Univerzita Palackého v Olomouci
Přírodovědecká fakulta
Katedra geoinformatiky

Filip JUNG

GEOINFORMATICKÉ ŘEŠENÍ PRO SLEDOVÁNÍ PROVOZU NA KŘIŽOVATKÁCH A ZAPLNĚNÍ PARKOVIŠŤ POMOCÍ WEBOVÝCH KAMER

diplomová práce

Vedoucí práce: Prof. RNDr. Vít Voženílek, CSc.

Prohlašuji, že jsem zadanou diplomovou práci řešil sám a že jsem uvedl veškerou literaturu. Všechna poskytnutá vstupní digitální data nebudu bez souhlasu školy poskytovat.

Olomouc, 3.května 2010 ……..……..

OBSAH

1 Úvod

Techniky a prostředky počítačového vidění nabízí široké možnosti při automatizovaném sběru a vyhodnocování kvalitativních i kvantitativních prostorových dat. Jejich použití v geoinformatice je tímto více než vhodné. Jedná se totiž obvykle o neinvazivní systémy schopné efektivně zpracovávat velké množství dat s možností automatické extrakce požadované informace.

Takováto data jsou samozřejmě geografickými informačními systémy již využívána, jako například data dálkového průzkumu Země. Ovšem schopnosti počítačového vidění a rozpoznávání obrazu nejsou v geoinformatice zdaleka použity v rozsahu jež odpovídají jejich možnostem, zvláště oblast kontinuálního monitoringu a sběru dat nabízí široké pole použití těchto systémů.

Na sběr kontinuálních a dynamických dat v geoinformatice úzce navazuje jejich zpracování a vizualizace. Efektivní uložení do databází, ať už prostorových, nebo neprostorových umožňuje jejich následné použití pro analýzy a vizualizace. Jelikož je dynamika podstatnou vlastností kontinuálních dat, je zde na místě použití dynamických vizualizačních metod. Animace jako jedna z těchto metod dovoluje získaná data efektivně vizualizovat a pomoci například pochopení jejich struktury, trendu či pouze jako nástroj ověření jinak dokázaných zákonitostí v datech.

Data o intenzitě silniční dopravy jsou důležitým zdrojem informací pro řízení silničního provozu, a také pro složky integrovaného záchranného systému, jejich kontinuální sledování může pomoci dispečerům nebo automatickým systémům vyhodnocovat aktuální dopravní situaci a reagovat na ni. Dlouhodobé sledování intenzity je prováděno v souvislosti s výzkumem zatížení dopravních komunikací.

2 Cíl práce

Cílem této práce je navržení, sestavení a otestování systému pro sledování provozu na křižovatkách a zaplnění parkovišť, jako aplikace geoinformačních technologií. Aplikace bude využívat webové kamery s bezdrátovým připojením ke sledování provozu (počty aut) na vybraných lokalitách, jimiž jsou parkoviště a křižovatka, v pravidelných intervalech.

Systém bude schopný na základě analýzy obrazového záznamu provádět záznamy do prostorových databází, statistické vyhodnocování sledovaných jevů a tvorbu různých výstupů (animace, mapy, tabulky, grafy, srovnání). Součástí řešení bude i vytvoření metodiky a postupu pro následné využití systému v libovolných lokalitách.

Ze zadaných cílů vyplývají dvě rozdílná řešení pro rozdílné lokality s odlišnou charakteristikou sledovaných dat. U parkoviště se jedná o jednotlivé snímky pořizované v daném intervalu. Provoz na křižovatce je zaznamenáván a zpracováván jako kontinuální videozáznam.

3 Metody a postup zpracování

Volba lokalit

První a velmi obtížnou částí práce byla volba lokalit pro provádění sběru dat. Jelikož byl požadavek na trvalé umístění kamery umožňující sledovat provoz kontinuálně, nebyl tento úkol vůbec jednoduchý. Jako modelové parkoviště bylo vybráno to na Palachově náměstí u Terezské brány v Olomouci. Ovšem po neúspěšném jednání s vlastníky o umístění kamery do budovy Církevního gymnázia Německého řádu, byla vybrána jiná lokalita, umožňující jednodušší instalaci kamery. Byla jím parkoviště ve dvoře budovy Přírodovědecké fakulty na tř. Svobody 26.

Pro druhou část práce, tedy sledování provozu na křižovatce, nebylo možné za daných podmínek najít takové místo, jež by umožňovalo instalaci webové kamery s adekvátním zázemím. Konečná volba padla na křižovatku ulic Foerstrova a Na Vozovce v Olomouci. Tato lokalita splňovala jednu ze základních podmínek a to tu, že křižovatku bylo možno zabrat jedinou kamerou z chodby 8.patra sousedícího panelového domu z chodby 8.patra sousedícího panelového domu.

Sestavení systému

Pro část práce zpracovávající data z kamery směřující na parkoviště byla zvoleno následující uspořádání systému: Bezdrátová webová kamera snímající v daném časovém intervalu parkoviště jež komunikuje s bezdrátovým směrovačem a odesílá data pomocí FTP na server. Na tomto serveru jsou pak data zpracovávána, ukládána a je k nim umožněn dálkový přístup.

Pro zpracování dat z křižovatky takové možnosti nebyly, tudíž se jedná o zpracování na desktopovém počítači nikoli souběžně se sběrem dat, ale až jako postprocessing.

Volba software a metod

Na základě studia problematiky a dostupných řešení bylo s pomocí testovacích dat rozhodováno a tom, jaké softwarové vybavení a algoritmy budou použity. Pro vyhodnocení obrazu z parkoviště byl vybrán open source statistický systém R s potřebnými knihovnami v této aplikaci běžící na operačním systému Linux. Pro analýzu video záznamu potom výkonné prostředí pro matematické výpočty Matlab.

Sběr dat, spuštění systému

Další fází bylo spuštění systému pro sledování provozu na parkovišti, jež zahrnovalo instalaci kamery a síťových prvků, konfigurace serveru a tvorbu programových kódů pro zpracování dat. Pro vyhodnocení provozu na křižovatce bylo potřeba získat a zpracovat potřebné množství videozáznamu.

Vyhodnocení a vizualizace dat

Po spuštění systému pro parkoviště a ověření jeho funkčnosti byla práce zaměřena na vyhodnocení a vizualizaci shromažďovaných dat. Mezitím byly voleny vhodné metody pro interpretaci dat získaných z videozáznamu provozu na křižovatce. V dalším kroku byla získaná data srovnána s daty již existujícími, například z daty ze sčítání dopravy.

4 Současné možnosti automatického sledování dopravy

Údaje o sledování dopravy a především její intenzity se využívají při koncepcích rozvoje komunikační sítě, návrhu komunikací, při úvahách o rozdělení finančních prostředků na opravy a rekonstrukce, kapacitních výpočtech, výpočtech negativních vlivů dopravy na životní prostředí apod. [33].

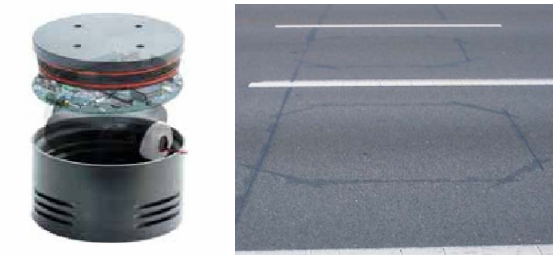
Následující kapitoly se zabývají stručným přehledem používaných senzorů pro automatické zaznamenávání údajů nejen o intenzitě dopravy, ale i dalších důležitých charakteristikách (např. rychlost, typ vozidla), jež mohou spolu se základními informacemi sloužit mimo jiné také k okamžitému vyhodnocování dopravy a následnému přizpůsobení systémů řízení provozu. V případě pokročilého zpracování signálu (kamerové systémy) mohou tyto systémy upozornit například i na nestandardní a nebezpečné situace a tyto informace dálkovým způsobem předat.

4.1 Senzory umístěné ve vozovce

Jedná se o senzory umístěné v krytu nebo podloží vozovky, ale také nějakým způsobem připevněné k jejich povrchu. Jako příklad těchto senzorů lze úvést často používané detekční indukční smyčky, které jsou umístěné v zářezech v krytu vozovky, váhové detektory zakomponované do krytu vozovky, magnetometry umístěné pod vozovkou nebo konstrukcí mostu, a nebo pneumatické trubky a piezoelektrické dráty umístěné na povrchu [22].

Výhody těchto senzorů spočívají v nízké pořizovací ceně, zavedenost většiny technologií a necitlivosti na vlivy počasí. Nevýhody těchto senzorů jsou hlavně v nutnosti přerušovat provoz na komunikaci a narušovat povrch vozovky při jejich instalaci, případně jsou citlivé na kvalitu povrchu silnice,

dále je také obvykle potřeba více senzorů pro pokrytí sledované oblasti. Omezení jsou také v oblasti pokročilé klasifikace vozidel a situací [9].



Obr. 1 Dvouosý magnetometr [22] a obr. 2 povrch vozovky s instalovanou detekční smyčkou [37].

4.2 Senzory umístěné nad vozovkou

Jedná se senzory připevněné nad povrchem silnice a to buď přímo nad ní, nebo vedle ní. Příklady tohoto druhu zařízení jsou systémy zpracovávající video (založené na strojovém vidění), mikrovlnné radary, ultrazvuk, pasivní infračervené, laserové radary a pasivní akustické senzory. Často také dochází ke kombinaci více technologií za účelem zpřesnění prováděné detekce, např. kombinace Dopplerova mikrovlnného radaru, ultrazvuku a pasivního infračerveného detektoru (obr. 3).

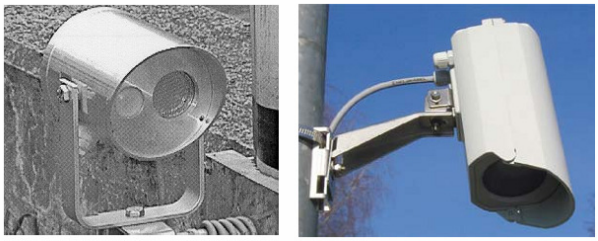
Tato skupina detektorů podává, stejně jako ty umístěné ve vozovce, informace o počtu projetých vozidel, jejich přítomnosti a průjezdu ale také mohou určovat rychlost vozidel, klasifikovat je a mohou pokrýt více pruhů či detekčních zón [22]. Většina těchto systémů je kompaktní a neinvazivní vůči vozovce. Srovnání vybraných detektorů z obou skupin ukazuje tab. 1.

Technologie	Výstupní data				Sledování více pruhů, více det. zón	Šířka kom. pásma	Pořizovací náklady[a]
	Sčítání	Přítomnost	Rychlost	Klasifikace			
Indukční smyčka	•	•	•[b]	•[c]	-	Nízká až střední	Nízké
Magnetometr (dvouosý)	•	•	•[b]	-	-	Nízká	Střední
Indukční cívka	•	•[c]	•[b]	-	-	Nízká	Nízké až střední
Mikrovlnný radar	•	•[c]	•	•[c]	•[c]	Střední	Nízké až střední

Aktivní infra	•	•[d]	•	•	•	Nízká až střední	Střední až vysoké
Pasivní infra	•	•[d]	•	-	-	Nízká až střední	Nízké až střední
Ultrazvuk	•	•	-	-	-	Nízká	Nízké až střední
Akustický detektor	•	•	•		•[c]	Nízká až střední	Střední
Videodetekční systém	•	•	•	•	•	Nízká až vysoká[e]	Střední až vysoké

a – do reálných nákladů je třeba započítat i prostředky nutné na instalaci, údržbu a opravy
b – rychlost může být měřena pouze za použití dvou senzorů
c – s použitím speciálních programových prostředků, případně zvláštního rozmístění senzorů
d – s použitím více detekčních zón
e – záleží jestli jsou posílána surová data, nebo už data nějak zpracovaná

Tab. 1 Typická výstupní data, šířka kom. pásma a pořizovací náklady běžně používaných dopravních senzorů, upraveno podle [22]



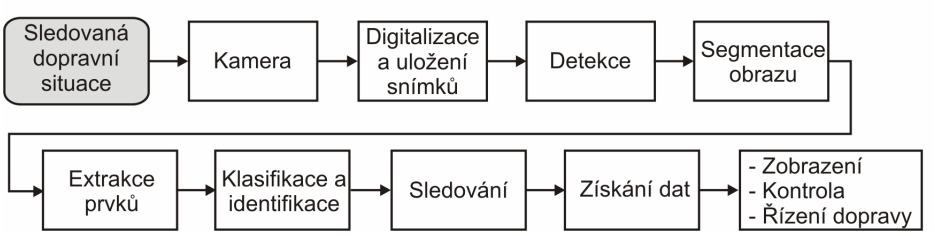
Obr. 3 Dopplerův radar kombinovaný vlevo s ultrazvukovým detektorem a infračerveným detektorem (vpravo).

4.2.1 Systémy zpracovávající videozáznam

První videokamery byly použity pro sledování dopravy kvůli jejich schopnosti přenášet obraz pro interpretaci k lidskému operátorovi. Dnešní systémy používají automatické zpracování videozáznamu pro získání informací pro sledování a ovládání dopravy. Tyto systémy se obvykle sestávají z jedné nebo více kamer, procesorové jednotky pro digitalizaci a zpracování obrazu a software pro interpretaci obrazu a jejich převedení na proud dat o dopravě [35].

Hardware je důležitou součástí systémů ale daleko větší vliv na kvalitu detekce a dalších úloh má programové vybavení zpracovávající obrazový záznam. Inteligentní dopravní systémy, jejích součástí je i práce s videozáznamem, se staly aktivním oblastí výzkumu v oblasti umělé inteligence. Klasické systémy (např. indukční smyčky) totiž poskytují omezené dopravní informace, a jsou složité a nákladné na instalaci. Je prokázáno, že

systémy založené na počítačovém vidění jsou při sledování dopravy flexibilní, pokud jsou vytvořeny dostatečně robustně a spolehlivě [2], [10], [18].



Obr. 4 Koncept zpracování obrazu pro detekci, klasifikaci a sledování vozidel, převzato z [9]

Následující část textu popisuje algoritmy používané pro detekci pohybu v obrazových datech. Další metody jako klasifikace a sledování, nejsou více popisovány. Většina názvů technik je ponechána v anglickém originále, v jakém se běžně používá.

4.2.2 Metody používané pro detekci pohybu ve videozáznamu

Možnost analyzovat pohyb má velký význam ve zpracování obrazu například v hlídacích kamerových systémech, rozpoznávání a sledování objektů nebo kompresi videosekvencí [8].

Pro analýzu pohybu se používá řada odlišných metod, jež mají své různé silné a slabé stránky. Rozvoj metod souvisel v poslední době hlavně s novými možnostmi technických prostředků, které se na zpracování obrazů podílejí. Při zpracovávání konkrétní úlohy hrají obvykle velkou roli různé kladené předpoklady. Např. zda byl pohyb zachycen statickou nebo mobilní kamerou, zda je posloupnost obrazů snímána v dostatečně krátkých časových intervalech, aby bylo možné ji považovat za reprezentaci spojitého pohybu [15]. Současně vyvíjené metody se snaží napodobit vnímání pohybu živými organismy.

Background subtraction

Neboli „odečítání pozadí“ je použitelné pro statickou kameru. Tento algoritmus určuje pohybující se objekt popředí jako rozdíl mezí aktuálním

snímkem a statickým snímkem pozadí scény. Problémem je automatické získaní snímku pozadí. To se totiž musí přizpůsobovat změnám intenzity osvětlení (ať už náhlým, nebo pozvolným), rušivým pohybům (třes kamery, kývající se strom), a změnám scény (zaparkované vozidlo). Další důležité faktory jsou například také stíny, „černá obrazovka“ a špatné počasí [8]

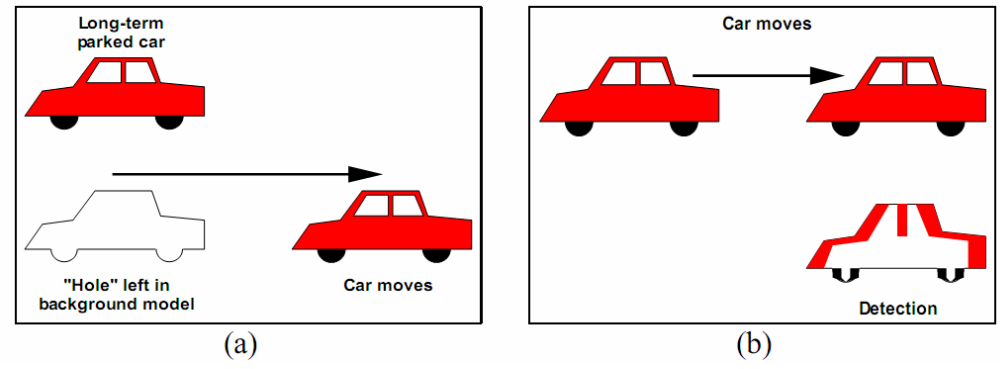
Obr. 5 Ukázka fungování odečítání pozadí [17]

Adjacent frame differencing (AFD)

Neboli „odečítání sousedících snímků“, představuje variantu předchozí metody, kde pozadí tvoří snímek předcházející snímku aktuálnímu. Obecný zápis je, že pixel patří k popředí, jestliže:

$$| \textit{aktuální snímek} - \textit{snímek pozadí} | > \textit{prahová hodnota}$$

Prahová hodnota stanovuje citlivost detekce změny. Ovšem její stanovení je prováděno empiricky, je totiž obtížné tuto hodnotu nějakým exaktním způsobem určit. Je tedy časté, že na rozdílovém obrazu se objevuje šum a také „díry“ v pohybujících se objektech [1].



Obr. 6 Problémy při použití background subtraction (vytvoření prázdného místa) a frame differencing (neúplné objekty) [1]

Pozadí jako medián nebo průměr posledních *n* snímků

V této metodě je obraz pozadí vypočítán statisticky z posledních *n* snímků, jako například:

$$\left| \frac{\sum_{i=t-n}^{t-1} obraz_i}{n} - obraz_i \right| \succ prahová\_hodnota$$

Tato metoda je v porovnání s jinými méně výpočtově náročná a má výhodu, že zastavené objekty se jen pomalu integrují do pozadí. Nevýhodou je opět použití konstanty v podobě prahové hodnoty a vyšší náročnosti na paměť, neboť je potřeba mít uloženo posledních *n* obrazů. Alternativou může být také vážený průměr (konkrétně exponenciálně vážený) [8].

Color mean and variance (CMV)

Pozadí tohoto algoritmu je reprezentováno jako průměr μ a variance σ^2 hodnot pixelů. Předpokládá se, že hodnoty lze modelovat pomocí normálního rozdělení. Nejjednodušší implementace používá standardní RGB barevný prostor. Pohyb je detekován skrz prahování na absolutních rozdílech mezi hodnotami pixelů a průměrem jednotlivých barevných kanálů. Když je rozdíl větší než směrodatná odchylka, je pixel označen jako popředí, v opačném případě jako pozadí.

$$\Delta_c = \left| x_c - \mu_c \right| \quad pro \ \ c \in R, G, B$$

$$Pokud \ \Delta_R \succ K_{\sigma R} \vee \Delta_G \succ K_{\sigma G} \vee \Delta_B \succ K_{\sigma B} \qquad pixel := popředí$$

$$Jinak \qquad\qquad pixel := pozadí$$

Tento přístup nepoužívá žádnou globální prahovou hodnotu a poskytuje proto lepší detekci než předchozí metody. Nedostatečná robustnost vůči změnám jasu a s tím spojené nesprávné detekce však omezují tento algoritmus. Existují ovšem také jeho další varianty, jež se snaží tato omezení limitovat [8].

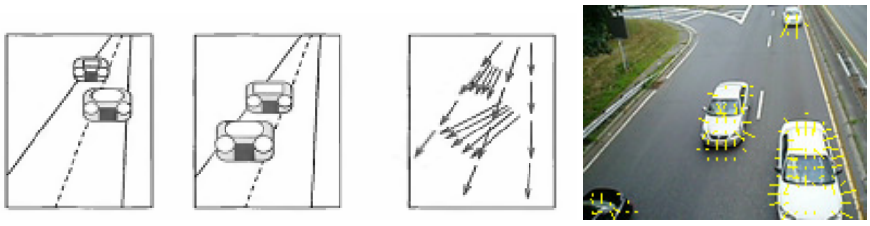
Gaussian mixture model (GMM)

Tento přístup k detekci pohybu je založen na myšlence, že pixel pozadí může v různý čas reprezentovat různé objekty (např. větrem se vlnící strom). Toho se dosáhne pomocí rozšíření CMV modelu. Místo Gaussova rozdělení pro pixel vezmeme mixturu z K, obvykle 3 až 5 vážených rozdělení aby charakterizovaly model pozadí. Pro snížení výpočetní a paměťové náročnosti se obvykle vychází ze statisticky nezávislých barevných kanálů, což znamená, že kovarianční matice Gaussova rozdělení existjí pouze diagonálně [8].

Výhoda tohoto algoritmu je, že nemá žádnou globální prahovou hodnotu a při detekci vzniká méně děr a objekty nejsou příliš fragmentované. Nevýhodou je však zdlouhavost výpočtu a velký vliv šumu na kvalitu detekce.

Optical flow

Změny způsobené pohybem lze zjišťovat také pomocí optického toku (optical flow), který zachycuje všechny změny obrazu za čas *dt*. Každému bodu obrazu optického toku odpovídá dvojrozměrný vektor rychlosti, vypovídající o směru a velikosti rychlosti pohybu v daném místě obrazu. Výpočet optického toku je nutným předpokladem zpracování vyšší úrovně, které dovoluje pracovat se statistickým i pohyblivým umístěním pozorovatele, umožňuje určit parametry pohybu, relativní rychlost předmětů v obraze apod. [15].



Obr. 7 Optický tok, schématické znázornění a ukázka detekce vozidel [14]

Výpočet optického toku je založen na dvou předpokladech: 1. pozorovaná světlost jakéhokoli bodu objektu je konstantní v čase, 2. blízké body v ploše

obrazu se pohybují podobným způsobem [14]. Z těchto omezení plyne, že samotný optický tok je třeba často doplnit o další pomocné algoritmy.

4.3 Parkovací systémy

V návaznosti na rostoucí počet automobilů rostou také problémy spojené s dostupnými parkovacími místy, jelikož jejich nabídka není schopná na tento fakt reagovat dostatečně rychle. Také z tohoto důvodu byly vyvinuty „chytré“ parkovací systémy, jež umožňují snadné a rychlé nalezení volného místa pro parkování [4].

Chytré parkovací systémy mohou být dle [13] rozděleny do pěti hlavních kategorií:

Parkovací naváděcí informační systémy

Mohou fungovat buď v rámci celého města, nebo jen v daném parkovacím zařízení. Poskytují informace, jež podporují rozhodování při výběru parkoviště z volným parkovacím místem. Systém se skládá se ze čtyř hlavních komponent: mechanismus diseminace informací, mechanismus sběru informací, kontrolní středisko a telekomunikační síť.

Do tohoto systému patří například statické a dynamické informační tabule, ale také lokace nejbližších volných parkovišť na základě GPS polohy uživatele. Pro distribuci dat je také možno použít mobilní telefony, PDA a Internet. Detekční senzory jsou obvykle instalovány na vjezdech a výjezdech, nebo probíhá detekce pro konkrétní parkovací místa. Implementace těchto technologií u nás může být například aplikace [16].

Tranzitně založené IS

Tato skupina systémů je se od předcházející odlišuje tím, že se soustřeďuje na „parkuj a jeď“ (park-and-ride) zařízení. Jedná se o napojení aktuálních informací o stavu parkovišť a veřejné dopravy, jako jsou jízdní řády a dopravní

podmínky. Cílem je zvýšit využití veřejné hromadné dopravy. Touto problematikou se zabývá například [4].

Chytré platební systémy

Tyto systémy jsou implementovány ve snaze překonat omezení konvenčních platebních metod pomocí například platebních karet, mobilních zařízení, automatickou identifikací vozidel. Problematika je dále rozebrána v [7].

E-parking

Hlavním rozdílem od ostatních systémů je zde možnost rezervace parkovacího místa na vybraném parkovišti ještě před příjezdem například pomocí SMS nebo Internetu. Takovéto funkční systémy jsou například [16] nebo [24].

Automatizované parkování

Tyto systémy obsahují počítačem řízené mechanismy, jež po zastavení auta samy dopraví vozidlo na vyhrazené místo. Tento systém nabízí maximální využití zabraného prostoru. Těmito systémy se zabývá například firma Robotic Parking [32]. Jejich nevýhodou jsou samozřejmě vysoké pořizovací náklady, a také vysoké nároky kladené na spolehlivost systému.

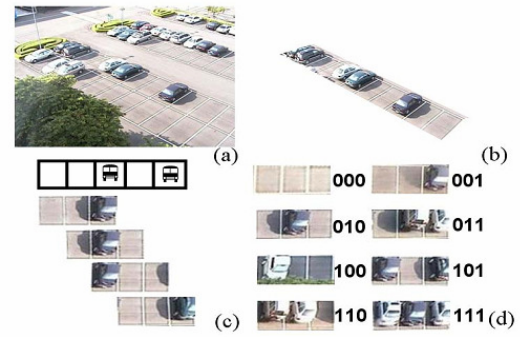
Obr. 8 Automatické parkování [32]

4.3.1 Rozpoznávání obrazu a parkovací systémy

V řadě „chytrých“ parkovacích systémů je v oblasti získávání dat použito kamerových systémů s automatickým vyhodnocením obrazu. Takto je možno efektivně zpracovávat aktuální data a ty dále distribuovat.

Tyto systémy mohou pracovat jak s předem definovanými parkovacími místy i s volným rozmístěním zaparkovaných vozidel. Při analýze obrazů jsou použity techniky segmentace a klasifikace obrazu. Vychází se z předpokladu, že automobily mají jiné obrazové vlastnosti než jejich okolí, proto se provádí například prahování, texturní analýza, detekce hran, analýza histogramu a další.

Například v [11] je pro detekci vozidel použita quad-tree dekmpozice. Pokročilejší a komplexnější klasifikace na základě analýzy histogramu a následného použití SVM (Support Vector Machine), pracující na základě distribuce pravděpodobnosti, jež je použito pro řízenou klasifikaci obrazu, představuje [21].



Obr. 9 Příprava obrazu pro klasifikaci [21]

4.4 Systémy používané v Olomouci

Město Olomouc se řadí mezi největší města v ČR a jako takové má samozřejmě problémy ze silniční dopravou, a to především v jeho centru. Magistrát města Olomouce (MmOl) spravuje ve městě křižovatkové systémy a také další prvky, jež mají za úkol řídit dopravu. Několik křižovatek (např.

Pražská /Tř.Míru a Polská /tř.17.listopadu) mají v povrchu vozovky zabudované indukční detekční smyčky, jež jsou napojeny na řadiče křižovatky a jsou schopny podle intenzity provozu upravovat signály na křižovatce.

Kamerové detektory jsou používány jak pro krátkodobé účely, jako např. počítání nákladních vozidel na ulici Okružní pro MmOl firmou Eltodo v loňském roce, ale také pro dlouhodobé, ke kterým patří sledování hustoty provozu na tř. Svobody pomocí systému kamer a zobrazením aktuálního stavu na informačních tabulích. Další trvale umístěnou kamerou s využitím pro řízení dopravy je systém Traficon, nainstalovaný firmou AŽD Praha, který vyhodnocuje dopravní kongesci v prostoru levého odbočení ze směru od Chválkovic na křižovatce Chválkovická - U Podjezdu a následně předává informaci řadiči světelné signalizace, který na dopravní kongesci reaguje úpravou algoritmu řízení.

V oblasti „chytrého“ parkování jsou v Olomouci nainstalované na několika místech (většinou příjezdové trasy do města) informační tabule se stavem několika parkovišť, jejichž systémy jsou schopny tuto informaci dálkově poskytnout. Jedná se o podzemní parkoviště u hlavního nádraží, kryté patrové parkoviště na Koželužské ulici a parkoviště u MmOl na ulici Hynaisova.

Obr 10 Kamrová jednotka systému Traficon [36]

Obr. 11 Instalace informačních tabulí o stavu parkovišť v Olomouci [28]

5 Systém pro sledování parkoviště

5.1 Výběr vhodného parkoviště

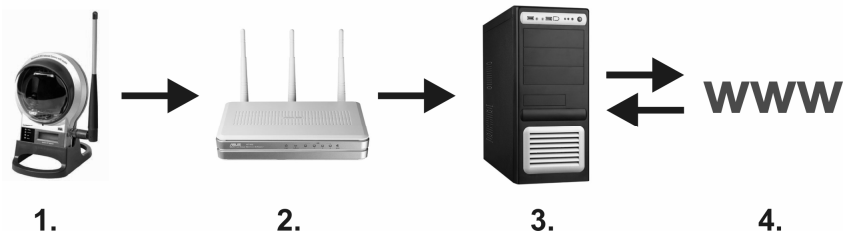
Volba lokality pro provádění sběru dat nebyla jednoduchá, jelikož byl požadavek na trvalé umístění kamery umožňující sledovat provoz kontinuálně. Jako modelové parkoviště bylo vybráno to na Palachově náměstí u Terezské brány v Olomouci. Webová kamera měla být umístěna v budově Církevního gymnázia Německého řádu, ovšem po dlouhém jednání se všemi stranami, jejichž souhlasu bylo potřeba pro umístění přístroje, nedošlo ke kladnému vyjádření ze strany Velmistrovské kanceláře, tudíž se musela hledat jiná varianta. Jako nejpřijatelnější bylo vybráno sledování parkoviště ve dvoře budovy Přírodovědecké fakulty UPOL na tř. Svobody 26. Zde nebyly takové problémy se získáním povolení pro umístění kamery, jelikož je kamera umístěna přímo na Katedře geoinformatiky. Tato změna lokality měla ovšem za následek problém s užitností celého systému, neboť kamera, tak jak byla umístěna na katedře nemůže zabrat celé parkoviště. U budovy gymnázia tento problém nebyl.

Změnou lokality došlo i ke změně cílové skupiny uživatelů, jelikož parkoviště ve dvoře budovy PřF není veřejné, ale pouze pro zaměstnance UP. Také odpadl původní záměr možného napojení výsledků vyšetřování zaplnění parkoviště na informační tabule od společnosti Eltodo.

Obr. 12 Pohled na sledované parkoviště z pozice kamery

5.2 Sestavení systému

Po zvážení možností a zjištění dostupného použitelného vybavení na katedře bylo vybráno následující složení systému hardwaru. Základem je bezdrátová webová kamera od firmy Linksys, jež je do sítě připojená pomocí bezdrátového routeru, z něj data proudí na server, kde jsou zpracována. Výsledek je potom publikován do sítě Internet (obr.13)



Obr. 13 Schéma základních prvků systému sledování provozu na parkovišti

5.3 Použitý hardware

Kamera

Nejdůležitější součástí je zařízení pro zachycení obrazu, zde katedrální webová kamera Linksys WVC 200. Jedná se o jednoduchou digitální webovou kameru s CMOS snímačem umožňující přenos dat buď pomocí drátového

napojení do datové sítě Fast Ethernet, nebo také použitím bezdrátového přenosu za použití standardu 802.11g. Snímání videa probíhá v rozlišení 640 na 480 pixelů. Důležitou schopností, jež nabízí nastavení kamery ve webovém rozhraní, je možnost odesílat v pravidelném intervalu sejmutý obraz pomocí protokolu FTP na server. Právě tato možnost je využita v aplikaci diplomové práce. Původně lze ve webovém rozhraní nastavit minimální interval pouze 30 minut. Pro změnu této hodnoty bylo použito doplňku Firebug do webového prohlížeče Mozzila Firefox, jež umožňuje úpravu html kódu stránky, v tomto případě prvku formuláře. Nyní bylo možné zadat požadovaný interval v minutách a uložit toto nastavení do paměti kamery . Velikost intervalu byla zvolena 2 minuty, aby data byla co nejaktuálnější.

Obr. 14 Použitá bezdrátová webová kamera Linksys WVC 200

Router

Pro sestavení systému byl katedrou poskytnut bezdrátový směrovač Asus WL-500W pracující v módu přístupového bodu (AP). Ten vytváří svou vlastní bezdrátovou síť do níž je kamera zapojena. Samotný router je potom zapojen do sítě LAN pomocí ethernetového kabelu. Přístroj umožňuje pokročilé zabezpečení bezdrátové sítě WPA2 s šifrováním TKIP nebo AES, ovšem

použitá bezdrátová kamera toto nepodporuje. Proto bylo použito jen slabší šifrování WEP se 128 bitovým klíčem.

Server

Pro potřeby diplomové práce byl poskytnut v dané době nezprovozněný PC od firmy Autocont, používaný jako server osazený procesorem Intel Pentium 4 3,2 GHz, s 512 MB RAM. Na počítači už byl operační systém a většina potřebných aplikací nainstalovaná.

5.4 Použitý software

Mandriva Linux 2006.0

Mandriva Linux (předtím též Mandrake Linux nebo Mandrakelinux) je francouzská distribuce Linuxu (GNU/Linux). První verze byla firmou Mandrakesoft postavena na Red Hat Linuxu 5.1 a grafickém prostředí KDE 1.0 v červenci roku 1998. Vyznačuje se množstvím nástrojů na snadnou konfiguraci systému.

Tento operační program byl už nainstalovaný na dříve provozovaném počítači poskytnutém pro diplomovou práci. Na operačním systému běží několik aplikací jež jsou popsány dále.

Apache HTTP server

Apache HTTP Server je softwarový webový server s otevřeným kódem pro GNU/Linux, BSD, Solaris, Mac OS X, Microsoft Windows a další platformy. V současné době dodává prohlížečům na celém světě většinu internetových stránek. WIKI Na webový server je napojeno několik dalších technologií, které rozšiřují jeho funkčnost [22].

MySQL

Pro uložení dat byla použita databáze MySQL, jež je rychlý a stabilní databázový systém. Nabízí bohatý a velmi užitečný soubor funkcí. Je podporován skripty PHP, CGI nebo ASP dle zvolené platformy virtuálního serveru. Databázi MySQL mohou využívat virtuální servery běžící na platformě Linux/Unix i Windows 2000 [3]. Pro svou snadnou implementovatelnost, výkon a především díky tomu, že se jedná o volně šiřitelný software, má vysoký podíl na v současné době používaných databázích [29]. V aplikaci byla použita verze 4.1.12 běžící na poskytnutém serveru.

PHP

Hypertextový preprocesor (původně Personal Home Page) je skriptovací programovací jazyk, určený především pro programování dynamických internetových stránek. Nejčastěji se začleňuje přímo do struktury jazyka HTML, XHTML či WML. PHP skripty jsou prováděny na straně serveru, k uživateli je přenášen až výsledek jejich činnosti. PHP je nezávislý na platformě, skripty fungují bez úprav na mnoha různých operačních systémech. Obsahuje rozsáhlé knihovny funkcí pro zpracování textu, grafiky, práci se soubory, přístup k většině databázových serverů [30].

JPGraph

JpGraph je přídavná knihovna do PHP, která není standardní součastí distribuce PHP. Je napsaná v prostředí UNIXu, ale odzkoušená a plně funkční též pod Windows. Je plně objektově orientovaná a naprogramována v PHP [26].

CRON

CRON je Linux/Unix systémový nástroj, který spouští různé programy v předem definovanou dobu a intervalu (obdoba naplánovaných úloh ve Windows).

R

Konečně pro vlastní zpracování obrazového záznamu je použito volně dostupného softwarového prostředí pro statistické výpočty a grafiku. Běží na rozličných platformách UNIX, Windows a MacOS. V práci byla použita verze 2.10.1. Samotný software si s obrazovými daty neporadí, ovšem jeho síla je ve velkém množství balíčků vytvořených převážně uživatelskou komunitou, mezi kterými je jich i několik právě pro úkoly zpracování obrazu. V případě popisované aplikace byl použit balíček *biOps: Image processing and analysis*, jež obsahuje základní metody a funkce pro zpracování a analýzu obrazových dat. Pro jeho úplné používání je potřeba mít nainstalovaný také balíček *fftw3* a knihovny *libjpeg* případně *libtiff*.

5.5 Instalace kamery

Pro spuštění systému automatické detekce vozidel na parkovišti bylo potřeba učinit několik dílčích kroků. Prvním byla instalace samotné kamery. Místo pro její umístění bylo vybráno v okně na chodbě Katedry geoinformatiky v druhém podlaží. Nabízela se možnost umístit kameru ještě výše, ovšem do prostor jiné katedry, což nebylo v dané situaci žádoucí.

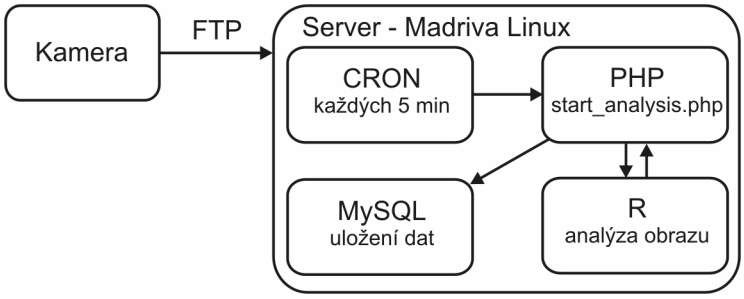
Pro uložení kamery byla vytvořena konzole, jež byla připevněna na strop ve výklenku okna. Napájecí kabel byl potom veden po stěně k nejbližší zásuvce. Kamera je umístěná v dostatečné výšce aby bylo zabráněno jejímu snadnému odcizení.

Obr. 15 Instalace kamery do okna na katedře

Obr. 16 Umístění kamery – pohled ze dvora PřF

5.6 Programová struktura systému

Po nainstalování hardwaru a softwaru se mohlo přistoupit ke zprovoznění celého systému. Postup od sejmutí obrazu kamerou až po uložení výsledku do databáze je následující: Kamera každé dvě minuty odešle snímek pomocí protokolu FTP na server, kde dojde k jeho uložení.



Obr. 17 Schéma fungování aplikace

Na serveru běží CRON, v jehož souboru crontab je zapsán příkaz:

`*/5 * * * * userName /usr/bin/php -f /var/www/.... /start_analysis.php ...`, dle nějž se každých 5 minut spustí php skript, provádějící další činnost.

Tento skript se odkazuje na několik dílčích funkcí jež vykonávají jednotlivé úkony. Pokud je splněna podmínka času, tj. že aktuální čas je mezi 6:00 a 20:00 pokračuje skript dále, jinak není vykonána žádná činnost. Nejprve se pomocí funkce `get_file_info_ftp()` zjistí nejaktuálnější snímek, z jeho názvu se vyextrahuje čas pořízení a tento snímek se uloží do pracovní složky a do složky určené pro zobrazení uživatelům. Na serveru je uložen vždy nejaktuálnější snímek, ostatní jsou průběžně mazány pomocí funkce `clean()`, která probíhá každou půlhodinu.

Další funkce, `analyse($time_filename)`, jejímž vstupem je čas získaný předchozí funkcí (ve formátu Unix time_stamp) otevře textový soubor obsahující programový kód určený pro R a zapíše do něj čas snímku. Potom pomocí php příkazu `exec` zavolá program R a jako vstup mu přidělí upravený textový soubor `code.txt`. Poté proběhne kód v R (popsaný v další kapitole), zapíše svůj výsledek do dalších textových souborů a vrátí zpět skriptu `start_analysis.php` čas vyextrahovaný z textového souboru výsledku.

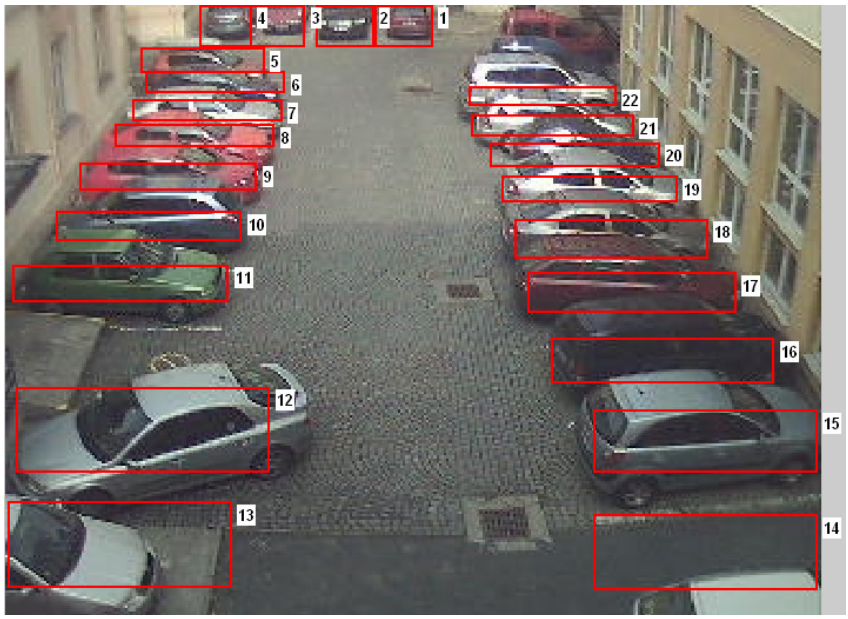
Dalším krokem je kontrola, kdy se srovná vstupní čas a čas vrácený funkcí `analyse()`. Pokud tyto časy souhlasí, pokračuje skript dále kontrolou relevantnosti získaných dat, kdy si z textového souboru `control.txt` vezme

průměrnou hodnotu pixelu. Pokud je hodnota v určeném rozmezí, vrátí se hodnota spolehlivosti 1 v opačném případě 0. Tato kontrola je kvůli případům, kdy by bylo například celé parkoviště zasněženo, nebo by obraz byl příliš tmavý (zakrytí kamery, špatné světelné podmínky).

Posledním krokem je potom vložení výsledků analýzy do databáze.

5.7 Analýza obrazového záznamu

Analýza obrazového záznamu se tedy odehrává v programu R, jež běží na linuxovém serveru. Pro potřeby analýzy bylo na snímku definováno celkem 22 oblastí zájmu, představující 22 parkovacích míst jež jsou pro kameru viditelná a jež jsou určena k používání (obr. 18).



Obr. 18 Definované oblasti zájmu na snímku parkoviště

Do analýzy v R, jíž zabezpečuje funkce `carDetection(timestamp)`, vstupuje jako jediná proměnná časová značka zpracovávaného snímku. V R kódu, jež je součástí příloh diplomové práce, se nejprve načte potřebný balíček biOps, po něm potom samotný obrázek, jež je převeden do odstínů šedi. Jako

první je provedena kontrola relevantnosti na základě průměrné hodnoty v ploše parkoviště, tato hodnota je zapsána do souboru `control.txt`.

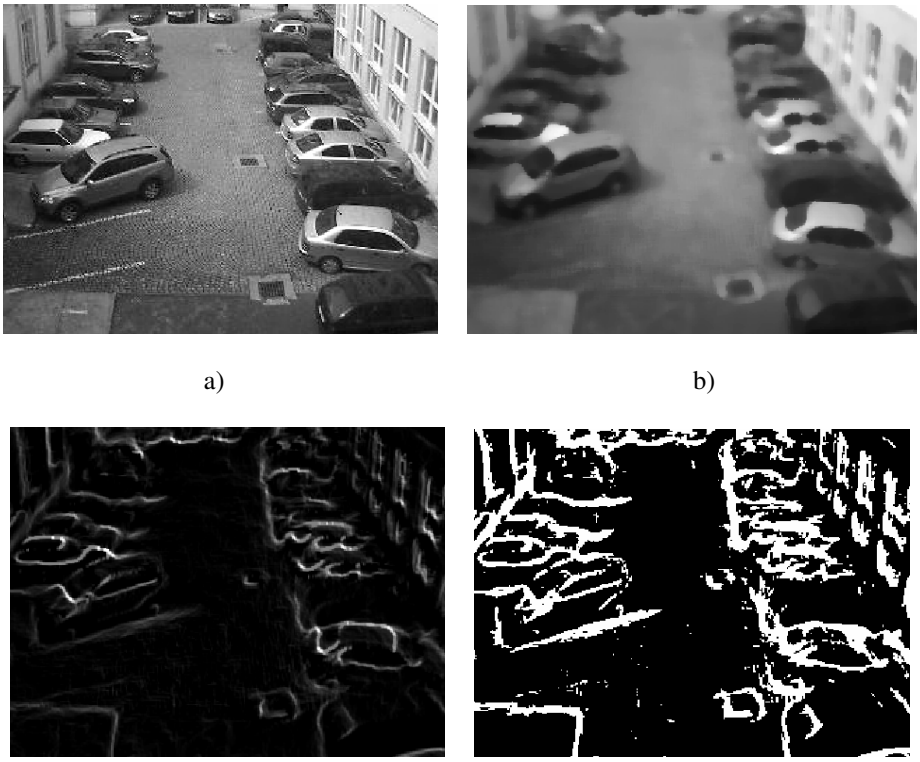
Dalším krokem je vylepšení kontrastu obrazu pomocí funkce `imgIncreaseContrast()`. Následně je obraz ošetřen mediánovým filtrem s velikostí okna 8, aby se odstranila textura povrchu parkoviště, jež by byla v dalších krocích na obtíž. Následně je provedeno další zvýraznění kontrastu a následně detekovány hrany pomocí Sobelova operátoru (obr X).

$$\mathbf{G}_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * \mathbf{A} \qquad \mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A}$$

Obr. 19 Sobelův operátor

Výsledné hrany jsou prahovány pro odstranění šumu. O tom zda bude parkovací místo detekováno jako obsazené, či jako volné rozhodují dvě kritéria. Prvním je hustota hran v daném parkovacím místě a druhým odlišnost průměrné hodnoty pixelu v parkovacím místě oproti průměru z plochy parkoviště. Pokud je rozhodnutí podle obou kritérií shodné, není potřeba dále nic kontrolovat. Pokud se však kritéria neshodnou, rozhoduje se na základě toho, které kritérium bylo určeno s větší jistotou, tj. s větším rozdílem od průměru nebo větší hustotou hran. Výsledek je zapsán do textového souboru, jehož obsah je následně importován do databáze.

Přesnost použité metody vychází dle provedených měření přibližně 80%. Velká část chyb je způsobena velkým překryvem vedle sebe stojících vozidel, toto je způsobeno nevhodným umístěním kamery. Také nízké rozlišení sejmutého obrazu kvalitě detekce nepřidává.



a) b)

C) d)

Obr. 20 Postup při zpracování obrazu: a) původní obraz se zvýšeným kontrastem, b)mediánový filtr, c) detekce hran Sobelovým operátorem, d) prahování hran

6 Sledování provozu na křižovatce

6.1 Výběr vhodné křižovatky

Požadavek při volbě lokality pro sledování provozu na křižovatce byl ještě striktnější, jelikož byla potřeba jednou kamerou zabrat celou křižovatku. Proto bylo potřeba kameru umístit co nejvýše, tento omezující požadavek nakonec vyústil ve výběr křižovatky ulic Na Vozovce a Foerstrova v Olomouci, na níž je dobrý pohled z přilehlého panelového domu na Foerstrově 33. Po jednání se sdružením vlastníků se nepodařilo dojednat trvalé umístění kamery, jež by vyžadovalo i instalaci potřebného zázemí. Muselo se tedy přistoupit k jiné formě sběru dat. Rozhodnuto bylo o použití digitálního fotoaparátu, jelikož na chodbě, kam byl poskytnut přístup, nebyla možnost zapojení do elektrické sítě. Tímto se také omezilo kontinuální snímaní pouze na zpracování několika záznamů. Zkoumaná křižovatka se nachází na frekventovaném průtahu městem, tudíž se zde očekávala vysoká hustota provozu.

Obr. 21 a 22 Panelový dům Foerstrova 33 a pohled na křižovatku z místa umístění fotoaparátu

6.2 Použitý hardware

Snímání obrazu

Jelikož nebylo možné nainstalovat trvale ani dočasně webovou kameru musela se najít jiná alternativa. Tou bylo použití fotoaparátu pro záznam videosekvencí. Jednalo se o fotoaparát značky Canon typ PowerShot A630. Ten je schopen zaznamenávat videosekvence do fotmátu Audio Video Interleave (AVI) s kompresí Motion JPEG. Maximální rozlišení videa je 640 na 480 px, což odpovídá běžnému rozlišení webové kamery. Ovšem optické a snímací vybavení fotoaparátu se s webkamerami těžko srovnává, u fotoaparátu je podstatně kvalitnější. Hlavní nevýhodou fotoaparátu je velký objem uložených dat, způsobený nevhodnou kompresí a také omezení maximální velikostí videosouboru na 1 GB. Z toho vyplývaly další opatření, jež byla provedena při získávání záznamu.

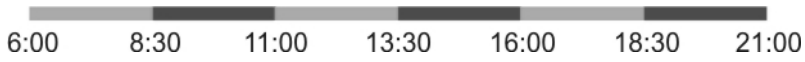
Pro stabilitu záznamu byl fotoaparát umístěn na stojan stativu, kalibrace proběhla dle značek umístěných na podlaze a také podle vzorového snímku uloženém ve fotoaparátu. Ovšem i při pečlivém nastavování není vždy zabraná identická scéna.

Analýza videozáznamů získaných fotoaparátem probíhala z důvodu relativně velké výpočtové náročnosti na dvou počítačích,. Prvním byl počítač s procesorem Intel Core 2 Duo T5800 2,0GHz s 4GB RAM, jako druhý počítač potom sloužil AMD Athlon 64 2,4GHz s 2GB RAM, oba s operačním systémem Windows XP Professional.

Obr. 23 Použitý fotoaparát Canon A630

6.3 Sběr dat

Nejdříve proběhlo několik natáčení pro různé světelné a povětrnostní podmínky, na nichž se provádělo testování a výběr vhodných algoritmů. Hlavním úkolem ale bylo vytvořit profil provozu v co největší části vybraného dne. Z důvodu omezených možností délky natáčení (byla použita 8GB paměťová karta, na níž se vlezlo přibližně 2,5 hodiny záznamu), bylo rozhodnuto, že se provede takové měření, aby byla zachycena co největší souvislá část běžného pracovního dne. Jelikož tedy nebylo možné natáčet v kuse déle než 2,5 hodiny, a také minimální maximální denní doba natáčení byla omezena, rozdělil se den od 6:00 do 21:00 na šest částí, z nichž vždy jedna byla natočena jeden vybraný den a následující zase den další.



Obr. 24 Rozdělení dne na jednotlivé úseky pro natáčení

První den měření byl čtvrtek 1.4.2010 a druhý den 22.4.2010. Oba tyto dny měli podobné počasí, bylo polojasno, se střídáním slunečního svitu. Ráno také byla mlha, jež prověřila schopnosti použitého algoritmu. Dny měření také odpovídají definici běžného pracovního dne, kterým je dle [34] úterý, středa nebo čtvrtek, pokud jsou pracovními dny a pokud jim předchází i po nich následuje pracovní den.

Celkem bylo natočeno 46 videosouborů ve formátu AVI (Audio Video Interleave), jejichž přehled je na tabulce 2. Během měření došlo k několika výpadkům ať už z technických nebo jiných důvodů, tato chybějící data byla později dopočítána (viz dále).

Sejmuto tedy mělo být celkem 15 hodin záznamu, ovšem poslední hodina (tj. od 20:00 do 21:00) nebyla z technických důvodů zabrána celá, tudíž je potom z následujících výpočtů vypuštěna.

video soubor	*počáteční čas*	*délka*	*koncový čas*	*rozdíl časů*
vid01	6:01:00	0:19:16	6:20:16	
vid02	6:20:18	0:24:26	6:44:44	0:00:02
vid03	6:44:46	0:29:16	7:14:02	0:00:02
vid04	7:14:05	0:29:22	7:43:27	0:00:03
vid05	7:43:29	0:27:15	8:10:44	0:00:02
vid06	8:12:28	0:20:31	8:32:59	**0:01:44**
vid07	8:35:32	0:18:45	8:54:17	**0:02:33**
vid08	8:54:20	0:19:06	9:13:26	0:00:03
vid09	9:13:28	0:18:54	9:32:22	0:00:02
vid10	9:32:24	0:18:56	9:51:20	0:00:02
vid11	9:51:23	0:18:41	10:10:04	0:00:03
vid12	10:10:07	0:18:43	10:28:50	0:00:03
vid13	10:28:52	0:18:50	10:47:42	0:00:02
vid14	10:47:56	0:08:25	10:56:21	0:00:14
vid15	11:00:59	0:19:28	11:20:27	**0:04:38**
vid16	11:20:29	0:19:15	11:39:44	0:00:02
vid17	11:39:47	0:19:22	11:59:09	0:00:03
vid18	11:59:11	0:18:49	12:18:00	0:00:02
vid19	12:19:19	0:18:58	12:38:17	**0:01:19**
vid20	12:38:19	0:18:47	12:57:06	0:00:02
vid21	12:57:09	0:18:30	13:15:39	0:00:03
vid22	13:15:41	0:08:19	13:24:00	0:00:02
vid23	13:25:53	0:18:57	13:44:50	**0:01:53**
vid24	13:44:52	0:19:13	14:04:05	0:00:02
vid25	14:04:07	0:19:08	14:23:15	0:00:02
vid26	14:23:22	0:18:56	14:42:18	0:00:07
vid27	14:42:21	0:19:07	15:01:28	0:00:03
vid28	15:01:39	0:17:44	15:19:23	0:00:11
vid29	15:19:31	0:17:49	15:37:20	0:00:08
vid30	15:37:26	0:10:55	15:48:21	0:00:06
vid31	15:53:59	0:18:30	16:12:29	**0:05:38**
vid32	16:12:44	0:18:34	16:31:18	0:00:15
vid33	16:34:28	0:18:44	16:53:12	**0:03:10**
vid34	16:53:15	0:19:03	17:12:18	0:00:03
vid35	17:12:21	0:19:19	17:31:40	0:00:03
vid36	17:31:48	0:19:23	17:51:11	0:00:08
vid37	17:51:14	0:19:25	18:10:39	0:00:03
vid38	18:10:42	0:08:34	18:19:16	0:00:03
vid39	18:19:16	0:18:45	18:38:01	0:00:00
vid40	18:38:05	0:18:11	18:56:16	0:00:04
vid41	18:56:20	0:17:08	19:13:28	0:00:04
vid42	19:13:33	0:18:34	19:32:07	0:00:05
vid43	19:32:11	0:17:52	19:50:03	0:00:04
vid44	19:50:20	0:18:31	20:08:51	0:00:17
vid45	20:09:04	0:18:29	20:27:33	0:00:13
vid46	20:27:46	0:11:10	20:38:56	0:00:13

Tab. 2 Natočená videa, šedě podbarvené jsou videa natočená 1.4.2010, ostatní jsou potom z 22.4.2010.

6.4 Použitý software

Pro práci s videem bylo použito několik programů. Freeware distribuovaný pod licencí GNU GPL **VirtualDub** a také freewareová verze programu **Any Video Converter** byly použity pro předzpracování videa.

Samotná analýza videozáznamu potom probíhala v prostředí **Matlab** verze 2009a, jehož licenci vlastní Katedra matematické analýzy a aplikací matematiky UP. Matlab je integrované prostředí pro vědeckotechnické výpočty, modelování, návrhy algoritmů, simulace, analýzu a prezentaci dat, měření a zpracování signálů, návrhy řídicích a komunikačních systémů. V práci se využívalo hlavně funkcí z Image Processing Toolbox. Pro následné zpracování výstupních dat bylo použito tabulkového procesoru MS Excel a statistického programu R.

Pro vizualizace potom bylo použito open source grafického editoru **Inkscape**, distribuovaného pod licencí GNU GPL. Na straně serveru, kde jsou produkovány vizualizace potom Apache web sever, MySQL, PHP, JPGraph, popsané v části o systému parkoviště. Další technologie využité pro vizualizaci jsou následující:

Scalable vector graphics (SVG) je jazyk, který popisuje dvojrozměrnou grafiku pomocí XML. Obraz je v nich reprezentován pomocí základních grafických objektů jako jsou úsečky, obdélníky, mnohoúhelníky apod., u kterých zaznamenáváme pouze jejich základní charakteristiky [3]. Zápis SVG je strukturou podobný HTML, ovšem se specifickými grafickými elementy.

Kartografové očekávali že SVG se stane novým standardem pro popis grafiky založené na XML. V blízké budoucnosti budou mapy na internetu budou ve formátu SVG [20]. To se ovšem vyplnilo jen částečně a to z toho důvodu nedokonalé podpory tohoto standardu u webových prohlížečů a také bezpečnostních omezení.

SVG také umožňuje definovat animace, které není potřeba ošetřovat žádným programovým kódem, tento způsob animace je kompatibilní se standardem SMIL (Synchronised Multimedia Integration Language).

Problémem u tohoto formátu je jeho nedokonalá podpora u téměř všech webových prohlížečích na trhu. Spolehlivé fungování SVG včetně animací je nyní pouze u prohlížečů Opera a Chrome. Mozzila Firefox SVG standard podporuje, ovšem bez intagrace SMIL animací. Hojně rozšířený Internet Explorer verze 8, nemá nativní podporu žádnou, ovšem chystaná verze 9 by už měla s podporou SVG přijít, otázka je v jakém rozsahu.

Pro ovládání, větší interaktivitu a funkčnost byl ve spojení s formátem SVG použit také **JavaScript**, což je multiplatformní, objektově orientovaný skriptovací jazyk pro WWW stránky, často vkládaný přímo do HTML (zde SVG) kódu stránky. K jeho spuštění dochází v prohlížeči, tj. na straně klienta.

V těsné návaznosti na JavaScript je použito standardu **Document object model** (DOM), což je specifikace W3C pro na platformě a jazyku nezávislý způsob přístupu k obsahu a struktuře dokumentu.

6.5 Analýza video záznamu – počítání vozidel

6.5.1 Předzpracování videozáznamu

V rámci předzpracování se nejprve jednalo o změnu kompresního formátu jelikož funkce `aviread()`, jež byla v programu Matlab použita, akceptovala pouze omezený počet formátů. Dalším důvodem pro konverzi videa byly kapacitní důvody, jelikož 2,5 hodiny záznamu zabíraly na disku celých 8 GB, bylo tedy vhodné zvolit formát s lepší kompresí. Výsledný formát komprese pro výstupní avi soubory byl Indeo video 5.10, převod byl proveden v programu VirtualDub.

Při provádění pokusů s delšími videozáznamy ovšem funkce `aviread()` vykazovala extrémní pokles výkonu při čtení snímků s vyšším číslem indexu. Vina není jen na straně funkce, ale také na použitém kompresním formátu, jež

je značně pomalý. Bylo tedy potřeba najít alternativu, jež by byla rychlejší. Tou se stala funkce `mmreader()`, vytvářející multimediální objekt pro načítání videa. Jako výstupní formát byl opět avi, ovšem nyní s kompresí VMW2, pro konverzi do tohoto formátu bylo potřeba použít programu Any Video Converter.

Pro další optimalizaci práce s videem v Matlabu bylo každé video otočeno o definovaný úhel. Důvodem bylo, že detekční zóny, používané pro počítání vozidel mají obdélníkový tvar definovaný souřadnicemi levého horního rohu, šířkou a výškou a bylo tedy potřeba obraz otočit tak, aby detekční zóny přibližně kopírovaly pruhy na komunikacích. Pokud se toto otočení provádělo v Matlabu, velice to snižovalo výkon algoritmu. Tomuto kroku by se dalo předejít natočením kamery již při samotném snímání obrazu.

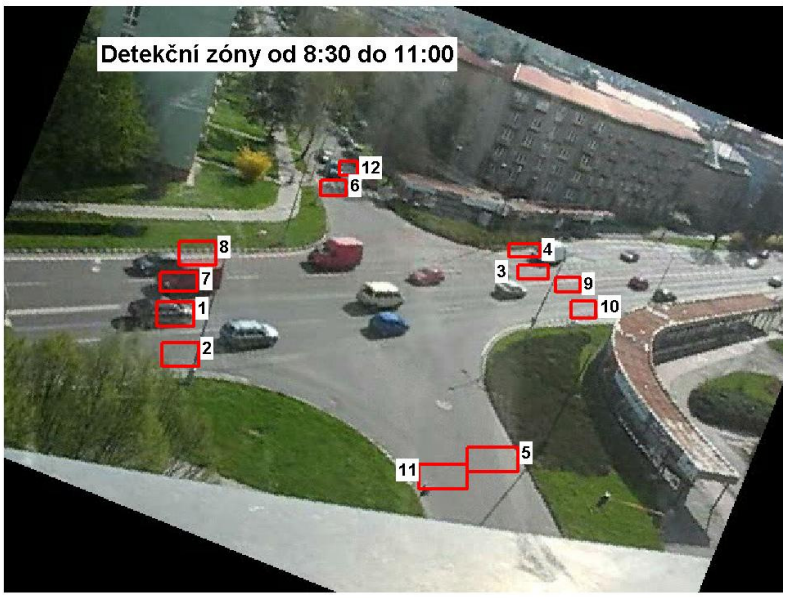
6.5.2 Počítání vozidel

Definování detekčních zón

Data nachystaná pro vstup do Matlabu bylo nyní možné zpracovat pro získání počtu projetých vozidel. Nejdříve ale bylo potřeba definovat detekční zóny (ROI – regions of interest). Jelikož umístění kamery bylo pro každou část natáčeného dne jiné, bylo také nutné definovat pro každou část dne zvláštní detekční zóny. Pro interaktivní a snadné určování souřadnic bylo použito objektu `imrect()`, která do obrazu vyextrahovaného z videa vykreslí uchopitelný a upravitelný obdélník. Po upravení jeho polohy a velikosti je možné pomocí kontextového menu kopírovat do schránky pole se souřadnicemi a rozměry. Takto byla vytvořena první sada ROI, viz ukázka:

```
r1 = [126 243 30 20];
...
r12 = [275 128 15 11];
r = [r1;r2;r3;r4;r5;r6;r7;r8;r9;r10;r11;r12];
```

Kde `r` je výsledné pole obsahující všechny ROI a jejich souřadnice. Pro další snadnější definování ROI u ostatních částí dne byla vytvořena funkce defineROIs(), jež vytvoří editovatelné obdélníky s umístěním a rozměry ze vstupního pole, ty je pak možno ručně upravit, aby vyhovovaly dané situaci. Další funkce drawROIs() potom vykreslí dané ROI na pozadí vybraného snímku určitého videa a výsledek uloží do rastrového obrázku. Výsledek je vidět na následujícím obrázku.



Obr. 25 Vykreslené ROI z funkce drawROIs()

Při definování detekčních zón byla brána na zřetel vzdálenost zóny od místa natáčení, ale také způsob, jakým vozidla daným směrem projíždí, tedy že například zóny 5 a 11 musí být širší, neboť vozidla zde nemají přesně definované pruhy a volí si různé trasy.

Algoritmus detekce

Vlastní algoritmus detekce je založen na metodě ADF (adjacent frame differencing). Jedná se tedy o odečítání po sobě následujících snímků a

prahování výsledku rozdílu. Pro samotné počítání byla vytvořena funkce `vehicleCounting()` jejíž fungování je dále popsáno.

Algoritmus byl zvolen kvůli své univerzálnosti, rychlosti ale také proto, že rozlišení videa a často také špatné povětrnostní podmínky neumožňovaly aplikaci pokročilejších rozpoznávacích znaků, jako například detekce hran nebo optického toku.

Vstupní parametry funkce jsou počáteční čas videa (zvláště zadány hodiny, minuty a vteřiny), cesta k videu jež se má zpracovat a také cesta k výstupnímu souboru. Parametry ovlivňující samotnou detekci jsou:

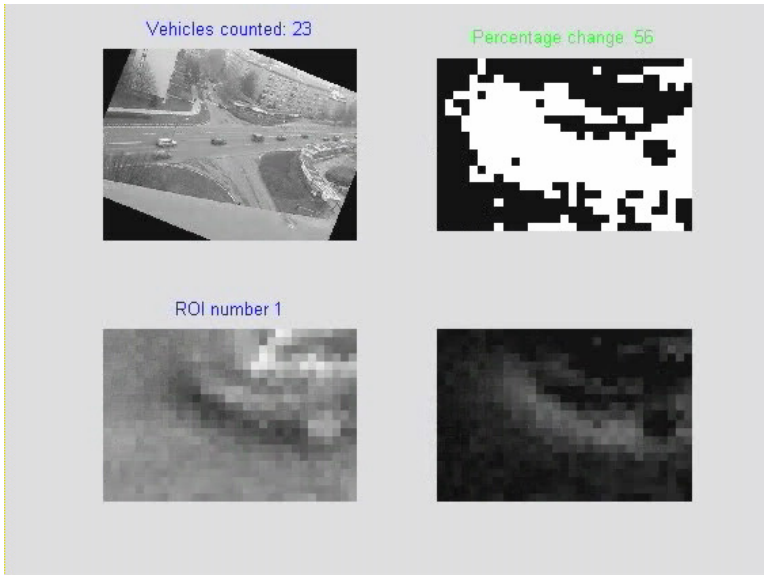
- pole obsahující souřadnice všech ROI (`r`) – jak bylo popsáno výše
- velikost kroku (`step`) – určuje kolik snímků bude v každém cyklu vynecháno, tj. určuje citlivost na rychle jedoucí vozidla ale také na vozidla jedoucí pomalu, blíže je vše objasněno dále
- prahová hodnota (`threshParam`) – definuje citlivost na šum a malé rozdíly mezi odečtenými snímky při převodu na binární obraz, nabývá hodnot `[0,1]`
- detekční parametr (`detectionParam`) – určuje citlivost detekce na základě definice plochy ROI, jež musí obsahovat změněné pixely, např. hodnota 4 říká, že musí být změněna minimálně ¼ pixelů v ROI aby bylo vozidlo detekováno

Při zpracování videa dojde nejprve k výpočtu správných časů, tj. na základě počátečního času, délky videa, rychlosti snímků se vypočítá počet celých minut a také případné necelé minuty obsažené ve videu. Výchozím časem je zde čas kdy se začalo snímat první video, tj. 6:01:00, což převedeno na minuty je 361.

Následně se pro aktuální počítanou minutu vytvoří pole obsahující indexy snímků videa vypočtené na základě délky zpracovávané minuty (první a poslední minuta nemusejí být celé) a velikosti kroku. V dalším cyklu se postupně prochází pole s indexy snímků a načítají se z videa. Dva následující snímky jsou postupně zpracovávány pro každou definovanou ROI. Snímky

jsou nejdříve ořezány a následně odečteny, a s použitím prahové hodnoty převedeny na binární obraz. Z tohoto obrazu je potom spočítána plocha změněných pixelů. Pokud je větší než definovaný detekční parametr a pokud předcházející stav detekce je `0`, je vozidlo detekováno. Z toho plyne omezení, že algoritmus pracuje až od druhého vyšetřovaného páru snímků.

Detekované počty jsou vždy pro každou ROI sečteny po skončení minuty a zapsány do pole, z nějž jsou po provedení detekce pro všechny snímky ve všech minutách zapsány do souboru. Zde bylo zvoleno formátování textového souboru pomocí čárek, čili soubor s příponou CSV (comma separated value). Takto může být soubor dále zpracováván tabulkovým procesorem nebo naimportován do databáze.



Obr. 26 Ukázka z výstupního videa detekce

Pro okamžitou kontrolu výsledků detekce byla vytvořena funkce `vehicleCountingVideo()`, jež má podobné vstupní parametry jako funkce předcházející, akorát nevytváří výstupní soubor textový ale video soubor s ukázkou detekce pro zvolenou ROI. Je možné také definovat časový rozsah detekce, tedy pokud například chceme zkoumat jen určitou problémovou situaci.

Vyhodnocení přesnosti

U každého algoritmu pro detekci vozidel je důležité vyhodnocení přesnosti. Zde byla spolehlivost vyšetřována pro různé světelné a povětrnostní podmínky, tedy nejen pro data natočená pro potřebu analýzy jednoho pracovního dne, ale také data sesbíraná dříve pro testování algoritmu, zde nechybí sníh, mlha a osvětlená vozovka lesklá po dešti.

Pro každou z vybraných podmínek bylo provedeno několik detekcí a to na různých ROI. Výsledky chyb jsou v tabulce 3. Uvedené ukázky jsou k nalezení na přiloženém DVD. Největší chyb jsou patrné za mlhy, kdy dochází k nezapočítání téměř 20% projetých vozidel. To je dáno velice nízkým kontrastem, který se zde díky mlze vyskytuje (obr. 27 b). Poněkud lepších výsledků bylo dosaženo pokud za mlhy byly rozsvíceny světla pouličního osvětlení. Další problémovou situací byla lesklá vozovka za tmy. Zde totiž docházelo k odrazům čelních světel automobilů, jež zasahovaly do jiných detekčních zón a způsobovaly tak opačnou situaci jako u mlhy (obr. 27 c), tedy větší počet detekovaných vozidel, a to až 17%. Použití algoritmu pro tyto situace tedy není příliš vhodné.

Ostatní situace nezpůsobovaly takové výrazné chyby, výrazné jsou také při dlouhých stínech, při nichž může docházet ke „spojování vozidel“ (obr. 27 f) a tím podhodnocení výsledků, nebo k zasahování stínu do sousední ROI a tím zase může dojít ke zvýšení počtu detekovaných vozidel.

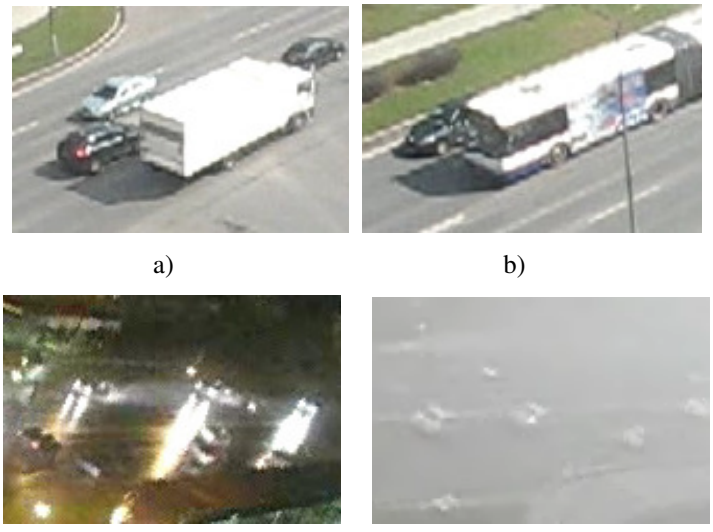
podmínky	*Z*	*R*	*chyba %*	*ukázka*
zataženo	79	77	2,6	uk07,uk08,uk09
slunečno, minimální stíny	44	42	4,8	uk16,uk17,uk18
slunečno, dlouhé stíny	61	58	5,2	uk13,uk14,uk15
nasněženo, lehké sněžení	64	60	6,7	uk10,uk11,uk12
tma, mlha	26	30	13,3	uk01,uk02,uk03
noc, mokrá vozovka	62	53	17,0	uk19,uk20,uk21
mlha	29	36	19,4	uk04,uk05,uk06

Tab. 3 Hodnocení přesnosti sčítání vozidel za rozdílných podmínek (Z - počet vozidel detekovaných algoritmem, R - skutečný počet projetých aut)

Menší chyb lze dosáhnout zvláštním vhodným nastavením vstupních parametrů funkce, ovšem to snižuje univerzálnost celého řešení. Vstupní parametry ovlivňující detekci byly pro celý vyšetřovaný den stejné, s výjimkou prvních pěti videí, jež byla pořízena za silné mlhy, a kde byla potřeba snížit chybu způsobenou špatnými světelnými a kontrastními podmínkami.

Zdroje chyb

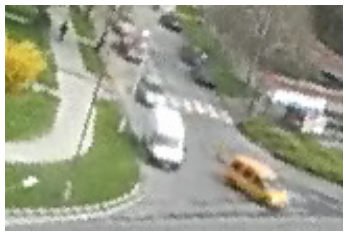
V návaznosti na předchozí kapitolu jsou popsány nejčastější příčiny chyb při detekci. Ty vycházejí z již popsaných povětrnostních vlivů a světelných podmínek (obr. 27 c, d, g), ale také působením dalších faktorů jako například při velké výšce vozidla zasahujícího do více ROI (obr. 27 a, b), nebo při těsné blízkosti po sobě jedoucích vozidel, kdy dochází k detekci pouze prvního vozidla (obr. 27 f). Dalším zdrojem chyby může být a započítání skupiny cyklistů jako vozidla (obr 27 h), nebo také neukázněnost a porušování předpisů ze strany řidičů (obr. 27 e).



a) b)

c) d)

e)



f)

g)

h)

Obr. 27 Ukázky nejčastějších příčin chyb

Určitý vliv na výskyt chyb může mít také volba velikosti ROI a také velikost kroku při načítání snímků videa. Na základě velikosti ROI a daném kroku snímků byly vypočítáno následující přibližné rozmezí rychlostí, při nichž dochází ke spolehlivé detekci.

délka ROI [m]	***min. rychlost [m/s] | [km/h]***		***min. rychlost [m/s] | [km/h]***	
2	1,9	6,8	22,5	81
3	2,8	10,1	26,3	94,5
5	4,7	16,9	33,8	121,5

Tab. 4 Vypočtené maximální a minimální rychlosti vhodné pro detekci

Výpočetní náročnost

Zpracování videa je relativně náročné na výpočetní výkon, jelikož je zde velký tok dat. Naprogramovaná funkce byla upravována tak aby, poskytovala co nejrychleji požadované výsledky. Optimalizace spočívala například v dopředném definování délky datových polí, znovu používání proměnných apod.. Následující tabulka ukazuje průměrné výpočetní časy pro oba použité počítače.

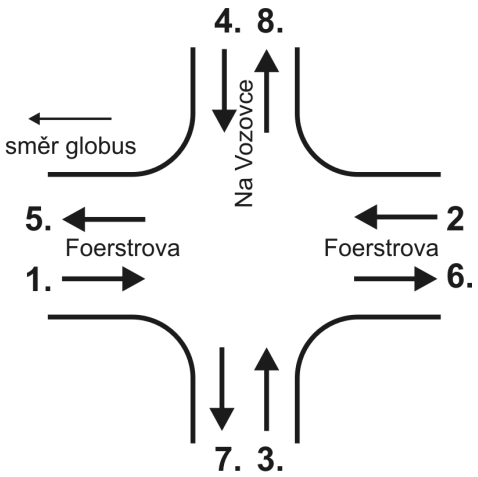
počítač	*doba analýzy 1 minuty videa při kroku 4 snímky*	*doba analýzy 1 minuty videa při kroku 5 snímků*
pc1	**54 s**	**34 s**
pc2	**1 min 10 s**	**57 s**

Tab. 5 Rychlost výpočtů, pc1: Intel Core 2 Duo T5800 2,0GHz s 4GB RAM, pc2: AMD Athlon 64 2,4GHz s 2GB RAM, oba WinXP Pro

Zpracování dat

Výstupní data pro každý video soubor ve formátu csv byla spojena do jednoho kompaktního souboru. Dále byla sečtena data z dvouproudých směrů pouze do jednoho sloupce. Výsledkem je celkem 8 směrů pro než je zaznamenán počet projetých vozidel (obr. 28). Byla také potřeba doplnit chybějící data. To bylo provedeno v MS Excel pomocí lineární regrese ze tří předcházejících a tří následujících záznamů. Došlo tedy k jistému shlazení dat v místech kde byla takto data dopočítána, jelikož tento dopočet příliš nerespektoval výraznou sezónní složku v datech. Pro další vizualizace a výpočty toto ovšem nebyl problém, protože s přesností na jednotlivé měření se už dále nepracovalo.

Takto upravená data byla dále připravena pro vyhodnocení. Jelikož byla ve formátu csv, nebyl je problém naimportovat do databáze MySQL pro vytváření webových vizualizací.



Obr. 28 Označení detekovaných směrů na křižovatce

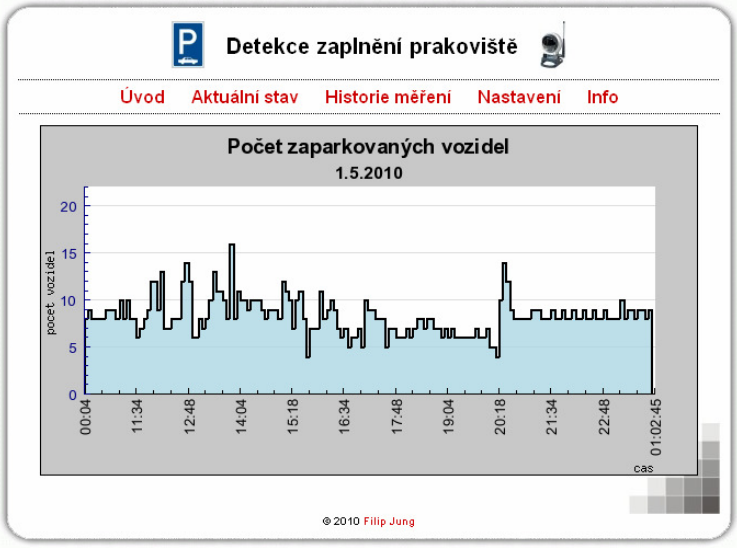
7 Výsledky

7.1 Vizualizace výsledků sledování parkoviště

Aktuální výsledek analýzy je možno vidět na Internetu na adrese: http://geohydro.upol.cz/parking_app/results.php, kde je v době od 6:00 do 20:00 zobrazen aktuální počet volných míst, čas posledního snímku a také obrázek obsahující nejaktuálnější snímek z kamery. Pokud uživatel na stránku vstoupí v době, kdy detekce neprobíhá, je mu tato informace sdělena. Rovněž pokud například došlo k nějakému výpadku kamery, tedy kdy je rozdíl mezi aktuálním časem a časem posledního snímku větší jak 7 minut (co 5 minut se spouští php skript a co dvě minuty je snímán obraz, v nejhorším případě je tedy maximální rozdíl 7 minut), je tato skutečnost zobrazena.

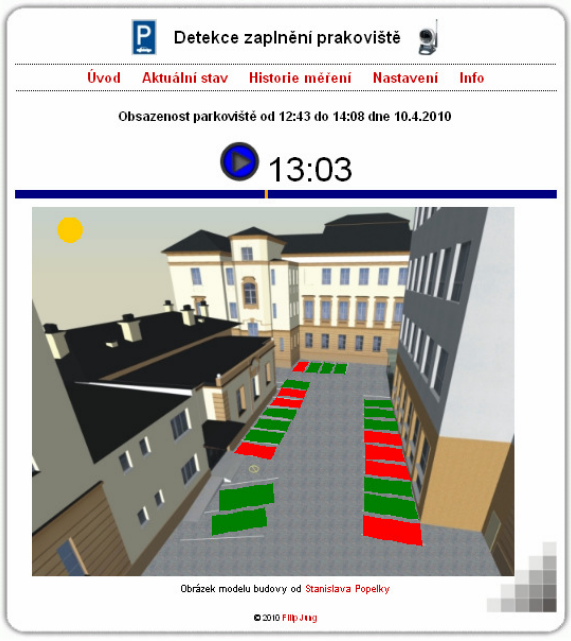
Obr. 29 Stránka s informacemi o stavu parkoviště

Jelikož jsou data po celou dobu běhu systému ukládána do databáze, je možné je zpětně prohlížet. Toto je na stránkách dostupné pod položkou menu Historie měření. Zde má uživatel možnost vybrat si ze dvou způsobů vizualizace pro daný den: grafu nebo jednoduché animace.



Obr. 30 Ukázka grafu ve webové aplikaci

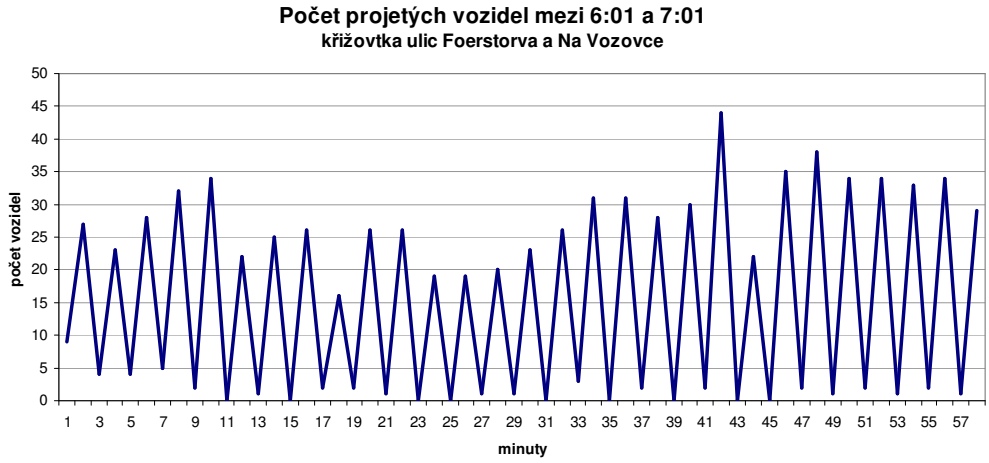
Graf je vytvářen dynamicky pomocí knihovny JPgraph a zobrazen jako rastrový obrázek. Animace je provedená tak, aby pro její prohlížení stačil samotný prohlížeč, a nebyla potřeba žádného pluginu. Jedinou podmínkou je zapnutí JavaScriptu. Animaci lze pomocí jednoduchých ovládacích prvků řídit a přesunout se tak na požadovanou denní dobu.



Obr. 31 Ukázka animace z webové aplikace

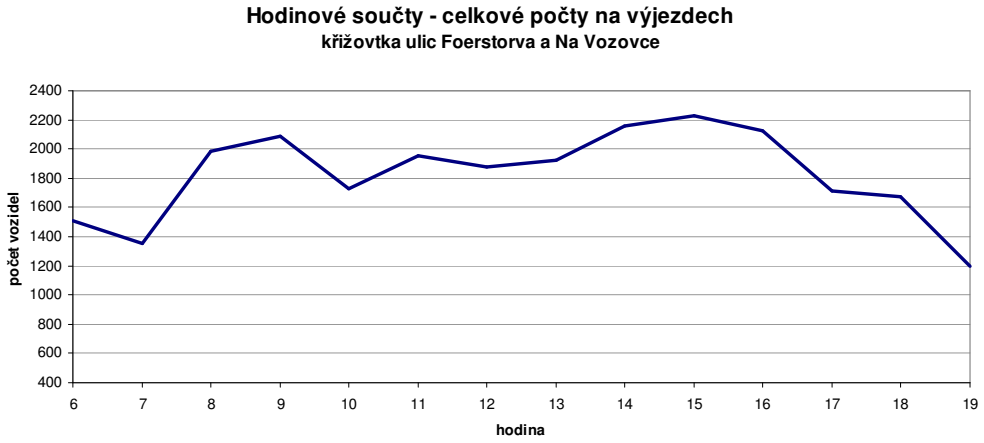
7.2 Vyhodnocení a vizualizace sčítání na křižovatce

Z naměřených a doplněných dat bylo možno vytvořit grafické a tabelární výstupy, jež by charakterizovaly denní provoz na sledované křižovatce. Měření vynesená do grafu po minutách mají charakter časové řady. Na první pohled je vidět sezónní složka, jež představuje interval světelné signalizace na křižovatce. Jelikož světelná signalizace byla v provozu po celou dobu měření, mají veškerá data shodný průběh.



Abychom data očistili od sezónní složky, a mohli modelovat jejich trend použijeme na data metodu klouzavých průměrů. Nyní je u následujícího grafu pro výjezdní směry patrnější trend. Data na počátku časové osy jsou vynechaná z důvodu průměrování oknem o velikost 30.

Můžeme provést srovnání s grafem vytvořeným z hodinových součtů pro výjezdní směry a vidíme, že trend je patrný i z něj.



Během dne můžeme identifikovat dvě výraznější maxima představující ranní a odpolední dopravní špičku.

Mezi počty aut v jednotlivých směrech lze očekávat závislost, vyjádříme ji pomocí korelace. Ta je velice nízká mezi hlavními příjezdními (1,2) směry a bočními výjezdními směry (7,8), vysoká naopak mezi hlavními příjezdními a výjezdními směry, což dokazuje, že převládající směr dopravy je po hlavní silnici a počet odbočených vozidel do bočních ulic je minimální. Také hledat závislost mezi bočními příjezdními směry a směry hlavními by bylo zbytečné.

příjezdní směr	*výjezdní směr*	*korelace*
1	6	0,89
1	7	-0,07
2	5	**0,96**
2	8	0,46

Tab. 6 Korelace pro vybrané směry

h	*s 1*	*s 2*	*s 3*	*s 4*	*součty vjezd*	*s 5*	*s 6*	*s 7*	*s 8*	*součty výjezd*
6	863	475	77	111	1526	595	758	69	88	1510
7	909	532	104	109	1654	753	500	70	31	1354
8	928	734	141	103	1906	828	985	105	69	1987
9	989	843	137	106	2075	902	1003	113	71	2089
10	823	766	95	82	1766	805	759	93	73	1730
11	885	837	139	93	1954	901	873	99	79	1952
12	952	803	105	105	1965	852	847	116	61	1876
13	990	778	107	107	1982	803	931	109	77	1920
14	953	902	106	164	2125	1011	944	121	80	2156
15	962	978	139	175	2254	1059	953	137	80	2229
16	922	943	101	138	2104	1027	877	132	91	2127
17	736	877	97	116	1826	794	733	117	71	1715
18	681	794	107	99	1681	776	723	87	86	1672
19	537	497	74	83	1191	528	544	59	67	1198
Σ	**12130**	**10759**	**1529**	**1591**	**26009**	**11634**	**11430**	**1427**	**1024**	**25515**

Tab. 7 Výsledná data pro jednotlivé směry

Srovnání s daty ze sčítání dopravy

U naměřených dat se přímo nabízí možnost srovnání s existujícími daty, jimiž je například údaj pro úsek procházející křižovatkou z celostátního sčítání dopravy 2005. V tomto úseku bylo v roce 2005 naměřeno **32 446** projetých vozidel [23]. Aby bylo možné z dat naměřených v rámci diplomové práce vytvořit odhad za 24 hodin, bylo použito tabulek v [34], jež určují jaká část celkové denní intenzity náleží které hodině v průběhu dne. Výsledkem je číslo **27 240**, jež je od staršího údaje značně odlišné. Tento pokles v intenzitě je velice pravděpodobně způsoben otevřením západního obchvatu Olomouce v roce 2007.

Vizualizace na webu

Na adrese http://geohydro.upol.cz/crossroads_app je pro uživatele k dispozici možnost tvorby animací z naměřených dat. Pro vizualizace v podobě animací byly vytvořeny součty po čtyřech minutách, jež jsou nahrány v MySQL databázi. Uživatel si pak může vybrat s jakou rychlostí chce animaci vidět a ta se mu následně zobrazí. Animace jsou interaktivní ve formátu SVG. Na webu jsou také k dispozici další grafy pro jednotlivé směry.

Obr. 32 Ukázka animace provozu na křižovatce

8 Diskuze

V obou řešených úkolech se pracovalo s analýzou obrazového záznamu, jež představuje zdroj dosud nedostatečně využívaných dat (s výjimkou dálkového průzkumu země) pro GIS systémy. Mnoho projektů a aplikací zmíněných v této práci ukázalo, že takto získaná data jsou spolehlivá a metody jejich získání jsou značně propracované a robustní. Při využívání nástrojů počítačového vidění je jejich hlavní síla v automatickém zpracování a také ve schopnosti přizpůsobovat se změnám, bez nutnosti zásahu člověka. Propracované systémy využívající těchto technologií tak představují účinný nástroj pro efektivní a sofistikované zpracování dat.

Systém sledování parkoviště – návrhy a vylepšení

Hlavní nevýhodou představené aplikace je špatné umístění kamery, jež ovšem v dané situaci nešlo lépe vyřešit. Vhodnějším umístěním kamery by se docílilo úplného pokrytí pozorovaného parkoviště a také snížení míry překrytu vedle sebe stojících aut, jež značně stěžuje spolehlivou detekci.

Také lepší kamera s větším rozlišením a kvalitnější optikou, by zajistila přesnější výsledky. Samotný použitý algoritmus má také své nedostatky, především při náhlých změnách osvětlení. Použití sofistikovanějšího algoritmu schopného se sám „učit“ by zajisté zvýšilo úspěšnost detekce.

Pokud by tedy poskytované výsledky byly úplné a správné, bylo by vhodné tento systém napojit na další zařízení. Nyní má uživatel možnost navštívit stránku ze svého mobilního zařízení a zjistit tak aktuální stav na parkovišti. Vhodné by bylo například napojení na GPS a technologii TMC nebo napojení na signalizační systém u vjezdu do parkoviště (obr.)

Použitá vizualizace představuje pro uživatele jednoduchou a atraktivní formu prohlédnutí naměřených dat. V případě grafů by bylo ještě možné nechat do jednoho grafu vykreslit více dní pro snadné srovnání. Představené animace

jsou jednoduchou dynamickou vizualizací, jež běží bez nutnosti instalace jakéhokoli pluginu do webového prohlížeče.

Obr. 33 Návrh možného dalšího napojení systému

Sledování provozu na křižovatce – návrhy a vylepšení

Počítání vozidel projetých křižovatkou představovalo komplexní úkol. Opět se jednalo o proces od sběru dat až po jejich interpretaci. Použitý algoritmus odpovídal kvalitě vstupních dat, jež neumožňovala efektivní implementaci složitějších metod. I přesto by bylo několik vylepšení možných, například detekce směru pohybu, aby se zamezilo chybným detekcím při překrytu vozidla do vedlejší detekční zóny. Také sledování detekovaných vozidel by pomohlo pro lepší interpretaci výsledků, neboť nyní není možné přesně zjistit jak se přesně vozidla po křižovatce pohybovala. I přesto jsou naměřená data zatížena vcelku malou chybou a lze je srovnávat s daty sesbíranými terénními pracovníky.

Naměřená data budou poskytnuta Magistrátu města Olomouce pro jejich případnou potřebu.

Natočených videí mělo být také použito i jako podkladů pro trestní oznámení, jelikož na křižovatce se dlouhodobě pohybovala těžká nákladní vozidla, jež zde měla zákaz vjezdu a jejich provoz znepříjemňoval žití nájemníkům nejen na Foerstrově 33 ale také zatěžoval boční komunikace nastavěné na takovéto zatížení. K využití dat pro tento účel nedošlo, neboť byly dodatečně nainstalovány dopravní značky, jež těmto průjezdům zamezily. Těžká nákladní doprava tak byla odkloněna jinam.

9 Závěr

Cíle stanoveného na počátku tvorby práce bylo dosaženo. Byly vytvořeny systémy zpracovávající obrazová data od sběru až po vizualizaci a interpretaci výsledků.

Sestavený systém pro sledování parkoviště, je schopen online poskytovat informaci o stavu zaplnění parkoviště, tuto informaci ukládat do databáze a v případě potřeby následně vizualizovat. Jako vylepšení je mimo jiné navrženo lepší umístění kamery, jelikož nynější umístění kamery bylo až náhradním řešením po selhání jednání o jiném lepším místě. Fungující systém používá otevřené softwarové prostředí a standardy, je tak snadno přenositelný na jiné aplikace.

V části práce zabývající se sledováním provozu na křižovatce byl sestaven algoritmus schopný sčítat vozidla celý den s přiměřenou chybou. Jedná se ale pouze postprocessingové zpracování, jelikož nebyla možnost trvalé instalace webové kamery. Bylo tedy použito záznamu na digitální fotoaparát a následného zpracování na stolním počítači. Výsledná data z měření na křižovatce ulic Foerstrova a Na Vozovce byla vyhodnocena a uživatelům umožněna na webu jejich vizualizace. Proběhlo také srovnání naměřených dat s oficiálními daty ze sčítání dopravy 2005.

Zdroje

1. COLLINS, R. T. et al.: A System for Video Surveillance and Monitoring – Final report [online]. [cit. 2009-05-10]. Carnegie Mellon University, 2000
2. CUCCHAIARA R., PICCARDI, M., MELLO, P.: Image analysis and rule-based reasoning for a traffic monitoring system. IEEE Intelligent Transportation Systems, vol. 1, no. 2, 2002. s. 119-130.
3. DOBEŠOVÁ, Z.: *Databázové systémy v GIS*. Olomouc, Univerzita Palackého, 2004, 76s., ISBN 80-244-0891-0.
4. FARHAN, B., MURRAY A.T.: Siting park-and-ride facilities using a multi-objective spatial optimization model. In Computers & Operations Research, vol. 35, 2008. s. 445-456.
5. HEJRAL, M.: Interval.cz - Průvodce SVG [online]. 2003-2006, [cit. 2009-04-20]. Dostupné z: http://interval.cz/webdesign/grafika/.
6. IDRIS, M.Y.I. et al.: Car Park System: A Review of Smart Parking System and its Technology. In Infromation Technology Journal 8 (2), 2009. s. 101-113. ISSN 1812-5638
7. JONES, W. D.: Parking 2.0. In IEEE Spectrum, vol. 43, 2006. ISSN: 0018-9235
8. KAMPEL, M.: Bildfolgen – Skriptum. Technische Universität Wien, Institut für Rechnergestützte Automation, Vers. 1.0, 2008.
9. KLEIN, L.A., GIBSON, D., MILLS, M.K.:Traffic Detector Handbook, FHWA-HRT-06-108, Federal Highway Administration, US Department of Transportation, Washington, DC, 2006.
10. KOLLER, D. , WEBER, J., MALIK, J.: Robust multiple car tracking with occlusion reasoning. In European Conference on Computer Vision, 1994, s. 189-196.
11. LIU, S. Robust vehicle detection from parking place images [online]. Boston University, 2006. [cit. 2009-10-23]. Dostupné z: http://iss.bu.edu/jkonrad/Publications/reports/Liu05-06buece.pdf
12. NEUMANN, A., WINTER, A. M.: Vector-based Web Cartography: Enabler SVG. In Carto.net [online]. [cit. 2009-10-20]. Dostupné z: http://www.carto.net/papers/svg/index_e.shtml.
13. SHAHEEN, S.A., RODIER, C.J., EAKEN, A.M.: Smart parking management field test: A bay area rapid transit (bart) district parking demonstration [online]. University of California, Institute of

Transportation Studies, 2005 Dostupné z: http://pubs.its.ucdavis.edu/download_pdf.php?id=44

14. SHAFIE, A. A., HAFIZ, F., ALI, M. H.: Motion Detection Techniques Using Optical Flow. Proceedings of World Academy of Science, Engineering and Technology, vol. 56, 2009. ISSN: 2070-3724.
15. ŠONKA, M., HLAVÁČ, V.: Počítačové vidění. Praha, Grada, 1992. 272 s. ISBN 80-85424-67-3.
16. ŠTOLFA, S. et al.: iParking – Inteligentní systém parkování [online]. In Sborník sympozia GIS Ostrava 2007. [cit. 2009-10-23]. ISSN 1213-239X
17. THOU-HO CHEN et al.: Intelligent Vehicle Counting Method Based on Blob Analysis in Traffic Surveillance. Proceedings of the Second ICICIC, 2007. ISBN:0-7695-2882-1
18. VEERARAGHAVAN, H., MASOUD, O., PAPANIKOLOPOULOS, N. P.: Computer vision algorithms for intersection monitoring. IEEE Intelligent Transportation Systems, vol. 4, no. 2, June 2003. s. 78-89.
19. VOŽENÍLEK, V. *Aplikovaná kartografie I. Tematické mapy.* Olomouc, Univerzita Palackého, 2. vyd., 2004. 187 s. ISBN 80-244-0270-X.
20. VOŽENÍLEK, V.: *Cartography for GIS,* Olomouc, Univerzita Palackého, 2005, 142s., ISBN 80-244-1047-8
21. WU, Q. et al.: Robust Parking Space Detection Considering Inter-Space Correlation. In IEEE International Conference on Multimedia and Expo, Beijing, 2007. s. 659-662. ISBN: 1-4244-1016-9
22. Apache Software Foundation, The [online]. [cit. 2009-10-10]. Dostupné z: http://www.apache.org/
23. Celostátní sčítání dopravy 2005 – ŘSD [online]. [cit. 2010-04-06]. Dostupné z: http://www.scitani2005.rsd.cz/mesta/ol/olomouc.jpg
24. Click and Park [online].Online aplikace. [cit. 2010-03-12]. Dostupné z: http://www.clickandpark.com
25. JavaScript vector draw library [online]. Stránky programu. [cit. 2009-01-20]. Dostupné z: http://www.c-point.com/javascript_vector_draw.htm
26. JpGraph [online]. [cit. 2009-01-10]. Dostupné z: http://aditus.nu/jpgraph/
27. Matlab documentation [online]. [cit. 2009-05-20]. Dostupné z: http://www.mathworks.com/access/helpdesk/help/helpdesk.html
28. Olomoucký deník 26.3.2009 [online]. [cit. 2009-04-07] Dostupné z: http://olomoucky.denik.cz/zpravy_region/zacpa-na-tride-svobody-ridicum-o-ni-reknou-cedule-.html

29. MySQL 5.0 Reference Manual [online]. [cit. 2009-10-10]. Dostupné z: http://dev.mysql.com/doc/refman/5.0/en/spatial-extensions.html.
30. PHP: Hypertext Preprocesor [online]. [cit. 2009-10-15]. Dostupné z: http://php.net/.
31. R-PHP - project pages [online]. [cit. 2009-09-12]. Dostupné z: http://dssm.unipa.it/R-php/
32. Robotic Parking [online]. Firemní stránky. [cit. 2010-03-12]. Dostupné z: http://www.robopark.com
33. Scalable Vector Graphics 1.1 Specification [online]. [cit. 2009-10-15]. Dostupné z: http://www.w3.org/TR/SVG11/.
34. Stanovení intenzit dopravy na pozemních komunikacích, Technické podmínky. EDIP s.r.o.. Mariánské Lázně, Koura publishing, 2007. 52s. ISBN 978-80-902527-7-6
35. Summary of Vehicle Detection and Surveillance Technologies used in Intelligent Transportation Systems, A [online] [cit. 2009-10-23]. The Vehicle Detector Clearinghouse, New Mexico State University, 2007.
36. Traficon NV [online]. Oficiální stránky společnosti. [cit. 2010-03-12]. http://www.traficon.com/
37. International Road Dynamics [online]. Oficiální stránky. [cit. 2010-02-15]. Dostupné z: http://www.irdinc.com/products/sensors_accessories/in_road_sensors/permanent_loops.php

Summary

This Master thesis is a conclusion of the Geoinformatics Masters degree study program at Department of Geoinformatics, Faculty of Science, Palacky University in Olomouc.

Means of computer vision, though widely developed, are not fully implemented and connected with GIS applications. Their main potential is an automated and intelligent data acquisition. The initial higher cost and infrastructure expenditures at the beginning are compensated with effectiveness and reliability. The thesis deals with using web cameras to monitor automobile traffic on parking areas and cross-roads. Parking area task is solved using wireless web camera which sends acquired image to processing server every five minutes. There the image segmentation and free and occupied places recognition takes place. Open source statistic software R is used for image processing. Resulting data are stored in database and can be subsequently visualized and examined through online creation of tables, graphs and animated maps, furthermore information about actual free places is accessible through web pages, where driver can see, whether he should drive to this particular parking area. Vehicle counting on cross-roads is more challenging, and due to the impossibility to set static camera for longer period and because of computing demands, only short times observations were made. However, particular observations were put together to simulate the daily flow of traffic. Video analysis is done in post processing in Matlab software using mainly optical flow algorithm implementation. Output data could be examined in tables, interactive graphs and animated maps. Created applications show connection of computer vision with GIS technologies and could be reusable to similar scenarios.

Přílohy

DVD 1

- videosoubory používané pro detekci

DVD 2

- videosoubory používané pro detekci
- videoukázky detekce
- textová část diplomové práce
- zdrojové kódy pro PHP, R a Matlab
- csv soubor s naměřenými daty
- webové stránky

ZDROJOVÉ KÓDY

a. drawROIs.m
b. vehicleCounting.m
c. code_r.txt

pozn. zbytek zdrojových kódů je z kapacitních důvodů na DVD

a) drawROIs.m

```
%FUNCTION
%drawROIs - shows and saves the image file with ROIs and their
numbers
%INPUT:
%fps - frames per second of input video
%frameTime - time of desired image to be drawn in seconds
%startTime - time to by displayed in title
%endTime - time to by displayed in title
%sourceVideo - path to video from which the image will be taken
%roiArray - name of variable holding array of current ROIs
coordinates
%outputImage - path to image file in which the produced image
will be saved

%EXAMPLE drawROIs(50, '6:01', '8:32', 'data\06_00-
08_30\vid01.avi', r, 'data\06_00-08_30\rois.jpg')

function[] = drawROIs(frameTime, startTime, endTime,
sourceVideo, roiArray, outputImage)

      %reads video into multimedia object and gets its frame
rate
      videoObj = mmreader(sourceVideo);
      fps = get(videoObj, 'frameRate');
      sampleFrame = frameTime * fps;

      %reads the choosen frame
      I = read(videoObj,sampleFrame);

      fig = figure;
      imshow(I);

      %prints the title
      title = sprintf('Detekční zóny od %s do %s', startTime,
endTime);

      t = text(80, 40, title, 'FontSize', 14, 'FontWeight',
'bold', 'BackgroundColor', 'white');

      %draws rectangles of ROIs and their numbers
      i = 1;
   while (i<=12)
     roi = roiArray(i,1:4);
     rectangle('EdgeColor','red','LineWidth',2,'Position',
roi);
      i = i + 1;
   end

   i = 1;
        while (i<=12)
     x = roiArray(i,1) + roiArray(i,3) + 3;
     y = roiArray(i,2) + 5;
```

```
      if (i == 11 || i == 3)
        x = roiArray(i,1) - 17;
           end
      k = num2str(i);
      text(x, y, k, 'FontSize', 10, 'FontWeight', 'bold',
'BackgroundColor', 'white', 'Margin', 1);
      i = i + 1;
         end

       %saves output image
       saveas(fig, outputImage);

    end
```

a) vehicleCounting.m

```
%FUNCTION
%vehicleCounting - counts vehicles in given video at defined
ROIs and writes result into csv file
%INPUT:
%step - defines how many frames will be skipped in every loop
%vidStartHour - hour part of time when the video starts
%vidStartMin - minutes part of time when the video starts
%vidStartSec - seconds part of time when the video starts
%roiArray - name of variable holding array of current ROIs
coordinates
%detectionParam - defines how much should be ROI covered by
object to be counted (4 = 1/4)
%thresParam - number in range [0,1], pixels greater than param
with the value 1 (white)
    %and replaces other pixels with the value 0 (black), i.e.
param 0.5 is
    %midway between black and white, purposed value is 0.05
with good contrast
    %conditions and 0.03 with low contrast
%filename - path to video from which the image will be taken
%outFilename - path to file in which the produced results will
be saved

%example: vehicleCounting(5, 6, 20, 18, r, 4, 0.05 'data\06_00-
08_30\vid01.avi','vid01_out.csv')

function [] = vehicleCounting(step, vidStartHour, vidStartMin,
vidStartSec, roiArray, detectionParam, thresParam, filename,
outFilename)

numROIs = length(roiArray);         %returns number of ROIs
from length of array containing ROIs
videoObj = mmreader(filename);      %reads video into
multimedia object
fps = round(get(videoObj, 'frameRate'));   %returns frame rate
of video in seconds
```

```
vidLen = round(get(videoObj, 'duration')); %returns length of
video in seconds

partOfFirstMinute = 60 - vidStartSec;  %returns length of part
of first minute in video (video usually does not start at 0
sec)

partOfLastMinute = vidLen - partOfFirstMinute - (60 *
floor((vidLen - partOfFirstMinute) / 60));

%returns number of minutes contained in analysed video includes
all minutes contained, not only whole
numMinutes = ceil((vidLen - partOfFirstMinute)/60) + (1 *
ceil(partOfFirstMinute/60));

startMinuteAll = 361;  %because first video starts at 6:01:00
startMinuteFromAll = (vidStartHour * 60 + vidStartMin + 1) -
startMinuteAll;

%prints calculated values into command line
fprintf('Start time of video is
%u:%u:%u\n',vidStartHour,vidStartMin,vidStartSec);
fprintf('Length of video %u sec\n',vidLen);
fprintf('Number of contained minutes %u\n',numMinutes);
fprintf('part Of First Minute: %u\n',partOfFirstMinute);
fprintf('part Of Last Minute: %u\n',partOfLastMinute);
fprintf('Minute from start: %u\n',startMinuteFromAll);

countedAll = zeros(numMinutes, numROIs + 1);  %defining array
for keeping actual minute and number of detected vehicles

m = 1;

%loop for every minute contained in the video
while (m <= numMinutes)

    %if it is first or last minute it takes its length from
counted
    %values, else it takes normal length of minute
    if (m == 1)
        lengthOfMinute = partOfFirstMinute;
    elseif (m == numMinutes)
        lengthOfMinute = partOfLastMinute;
    else
        lengthOfMinute = 60;
    end

    %counts min and max index of frame
    if (m == 1 || m == numMinutes)
        minFrame = (m - 1) * lengthOfMinute * fps;
    else
        minFrame = maxFrame;
    end
    maxFrame = minFrame + (lengthOfMinute * fps);
```

```
    %counts and displays number of frames, according to the
defined step
    numFrames1 = floor((maxFrame - minFrame)/step);
    fprintf('Num frames 1: %u\n',numFrames1);

    frames = zeros(1, numFrames1); %defining array for keeping
frame index
    o = minFrame;
    %if minFrame is 0 it adds 1 to it, bacause avi index starts
with 1
    if (o == 0 || o == 1)
       frames(1) = 1;
       q = 1;
    else
       frames(1) = o;
       q = 0;
    end

    %while maxFrame it adds a index value of frame into the
defined array
    k = 1;
    while (o < maxFrame-step)
        frames(k+1) = o + step - q;
        o = o + step;
        k = k + 1;
    end

    %counts and displays number of frames, according to the
length of the
    %array holding frame indexes
    numFrames = length(frames);
    fprintf('Num frames 2: %u\n',numFrames);

    counted = zeros(numROIs,numFrames);   %defining array for
keeping actual counted vehicles
    states = zeros(numROIs,numFrames);

    %reads the first frame from avi file
    video1 = read(videoObj, frames(1));
    j = 1;
    while (j <= (numFrames-1))
        %reads the following frame from avi file
        video2 = read(videoObj, frames(j+1));

        %converts images into greyscale
        img1 = rgb2gray(video1);
        img2 = rgb2gray(video2);

        %detects vehicles for all ROIs
        n = 1;
        while (n <= numROIs)
            width = roiArray(n,3);
            height = roiArray(n,4);
```

```
            %crops input images according to the ROI
coordinates
            img1crop = imcrop(img1, [roiArray(n,1)
roiArray(n,2) width height]);
            img2crop = imcrop(img2, [roiArray(n,1)
roiArray(n,2) width height]);

            %subtracts cropped images
            imgDiff = img2crop - img1crop;
            imgDiffAbs = abs(imgDiff);

            %thresholds subtracted image with threshold
parameter
            imgDiffBw = im2bw(imgDiffAbs, thresParam);

            %returns number of pixels that showed significant
change
            area = bwarea(imgDiffBw);

            %tolerance defines how much should be ROI covered
by object to be counted
            tolerance = (width*height)/detectionParam;

            %writes into arrays 1 if vehicle is present 0 if
not,
            %another vehicle is counted if there was at least
one empty
            %frame before it
            if (area > tolerance)
                if (j > 1)
                    states(n, j) = 1;
                    if (states(n, j-1) == 0)
                        counted(n, j) = 1;
                    else
                        counted(n, j) = 0;
                    end
                else
                    counted(n, j) = 1;
                    states(n, j) = 1;
                end
            else
                counted(n, j) = 0;
                states(n, j) = 0;
            end

            n = n + 1;
        end

        %reuse of the second image
        video1 = video2;
        fprintf('%u\n',j);
        j = j + 1;

    end
```

```
    %gets actual real minute (defined from start of measuring -
6:01)
    actualMinute = startMinuteFromAll + m - 1;

    %counts number of vehicles for processed minute and writes
it into an array
    n = 1;
    while (n <= numROIs + 1)
        if (n == 1)
            countedAll(m,n) = actualMinute;
        else
            countedAll(m,n) = sum(counted(n - 1,1:numFrames));
        end
        n = n + 1;
    end

    fprintf('Actual minute from start: %u\n',actualMinute);

    m = m + 1;
    fprintf('Actual minute of video: %u\n',m);
end
    %writes nummbers of counted vehicles for each minute into
definedfile
    i = 1;
    rowLen = numROIs + 1;
    fid = fopen(outFilename, 'w');
    while (i <= numMinutes)
        fprintf(fid,
'%u;%u;%u;%u;%u;%u;%u;%u;%u;%u;%u;%u;%u\n',
countedAll(i,1:rowLen));
        i = i + 1;
    end
    fclose(fid);
end
```

c)code_r.txt

```
timestamp = 1272946560

carDetection = function(timestamp) {

    #sets working directory and loads needed package
    setwd('/var/www/html/parking_app/r/')
    library(biOps)

    #reads image and converts it into greyscale
    i1 =
readJpeg("/var/www/html/parking_app/r/out/last_img.jpg")
    i1 = imgRGB2Grey(i1)
    xStart = 115
    yStart = 19
    xDim = 65
    yDim = 220
```

```
    #extracts the mean value from parking place surface for
later control
    meanOverall = mean(i1)
    write(meanOverall,
"/var/www/html/parking_app/r/out/control.txt", ncolumns=1,
append = FALSE, sep="")

    b = imgCrop(i1, xStart, yStart, xDim, yDim)

    bMean = mean(b)
    bSd = sd(b)

    #increases image contrast
    c = imgIncreaseContrast(i1, (min(i1)+(mean(i1)/4)),
(max(i1)-(mean(i1)/4)))

    #plot(c)

    #calls block median filter
    block = imgBlockMedianFilter(c, 8)

    #plot(block)

    #increases image contrast
    c1 = imgIncreaseContrast(block,
(min(block)+(mean(block)/4)), (max(block)-(mean(block)/4)))
    #detects image edges using Sobel operator
    edges = imgSobel(c1)

    eMean = mean(edges)
    eTol = sd(edges)/2

    #thresholds edges for noise reduction
    thEdges = imgThreshold(edges, eMean + eTol)

    #plot(thEdges)

    bEdges = imgCrop(thEdges, 48, 0, 192, 240)
    meanBEdges = round(mean(bEdges))

    coordsX =
c(146,123,97,77,54,56,51,44,30,21,4,5,2,232,232,215,206,201,196
,191,184,183);
    coordsY =
c(1,1,1,1,18,27,38,48,63,82,103,151,196,201,160,132,106,85,68,5
5,44,33);
    dimsX =
c(22,22,21,20,48,54,58,62,69,72,84,99,87,87,87,87,81,75,68,66,6
3,57);
    dimsY =
c(16,16,16,16,9,8,8,8,10,11,14,33,33,29,24,17,15,15,10,9,8,7);

    row = array("", c(1, length(coordsX)))
```

```
    i = 1
    tolerance = 13

    #do check for all parking places
    while (i <= length(coordsX)) {
      xStart = coordsX[i]
      yStart = coordsY[i]
      xDim = dimsX[i]
        yDim = dimsY[i]
      imgRawCropped = imgCrop(i1, xStart, yStart, xDim, yDim)

      imgEdgesCropped = imgCrop(thEdges, xStart, yStart, xDim,
yDim)
      meanEdgesCropped = round(mean(imgEdgesCropped))

        #plot(imgRawCropped)
        #plot(imgEdgesCropped)

        meanDiff = abs(mean(imgRawCropped) - bMean)

        p1=0;
        p2=0;

        #if the mean value is greater then tolrence, there is
detected vehicle
        if (meanDiff > tolerance) {
       stat = "is"
      }
      else {
       stat = "isnot"
      }

      #if the mean value is greater than mean in edges from
background, there is detected vehicle
      if (meanEdgesCropped > (meanBEdges+(meanBEdges/2))) {
       stat1 = "is"
      }
      else {
       stat1 = "isnot"
      }

        if (stat == stat1) {
          stat2 = stat
        }

        #decides which criteria is more significant
        else {

          p1 = abs(round(bMean)-
round(mean(imgRawCropped)))/round(bMean)
          p2 = abs(meanEdgesCropped-meanBEdges)/meanBEdges

          if (p1 > p2) {
```

```
            stat2 = stat
          }
          else {
            stat2 = stat1
          }
        }

        if (stat2 == "is") {
          s = 1
        }
        else {
          s = 0
        }

        row[,i] = paste("'",s,"'", sep="")

        i = i + 1;
    }

    #writes output into a text file
    write(row, "/var/www/html/parking_app/r/out/output.txt",
ncolumns=length(coordsX), append = FALSE, sep=",")
    return(timestamp)
}

#calls car detection function
carDetection(timestamp)

#quits the R
q("no")
```