PhD., za ochotu, pomoc a dobré vedení.

OBSAH

ÚVOD

Robotika prodělala v posledním desetiletí bouřlivý vývoj. Stále se objevují nové aplikace, které se už neomezují jen na vědecké či průmyslové využití, ale našly si cestu i do života obyčejných lidí. Autonomní roboty můžeme nalézt na mezinárodních letištích jako mobilní interaktivní informační panely, jako bezpečnostní systémy hlídající budovy a průmyslové komplexy nebo jako vysavače v běžných domácnostech.

Stejně jako je zrak důležitým smyslem člověka, podle kterého se dokáže orientovat v prostředí a rozpoznávat důležité signály moderního světa, tak i kamera a softwarové zpracování obrazu jsou stejně důležité pro autonomního robota. Data získaná z kamery mohou při správném zpracování poskytnout všechny potřebné informace pro navigaci a orientaci robota a pro rozpoznávání objektů a interakci s okolím. Kamera je navíc pasivním snímačem, k jehož provozu není třeba mnoho energie. Proto hraje zpracování obrazu robota, který se má pohybovat v prostředí vytvořeným člověkem, důležitou roli.

Konkrétní úloha zpracování obrazu je závislá na účelu, pro který je robot použit. Ve své práci se zabývám zpracováním obrazu autonomního robota pohybujícího se po cestách ve venkovním prostředí. Robot musí být tedy schopen cestu v obraze rozpoznat a určit úhel natočení kol tak, aby se na této cestě udržel. Obojího jsem ve své práci dosáhl.

FORMULACE PROBLÉMU A CÍLE ŘEŠENÍ

Požadavek na segmentační algoritmus vznikl při Robotour 2009, jejímž cílem bylo, aby autonomní robot projel zadanou trasu vytyčenou GPS souřadnicemi a po celou dobu zůstal na cestě. Soutěže se zúčastnil robot Bender II, vyvinutý a zkonstruovaný na ÚMTMB společně s ÚAI na fakultě FSI VUT v Brně. Robot byl vybaven kamerou, GPS navigací, laserovým senzorem SICK pro detekci překážek, odometrií (IRC snímače na předních kolech) a kompasem. Navigace podle údajů z kamery byla založena na detekci zelené barvy v obraze. Účast týmu na robotické vytyčila některé cíle, které je potřeba pro navigaci autonomního robota zvládnout. Jedním z nich je navigace robota na základě robustního zpracování obrazu.

Cílem řešení práce je tedy prozkoumat metody zpracování a segmentace obrazu (i z hlediska dalšího využití na mobilním robotu) a vytvořit algoritmus, který v obraze rozpozná cestu a tyto informace o cestě použije pro navigaci robota. Rozpoznáním cesty se rozumí rozdělit množinu všech bodů v obraze na dvě samostatné podmnožiny, z nichž jedna odpovídá cestě a druhá okolí. Navigací robota se rozumí výpočet úhlu natočení kol robota ze získaných segmentovaných dat.

REŠERŠNÍ STUDIE

Použití obrazu v navigaci robotů (a silničních vozidel) je častým předmětem výzkumu posledních dvacet let. Oblasti výzkumu mohou být děleny podle charakteru cesty, použitých senzorů a algoritmů zpracování obrazu. Ve své práci jsem se zaměřil pouze na zpracování obrazu z monokulární kamery. Výpočetní čas binokulárních systémů, které potřebují dobře zvládnutou synchronizaci obou kamer, je stále relativně vysoký. I výpočetní čas zpracování obrazu pro binokulární systémy je stále problém.

Pokud se robot pohybuje po dobře strukturovaných cestách, jako například vozovky ve městech, je největší pozornost zamřena na detekci a stopování střední dělící čáry nebo krajnice v obraze. Protože je vozovka poměrně stejnorodý (uniformní) povrch, jsou s úspěchem použity techniky segmentace na základě barvy [1] a techniky na základě detekce středního dělícího pásu na vozovce a detekce hran [2]. I na dobře rozpoznatelných vozovkách je třeba, aby segmentační algoritmus byl invariantní na stíny a osvětlení. Pro tento účel používá spousta autorů transformaci původního obrazu do jiného barevného prostoru [3, 4, 5].

Pokud se robot pohybuje v nestrukturovaném prostředí (venkovní cesty), hlavní pozornost je zaměřena na vyhýbání překážek a klasifikaci terénu. Například metoda prezentovaná v [4] je založena na konstrukci 2D modelu scény venkovního prostředí. Obraz je konvertován do barevného prostoru invariantního na osvětlení, dále je segmentován metodou růstu regionů a získané regiony (oblasti stejných bodů) jsou klasifikovány do předučených tříd reprezentujících krajinu. Algoritmus dává dobré výsledky, ale je silně závislý na předučených znalostech. Ve svém řešení jsem se chtěl vyhnout implementování nějakých znalostí o okolním prostředí.

Výzkumný tým [5] motivován soutěží DARPA vyvinul metodu pro extrakci směru v pouštním terénu. Jejich přístup používá transformaci obrazu do c1c2c3 barevného prostoru publikovaném v [6], který je výhodný pro eliminaci stínů z obrazu a lepší segmentaci obrazu. Pro plánování pohybu svého autonomního motocyklu používají obrazový vektorový prostor, který reprezentuje bezkolizní směry v obraze. Tento vektorový prostor je promítnut na předem připravený soubor trajektorií v obraze. Trajektorie, která má největší shodu s vektorovým prostorem je vybrána pro navigaci robota.

Zajímavým přístupem je použití algoritmu mean-shift pro segmentaci a použití techniky „graph-cut“ pro sjednocení podobných částí v obraze. Metoda prezentovaná v práci [7] nejdříve segmentuje obraz na homogenní části s přesnou hranou metodou založenou na algoritmu mean-shift se zabudovaným detektorem hran. Poté se s jednotlivých regionů sestaví graf, kde uzly jsou statistické informace celého regionu a hrany jsou vzdálenosti mezi nimi. Z předchozích průběhů algoritmu se uchová barevný model cesty a prostředí, který pomůže získat finální bitový obraz, který odpovídá cestě. Algoritmus kombinuje výhody precizní mean-shift segmentace s graph-cut algoritmem a použití výsledků z minulých průběhů celého algoritmu zajišťuje dostatečnou adaptabilitu. Bohužel mean-shift segmentace se ukázala jako velice náročná na výpočetní čas, ale je možné tuto metodu použít např. v inicializační fázi robota nebo urychlit její výpočet pomocí GPU.

1. ANALÝZA PROBLÉMU

V této části postupně analyzuji problém, přijmu určitá zjednodušení a navrhnu řešení. Nejdříve charakterizuji prostředí, ve kterém se mobilní robot pohybuje. Na základě charakteru prostředí určím některé obecné rysy řešení.

Další část probírá jednotlivé metody úprav a segmentace obrazu, které při zpracování použijeme, nebo jsou z hlediska budoucího využití pro robota významné.

1.1. Charakteristika prostředí

Obecně se prostředí, ve kterém se autonomní roboti pohybují, dělí na prostředí vnitřků budov (indoor) a prostředí venkovní (outdoor). Vnitřní prostředí má oproti vnějšímu z hlediska detekce cesty několik výhod. Povrch podlah vnitřku budov bývá většinou vodorovný a homogenní z hlediska barvy i z hlediska členitosti povrchu. Další nespornou výhodou je stálost osvětlení. Ve vnitřku budov se téměř nevyskytují kontrastní přechody světlo/stín. Poslední výhoda spočívá v ostrém přechodu mezi podlahou a stěnou. Cesta se od okolního prostředí odlišuje jasnou hranou, kterou v obraze můžeme detekovat.

Oproti tomu vlastnosti venkovního prostředí jsou více obecné. Povrch, po kterém se robot pohybuje, může své vlastnosti měnit v závislosti na pozici robota. Světelné podmínky se ve venkovním prostředí se také mění s pozicí robota a dokonce i s časem, vyskytují se ostré přechody světlo/stín. Další nevýhodou je pozvolný přechod mezi povrchem cesty a okolním prostředím. Z tohoto důvodu není možné detekovat jasnou hranu přechodu a určit tak hraniční oblast cesty. Úloha zpracování obrazu ve venkovním prostředí je tedy obtížnější.

1.2. Předpoklady řešení a přijatá zjednodušení

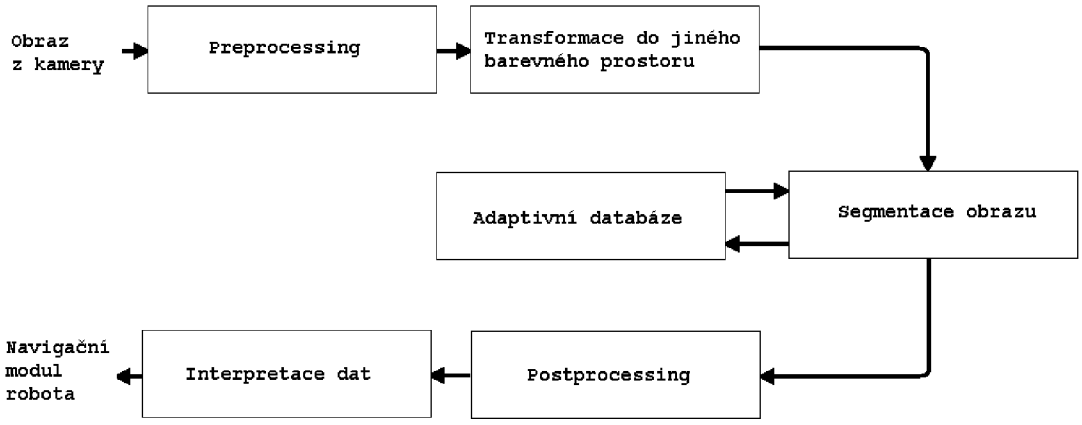
Na základě charakteru prostředí je možno říci, že úspěšné řešení bude muset být schopno detekovat cestu v obraze, ve kterém jsou přítomné stíny. Z toho vyplývá použití jiných barevných prostorů, než je RGB systém souřadnic, které jsou invariantní na změnu osvětlení a přítomnost stínů v obraze.

Další charakteristikou detekované cesty je její proměnlivost. Cesta ve venkovním prostředí se může spojitě i náhle měnit. Cesta může mít více podob (asfalt, zemina, žulové kostky, atd.). Na základě této vlastnosti prostředí lze říct, že algoritmus bude muset být

adaptivní, aby se mohl přizpůsobovat měnícím se podmínkám. Pokud se chceme vyhnout předučeným znalostem o venkovním prostředí, tak vhodné řešení je uchovávání informací o projeté cestě v databázi a jejich použití v průběhu algoritmu. Z toho vyplývá nutnost tuto databázi inicializovat při spuštění robota.

Zároveň pro jednoduché a dostačující řešení předpokládejme, že země, po které se robot pohybuje, je plochá. Toto zjednodušení umožňuje z řešení vynechat detekci horizontu, který se nebude měnit s pozicí robota a bude stále na stejné pozici v obraze.

Obecné řešení bude mít tuto strukturu:



Obr.1. Obecné schéma řešení segmentovacího algoritmu.

Algoritmus rozpoznávání cesty funguje následovně. Data získané kamerou jsou filtrována za účelem potlačení aditivního šumu a následně jsou převedena do vhodného barevného prostoru, invariantního vůči stínům a změně osvětlení. Takto upravený obraz vhodnou segmentační metodou rozdělíme na stejné oblasti a na základě informací z databáze vyhodnotíme, která část segmentovaného obrazu odpovídá cestě. Rozčleněná data se dále upravují a nakonec se v interpretačním modulu extrahují parametry důležité pro navigaci robota.

V dalším textu se budu zabývat jednotlivými bloky schématu na obr. 1.

1.3. Barevné prostory

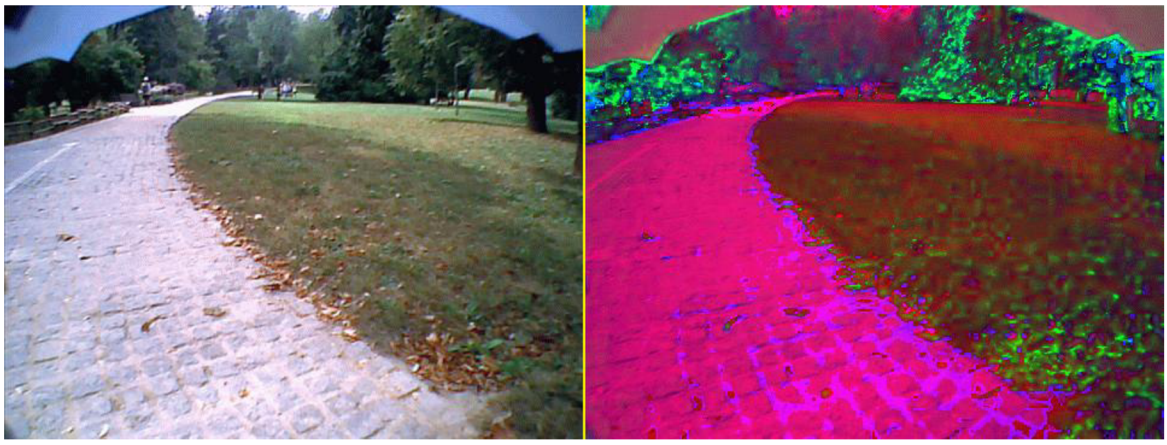
Pro segmentaci obrazu ve venkovním prostředí je standardní RGB barevný prostor nevyhovující. Problém spočívá v tom, že dva body reprezentující stejnou barvu s různým jasem jsou od sebe v RGB prostoru poměrně vzdáleny. Řešením může být použití takového barevného prostoru, jehož barevná složka je oddělená od složky jasové. V této práci jsem se zabýval dvěma barevnými prostory, které jsem vyhledal v literatuře a které se pro daný problém používají. Jedná se o barevné modely HSI, c1c2c3 publikovaným v [6].

1.3.1. Barevný prostor HSI

HSI (Hue, Saturation, Intensity) barevný prostor je velice rozšířen a s úspěchem použit v mnoha aplikacích segmentace obrazu a detekce cesty [1, 3]. Jak název napovídá, barevná složka HSI prostoru je oddělena od složek intenzity a sytosti. Tato vlastnost barevného prostoru nám dává možnost kvantifikovat podobnost dvou bodů v obraze nezávisle na jejich složkách intensity. Jinými slovy bod cesty v obraze, na kterou dopadl stín, bude mít stejnou „barevnou“ hodnotu jako bod, na který stín nedopadl, a body se budou lišit pouze v hodnotách saturace a intensity. Tento předpoklad ovšem platí pouze pro chromatické barvy. Transformace z RGB do HSI vypadá následovně:

$$
\begin{aligned}
H &= \tan^{-1}\left(\frac{\sqrt{3}(G-B)}{2R-G-B}\right) \\
I &= \frac{R+G+B}{3} \\
S &= \begin{cases} 0 & \textit{jestliže } \min(R,G,B) = \max(R,G,B) \\ 1-\dfrac{\min(R,G,B)}{I} & \textit{pokud jinak} \end{cases}
\end{aligned}
\tag{1}
$$

kde R, G, B jsou hodnoty obrazového bodu pro jednotlivé jasové složky v rozsahu $\langle 0, 255 \rangle$, H je úhel v cylindrickém HSI prostoru v rozsahu $\langle 0, 2\pi \rangle$, I intensita barvy a S je saturace barvy, obě v rozsahu $\langle 0, 1 \rangle$. Při výpočtu je třeba dávat pozor na fakt, že pro některé hodnoty R G B není úhel H definován.



Obr. 2. Konverze do HSI prostoru. Vlevo je původní obrázek, vpravo je obrázek v HSI souřadnicích, které jsou zobrazeny v RGB.

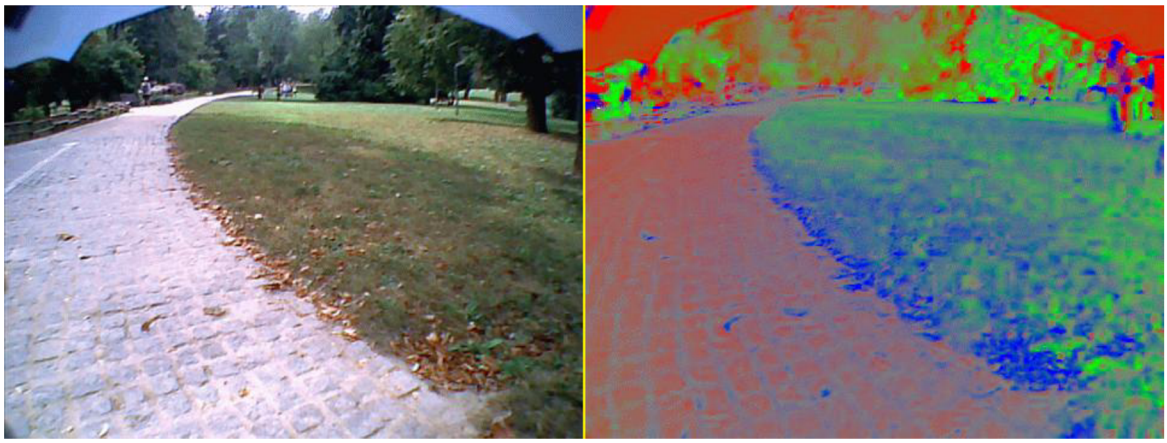
1.3.2. Barevný prostor c1c2c3

Tento barevný prostor byl vyvinut speciálně za účelem získání barevného prostoru invariantního na úhlu pohledu kamery, geometrii sledovaného objektu a stíny [6]. Původní použití tohoto barevného prostoru bylo ve vnitřních prostředích, ale někteří autoři tento prostor s úspěchem používají v prostředích venkovních [5]. Z tohoto důvodu jsem se rozhodl jej otestovat.

Transformační vztah mezi barevnými prostory:

$$\begin{aligned} c1 &= \tan^{-1}\left(\frac{R}{\max(G,B)}\right) \\ c2 &= \tan^{-1}\left(\frac{G}{\max(R,B)}\right) \\ c3 &= \tan^{-1}\left(\frac{B}{\max(R,G)}\right) \end{aligned} \tag{2}$$

kde R, G, B jsou hodnoty obrazového bodu pro jednotlivé jasové složky v rozsahu $\langle 0,255 \rangle$. Výsledné koeficienty jsou v rozsahu $\left\langle 0,\frac{\pi}{2} \right\rangle$ a je třeba je převést na rozsah $\langle 0,255 \rangle$. Při výpočtu je třeba dávat pozor na dělení nulou.



Obr. 3. Konverze do c1c2c3 barevného prostoru. Vlevo je původní obrázek, vpravo je obrázek konvertovaný do c1c2c3 barevného prostoru, zobrazený v RGB.

1.4. Preprocessing

Preprocessing znamená úpravu dat před jejich samotným zpracováním, nebo též úpravu dat pro potřeby následujícího (většinou hlavního) zpracování. V obrazové analýze se preprocessing zabývá hlavně odstraněním šumu a ořezáváním obrazu. Hlavní metodou filtrace šumu v obrazové analýze je diskrétní dvourozměrná konvoluce.

1.4.1. Diskrétní dvourozměrná konvoluce

Konvoluce je matematická operace dvou funkcí (zpracovávané funkce a jádra), jejímž výsledkem je funkce, na kterou může být nahlíženo jako na upravenou funkci původní. Nechť A je obrazová matice o rozměru $w \times h$ a C nechť je matice zvaná konvoluční jádro nebo též kernel o rozměru $(2n+1)^2$. Konvolucí vznikne matice B o stejném rozměru jako matice A, přičemž platí:

$$A = \left[a_{i,j} \right]_{j \in \langle 0,h \rangle}^{i \in \langle 0,w \rangle} \qquad C = \left[c_{i,j} \right]_{j \in \langle -n,n \rangle}^{i \in \langle -n,n \rangle}$$

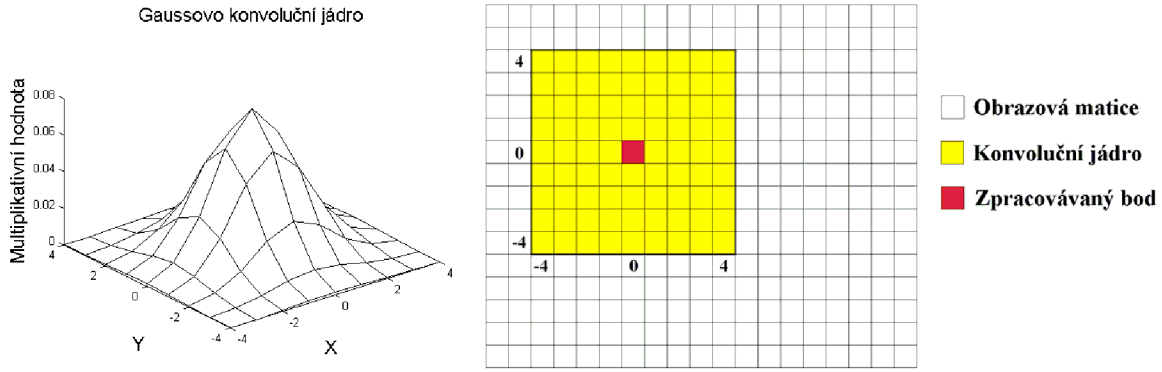
$$A * C = B \left[b_{i,j} \right]_{j \in \langle 0,h \rangle}^{i \in \langle 0,w \rangle} \tag{3}$$

$$b_{i,j} = \sum_{l=-n}^{n} \sum_{k=-n}^{n} a_{i+k,j+l} \cdot c_{k,l}$$

Existuje celá řada konvolučních jader, každé pro specifický účel. Pro potlačení hran a šumu v obraze se často používá Gaussovo konvoluční jádro, jehož koeficienty se vypočítají následovně.

$$G\left[g_{x,y}\right]_{y\in\langle -n,n\rangle}^{x\in\langle -n,n\rangle} \qquad g(x,y)=\frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4}$$

kde $G\left[g_{x,y}\right]$ je matice o rozměru $(2n+1)^2$, x a y jsou relativní pozice v kernelu od prostředního prvku a σ je směrodatná odchylka. Součet všech prvků v konvolučním jádře musí být roven jedné, aby nedocházelo ke zvyšování jasu obrazu.

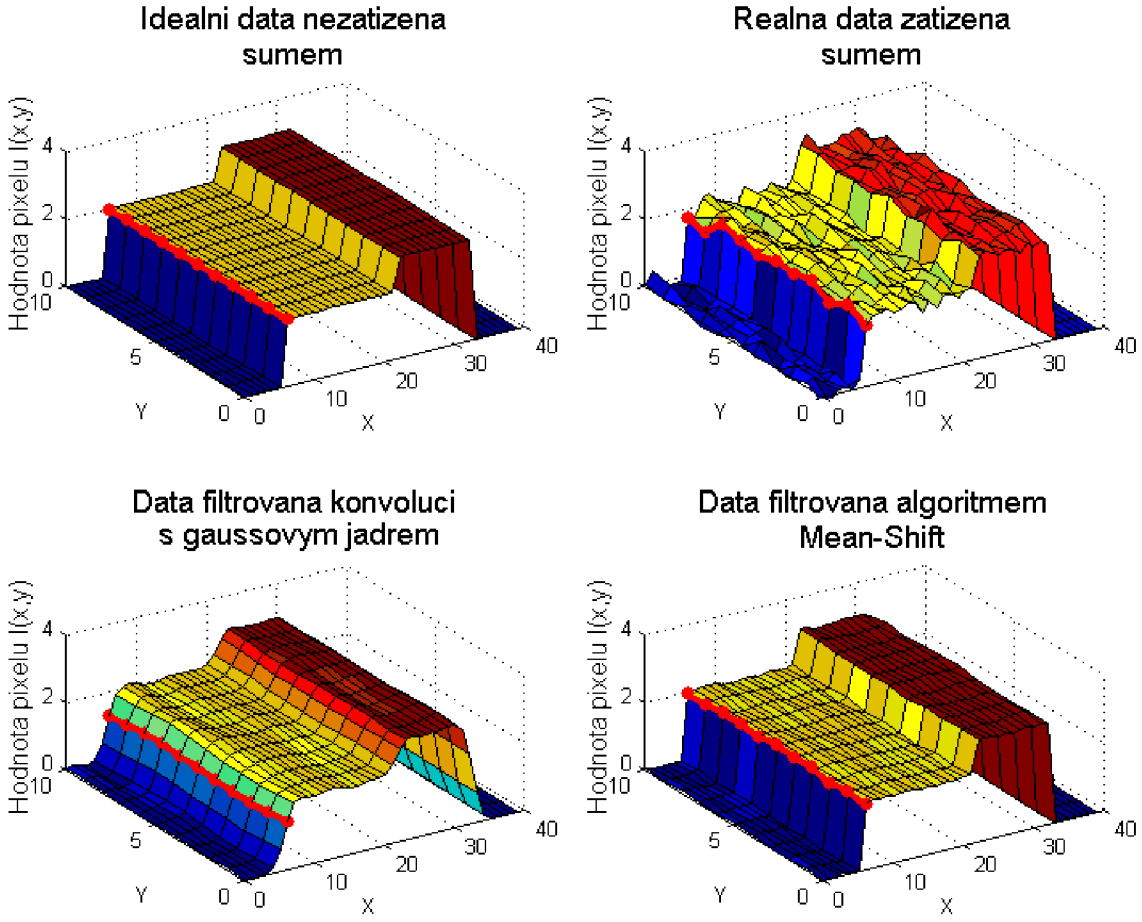


Obr. 4. Princip konvoluce. Obrázek vlevo ukazuje gaussovo jádro, na obrázku vlevo je naznačena matice jádra, která se postupně posouvá po celé obrazové matici.

Obr. 5. Příklad konvoluce gaussovým jádrem. Vlevo je originální obraz a vpravo je zpracovaný obraz konvolucí gaussovým jádrem o rozměru 9x9.

1.4.2. Filtr založený na algoritmu Mean-shift

Speciálním případem diskrétní konvoluce je technika zvaná „Discontinuity Preserving Filtering“ [8] (filtrace zachovávající ostré přechody). Tato metoda je založená na algoritmu mean-shift, jehož princip je podrobně popsán v kapitole o segmentačních metodách. Algoritmus funguje na principu posouvání barevných hodnot aktuálního obrazového bodu směrem ke střední hodnotě barevných hodnot okolních bodů. Algoritmus počítá lokální odhad hustoty příbuzných bodů v barevném prostoru a posouvá hodnoty aktuálního bodu ve směru gradientu této hustoty tak dlouho, dokud není dosaženo lokálního maxima (střední hodnoty). Příbuznost bodů je definována barevným rozsahem a velikost okolí je určena velikostí kernelového jádra. Výjimečnost algoritmu oproti například gaussově konvoluci spočívá v jeho schopnosti zachovat ostrost hran mezi barevně odlišnými objekty v obraze. To je z hlediska segmentace dat velice užitečná vlastnost.



Obr. 6. Porovnání filtrace šumu konvolucí gaussovým jádrem a Mean-Shift algoritmem.

1.5. Metody segmentace obrazu

Segmentací obrazu se rozumí takový proces, který podle zvoleného kritéria sdružuje body v obraze do množin s podobnými vlastnostmi. Pokud v obraze hledáme objekt se známými vlastnostmi, můžeme použít dva kvalitativně odlišné přístupy. Prvním je hledat objekt v obraze přímo, kdy obrazové body dělíme na základě jejich vlastností do dvou množin: hledaný objekt a pozadí. Druhým přístupem je nejdříve rozčlenit celý obraz na spojité oblasti podobných bodů a pak s těmito celky provést klasifikaci do množin objekt/pozadí. Oba přístupy mají své využití.

Segmentaci obrazu dále můžeme dělit na tzv. segmentaci „top-down“ (od shora dolů). V tomto případě celkový obraz dělíme na postupně menší části, které klasifikujeme podle zvolených kritérií nebo na základě znalostí o objektu (např. křivky), který v obraze hledáme. Opačný přístup je tzv. segmentace „bottom-up“ (od spodu nahoru), který posuzuje každý bod v obraze zvlášť a klasifikuje jej do skupiny podobných bodů podle jeho vlastností. Tyto vlastnosti, dle kterých klasifikace probíhá, mohou být dvojího druhu. Buď posuzujeme přímo barevné hodnoty bodu, nebo bereme v úvahu definované okolí bodu a jeho statistické vlastnosti.

Jako vhodný typ metod jsem zvolil segmentační metody od spodu nahoru, které klasifikují body pouze na základě jejich barevných vlastností. Výpočet statistických vlastností okolí bodu se ukázalo jako výpočetně náročné, i když některá řešení snížení výpočetního času na úkor přesnosti existují [10].

1.5.1. Metoda prahování

Metoda prahování je jednoduchá a efektivní metoda segmentace obrazu. Její princip spočívá ve zvolení metriky pro testování vzdálenosti dvou bodů obrazu a zvolení prahové hodnoty určující maximální vzdálenost dvou prvků, které náleží do stejné třídy.

Nechť $A\left[a_{i,j}\right]$ je obrazová matice, jejíž prvky jsou body v třírozměrném prostoru určené souřadnicemi $a_{i,j}\left[a_{i,j,0}, a_{i,j,1}, a_{i,j,2}\right]$ a nechť bod $r_{obj}\left[r_{obj,0}, r_{obj,1}, r_{obj,2}\right]$ je bod v třírozměrném prostoru reprezentující libovolný objekt O, v našem případě cestu. Pak je možno říci, že prvek matice $a_{i,j}$ patří do množiny bodů O tehdy, když splňuje následující podmínku:

$$dist\left(a_{i,j}, r_{obj}\right) \leq t_{diss} \tag{5}$$

kde t_{diss} je tzv. práh maximální vzdálenosti, nebo též práh odlišnosti (dissimilarity), a kde $dist\left(a_{i,j}, r_{obj}\right)$ je vzdálenost bodů ve zvolené metrice. Nejčastěji bývá pro měření vzdálenosti bodů používána Euklidovská metrika, ale i jiné metriky mohou být použity (např. Mahalanobisova metrika):

$$\sqrt{\left(a_{i,j,0} - r_{obj,0}\right)^2 + \left(a_{i,j,1} - r_{obj,1}\right)^2 + \left(a_{i,j,2} - r_{obj,2}\right)^2} \leq t_{diss} \tag{6}$$

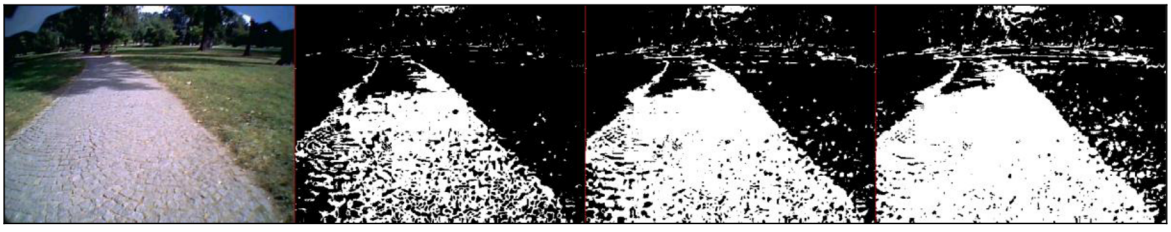
Při segmentaci v HSI barevném prostoru není vhodné použít Euklidovskou metriku, protože tento barevný prostor má cylindrický charakter. Z tohoto důvodu se vzdálenost dvou bodů v HSI prostoru měří dle literatury [1] následovně:

$$dist(a,b) = \sqrt{\left(S_a\right)^2 + \left(S_b\right)^2 - 2S_aS_b\cos\theta} \tag{7}$$

kde *a, b* jsou body v HSI barevném prostoru, určené souřadnicemi $a\left[H_a, S_a, I_a\right]$ a kde úhel θ je definován následovně:

$$\theta = \begin{cases} \left|H_a - H_b\right| & if\ \left|H_a - H_b\right| < 180° \\ 360 - \left|H_a - H_b\right| & if\ \left|H_a - H_b\right| > 180° \end{cases} \tag{8}$$

Algoritmus jednoprůchodového prahování vypočítá vzdálenost každého bodu od reprezentanta objektu, a pokud je menší než prahová hodnota, bod je přiřazen k množině bodů, reprezentované $\vec{r}_{obj}$. V opačném případě je bod klasifikován jako pozadí. Hledaný objekt v obraze může být reprezentován více než jedním bodem. V tom případě je bod klasifikován jako cesta, pokud splní podmínku (6) alespoň pro jednoho z nich.

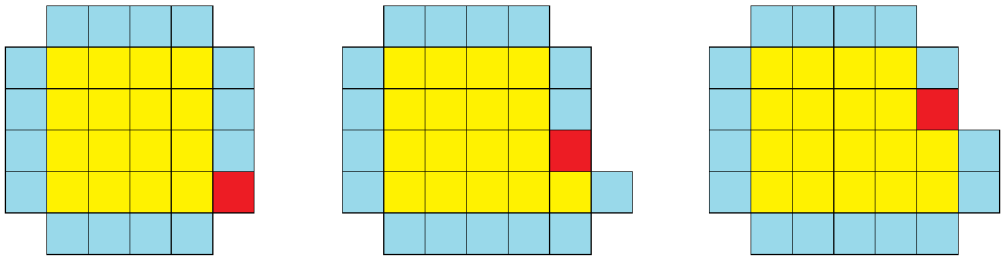


Obr. 7. Příklad segmentace obrazu pomocí prahování. Vlevo je původní obraz. Obrazy vpravo jsou výsledky segmentace prahováním postupně pro prahové hodnoty t_{diss} *= 10, 15, 20. Bílé body odpovídají cestě, černé pozadí. Jako referenční bod byla zvolena střední hodnota barev trapézovitého regionu ve spodní části obrazu (viz. inicializace robota).*

Segmentace obrazu prahováním tedy přímo rozdělí obraz na dvě části: hledaný objekt a okolí. Výhoda tohoto algoritmu spočívá v jeho jednoduchosti a rychlosti. Určitá nevýhoda však spočívá v tom, že je aplikována globálně a zcela ignoruje prostorové rozložení objektu v obraze. To způsobuje, že jako objekt jsou klasifikovány body, jejichž vlastnosti sice objektu odpovídají, ale jejich umístění v obraze nikoliv. Tato nevýhoda se dá odstranit dalším zpracováním obrazu (postprocessingem).

1.5.2. Metoda rostoucích regionů

Jedná se o segmentační metodu „bottom-up“, která částečně odstraňuje nevýhodu předchozí metody. Metoda začíná inicializací. Jsou definovány počáteční regiony umístěním takzvaných „semen“ do obrazu (seeds). Princip metody spočívá v iterativním testování bodů, které s regionem sousedí. Pokud sousední bod splní námi definované kritérium homogenity, je zahrnut do regionu, upraví se vlastnosti celého regionu a vypočítají se noví sousedi. Pokud je bod již členem jiného regionu, jsou oba regiony sloučeny. Při určování sousedů můžeme za sousední body považovat buď pouze čtyři body, nebo všech okolních osm. V literatuře se uvádí tzv. „4-connectivity“ nebo „8-connectivity“. Tímto iterativním procesem region v obraze „roste“.



Obr. 8. Princip segmentační metody. Obrázek ukazuje dvě iterace čtyř-konektivní metody růstu regionů. Žluté čtverce představují body náležící do regionu, červený čtverec je právě zkoumaný bod a světle modré jsou sousední body, které se teprve budou testovat.

Při každém přidání nového bodu do regionu jsou všechny vlastnosti regionu upraveny váženým průměrem.

$$r_N = \frac{(n-1)r_N + r_i}{n}; \qquad g_N = \frac{(n-1)g_N + g_i}{n}; \quad b_N = \frac{(n-1)b_N + b_i}{n} \tag{9}$$

kde n je počet bodů v regionu, r_N g_N b_N jsou průměrné souřadnice všech bodů regionu v RGB barevném prostoru (ale může se jednat souřadnice v libovolném barevném prostoru) a r_i g_i b_i jsou souřadnice právě přidaného bodu. Podobným způsobem je vypočítáno i těžiště regionu.

Nejčastějším kritériem homogenity bývá podmínka velice podobná rovnici (5), která vyžaduje, aby euklidovská vzdálenost průměrné hodnoty souřadnic regionu a souřadnic zkoumaného bodu byla menší než zvolená prahová hodnota.

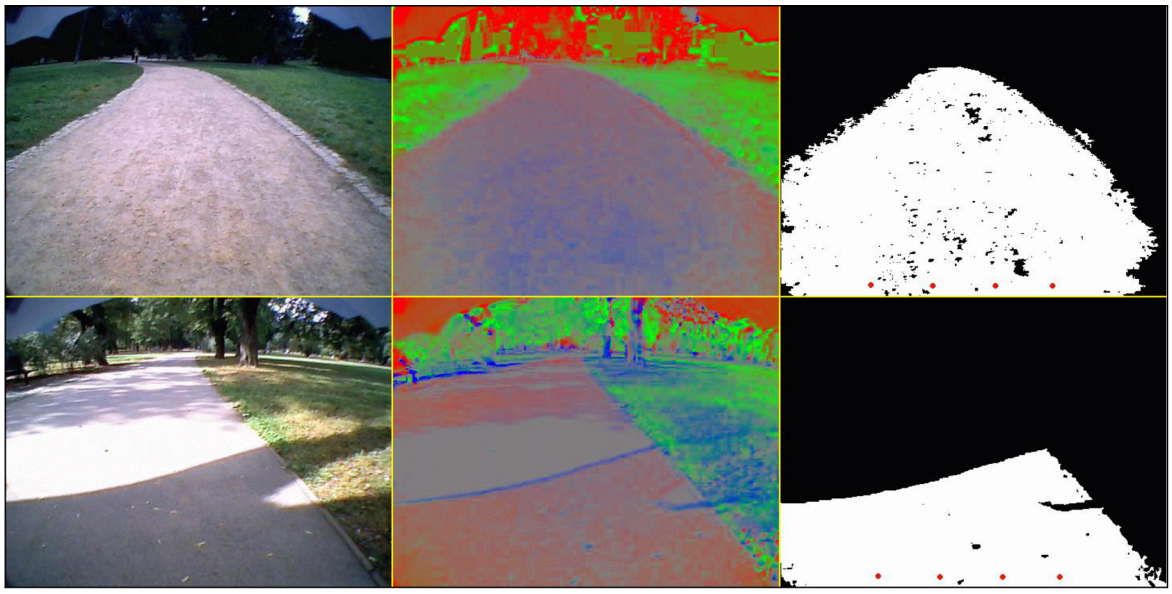
Pokud zkoumaný bod kritérium nesplní, mohou nastat dva případy. V prvním případě založíme nový region, jehož počáteční bod (semeno) je právě vyloučený kandidát. V tomto případě je metoda růstu regionů segmentovací metodou úplnou, protože každý pixel v obraze bude klasifikován do nějakého regionu podobných bodů. Celkový počet regionů v obraze je závislý kritériu homogenity a jeho prahové hodnotě. Ještě je třeba říci, že výsledky segmentace jsou závislé i na poloze počátečních semen. Pro různé počáteční podmínky mohou být výsledky různé.

Obr. 9. Příklad segmentace obrazu metodou růstu regionů v RGB barevném prostoru. Vlevo je původní obraz, uprostřed je segmentovaný obraz pro prahovou hodnotu $t_{min} = 30$ a vpravo je segmentovaný obraz pro prahovou hodnotu $t_{min} = 50$.

V druhém případě, při nesplnění kritéria homogenity, není další region založen a pokračuje se ve výpočtu aktuálního regionu. V tomto případě segmentujeme obraz pouze částečně a výsledný počet regionů bude maximálně odpovídat počtu počátečních semen.

Oba přístupy mají své výhody i nevýhody. Výhoda částečné segmentace je v její rychlosti, zvláště pokud umístíme počáteční bod do oblasti, o které jsme přesvědčeni, že patří k námi hledanému objektu. Nejčastěji to bývá spodní kraj obrazu, kde očekáváme cestu. Takto používaná metoda růstu regionů je používána v mnoha aplikacích navigace mobilních robotů v indoor prostředích a na silnicích [4]. Oproti metodě prahování je výhoda metody růstu regionů v lokální spojitosti segmentovaného objektu v obraze. Tato

výhoda ovšem platí pouze pro homogenní typy povrchu. Tento přístup se neosvědčí, pokud se pohybujeme v členitém terénu, ve kterém se mohou vyskytnout různé překážky, nebo pokud se barevné vlastnosti povrchu mění skokově.



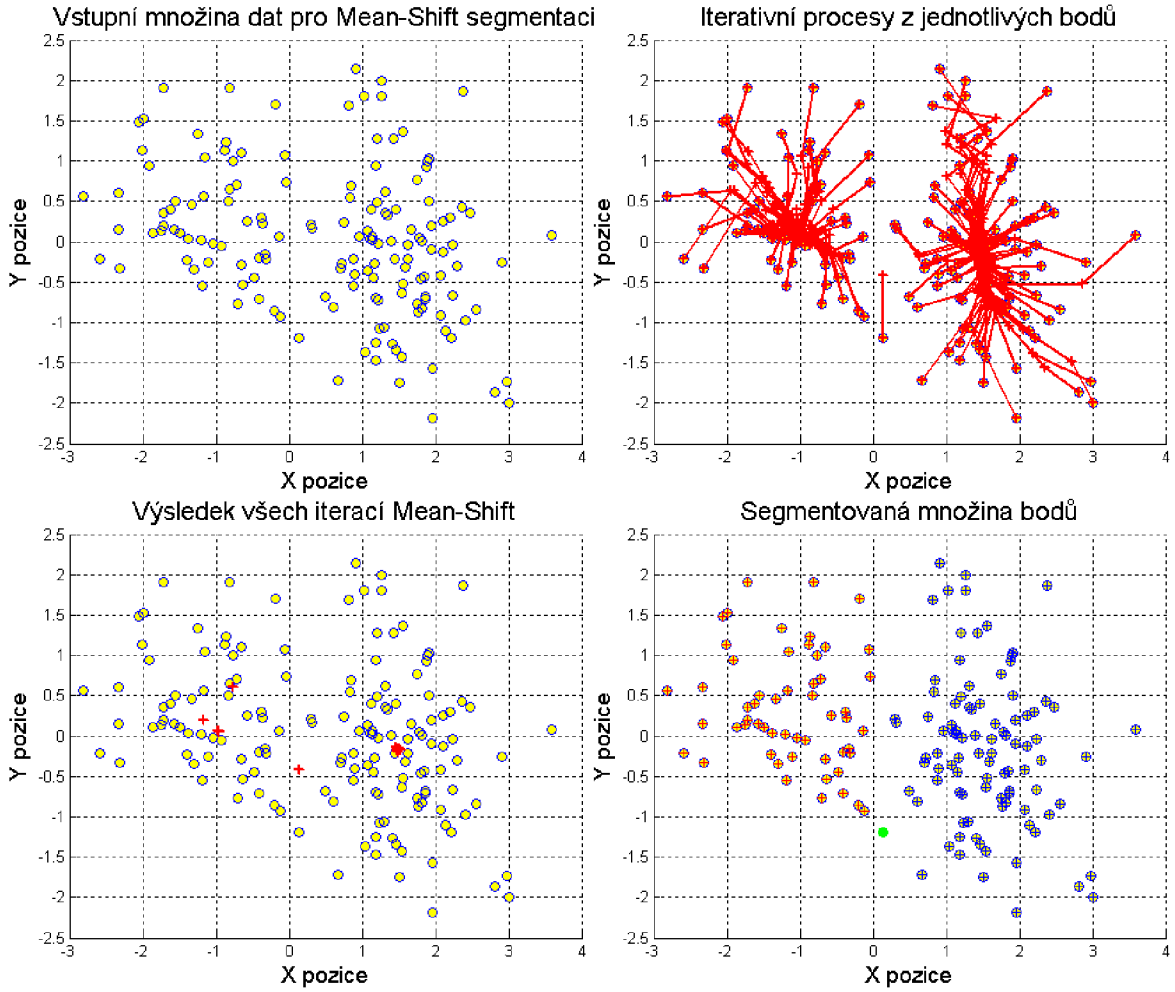
Obr. 10. Příklad částečné segmentace metodou růstu regionů v c1c2c3 barevném prostoru. Levý sloupec zachycuje původní obraz, prostřední sloupec je obraz převedený do c1c2c3 barevného prostoru, pravý sloupec ukazuje částečnou segmentaci růstem regionů, kde červené body jsou počáteční semínka. Vrchní řada ukazuje dobrý výsledek segmentace cesty. Spodní řada ukazuje špatný výsledek segmentace vlivem nespojitosti barevných vlastností cesty.

1.5.3. Metoda segmentace založená na algoritmu Mean-Shift

Algoritmus Mean-Shift je neparametrická iterativní metoda, kterou můžeme s úspěchem použít v různých aplikacích, např. ve filtraci a segmentaci obrazu nebo ke stopování objektů v obraze (object tracking).

Zjednodušeně lze metodu segmentace algoritmem mean-shift popsat jako iterativní metodu, která v každé iteraci pro určitou pozici počítá gradient hustoty bodů v blízkém okolí a pohybuje touto pozicí ve směru gradientu tak dlouho, dokud není dosaženo lokálního maxima (lokální střední hodnoty). Tento iterativní proces je inicializován v každém bodě a jeho výsledkem jsou souřadnice lokálního maxima, ve kterém se iterativní proces zastavil. Body, které konvergují do stejného lokálního maxima,

považujeme za členy stejné oblasti v obraze. Princip algoritmu na dvourozměrných datech je názorně ilustrován na obr. 11.



Obr. 11. Obrázek ilustruje mean-shift segmentaci pro data v dvourozměrném prostoru. Vlevo nahoře je obrázek vstupní množiny dat, kterou chceme segmentovat. Obrázek vpravo nahoře zobrazuje průběh iterací pro všechny body. Obrázek vlevo dole zobrazuje konečné pozice, ve kterých se iterační procesy zastavily. Konečně obrázek vpravo dole zobrazuje segmentovaná data. Zelený bod se nezařadil do žádné z množin a jeho přiřazení proběhne v postprocessu.

Metoda vychází z odhadu pravděpodobnosti hustoty bodů $f(x)$ v okolí současné pozice definované kernelovým jádrem o rozměru $(2h+1)^2$ v obecném n-rozměrném prostoru. Gradient této hustoty $\nabla f(x)$ udává vektor směřující k vyšší hustotě bodů, přičemž nejvyšší hustota bodů je právě v lokálním maximu, kde je velikost gradientu

nulová. Obecný zápis odhadu hustoty pravděpodobnosti pro bod v obecném n-rozměrném prostoru je

$$f(x) = \frac{1}{N}\sum_{i=1}^{N} K(x_i - x) \tag{10}$$

kde body x, x_i jsou body v n-rozměrném prostoru, N je počet bodů x_i v okolí bodu x a kde $K(x)$ je radiálně symetrická funkce označovaná jako jádro nebo též kernel. Vzhledem k vlastnostem kernelu (radiální charakter, symetričnost) je často ve vzorcích výhodnější použít tzv. profil kernelu, který počítá s délkou vektoru.

$$K(x) = c_k k\left(\|x\|^2\right) \tag{11}$$

kde c_k je normalizační konstanta a $\|x\|$ je délka vektoru x. Mezi tři nejpoužívanější typy kernelu patří uniformní jádro, epanečnikovo jádro a gaussovo jádro, jejichž vlastnosti byly podrobně popsány v literatuře. Pro potřeby diplomové práce bylo zvoleno epanečnikovo jádro, které je kombinací mezi rychlostí uniformního jádra a přesností gaussova jádra.

$$K_e(x) = c_k k_e\left(\|x\|^2\right) = \begin{cases} c_k\left(1 - \|x\|^2\right) & \text{pro } \|x\| \le 1 \\ 0 & jinak \end{cases} \tag{12}$$

Z rovnice (12) je patrné, že profil kernelu je definován jen na intervalu $\langle 0,1 \rangle$. Z tohoto důvodu se všechny body použité pro výpočet normalizují šířkou jádra h, která definuje velikost okolí bodu x. Dosazením šířky kernelu h a rovnice (12) do rovnice (10) dostáváme tvar

$$f(x) = \frac{c_k}{N} \cdot \sum_{i=1}^{N} k\left(\left\|\frac{x_i - x}{h}\right\|^2\right) \tag{13}$$

Derivace odhadu rozložení hustoty:

$$\nabla f(x) = \frac{2c_k}{N \cdot h^2} \cdot \sum_{i=1}^{N} (x_i - x)\, k'\left(\left\|\frac{x_i - x}{h}\right\|^2\right) \tag{14}$$

při úpravě a zavedení substituce $g(x) = -k'(x)$ vztah (14) přejde na tvar:

$$\nabla f(x) = \frac{2c_k}{N \cdot h^2} \cdot \sum_{i=1}^{N} g\left(\left\|\frac{x - x_i}{h}\right\|^2\right) \cdot \left[\frac{\sum_{i=1}^{N} g\left(\left\|\frac{x - x_i}{h}\right\|^2\right) \cdot x_i}{\sum_{i=1}^{N} g\left(\left\|\frac{x - x_i}{h}\right\|^2\right)} - x\right] \tag{15}$$

Toto je výsledný obecný vztah pro výpočet gradientu hustoty uvedený v literatuře [8]. Vztah se v podstatě skládá ze dvou částí. Výraz v hranaté závorce je vektor a udává směr gradientu hustoty bodů. Výraz před hranatou závorkou pak udává jeho velikost.

Pro obrazovou analýzu se často používá kombinovaný kernel, pro který se definují dvě různé šířky pásma. První se týká šířky kernelové matice v obraze a říká se jí „územní“ (spatial) šířka pásma h_s. Druhé se říká šířka pásma rozsahu h_r a definuje maximální vzdálenost bodu x_i od bodu x v n-rozměrném barevném prostoru (pro RGB n=3). Kombinovaný kernel se pak definuje jako součin dvou samostatných kernelových funkcí. Výhoda je právě v použití odlišných šířek pásma pro okolí bodu a pro barevný rozsah bodů, které považujeme za blízké.

$$K_e(x) = c_k \cdot k_s\left(\left\|\frac{x^{(s)} - x_i^{(s)}}{h_s}\right\|^2\right) \cdot k_r\left(\left\|\frac{x^{(r)} - x_i^{(r)}}{h_r}\right\|^2\right) \tag{16}$$

Nyní nastávají dva kvalitativně odlišné přístupy závislé na tom, jestli chceme algoritmus mean shift použít pro segmentaci dat nebo jejich filtraci a vyhlazení.

Segmentace

Pokud chceme obrazová data algoritmem mean-shift segmentovat, musíme pro každý obrazový bod provést iterativní výpočet rovnice (15). Pro výpočet používáme jak pozice okolních bodů $x_i^{(s)}$ v obraze, tak i jejich barevné hodnoty, ale výsledný gradient počítáme pouze pro pozici v obraze. Jinými slovy upravujeme pouze souřadnice v obraze a barevné hodnoty aktuálně vzatého bodu jsou po celou dobu výpočtu stejné. Výsledkem rovnice (15) bude dvourozměrný vektor.

Výpočet provádíme tak dlouho, dokud velikost tohoto vektoru nebude menší než námi definované ε. Poté považujeme výpočet bodu za uzavřený, uložíme aktuální pozici bodu do které algoritmus zkonvergoval a celý výpočet opakujeme pro následující obrazový bod. Výsledky algoritmu ukazuje obr. 12.

Obr. 12. Příklad segmentace algoritmem Mean-Shift. Vpravo je původní obraz, vprostřed je segmentovaný obraz s hodnotami $h_s = 3$ a $h_r = 40$, vlevo je obraz segmentovaný s hodnotami $h_s = 6$ a $h_r = 25$. Pro oba obrazy bylo použito $\varepsilon = 0{,}02$. Zpracované obrazy obsahují barevné hodnoty bodů, do kterých algoritmus z dané pozice zkonvergoval.

Výstupem ze segmentace pomocí algoritmu mean-shift je dvourozměrné pole dvojicí čísel o velikosti původního obrazu. Každá dvojice čísel na dané pozici odpovídá místu, kam algoritmus mean-shift z této pozice zkonvergoval. Tyto výsledky je třeba dále zpracovat.

Oblasti v obraze, které jsou si z hlediska svých vlastností podobné, budou mít výsledky mean-shift algoritmu koncentrovány na jednom místě. Této oblasti se říká „basin of attraction“ [8] (mísa atrakce). Pak je možné říci, že dva body mají stejnou mísu atrakce, pokud splňují následující podmínku:

$$dist\left(z_i, z_j\right) \leq 2 \cdot h_s \tag{17}$$

kde z_i a z_j jsou dvourozměrné souřadnice výsledků mean-shift algoritmu bodů, které právě posuzujeme, $dist(z_i, z_j)$ je euklidovská vzdálenost určující vzdálenost těchto dvou bodů a h_s je územní šířka pásma, kterou jsme použili pro segmentaci. Tímto způsobem klasifikujeme každý bod podle jeho výsledku do příslušné množiny sdílející stejnou mísu atrakce. Počet těchto mís se liší podle hodnot použitých parametrů při segmentaci. Nakonec segmentovací procedury se řadí podmínka, aby každá mísa atrakce (oblast obrazu) měla větší počet bodů než námi zvolené minimum. Pokud tuto podmínku nesplní, je přiřazena k nejbližší oblasti a přebírá její hodnoty. Toto opatření eliminuje všechny malé oblasti v obraze (například i šum).

Výsledkem jsou tedy množiny bodů v obraze, které sdílejí stejnou mísu atrakce. Nyní je třeba rozhodnout, zda oblasti bodů náleží do objektu cesta či nikoliv. K tomu nám

opět slouží kritérium vyjádřené rovnicí (5), kdy porovnáváme průměrné hodnoty barevných složek všech bodů náležící stejné oblasti s referenčním bodem poskytnutým adaptivní databází.

Výpočetní čas metody segmentace pomocí mean-shift algoritmu se pohybuje řádově ve stovkách sekund, proto se jeho použití zatím omezuje pouze na offline zpracování obrazu. Nutno říci, že nebyly provedeny žádné optimalizace ani implementovány některé zrychlující techniky obsažené v literatuře.

Filtrace dat

Pokud obrazová data budeme algoritmem mean-shift filtrovat, postup je obdobný pouze s tím rozdílem, že iterativní výpočet rovnice (15) budeme provádět pro barevné hodnoty a pozice v obraze bude konstantní. Výsledkem rovnice (15) tedy bude třírozměrný vektor. Výpočet opět provádíme tak dlouho, dokud hodnota gradientu nepoklesne pod definované ε. Výsledky algoritmu ukazuje obr. 13.

Obr. 13. Příklad filtrace algoritmem Mean-Shift. Vpravo původní obraz, vlevo filtrovaný obraz s použitými hodnotami $h_s = 10$ a $h_r = 8$.

1.6. Datová struktura segmentovaných dat

Výsledná segmentovaná data mají určitou strukturu, která je závislá na použité metodě segmentace a která vyplývá z charakteru metody. Tyto data je třeba z hlediska dalšího zpracování převést na takovou strukturu, která je pro další zpracování nejvýhodnější.

Výstupem ze segmentace prahováním je dvourozměrné bitové pole, nebo též bitová maska o rozměru původního obrazu. Pozice v tomto dvourozměrném poli odpovídá pozici v původním obraze a může nabývat hodnoty 1, pokud je bod klasifikován jako cesta, nebo hodnoty 0, pokud je tomu naopak. Tato datová reprezentace je z hlediska rychlosti dalšího zpracování vhodná.

Výstupem ze segmentační metody růstu regionů je seznam, který obsahuje informace o jednotlivých členech regionu, o střední hodnotě barevných hodnot celého regionu a o střední hodnotě pozic všech bodů, kteří jsou členy regionu (těžišti). U každého regionu je pak třeba určit, zda patří do objektu cesta či nikoliv. Opět nám může posloužit kritérium příbuznosti dané v rovnici (5), kam místo jednotlivých bodů obrazu budeme dosazovat střední hodnotu barev regionů. Výslednou datovou strukturou bude opět bitové pole. Rozdíl oproti předchozí metodě je v segmentovaných datech, která jsou spojitá a neobsahují takové množství malých samostatných uzavřených oblastí, které je třeba z obrazu odstraňovat, což šetří čas v postprocesingu. Nicméně metoda sama je výpočetně náročnější než metoda prahování.

Výstupem ze segmentace pomocí algoritmu mean-shift je vlastně také seznam množin bodů, které náleží stejné oblasti. Tato struktura může být dále zpracovávána. V kapitole o mean-shift segmentaci bylo popsáno, jak se s touto strukturou nakládá, ale existují i jiné přístupy. V literatuře [7] se objevuje technika „graph-cuts“, která tuto strukturu převede na graf, kde uzly jsou průměrné barevné hodnoty množiny a hrany jsou euklidovské vzdálenosti mezi nimi. S pomocí referenčního bodu se dá tato struktura rozčlenit na bitový obraz cesta/pozadí.

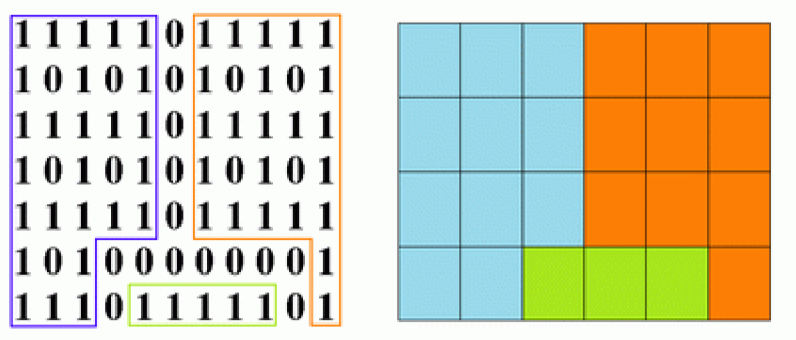
1.7. Postprocessing

Termínem „postprocessing“ máme na mysli dodatečné zpracování dat. Do této fáze patří úprava výstupní datové struktury ze segmentační metody, oprava špatně klasifikovaných pixelů, slučování malých oblastí v obraze s většími a zaplňování děr.

Úprava obrazu po segmentaci se zpravidla liší dle použité datové struktury, ve které jsou výsledky ze segmentovací metody uchovávány a které je nutné zpracovat na vhodný formát.

Jako výhodnou datovou strukturou pro další zpracování je bitové pole. Toto pole může být jednoduché, kdy rozměr bitového pole odpovídá rozměrům obrazu a hodnoty v poli pouze vyjadřují svou příslušnost k hledanému objektu (ano/ne). Tato struktura je například výsledek segmentace prahováním.

Pokud ale máme v obraze více oblastí, které je třeba dále zpracovat, je výhodné použít bitové pole o dvojnásobné délce. V tomto poli liché sloupce a řádky odpovídají bodům v obraze a jsou nastaveny na hodnotu 1 a bitové hodnoty mezi nimi vyjadřují, zda tyto body patří do stejné segmentované množiny či nikoliv. Příklad takové bitové struktury je ilustrován na obr.14.



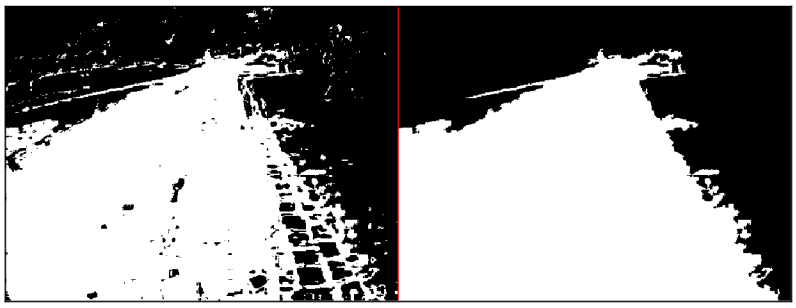
Obr. 14. Příklad použití rozšířené bitové struktury pro náročnější postprocessing.

1.7.1. Vyplňování děr v obraze

Vlivem šumu nebo nehomogenity prostředí se v segmentovaném obraze vyskytují špatně klasifikované oblasti pixelů. Leží-li na cestě nějaká malá překážka (například spadlý list) segmentační algoritmus ji nerozpozná jako cestu. Z tohoto důvodu se do fáze postprocessingu zařazuje algoritmus „vyplňování děr“ (z angl. hole fill). Tento algoritmus

rozpozná v segmentovaném obraze spojené oblasti se stejnými vlastnostmi metodou connected component labeling (volně přeloženo jako „označování spojených částí“) a pokud je počet členů (pixelů) v této oblasti menší než zvolené minimální kritérium, oblast je přiřazena k nejbližší větší oblasti a přebírá její hodnoty. Tato technika se používá k obecnému sloučení malých oblastí v obraze s velkými.

Connected component labeling je algoritmickou aplikaci teorie grafů, která všem propojeným částím obrazu přiřadí stejné označení. Dva sousední pixely považujeme za propojené, pokud byly segmentačním algoritmem klasifikovány do stejné množiny. Výstupem z této procedury je seznam všech propojených oblastí v obraze včetně seznamu bodů, které do této oblasti náleží.



Obr. 15. Příklad využití algoritmu zaplňování děr v obraze.

1.7.2. Dilatace – Eroze

Metody dilatace a eroze se používají hlavně pro dvousložkový obraz. Jedná se o základní metody matematické morfologie, která se používá pro zpracování digitálního obrazu. Podstata dilatace a eroze je jednoduchá. Dilatace je proces, který přiřadí zkoumanému bodu v obraze maximální hodnotu bodu z blízkého okolí, proto se mu též říká maximální filtr. Eroze je opakem dilatace a přiřazuje aktuálnímu bodu minimum z blízkého okolí bodů a nazývá se tedy minimálním filtrem.

Matematický zápis je následující. Označme obrazovou matici o rozměru $w \times h$ jako $A\left[a_{i,j}\right]$ a označme $O_{i,j}$ okolí bodu $a_{i,j}$. Výsledný zpracovaný obraz označme $B\left[b_{i,j}\right]$. Položíme-li všem prvkům matice B podmínku $b_{i,j} = \max O_{i,j}$, pak řekneme, že obraz A byl zpracován maximálním filtrem. Platí-li $b_{i,j} = \min O_{i,j}$, pak byl obraz zpracován minimálním filtrem.

Metody dilatace a eroze se v digitálním zpracování obrazu používají například k určení hranice objektu. Označme původní obraz jako A, dilatovaný obraz jako A_D a erodovaný obraz jako A_E. Pak hledanou hranici objektu získáme odečtením obrazů od sebe $B = A_D - A_E$.

Obr. 16. Ukázka získání hranice segmentovaného objektu. Vlevo je původní obrázek, uprostřed segmentovaný objekt po odstranění malých oblastí, vpravo hranice objektu.

Dalším využitím metod dilatace a eroze je v odstranění šumu, artefaktů a špatně klasifikovaných bodů, nebo propojování oddělených oblastí. Kombinaci metod se říká „otevření“, pokud na obraz použijeme metodu eroze a následně dilatace v tomto pořadí. Tato kombinace odstraňuje malé oblasti a artefakty. Opačný postup, posloupnost metod dilatace a následně eroze, se nazývá „uzavření“ a slouží ke spojování blízkých objektů v obraze v jeden větší celek.

2. IMPLEMENTACE VYBRANÝCH METOD

V této části se budu zabývat výběrem jednotlivých metod pro řešení zadaného problému. Postupně proberu schéma na obr. 1. a přiřadím jednotlivým blokům konkrétní funkce včetně implementačního řešení.

2.1. Preprocessing

2.1.1. Odstranění horizontu

Účelem algoritmu je rozpoznat cestu. Z tohoto důvodu je zbytečné, aby zpracovávaný obraz obsahoval data, která se nacházejí nad horizontem. Jinými slovy, obraz obsahující oblohu nám k rozpoznání cesty nepomůže. Pokud známe přibližné umístění horizontu v obraze, můžeme ho z dalšího zpracování vypustit.

Existuje několik technik detekce výšky horizontu pro silniční vozidla [9]. Nám stačí přijmout předpoklad, že povrch, po kterém se robot pohybuje, je relativně plochý. Pozice horizontu v obraze je potom konstantní a je dána konstrukčními parametry robota. Je dobré podotknout, že například pro detekci lidí nebo obličejů v obraze by bylo nutné obraz zachovat celý.

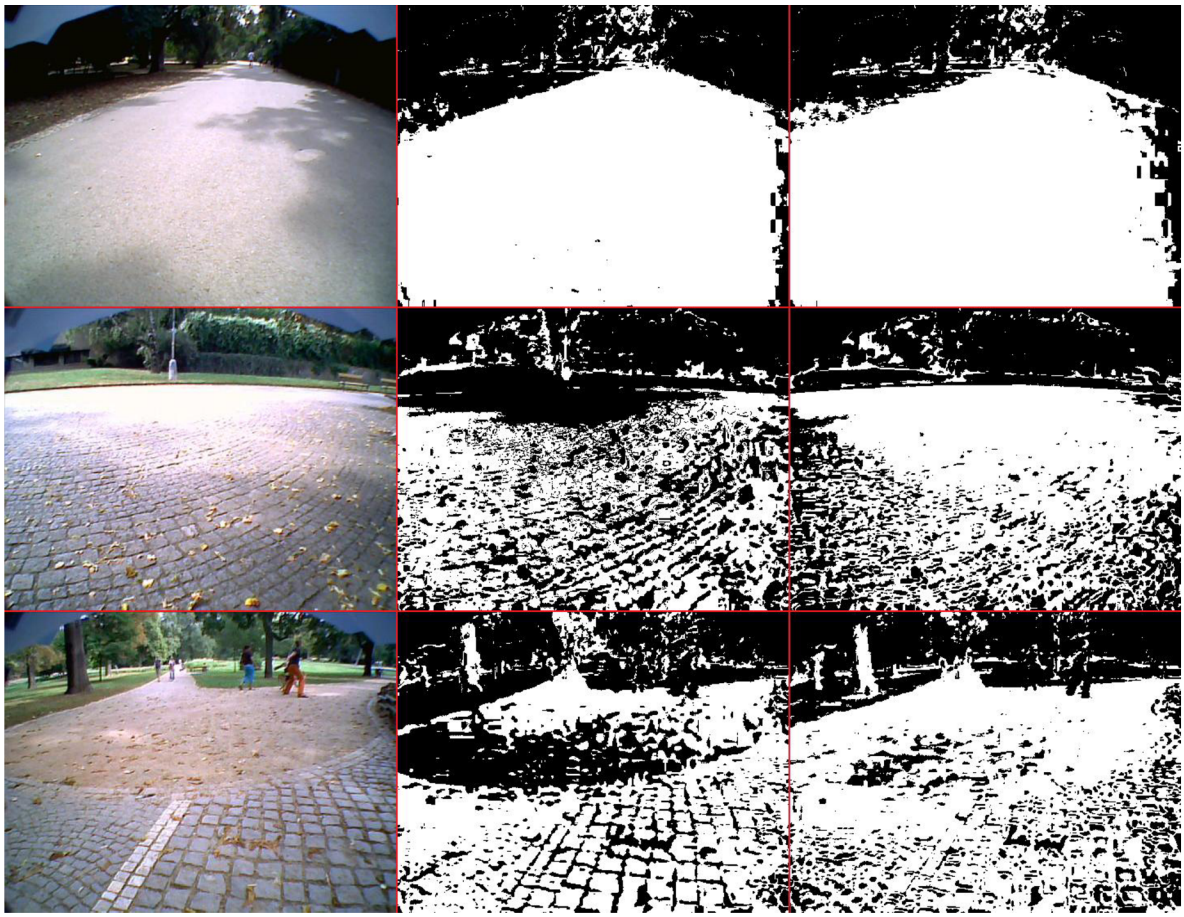
2.1.2. Filtrace šumu

Jako vhodná metoda filtrace šumu byla vybrána výše popsaná konvoluce gaussovým jádrem.

2.2. Výběr barevného prostoru

Srovnání proběhlo mezi dvěma barevnými prostory, HSI a c1c2c3. Jejich vlastnosti byly zkoumány na prahovací segmentovací metodě. Referenční bod byl pečlivě vybrán jako střední hodnota barev z trapézového regionu před robotem (viz kapitola o inicializaci robotu). Pro cesty s homogenním povrchem jsou výsledky obou prostorů srovnatelné, dokonce HSI se ukazuje jako mírně lepší. Proto HSI barevný prostor bývá často používán pro aplikace jízdních asistentů nebo přímo pro navigaci automobilu [1]. Bohužel HSI má určité potíže při segmentaci míst v obraze s nízkou saturací barvy. Vyplývá to z jeho cylindrického charakteru. Při nízké saturaci nebo intenzitě, tedy při hodnotách odpovídající stupňům šedi, je barevná (Hue) složka nestabilní. Řeší se to implementací dodatečných

podmínek, kdy algoritmus rozlišuje chromatické a achromatické barvy na základě hodnot saturace a intensity. HSI prostor bohužel neobstál ani v komplexních podmínkách přechodu mezi jednotlivými typy povrchu. Srovnání obou prostorů ukazuje Obr. 17.



Obr. 17. Porovnání výsledků segmentace pro barevné prostory HSI a c1c2c3. Sloupec vlevo je originální obrázek, prostřední sloupec zachycuje segmentaci v HSI barevném prostoru a sloupec vpravo ukazuje segmentaci v c1c2c3 barevném prostoru. Horní řada zachycuje asfaltovou cestu, prostřední řada zachycuje obraz s oblastí s nízkou saturací a spodní řada ukazuje změnu typu cesty. Bíle jsou označeny body, které algoritmus rozpoznal jako cestu.

Výpočetní čas potřebný na konverzi obrazu z RGB prostoru do HSI je zhruba dvakrát až třikrát rychlejší než konverze do prostoru c1c2c3, což je dáno jednodušším transformačním vztahem. Na druhou stranu je výpočetní čas segmentace v HSI prostoru delší kvůli cylindrické metrice HSI (rovnice (7) a (8)), takže výpočetní náročnost obou

barevných prostorů je ve výsledku stejná. Ve své práci jsem se rozhodl používat barevný prostor c1c2c3 pro jeho robustnost.

2.3. Výběr segmentační metody

Za nejvhodnější metodu segmentace považuji pro daný problém metodu prahování. Tato metoda má výhodu ve své jednoduchosti a robustnosti. Výpočetní čas je oproti metodě rostoucích regionů a metodě mean-shift krátký. Tato vlastnost je pro použití na mobilním robotu s omezenými výpočetními zdroji rozhodující.

Metoda růstu regionů dosahuje v dílčích případech z hlediska segmentace lepších výsledků než metoda prahování. Jestliže se jedná o obraz s homogenní cestou (např. asfaltovou) a metoda růstu regionů je použita jako částečná segmentace s vhodně umístěnými počátečními podmínkami (semeny), tak v tomto specifickém případě je výsledkem spojitý region zcela popisující cestu. Celkový výpočetní čas metody růstu regionů je ale větší než u metody prahování. Podstatné zlepšení výpočetní rychlosti lze dosáhnout optimalizacemi. Navíc výstupní segmentovaný obraz obsahuje jen malé množství děr v obraze a téměř žádné špatně klasifikované body. Tento fakt šetří čas v postprocessingu, protože tyto malé oblasti a špatné pixely není nutné opravovat.

Bohužel segmentace metodou rostoucích regionů havaruje v případě, že povrch cesty není homogenní a je-li například kombinací více typů cest, nebo je-li na povrchu cesty rozptýlen nějaký barevně odlišný materiál (kamení, listí). V tomto případě bychom museli rozsegmentovat celý obraz, což by mělo za následek další zvýšení výpočetního času.

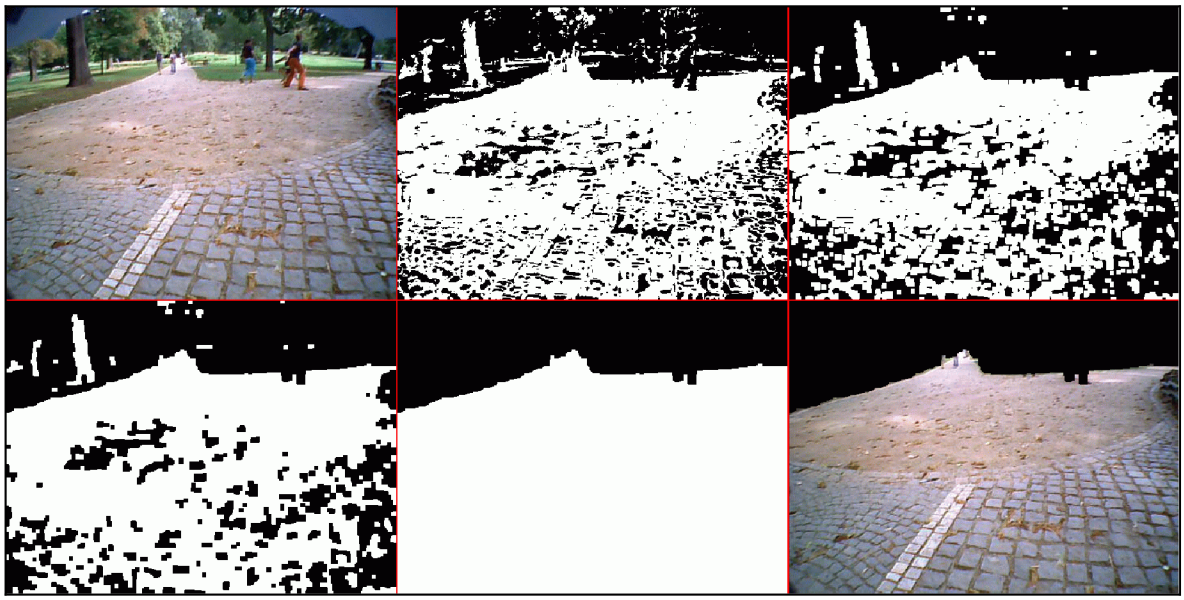
Obr. 18. Porovnání segmentačních metod rostoucích regionů a prahování. Obrázek vlevo zobrazuje cestu s navátým listím, prostřední obrázek je výsledek segmentace rostoucích regionů a obrázek vpravo je výsledek segmentace prahováním.

2.4. Postprocessing

Konkrétní podoba metod v postprocessingu je závislá na volbě segmentační metody. Pro námi zvolenou metodu prahování se jako nejlepší způsob ukázalo nejdříve aplikovat na obraz kombinaci metod dilatace a eroze a ve výsledném obraze odstranit malé objekty algoritmem vyplňování děr.

Pořadí metod dilatace a eroze aplikovaných na segmentovaná data závisí na velikosti prahovacího kritéria t_{diss}. Pokud je toto kritérium „malé“, tak celkový počet bodů klasifikovaných jako cesta je méně než v ideálním případě, protože ne všechny body jsou kvůli nízkému prahu do cesty zahrnuty. V tomto případě je výhodné aplikovat nejdříve metodu uzavření (dilatace-eroze), která spojí blízké body a vytvoří větší celky, a až následně metodu otevření (eroze-dilatace), která odstraní z obrazu artefakty a šum.

Pokud je prahovací kritérium „velké“, tak lze očekávat, že jako cesta budou klasifikovány i body, které do objektu cesta nepatří. V tomto případě se použije postup opačný a nejdříve je aplikována metoda otevření a následně uzavření. Experimenty ukázaly, že použití většího prahovacího kritéria dává lepší výsledky.



Obr. 19. Ukázka postprocessingu pro metodu prahování (t_{diss}=20). Nahoře vlevo je původní obraz, nahoře uprostřed je segmentovaný obraz, nahoře vpravo je obraz po otevření, dole vlevo je obraz po metodě uzavření, dole uprostřed je obraz po odstranění malých objektů a konečně vpravo dole je zobrazen celkový výsledek postprocessingu prolnutý s původním obrazem.

Takto dilatací a erozí upravený obraz je předán algoritmu vyplňování děr, který z obrazu odstraní všechny objekty, jejichž velikost je menší než minimální počet bodů. Toto kritérium jsem zvolil na jednu dvacetinu celkového počtu bodů v obraze.

2.5. Adaptivní databáze

Adaptivní databáze uchovává jeden nebo více reprezentativních bodů cesty, které se používají při segmentaci pro rozlišení cesty od pozadí, a vybírá nejvhodnějšího kandidáta pro aktuální segmentační krok. Inteligentní správou adaptivní databáze můžeme dosáhnout dobrých výsledků segmentace. Jedním z přístupů je uchovávat všechny použité reprezentanty a vyčlenit tak v barevném prostoru oblast odpovídající všem cestám, po kterých robot jel. Nevýhoda tohoto přístupu se projeví, pokud robot projede takovým prostředím, kde barevné hodnoty pozadí budou odpovídat některému z typu cest v databázi. Druhým z přístupů je uchovávat v databázi vždy dvojici bodů reprezentujících cestu i pozadí. Realizací adaptivní databáze může být celá řada a v budoucnu mohou být jednoduše implementovány do robota jako samostatný inteligentní softwarový modul.

Za dostatečné a robustní řešení považuji extrahovat reprezentanta z aktuálního průchodu algoritmu a použít ho pro příští segmentaci obrazu. Samotné získání reprezentativního bodu probíhá tak, že se zprůměrují všechny hodnoty barevných složek všech bodů klasifikovaných jako cesta. Po testování tohoto přístupu jsem zavedl podmínku, že počet bodů, ze kterých je reprezentant získáván musí být větší než určité minimální množství. Testoval jsem i jiné metody správy adaptivní databáze, ale žádná se bohužel neukázala jako dostatečně robustní.

Adaptivní databázi je nutné při spuštění robota inicializovat.

Obr. 20. Obrazové body použité pro extrakci reprezentanta cesty.

2.6. Inicializační fáze

Na celý algoritmus segmentace obrazu se dá dívat jako na zpětnovazební systém. Databáze reprezentantů poskytne kandidáta pro segmentaci, jejíž výsledkem je zpracovaný obraz, který poskytne nového kandidáta. Pro první segmentaci však tento reprezentant chybí. Z toho důvodu je třeba pro tento systém nastavit počáteční podmínky a k tomuto účelu slouží inicializační fáze robota.

Předpokládejme, že robot při své inicializaci stojí na cestě. Tento předpoklad je nutný pro získání reprezentanta, který se získá jako střední hodnota bodů z trapézovité oblasti před robotem. Tento postup bývá často používán v literatuře [1,5]

Obr. 21. Obrázek se zvýrazněným trapézovitým regionem v oblasti před robotem.

Další experimenty se týkaly volby prahovacího kritéria. Byla odzkoušena metodika porovnávání výsledků segmentace pro různá prahovací kritéria s výsledky jiných segmentovacích algoritmů, nebo s obrazem upraveným gradientním filtrem. Idea byla založena na předpokladu, že vhodná volba prahovacího kritéria vytvoří takový segmentovaný obraz, jehož hrany budou korespondovat s hranami vytvořenými gradientním filtrem nebo jinou segmentační metodou. Tento postup však nedával dobré výsledky, proto jsem se rozhodl nastavit prahovací kritérium na pevnou hodnotu $\mathbf{t_{diss}} = 20$, což je dostatečné, jednoduché a zároveň robustní řešení.

2.7. Interpretace segmentovaných dat

Navigace robota Bender II na soutěži Robotour 2009 se skládala ze dvou částí dle situace, ve které se robot nacházel. Pokud byl robot na křižovatce nebo v otevřeném prostoru, byl primární důraz kladen na navigaci pomocí GPS souřadnic. Pokud se však robot nacházel na rovném úseku cesty, nepřesnost GPS by mohla způsobit, že by robot sjel z cesty. Proto byla navigace na takovýchto úsecích primárně zajišťována daty z kamery.

Segmentovaná data ve formátu bitového pole jsou použita v plánovači cesty mobilního robota. Ze segmentovaných dat můžeme určit volný prostor před robotem a vypočítat úhel pro navigaci. Hlavní důraz je kladen na požadavek, aby se robot udržel na cestě. Dále je možno vzhledem k prostorovému rozložení dat říci o jakou topologii cesty se jedná. Můžeme rozlišovat tyto typy: otevřený prostor, křižovatka, rovná cesta.

2.7.1. Extrakce navigačního úhlu robota

Pro extrakci navigačního úhlu robota jsem navrhl metodu inspirovanou prací [5]. Jedná se o ohodnocení předpočítaných trajektorií robota, promítnutých do segmentovaného obrazu. Na rozdíl od [5] předpokládejme, že robot se pohybuje malou rychlostí. To nám umožní přijmout zjednodušení, že výsledek segmentace obrazu má robot k dispozici se zanedbatelným zpožděním. Toto zpoždění se nijak neprojeví na navigaci robota. Druhým zjednodušením je, že trajektorii robota považujeme za přímku. Dále předpokládejme, že kamera je připevněna ve střední rovině robota, takže svislá osa obrazu může být považována za projekci aktuální trajektorie robota do obrazu.

Nyní je potřeba formulovat problém navigace robota kvantitativně. Nechť $w(\alpha)$ je váhová funkce pro trajektorii v obraze ve směru určeném úhlem natočení kol α.

$$w(\alpha) = d(\alpha)^k \tag{18}$$

kde $d(\alpha)$ je funkce, která určuje součet všech bodů v obraze, které jsou v daném směru klasifikovány jako cesta. Jedná se o jistou analogii s maximální vzdáleností, kterou může robot v daném směru definovaném úhlem α ujet. Konstanta k je nelinearita zvýhodňující větší $d(\alpha)$ v obraze. Jinými slovy úhel, ve kterém má robot šanci ujet větší vzdálenost, bude mít větší váhu než úhel s menší ujetou vzdáleností. Funkce $d(\alpha)$ je spočítána následovně:

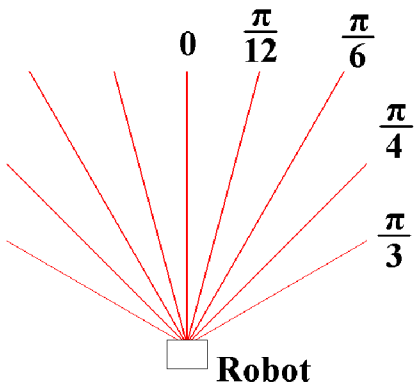
$$d(\alpha) = \sum B(x, y) \tag{19}$$

Funkce *B(x,y)* nabývá hodnot 1, pokud je bod na souřadnicích *x,y* v bitové mapě klasifikován jako cesta, a nabývá hodnoty 0 pokud je tomu naopak. Souřadnice *x, y* počítáme vždy pro určitý úhel α:

$$\begin{aligned} y &= height - u \\ x &= \frac{width}{2} + v \\ v &= u \cdot \tan(\alpha) \end{aligned} \tag{20}$$

kde *u* a *v* udávají trajektorii robota pro daný úhel alfa. Tato trajektorie vystupuje ze spodního okraje obrazu pod daným úhlem. *width* a *height* v rovnici označují šířku a výšku obrazu.

Nyní jsme převedli problém navigace robota na optimalizační úlohu, ve které hledáme takový úhel, jehož váha bude pro daný obraz největší. Řešení této úlohy existuje nekonečně mnoho a výpočet by byl časově náročný. V našem případě se spokojíme s množinou předpřipravených trajektorií v paměti robota, které dostatečně pokryjí manévrovací prostor.



Obr. 22. Testované trajektorie pro získání navigačního úhlu robota.

Provedeme tedy výpočet vah *w(α)* pro všechny tyto trajektorie. Výsledný navigační úhel vypočítáme jako vážený průměr úhlů a jejich vah.

$$\alpha = \frac{\sum_i w(\alpha_i) \cdot \alpha_i}{\sum_i w(\alpha_i)} \tag{21}$$

Možným vylepšením této metody je možné dle [5]. Toto řešení vypočítá každému úhlu určující předdefinovanou trajektorii míru shody s navigačním úhlem, který je určen GPS souřadnicemi cíle, GPS souřadnicemi robota a aktuálním úhlem robota z kompasu.

Na váhovou funkci se pak můžeme dívat jako na součin dvou samostatných váhových funkcí. První udává výše uvedenou hodnotu vyjadřující bezkolizní vzdálenost v obraze v daném směru α. Druhá váhová funkce w_{GPS} udává hodnotu v rozmezí $\langle -1,1 \rangle$ vyjadřující shodu daného úhlu α s navigačním úhlem poskytnutým GPS navigací. Součin obou váhových funkcí můžeme napsat následovně:

$$w(\alpha) = d(\alpha)^k \cdot \cos(\alpha - \phi) \tag{22}$$

Toto řešení ale předpokládá dostatečně přesnou GPS navigaci nebo dlouhé rovné úseky cesty. Výsledky algoritmu prezentuje obr. 22.

Obr. 23. Výsledek algoritmu extrakce směru.

2.8. Použitý programovací jazyk

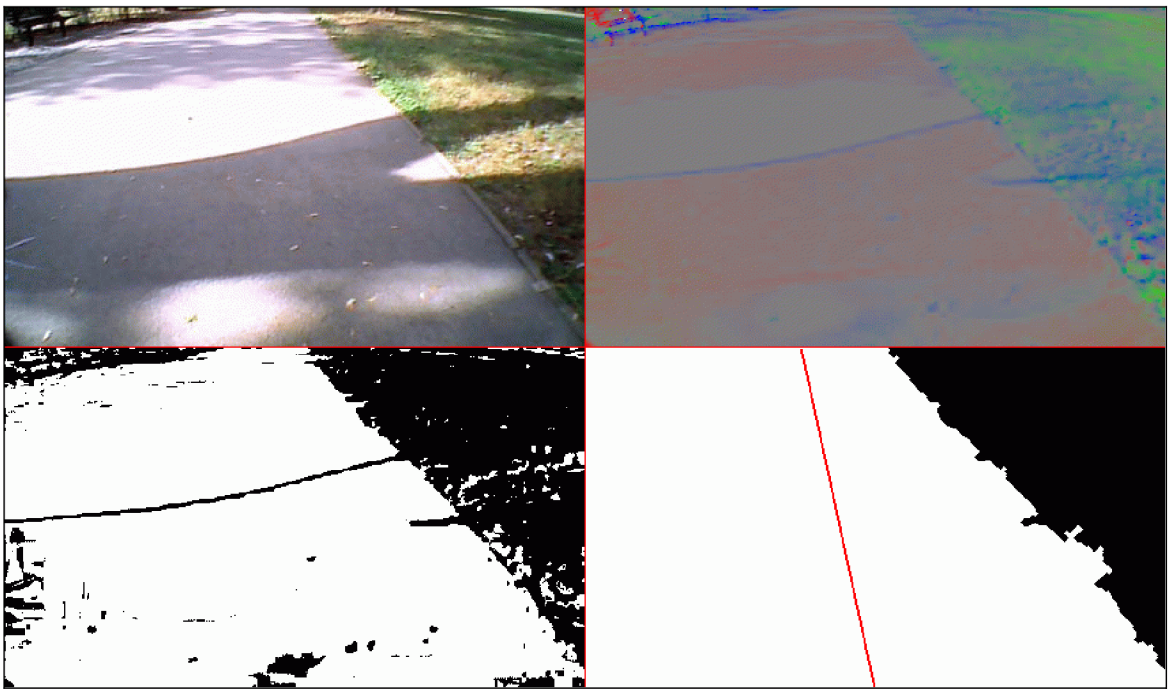
Algoritmus byl vyvíjen v jazyku Delphi a postupně přepsán do jazyka C#. Důvodem je, že hlavní řídící program pro robot Bender II je také psán v jazyce C#. Při vývoji programu byl použit soubor knihoven EmguCV [13] pro snazší přístup k datům obrazové matice a pro některé další funkce, které knihovna poskytuje. Veškerý mnou napsaný kód pro účely diplomové práce je open source.

3. EXPERIMENT NA REÁLNÝCH DATECH

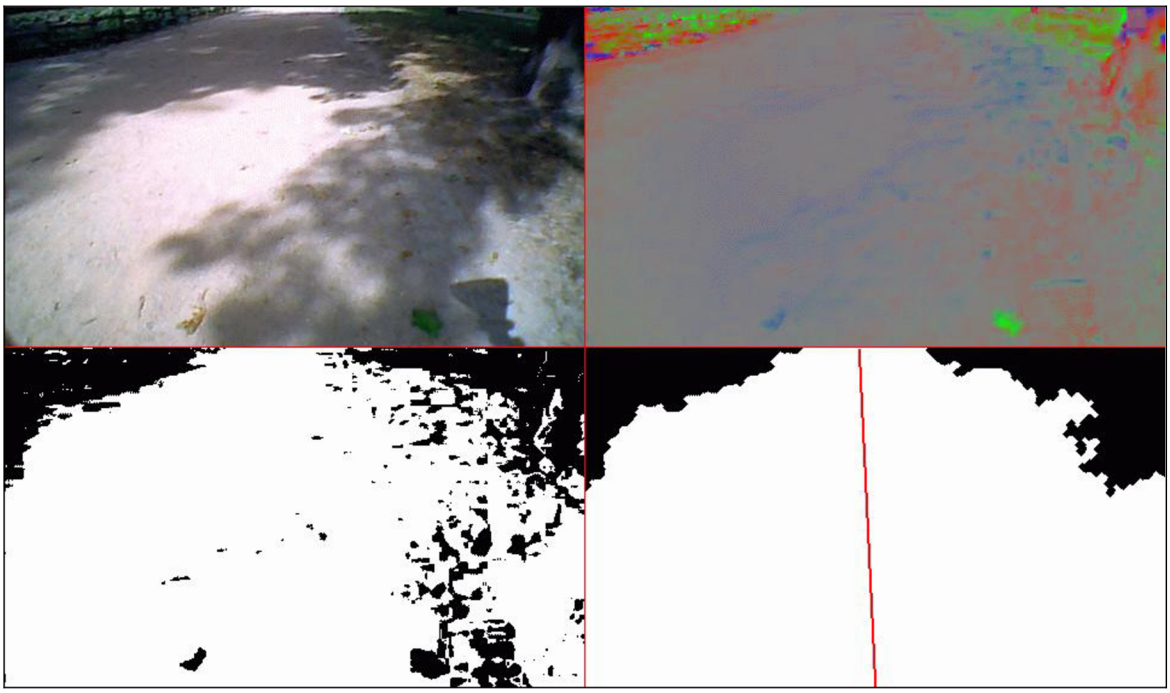
Obrázky byly získány na soutěži Robotour 2009 v Brně v Lužáneckém parku, kamerou Megapixel USB2 Wide Angle Webcam Live WB-6200p, vzorkovací frekvencí 1 snímek za vteřinu a byly použity jako vstupní data. Na následujících obrázcích jsou vidět výsledky průběhu celého algoritmu v určitých situacích. Každý obrázek se skládá ze čtyř menších obrázků, které ukazují oříznutý nezpracovaný obraz (vlevo nahoře), obraz transformovaný do c1c2c3 barevného prostoru (vpravo nahoře), segmentovaný obraz (vlevo dole) a nakonec upravený obraz s extrakcí směru (vpravo dole).

Obr. 24. zachycuje výsledek algoritmu pro obraz s ostrým přechodem světlo-stín. Algoritmus se v těchto světelných podmínkách zachoval dobře. Následující obr. 25. zachycuje nezřetelnou hranici mezi cestou a okolím a ukazuje složitost problému segmentace. Obr. 26. je průběh algoritmu s překážkou v obraze. Navigační úhel se snaží robota navést, aby se překážce vyhnul. Obr. 27. zachycuje nevýhodu ignorování některých překáže prezentovaného algoritmu, která vyplývá ze způsobu výpočtu navigačního úhlu. Obr. 28. zachycuje přechod mezi dvěma typy cest za obtížných světelných podmínek, na které algoritmus reaguje správně.

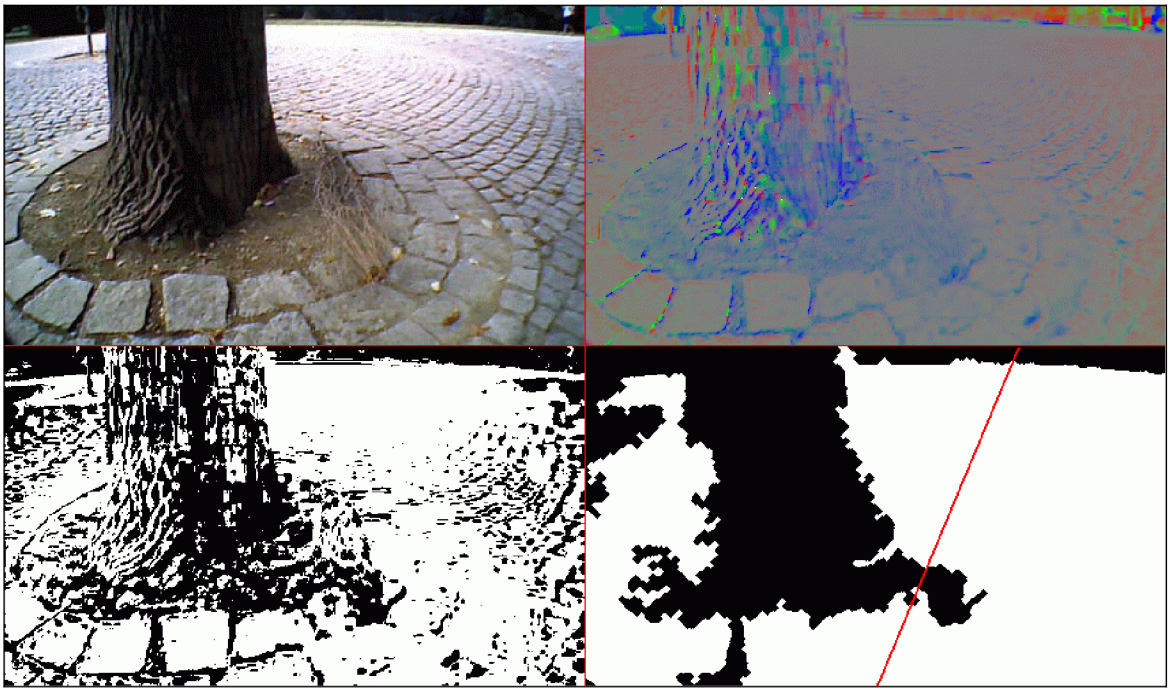
Do výsledné extrakce směru nebyla zahrnuta syntéza dat z GPS.



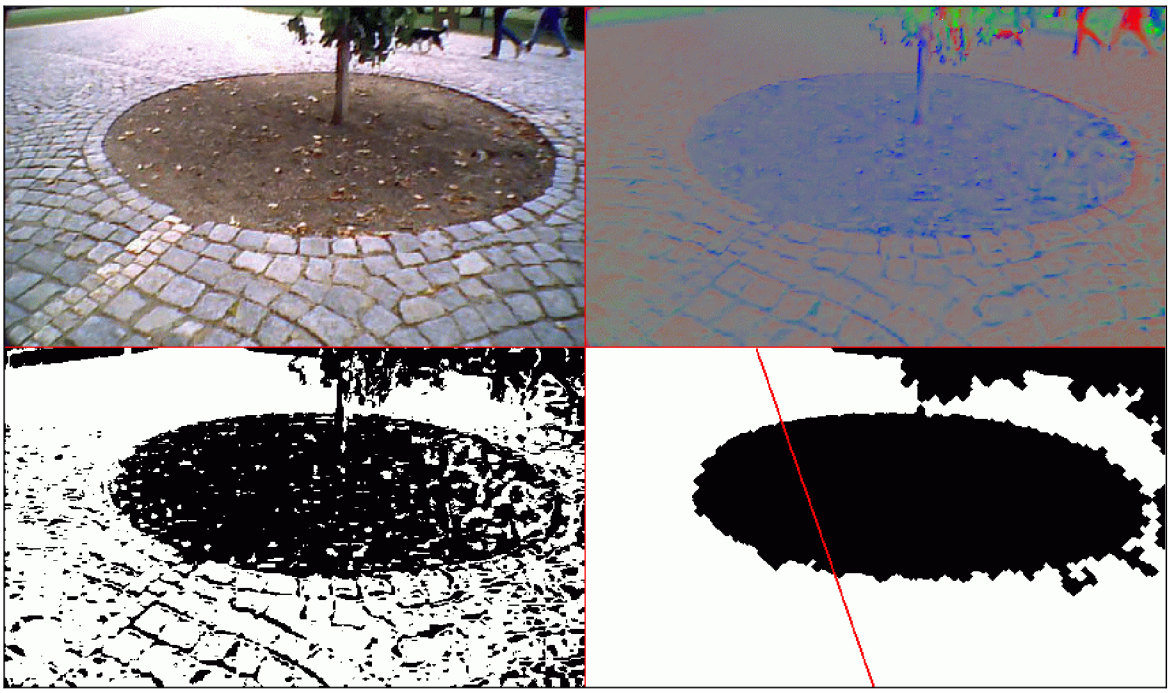
Obr. 24. Průběh algoritmu na asfaltové rovné cestě s výrazným stínem.



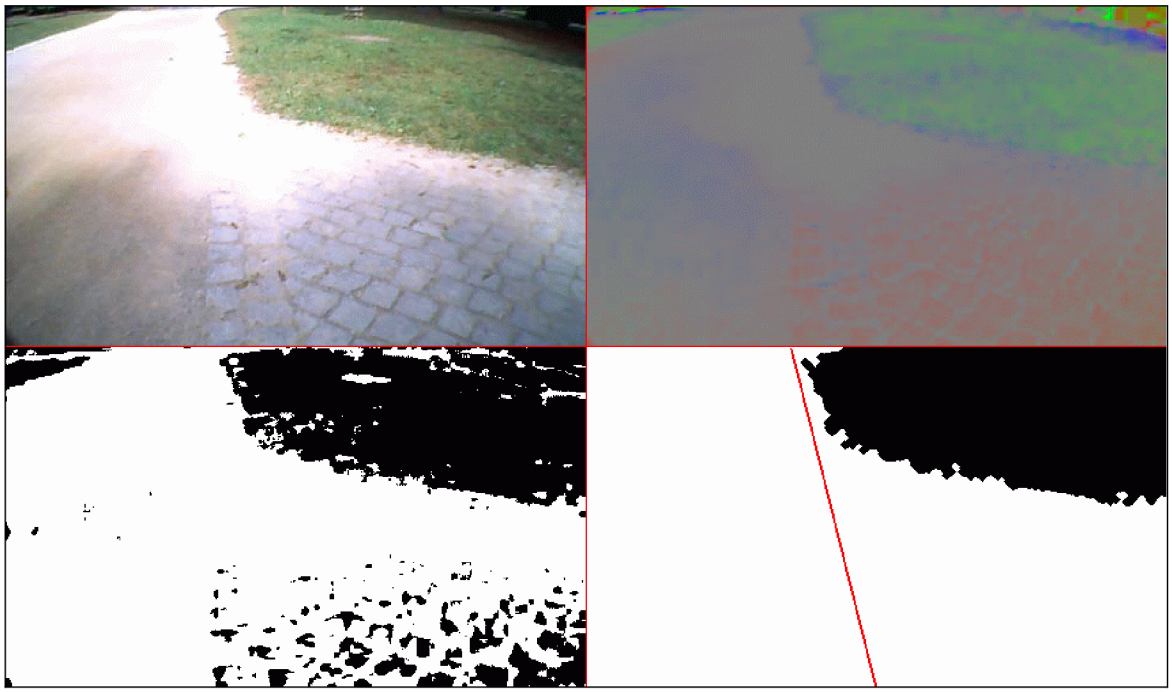
Obr. 25. Obraz prašné cesty. Hranice označující cestu je na pravé straně nezřetelná.



Obr. 26. Obraz s překážkou. Algoritmus reaguje správně a extrahovaný úhel se překážce vyhýbá.



Obr. 27. Obraz s kruhovou překážkou.



Obr. 28. Obraz s přechodem mezi dvěma typy cest.

ZÁVĚR

Ve své práci jsem se zabýval zpracováním obrazu pro autonomního mobilního robota pohybujícího se ve venkovním prostředí. Na základě charakteru prostředí jsem určil hlavní rysy řešení. Podrobně jsem prozkoumal vhodné barevné prostory a metody zpracování a segmentace obrazu. Z prozkoumaných metod jsem vybral dle mého soudu nejvhodnější kombinaci metod pro řešení. Obraz je oříznut a je z něj filtrován šum gaussovou konvolucí. Metodou prahování je obraz segmentován na dvě části: cesta a okolí. Kombinací metod dilatace a eroze a algoritmem vyplňování děr je obraz dále zpracováván tak, aby se dosáhlo lepších výsledků segmentace. Výsledný segmentovaný obraz se použije pro určení referenčního bodu pro segmentaci obrazu v příštím průběhu algoritmu. Zpracovaný obraz je použit pro algoritmus extrakce směru, který naviguje robota tak, aby vždy jel po cestě za cílem vytyčeným GPS souřadnicemi.

Testy na reálných datech potvrdily použitelnost metody pro robota ve venkovním prostředí. Výpočetní čas celého zpracování jednoho obrazu o rozměrech 480x360 pixelů je okolo 200 milisekund bez optimalizací kódu. Vhodnou optimalizací je možné výpočetní čas snížit o čtvrtinu až polovinu. Algoritmus byl vyvíjen v jazyce Delphi a poté přepsán do jazyka C# z důvodu lepší komunikace mezi vývojáři pro Bendra II, jehož hlavní řídící algoritmus je psán právě v tomto jazyce.

Výhoda metody spočívá v její rychlosti, jednoduchosti a robustnosti. Nevýhoda metody je právě v její obecnosti, která sice zajišťuje robustnost, ale má za následek časté špatné klasifikace bodů v obraze, které se pak negativně promítnou do extrakce nového bodu.

Možným rozšířením algoritmu může být inteligentní správa adaptivní databáze. Robot po celou dobu své jízdy shromažďuje informace o cestě a o okolí. Tyto informace mohou být dále využívány, zvlášť pokud se robot pohybuje delší dobu v typově stejné oblasti. Dalším rozšířením může být implementace předučených informací a typových prostředí z naměřených dat.

Předmětem dalšího výzkumu je zejména použití mnohonásobně rychlejších procesorů na grafických kartách pro výpočty časově náročných algoritmů zpracování obrazu pomocí jazyka OpenCL. S touto technologií se výpočetní časy těch nejnáročnějších algoritmů zpracování obrazu budou blížit použití v reálném čase.

SEZNAM POUŽITÉ LITERATURY

[1] SOTELO, M., RODRIGUEZ, F., MAGDALENA, L., BERGASA, L., BOQUETE, L., *A color vision-based lane tracking system for autonomous driving in unmarked rous,* Autonomous Robots, vo.16, no. 1, 2004

[2] EKINCI, M., GIBSS, F. W. J., THOMAS, B. T., *Konwledge-based navigation for autonomous road vehicles*, Turkish Journal of Electrical Engineering & Computer Sciences, 2000

[3] ALVAREZ, J. M., LOPEZ, A. M., BALDRICH, R., *Shadow resitstant road segmentation from a mobile monocular systém*, Iberian Conference on Pattern Recognition and Image Analysis, vol. II, 9-16, 2007

[4] CERVANTES, J. G. A, DEVY, M., *Scene Modeling by ICA and Color Segmentation*, MICAI 2004: 574-583

[5] SONG, D., LEE, H. N.,YI, J., LEVANDOWSKI, A., *Vision-based motion planning for an autonomous motorcycle on ill-structured roads*, Auton Robot, vol. 23, pp. 197–212, 2007.

[6] GEVERS, T., SMEULDERS, A. W. M. *Colour based object recognition*, Pattern Recognition, 32, 453–464. 1999

[7] LIN, L. ZHOU, W., *A Robust and Adaptive Road Following Algorithm for Video Image Sequence*, Lecture Notes in Computer Science, 2007, NUMB 4681, 1041-1049

[8] COMANICIU, D., MEER, P., *Mean Shift: A Robust Approach Toward Feature Space Analysis*, IEEE Tranactions on pattern analysis and machine inteligence, vol. 24, no. 5, May 2002

[9] RASMUSSEN, C., *Grouping dominant orientations for illstructured road following*, In Proceedings of IEEE computer society conference on computer vision and pattern recognition, June 2004.

[10] UNSER, M., *Sum and Difference Histograms for Texture Classification,* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 8, 118-125,1986

[11] GRUDIC, G., MULLIGAN, J., *Outdoor path labeling using polynomial mahalanobis distance*, In Robotics: science and systems, Philadelphia, PA, June 2006.

[12] FORSYTH, PONCE, *Computer Vision: A Modern Approach*, Prentice Hall, 2002

[13] *EmguCV: OpenCV in .NET*, [online], 10.5.2008, 6.4.2010 [cit. 24. 5. 2010], Dostupné z: <http://www.emgu.com>

SEZNAM OBRÁZKŮ

OBSAH PŘÍLOHY

Přílohou diplomové práce je trvanlivý CD nosič se spustitelnými programy a veškerým programovým kódem použitým k vytvoření diplomové práce. Program ke svému spuštění potřebuje Windows XP, Windows Vista nebo Windows 7 (32bit) s podporou pro .NET. CD dále obsahuje obrazová data použitá pro testování algoritmu, která robot shromáždil během soutěže Robotour 2009.

Adresářová struktura na nosiči je následující:

/text – text práce
/exe – spustitelný program
/code – zdrojové kódy
/data – bitmapy z Robotour 2009