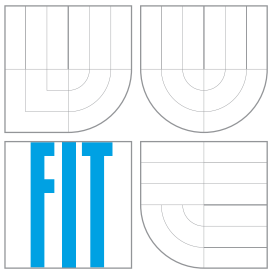
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ODHAD PARAMETRŮ OBJEKTŮ Z OBRAZŮ

ESTIMATION OF OBJECT PARAMETERS FROM IMAGES

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. BRONISLAV PŘIBYL

VEDOUCÍ PRÁCE
SUPERVISOR

Doc. Dr. Ing. PAVEL ZEMČÍK

BRNO 2010

Abstrakt

Rapidní rozvoj komunikačních technologií v posledním desetiletí zapříčinil zvýšení objemu informací, které lidé a organizace generují a sdílejí. V současné spleti je stále těžší identifikovat relevantní zprávy, protože ještě neexistují nástroje a techniky pro inteligentní správu informace v masovém měřítku. Obrazová informace je vzhledem k multimediální povaze dnešních médií stále frekventovanější a důležitější. Tato práce popisuje software pro automatický odhad předem definovaných vlastností objektů v obraze. Je také popsána implementace tohoto algoritmu v jazyce C++.

Abstract

Rapid expansion of communication technologies in last decade caused increased volume of information which is beeing generated and shared by people and organisations. It is permanently harder to identify relevant content today because of absence of tools and techniques, which may support mass information management. As today's media have rather multimedial character image information is even more important. This project describes software for automatic estimation of predefined object parameters from images. A C++ implementation of this algorithm is also described.

Klíčová slova

Detekce objektů, rozpoznávání, kalibrace scény z jednoho snímku, automatický odhad parametrů.

Keywords

Object detection, pattern recognition, scene calibration from single image, automatic parameters estimation.

Citace

Bronislav Přibyl: Odhad parametrů objektů z obrazů, diplomová práce, Brno, FIT VUT v Brně, 2010

Odhad parametrů objektů z obrazů

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Doc. Dr. Ing. Pavla Zemčíka. Další informace mi poskytli Ing. Michal Hradiš a Ing. Roman Juránek. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Bronislav Přibyl
25. května 2010

Poděkování

Děkuji především vedoucímu mé diplomové práce Doc. Dr. Ing. Pavlu Zemčíkovi za trpělivost a odborné vedení, dále Ing. Michalu Hradišovi a Ing. Romanu Juránkovi za konzultace a poskytnutí detekčního softwaru. Děkuji také Ing. Michalu Seemanovi a Ing. Pavlu Žákovi za konzultace.

Obsah

Kapitola 1

Úvod

Vynález písma umožnil člověku jednoduše přenášet a hlavně uchovávat informace nutné k jeho přežití. Postupem času se začaly přenášet i jiné zprávy, než jen ty nutné k přežití, což ovšem nesnížilo jejich důležitost. Díky rapidnímu rozvoji komunikačních technologií v posledním desetiletí je dnes velmi snadné generovat a sdílet velké množství informací. Jednotlivci i celé organizace produkují obsah, který je následně sdílen individuálně i v různých komunitách. Objem informací však roste tak závratným tempem, že je již velmi těžké se v této spleti vyznat a identifikovat relevantní zprávy. V současné době ještě neexistují efektivní nástroje a techniky, které by umožnily inteligentní zpracování a správu informace ve zmíněném měřítku. Tím pádem jsou uživatelé ochuzeni o znalosti a z nich plynoucí výhody, které by z vydolovaných informací mohli získat.

Projekt WeKnowIt[1] si klade za cíl prozkoumat a vyvinout právě techniky, které by umožnily inteligentní správu a rozpoznávání důležitých a relevantních informací ve velkém měřítku. Povaha úlohy vyžaduje využití znalostí a technik z různých oborů informačních technologií. Je totiž nutné zpracovávat nejen text, ale i zvukové a obrazové materiály či videosekvence, protože velká část dnešních komunikačních kanálů má multimediální charakter.

Počítačové vidění je jedním z pilířů výše popsané úlohy. Grafická podoba informace totiž dokáže o určité skutečnosti vypovědět mnohem více než běžný textový popis a také minimalizuje možnou dezinterpretaci zobrazené skutečnosti. Pokud je třeba scénu zachycenou v obraze popsat stručně a při tom výstižně, je vhodné extrahovat z obrazu vysokoúrovňové informace, jako např. typy objektů vyskytujících se v obraze, jejich vzájemnou polohu a vztah nebo jejich metrické vlastnosti. Tato práce si klade za cíl navrhnout a vyvinout software pro pokud možno automatický odhad metrických vlastností předem specifikovaných zajímavých objektů v obraze. Může se jednat například o odhad velikosti, plochy nebo vzájemné vzdálenosti objektů jako jsou automobily, lidé, psi, domy, vodní plochy atd.

Dokument je členěn do několika částí. V kapitolách 2 a 3 je čtenář seznámen se základními přístupy k detekci objektů a kalibraci scény, ve 4. kapitole je diskutován současný stav odhadu geometrie scény z jediného snímku. V kapitole 5 je rozebrána implementace softwaru pro automatický odhad parametrů objektů v obraze spolu s jeho aplikací na reálné obrazy. Závěrečnou kapitolu tvoří zhodnocení dosažených výsledků a možné pokračování práce v budoucnu.

[1] Viz http://www.weknowit.eu.

Kapitola 2

Detekce objektů

Metody pro detekci objektů v obraze využívají různé přístupy, z nichž některé není možné zcela přesně kategorizovat. Pro lepší představu je však lze rozdělit např. do těchto kategorií [27]:

- Metody pracující shora dolů – na vysoké úrovni detailu se hledají kandidátní objekty, které se verifikují podle nízkoúrovňových detailů. Například při hledání obličeje se nejprve naleznou oblasti s barvou podobnou lidské kůži a tyto se následně testují na přítomnost očí, nosu a úst.
- Metody pracující zezdola nahoru – nejprve se vyhledají části objektů (např. podle barvy, textury, tvaru) u kterých se poté ověřuje jejich vzájemná pozice a velikost.
- Metody založené na modelu – model objektu popisuje jeho vizuální vlastnosti a zjišťuje se, jak dobře model odpovídá zkoumanému místu v obraze.
- Metody založené na vzhledu – nejprve se natrénuje tzv. klasifikátor, kterým se pak skenuje celý obraz. Klasifikátor pro každou zkoumanou pozici v obraze provede rozhodnutí, zda se na dané pozici hledaný objekt vyskytuje či nikoliv.
- Metody využívající neuronových sítí – neuronová síť je schopna detekovat nejen objekty přesně odpovídající trénovací množině, ale i objekty různým způsobem natočené a zkreslené (např. lidské obličeje z profilu apod).

Níže jsou podrobněji popsány vybrané metody detekce objektů, jež jsou pro tuto práci relevantní.

2.1 Klasifikace pomocí AdaBoostu

AdaBoost je algoritmus strojového učení s učitelem, který patří mezi tzv. boosting metody. Pod pojmem boosting se v oboru strojového učení myslí obecná metoda pro dosažení vyšší přesnosti jakéhokoliv učícího algoritmu s učitelem [19]. Prakticky jde o vhodné zkombinování tzv. slabých klasifikátorů, z nichž každý klasifikuje data jen o něco lépe než náhodně, v jediný silný klasifikátor, který vstupní data klasifikuje velmi dobře. Ve většině případů jde o lineární kombinaci slabých klasifikátorů.

Algoritmus byl poprvé publikován v [18] a k detekci objektů v obraze ho jako první použili Viola a Jones v [45]. Výsledný silný klasifikátor je tvořen lineární kombinací slabých

klasifikátorů. Ke kladům algoritmu patří garance exponenciálního snižování chyby na trénovacích datech až na libovolně nízkou úroveň za předpokladu, že množina slabých klasifikátorů obsahuje jen takové, které klasifikují lépe než náhodně [19].

Vstupem algoritmu je množina trénovacích dat $S = \langle (x_1, y_1), \ldots, (x_m, y_m) \rangle, x_i \in X, y_i \in Y = \{-1, +1\}, i \in \langle 1, m \rangle$ a množina slabých klasifikátorů. AdaBoost opakovaně vybírá slabé klasifikátory v iteračních krocích $t \in \langle 1, T \rangle$. Úkolem slabého klasifikátoru je nalézt hypotézu $h_t : X \to \{-1, +1\}$. V každém kroku je vybrán jeden slabý klasifikátor, kritériem výběru je minimální chyba na trénovacích datech s ohledem na jejich distribuci $D_t(i)$. Chyba je součtem vah špatně klasifikovaných vzorků:

$$\varepsilon_t = \sum_{i:h_t(x_i) \neq y_i} D_t(i) \tag{2.1}$$

Vybraná hypotéza je přidána k silnému klasifikátoru s koeficientem α_t, který je závislý na chybě ε_t hypotézy h_t při aktuální distribuci vstupních dat D_t:

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \varepsilon_t}{\varepsilon_t} \right) \tag{2.2}$$

Výsledný silný klasifikátor $H(x)$ je lineární kombinací vybraných slabých hypotéz:

$$H(x) = \operatorname{sign} \left(\sum_{t=1}^{T} \alpha_t h_t(x) \right) \tag{2.3}$$

V každém iteračním kroku proběhne převážení trénovacích dat tak, že dobře klasifikovaným vzorkům se váha sníží, naopak chybně klasifikovaným vzorkům se váha zvýší. Nová distribuce D_{t+1} se spočte jako

$$D_{t+1}(x_i) = \frac{D_t(x_i) \cdot e^{-\alpha_t y_i h_t(x_i)}}{Z_t}, \tag{2.4}$$

kde Z_t je normalizační faktor zvolený tak, aby funkce D_{t+1} zůstala pravděpodobnostním rozložením – musí platit $\sum_{i=1}^{m} D_{t+1}(x_i) = 1$. Přepočítávání distribuce $D(t)$ je jedním ze základních principů AdaBoostu. Váha $D_t(x_i)$ vyjadřuje, jak dobře je vzorek x_i klasifikován v kroku t všemi slabými klasifikátory vybranými v předchozích krocích. Pseudokód je ukázán v algoritmu 2.1.

Viola a Jones v [45] představili několik zásadních vylepšení výše popsaného přístupu pro účely detekce objektů v obraze. Upravili algoritmus AdaBoost tak, aby vybíral pouze několik málo slabých klasifikátorů, které reprezentují nejdůležitější vizuální rysy hledaného objektu (jako slabé klasifikátory použili příznaky připomínající Haarovy vlnky). Pro zrychlení výpočtu těchto příznaků zavedli tzv. integrální obraz a výsledné silné klasifikátory pak ještě zapojili do tzv. kaskády. Vysvětlení uvedených pojmů následuje.

Modifikace AdaBoost algoritmu spočívá ve změně kódování ohodnocení vzorků z bipolárního na binární $Y = \{0, 1\}$ pro negativní, resp. pozitivní vzorky a v mírně odlišné práci s vahami vzorků. Na počátku jsou váhy pozitivních vzorků inicializovány na hodnotu

Algoritmus 2.1 AdaBoost.

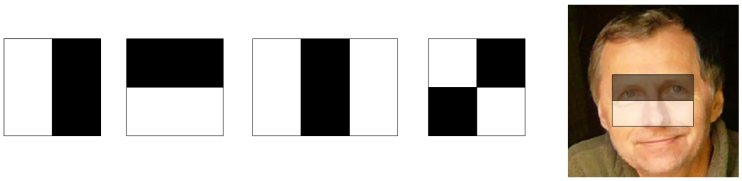
1: **Vstup**: $S = \langle (x_1, y_1), \ldots, (x_m, y_m) \rangle, x_i \in X, y_i \in Y = \{-1, +1\}, i \in \langle 1, m \rangle$
2: **Inicializace**: $\forall i \in \langle 1, m \rangle : D_1(x_i) = \frac{1}{m}$
3: **pro** $t = 1$ do T **proveď**
4: Vyber slabý klasifikátor $h_t : X \to \{-1, +1\}$ s nejmenší chybou ε_t pro distribuci D_t.
5: Spočti jeho váhu $\alpha_t = \frac{1}{2} \ln \left(\frac{1-\varepsilon_t}{\varepsilon_t} \right)$.
6: Proveď převážení vzorků trénovací množiny $D_{t+1}(x_i) = \frac{D_t(x_i) \cdot e^{-\alpha_t y_i h_t(x_i)}}{Z_t}$,
kde Z_t je normalizační faktor vysvětlený u rovnice 2.4.
7: **konec** (pro každé)
8: **Výstup**: Silný klasifikátor $H(x) = \operatorname{sign} \left(\sum_{t=1}^{T} \alpha_t h_t(x) \right)$

$w_{1,i} = \frac{1}{2m}$ a váhy negativních vzorků na hodnotu $w_{1,i} = \frac{1}{2l}$, kde m a l jsou počty pozitivních, respektive negativních vzorků. Jelikož rozložení w není pravděpodobnostním rozložením, je nutné na začátku každé iterace algoritmu provést normalizaci tohoto rozložení. Pseudokód může čtenář vidět v algoritmu 2.2.

Algoritmus 2.2 AdaBoost podle Violy a Jonese [45].

1: **Vstup**: $S = \langle (x_1, y_1), \ldots, (x_n, y_n) \rangle, x_i \in X, y_i \in Y = \{0, 1\}, i \in \langle 1, n \rangle$
2: **Inicializace**: $\forall i \in \langle 1, n \rangle : w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$
3: **pro** $t = 1$ do T **proveď**
4: Normalizuj váhy tak, aby w_t bylo pravděpodobností rozložení: $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$.
5: Pro každý příznak j natrénuj slabý klasifikátor h_j, jenž používá pouze jediný příznak.
Spočti jeho chybu ε_j vzhledem k w_t: $\varepsilon_j = \sum_i w_i |h_j(x_i) - y_i|$.
6: Vyber slabý klasifikátor h_t s nejnižší chybou ε_t.
7: Proveď převážení vzorků trénovací množiny $w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$,
kde $e_i = 0$ pokud je vzorek x_i klasifikován správně, jinak $e_i = 1$ a $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$
8: **konec** (pro každé)
9: **Výstup**: Silný klasifikátor $H(x) = \begin{cases} 1 & \text{pro } \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{jinak} \end{cases}$,
kde $\alpha_t = \log \frac{1}{\beta_t}$

Uvedený modifikovaný algoritmus pracuje se slabými klasifikátory, které nevyhodnocují přímo hodnoty pixelů, ale tzv. Haarovy příznaky. Jedná se o příznaky založené na Haarových vlnkách, které publikoval již v roce 1909 Alfred Haar [22]. Tyto vlnky jsou jednoduché skokové funkce lokálního charakteru. Jejich rozšířením do Haarových příznaků vzniknou primitivní dvourozměrné konvoluční filtry ilustrované na obrázku 2.1. Princip jejich použití spočívá ve vyčíslení rozdílu intenzity mezi pixely obrazu v bílé a černé oblasti filtru. Haarovy příznaky poskytují informace o obraze na abstraktnější úrovni než hodnoty pixelů, což je pro detekční úlohy výhodné. Druhou nezanedbatelnou výhodou je rychlost jejich vyhodnocení – při použití integrálního obrazu je doba výpočtu konstantní nezávisle na velikosti příznaků.

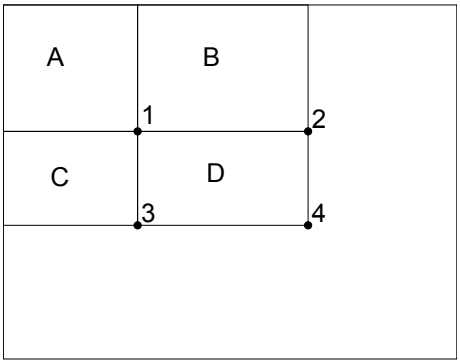


Obrázek 2.1: Ilustrace Haarových příznaků (vlevo) a jejich význam při detekci obličeje (vpravo). Převzato z [41].

Termín „integrální obraz“ poprvé použili Viola a Jones v roce 2001 v [44], nicméně se netají tím, že se inspirovali podobnou datovou strukturou již dříve používanou v počítačové grafice. Integrální obraz ii je rastr hodnot o stejné velikosti jako obraz i, ze kterého je vypočten. Jednotlivé hodnoty se spočtou podle rovnice 2.5.

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x', y') \tag{2.5}$$

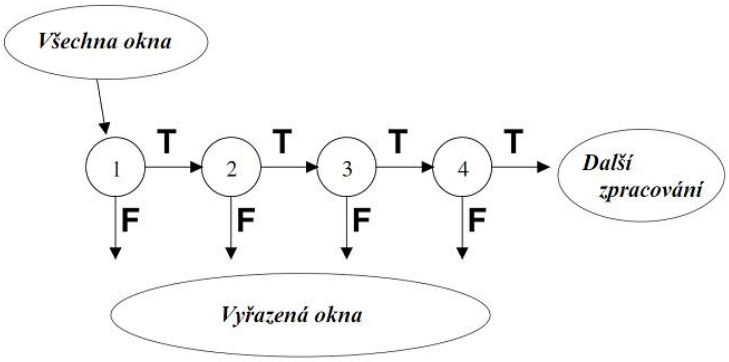
To znamená, že každá z hodnot integrálního obrazu je rovna součtu hodnot pixelů obdélníkové oblasti vstupního obrazu, která je vymezena onou hodnotou a levým horním rohem obrazu. Ilustrace viz obrázek 2.2. Sumu pixelů libovolné obdélníkové oblasti tak lze spočítat v konstantním čase, čehož je využito při výpočtu Haarových příznaků.



Obrázek 2.2: Integrální obraz. Součet pixelů v obdélníku D může být vypočítán jen se 4 přístupy do paměti. Hodnota integrálního obrazu na pozici 1 je suma pixelů obdélníku A. Hodnota na pozici 2 je suma obdélníků $A+B$, na pozici 3 suma $A+C$ a na pozici 4 suma $A+B+C+D$. Součet pixelů v obdélníku D se spočítá jako $4+1-(2+3)$. Převzato z [45].

Posledním ze jmenovaných vylepšení, které Viola a Jones v [45] navrhli, je tzv. kaskáda klasifikátorů. Stojí za ní zjištění, že klasifikátor při skenování obrazu projde obrovské množství výřezů s pozadím oproti relativně málo výřezům s hledaným objektem. Celková rychlost detekce proto záleží především na rychlosti detekce pozadí. Cílem je tedy vytvořit klasifikátor, který je schopen velmi rychle detekovat pozadí na základě několika málo příznaků (např. 2) a při tom nezamítne téměř žádný výřez s hledaným objektem (např. s pravděpodobností 0,999). Takový klasifikátor umožní rychle vyřadit velké procento výřezů s pozadím. O zbylých výřezech je možné prohlásit, že klasifikátor o nich nemá dostatečné

informace na to, aby rozhodl, zda je v nich hledaný objekt přítomen či nikoliv. Tyto výřezy jsou postoupeny dalšímu, složitějšímu klasifikátoru, který celý proces opakuje. Vzniká tak tzv. kaskáda klasifikátorů (viz obrázek 2.3), což je vlastně degradovaný rozhodovací strom. Každý z klasifikátorů má možnost skenovaný výřez buď vyřadit z dalšího zpracování (hledaný objekt není přítomen) nebo poslat dál (hledaný objekt může být přítomen). Teprve poslední klasifikátor v kaskádě může rozhodnout, zda hledaný objekt přítomen je.



Obrázek 2.3: Kaskáda klasifikátorů. Převzato z [32].

2.2 Robustní detekce objektů s prokládanou kategorizací a segmentací

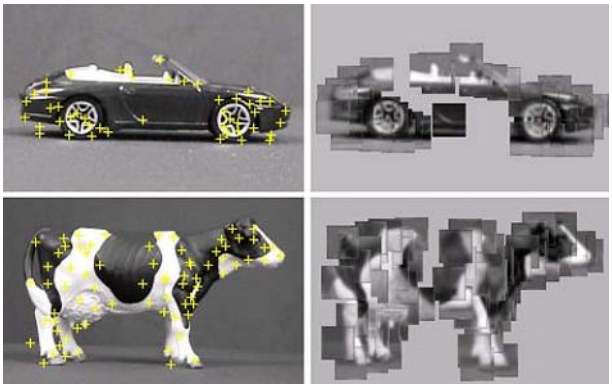
Segmentace a detekce objektů jsou většinou považovány za dvě samostatné oblasti počítačového vidění. K jejich separaci mimo jiné přispívá neexistence segmentačních metod, které by byly nezávislé na úloze, a také úspěch detekčních metod založených na vzhledu (viz např. klasifikátory na principu AdaBoost popsaném v sekci 2.1). Obě oblasti se z tohoto důvodu vyvíjely odděleně [34] i přes to, že jak v oblasti počítačového vidění [4], tak v oblasti lidského vnímání [43] bylo prokázáno, že rozpoznávání a segmentace objektů jsou silně propojené procesy. Toho je využito v systému pro detekci, kategorizaci a segmentaci objektů, který je popsán Leonardisem a kol. v [34]. Detekce a segmentace jsou zde chápány jako synergicky spolupracující procesy.

Aby se systém naučil poznávat objekty jednotlivých kategorií, je nutné nejprve pro každou kategorii vytvořit tzv. slovník, ve kterém jsou zaznamenány informace o výskytech rysů charakteristických pro objekty dané kategorie. Toho je dosaženo nalezením zájmových bodů [37] na objektech a extrakcí lokálních příznaků v jejich okolí. Následně jsou příznaky sloučeny do shluků kvůli jejich kompaktnější reprezentaci. Na základě slovníku je zkonstruován model objektu určující, na kterém místě objektu (či modelu) se může nacházet příslušná položka slovníku. Nejde o explicitní definici modelu všech myslitelných objektů dané kategorie, nýbrž o implicitní definici „povolených“ rysů, které se obvykle vyskytují pohromadě. Ruku v ruce s detekcí je prováděna pravděpodobnostní segmentace využívající informace o detekované kategorii objektu spolu s dalšími informacemi obsaženými v obraze. Výstupem segmentace je pravděpodobnost, s jakou ten který pixel přísluší objektu nebo pozadí a důvěryhodnost provedené segmentace. Tyto informace jsou zpětně využity pro vylepšení detekce objektu: Detekční mechanismus může být odstíněn od vlivů pozadí a na

základě hypotézy, kde v obraze se objekt nachází, mohou být vyřešeny nejasnosti způsobné překrývajícími se objekty.

Prvním krokem při detekci a identifikaci objektů využívající lokální příznaky je nalezení předem natrénovaných vzorů v nových, dosud neznámých obrazech, ve kterých se objekty vyskytují za rozdílných podmínek (jako např. úhel záběru, osvětlení). Pokud jsou hledány objekty přesně definovaného vzhledu, extrahované příznaky mohou být velmi specifické. Navíc, pokud se jedná o objekty stálého tvaru, je třeba k úspěšné detekci pouze malého počtu příznaků. Pokud je však třeba hledat všechny objekty patřící do určité kategorie, je situace složitější. Detekci neztěžují jen různé podmínky, za kterých mohl být objekt zachycen, ale také měnící se konfigurace lokálních příznaků mezi různými objekty téže třídy. Důsledkem výše uvedeného je značně redukovaný počet příznaků, které jsou přítomny na všech objektech dané třídy.

Jak už bylo zmíněno dříve, lokální příznaky se extrahují pouze z oblastí kolem zájmových bodů, protože je v jejich okolí koncentrováno více obrazové informace. Zájmové body jsou získány po aplikaci některého z detektorů zájmových bodů, jako je např. Harris [23], Harris-Laplace [38], Hessian-Laplace [38] nebo DoG[1] [36]. Extrakce lokálních příznaků je provedena obdobně aplikací některého z deskriptorů, jako např. Greyvalue Patches [1], SIFT[2] [36], SURF[3] [6] nebo Local Shape Context [7]. Ilustraci extrahovaných lokálních příznaků je možno vidět na obrázku 2.4. Je zřejmé, že extrahované příznaky hustě pokrývají zobrazený objekt, zatímco monotónní plochy, které obsahují minimum informací, pokryty nejsou. Detektory zájmových bodů ani deskriptory lokálních příznaků nejsou předmětem této práce. Pro více informací nechť čtenář použije odkazované literatury.



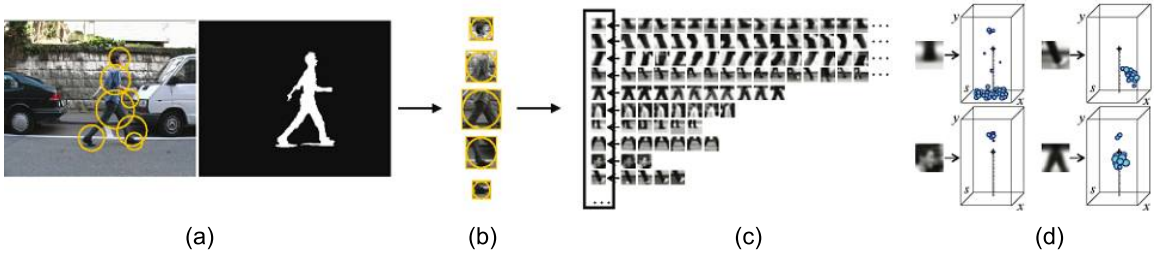
Obrázek 2.4: Extrahované lokální příznaky použité při generování slovníku: Vlevo zájmové body (získané Harrisovým detektorem), vpravo příznaky extrahované v okolí zájmových bodů (ilustrovány pomocí korespondujících čtvercových výřezů obrazu). Převzato z [34].

Výše popsaný princip extrakce lokálních příznaků je aplikován postupně na všechny trénovací obrazy shodné třídy. Následně jsou vizuálně podobné příznaky seskupeny do shluků reprezentujících typické znaky objektů. Aby byl snížen objem uchovávaných dat, je každý shluk příznaků reprezentován pouze svým středem. Nutnou podmínkou však je, aby středy shluků vždy smysluplně reprezentovaly celý shluk – není tedy účelem vytvořit co nejmenší počet shluků, ale zajistit, aby shluky byly kompaktní a obsahovaly pokud

[1] Zkratka z anglického Difference of Gaussian.
[2] Zkratka z anglického Scale Invariant Feature-Transform.
[3] Zkratka z anglického Speeded Up Robust Features.

možno stejný typ příznaků. To je nutné zohlednit při výběru shlukovací metody. Leonardis a kol. [34] doporučuje používat metody k-means [2], aglomerační shlukování [33] nebo jeho vlastní algoritmus typu average-link [21].

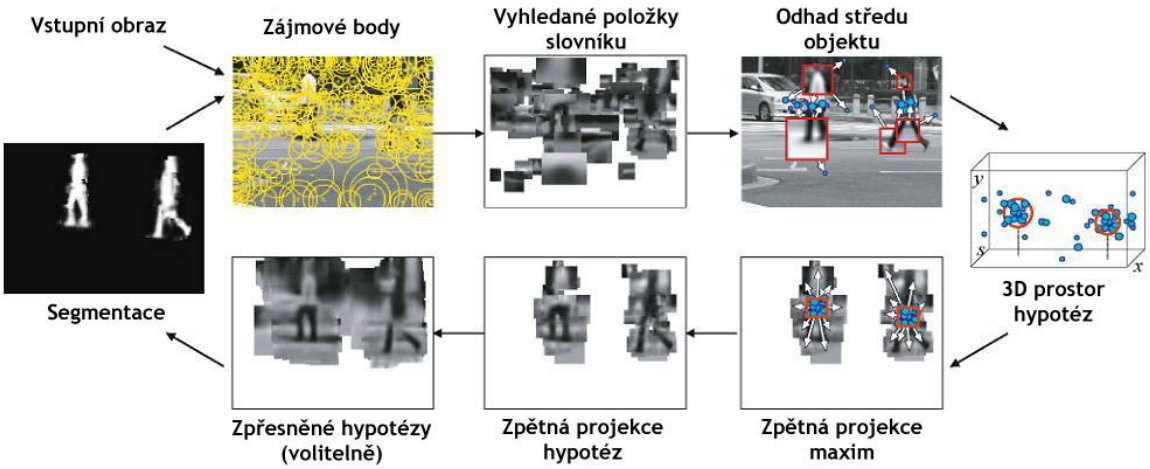
Jednotlivé třídy objektů jsou popsány pomocí tzv. „modelu implicitního tvaru“[4] $ISM(C) = (\mathcal{C}, P_{\mathcal{C}})$, který se skládá z třídně specifického slovníku lokálních příznaků $\mathcal{C}$ a prostorového pravděpodobnostního rozložení $P_{\mathcal{C}}$. Toto rozložení udává, na kterém místě v objektu se může daná položka slovníku vyskytovat. Celý koncept připomíná model hvězdy, kdy je poloha každé položky slovníku závislá pouze na středu objektu. Způsob tvorby slovníku $\mathcal{C}$ byl popsán výše. Rozložení $P_{\mathcal{C}}$ se generuje při druhém průchodu množinou trénovacích obrazů, kdy se pro každý obraz hledají odpovídající položky slovníku. Nehledají se pouze nejlépe odpovídající položky, ale všechny položky, jejichž podobnost je vyšší než určitý práh t. Pro každou položku slovníku se ukládají všechny pozice, na kterých byla nalezena relativně ke středu objektu; každá pozice je ovšem vážena pravděpodobností, s jakou se zde může daná položka slovníku nacházet. Celý trénovací proces je znázorněn na obrázku 2.5.



Obrázek 2.5: Proces trénování. (**a**) Vstupem jsou trénovací obrazy objektů spolu s jejich referenčními segmentacemi. Na objektech jsou detekovány zájmové body (žluté kružnice). (**b**) V okolí zájmových bodů jsou extrahovány lokální příznaky. (**c**) Extrahované příznaky jsou zařazeny do slovníku. (**d**) Pro každou položku slovníku je natrénováno prostorové pravděpodobnostní rozložení jeho možných výskytů relativně vzhledem ke středu objektu. Převzato z [34] a upraveno.

Proces detekce objektů začíná, stejně jako při trénování, aplikací detektoru zájmových bodů a následnou extrakcí lokálních příznaků v jejich okolí. Ve slovníku jsou poté vyhledány položky, jejichž míra podobnosti s extrahovanými příznaky je větší než stanovený práh t. Ze všech nalezených položek jsou pomocí obecné Houghovy transformace [5] vyfiltrovány pouze jejich smysluplné konfigurace. Každá takto vybraná položka přispívá k odhadu středu objektu, přičemž odhady jsou váženy pomocí prostorového rozložení $P_{\mathcal{C}}$. Správné hypotézy reprezentující středy objektů jsou vyhledány jako maxima v prostoru všech hypotéz, k čemuž se používá algoritmu Mean-Shift Mode Estimation [10]. Všechny příznaky, které přispěly k vítězné hypotéze jsou vybrány a jejich sjednocením vzniká vizualiace objektu, který systém detekoval. Výsledkem je tak reprezentace objektu spolu s jeho hraniční oblastí. Tato reprezentace může být eventuálně zpřesněna použitím více lokálních příznaků nebo může být použita jako podklad pro segmentaci objektu pixel po pixelu. Celý proces detekce je znázorněn na obrázku 2.6.

[4] Anglicky „Implicit Shape Model“.



Obrázek 2.6: Proces detekce. Ve vstupním obraze jsou detekovány zájmové body a v jejich okolí jsou extrahovány lokální příznaky, které jsou porovnány s položkami ve slovníku. Odpovídající položky přispívají k odhadu středu objektu. Vítězné hypotézy mohou být později volitelně zpřesněny použitím více lokálních příznaků. Na základě zpětně promítnutých hypotéz je spočtena segmentace objektu. Převzato z [34] a upraveno.

2.3 Template matching

Pojmem template matching se označuje skupina metod pro detekci objektů v obraze, které využívají model hledaného objektu popisující jeho tvar, barvu či texturu. V určitém místě obrazu se pak zkoumá, jak dobře toto místo odpovídá vytvořenému modelu. Mezi zástupce těchto metod patří např. jednoduché šablony (vzory), Active Shape Models nebo Active Appearance Models [27].

Základním principem metody jednoduchých šablon je postupné přikládání šablony, která obsahuje hledaný objekt, na vstupní obraz. Pro každou pozici šablony je překrytá oblast obrazu pixel po pixelu porovnána se šablonou, poté je šablona posunuta o určitý malý počet pixelů dál a porovnání se opakuje. Jako srovnávací funkce je obvykle použita normalizovaná vzájemná korelace mezi šablonou a odpovídající oblastí obrazu. Hledaný objekt se vyskytuje na pozicích, kde korelace dosahuje maxim [9].

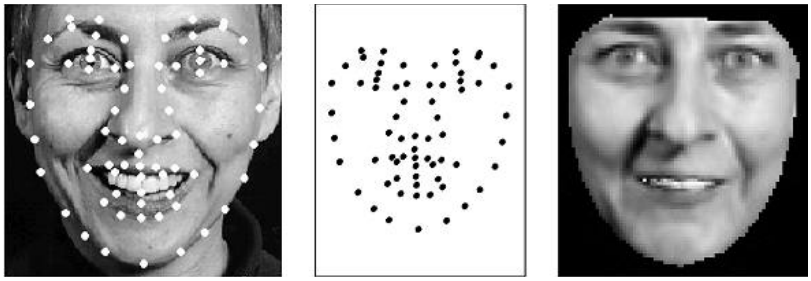
Hlavní myšlenkou Active Shape Models [12] je strojové učení tvarů určité třídy. Vstupem je trénovací množina obrazů, ve kterých jsou objekty opatřeny značkami, které označují jejich výrazné rysy a tvar. Trénovací množina je zpracována pomocí Procrustesovy analýzy[5] (jsou zjištěny dovolené variace tvarů v rámci trénovací množiny) a je vytvořen statistický model rozložení bodů v prostoru. Detekce objektů pak probíhá iterativním upravováním modelu tak, aby co nejlépe odpovídal tvaru v obraze. Výsledné parametry modelu charakterizují detekovaný objekt.

[5]Procrustesova analýza je druh statistické analýzy používané ke zjištění distribuce tvarů objektů v rámci nějaké množiny tvarů. Procrustes je kovář z řecké mytologie, který zval kolemjdoucí na přenocování do jeho železné postele. V noci je potom nutil, aby se natáhli přes celou postel; Když to nedokázali, pomáhal si při tom kovářským kladivem nebo jim usekl končetiny. [48]

Active Appearance Models

Active Appearance Models jsou podle [40] přímým rozšířením Active Shape Models. Kromě informace o tvaru objektu pracují také s texturní informací, resp. s intenzitami jednotlivých pixelů v objektu.

Pro trénování je tedy také požadována sada anotovaných obrazů, kde jsou v každém z nich označeny odpovídající si body, viz obrázek 2.7 (a). Na množiny bodů, každou reprezentovánu vektorem $\mathbf{x}$, je aplikována Procrustesova analýza (viz začátek sekce 2.3) a je vytvořen statistický model tvaru objektu. Následně je každý trénovací obraz zdeformován tak, aby jeho body odpovídaly bodům průměrného tvaru, čímž je vytvořen tzv. „beztvarý výřez“, viz např. obrázek 2.7 (c). Rastr výřezu je řádek po řádku načten do vektoru textury $\mathbf{g}$, která je poté normalizována podle vzorce $\mathbf{g} \leftarrow (\mathbf{g} - \mu_g \mathbf{1}) / \sigma_g$, kde $\mathbf{1}$ je vektor jedniček a μ_g a σ_g^2 jsou průměr a rozptyl prvků vektoru $\mathbf{g}$. Po normalizaci platí $\mathbf{g}^T\mathbf{1} = 0$ a $|\mathbf{g}| = 1$. Pomocí analýzy vlastních čísel a vektorů je vytvořen texturní model a na základě korelace textur a tvarů je vypočten výsledný model třídy objektů.



Obrázek 2.7: Z (**a**) anotovaného trénovacího obrazu je získána (**b**) množina bodů a (**c**) beztvarý výřez. Převzato z [11] a upraveno.

Výsledný model má parametr $\mathbf{c}$, který ovlivňuje jeho tvar a vnitřní texturu:

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{Q}_s \mathbf{c} \tag{2.6}$$

$$\mathbf{g} = \bar{\mathbf{g}} + \mathbf{Q}_g \mathbf{c}, \tag{2.7}$$

kde $\bar{\mathbf{x}}$ je průměrný tvar, $\bar{\mathbf{g}}$ průměrná textura v průměrném tvaru a $\mathbf{Q}_s$, $\mathbf{Q}_g$ matice popisující variace příslušných modelů trénovací množiny.

Tvar $\mathbf{X}$ může být vytvořen vhodnou transformací S_t z bodů $\mathbf{x}$: $\mathbf{X} = S_t(\mathbf{x})$. S_t bude typicky podobnost složená ze změny měřítka s, rotace θ a posunutí (t_x, t_y). Kvůli linearitě je změna měřítka a rotace sloučena do dvojice (s_x, s_y), kde $s_x = (s\cos\theta - 1), s_y = s\sin\theta$. Pro identitu je pak vektor parametrů příslušného tvaru $t = (s_x, s_y, t_x, t_y)^T$ nulový. Platí také skládání transformací: $S_{t+\delta t}(\mathbf{x}) \approx S_t\left(S_{\delta t}(\mathbf{x})\right)$.

Textura uvnitř tvaru $\mathbf{g}_{im}$ je generována z trénovacího obrazu změnou měřítka a posunutím: $\mathbf{g}_{im} = T_u(\mathbf{g}) = (u_1 + 1)\mathbf{g}_{im} + u_2\mathbf{1}$, kde $\mathbf{u}$ je vektor parametrů transformace definovaný pro identitu jako $\mathbf{u} = \mathbf{0}$, jinak $T_{u+\delta u}(\mathbf{g}) \approx T_u\left(T_{\delta u}(\mathbf{g})\right)$.

Celá rekonstrukce objektu je provedena nagenerováním textury do rastru uvnitř průměrného tvaru s následnou deformací tohoto rastru tak, aby se body $\mathbf{x}$ modelového tvaru shodovaly s body $\mathbf{X}$ tvaru v obrazu.

Pro nalezení detekovaného objektu v obraze je zapotřebí hrubého odhadu jeho pozice, orientace a měřítka, aby mohl být model umístěn do vhodné výchozí pozice a později konvergovat ke skutečnému objektu. Parametry modelu $\mathbf{c}$ a parametry transformace modelového tvaru $\mathbf{t}$ definují pozici bodů objektu v obraze $\mathbf{X}$. Tyto body tedy určují tvar obrazového výřezu, který má být reprezentován modelem. Během procesu detekce jsou body obrazového výřezu $\mathbf{g}_{im}$ vzorkovány a promítány do textury modelu: $\mathbf{g}_s = T_u^{-1}(\mathbf{g}_{im})$. Aktuální textura modelu je vyjádřena jako $\mathbf{g}_m = \bar{\mathbf{g}} + \mathbf{Q}_g\mathbf{c}$. Aktuální rozdíl mezi normalizovanou texturou modelu a obrazu je tudíž

$$\mathbf{r}(\mathbf{p}) = \mathbf{g}_s - \mathbf{g}_m, \tag{2.8}$$

kde $\mathbf{p}$ jsou parametry modelu; $\mathbf{p}^T = (\mathbf{c}^T|\mathbf{t}^T|\mathbf{u}^T)$.

Jednoduchá metrika výše uvedeného rozdílu je suma čtverců prvků vektoru $\mathbf{r}$, $E(\mathbf{p}) = \mathbf{r}^T\mathbf{r}$. Aplikací Taylorova rozvoje prvního řádu na rovnici 2.8 dostaneme

$$\mathbf{r}(\mathbf{p} + \delta\mathbf{p}) = \mathbf{r}(\mathbf{p}) + \frac{\partial \mathbf{r}}{\partial \mathbf{p}}\delta\mathbf{p}, \tag{2.9}$$

kde $\frac{\partial \mathbf{r}}{\partial \mathbf{p}}$ je matice, jejíž prvek s indexem i, j je roven $\frac{dr_i}{dp_j}$.

Nechť aktuální rozdíl normalizovaných textur je $\mathbf{r}$. Cílem je zvolit $\delta\mathbf{p}$ tak, aby výraz $|\mathbf{r}(\mathbf{p} + \delta\mathbf{p})|^2$ byl minimální. Je-li rovnice 2.9 položena rovna nule, střední kvadratická hodnota $\delta\mathbf{p}$ je následující:

$$\delta\mathbf{p} = -\mathbf{R}\mathbf{r}(\mathbf{p}), \quad \text{kde} \quad \mathbf{R} = \left(\frac{\partial \mathbf{r}}{\partial \mathbf{p}}^T \frac{\partial \mathbf{r}}{\partial \mathbf{p}}\right)^{-1} \frac{\partial \mathbf{r}}{\partial \mathbf{p}}^T. \tag{2.10}$$

S využitím rovnice 2.10 může být nalezena vhodná korekce parametrů modelu na základě změřeného rozdílu $\mathbf{r}$. To umožňuje konstrukci iterativního algoritmu pro řešení výše popsaného optimalizačního problému. Pro aktuální odhad parametrů modelu $\mathbf{c}$, parametrů tvaru $\mathbf{t}$, parametrů transformace textury $\mathbf{u}$ a obrazového výřezu $\mathbf{g}_{im}$ je jeden krok iterativního algoritmu následující:

Algoritmus 2.3 Jeden krok iterativního přizpůsobení modelu.

1: Promítni obrazovou texturu do modelového rastru použitím $\mathbf{g}_s = T_u^{-1}(\mathbf{g}_{im})$.
2: Vypočti chybový vektor $\mathbf{r} = \mathbf{g}_s - \mathbf{g}_m$ a aktuální chybu $E = |\mathbf{r}|^2$.
3: Vypočti odhadovaná posunutí $\delta\mathbf{p} = -\mathbf{R}\mathbf{r}(\mathbf{p})$.
4: Aktualizuj parametry modelu $\mathbf{p} \leftarrow \mathbf{p} + k\delta\mathbf{p}$, kde zpočátku $k = 1$.
5: Vypočti nové body $\mathbf{X}'$ a modelovou texturu $\mathbf{g}'_m$.
6: Vzorkuj obraz v nových bodech pro výpočet $\mathbf{g}'_{im}$.
7: Vypočti nový chybový vektor $\mathbf{r}' = T_{u'}^{-1}(\mathbf{g}'_{im}) - \mathbf{g}'_m$.
8: Pokud $|\mathbf{r}'|^2 < E$, pak přijmi nový odhad; jinak zkus počítat s $k = 0.5, k = 0.25$ apod.

Algoritmus 2.3 je opakován dokud se zmenšuje chyba $|\mathbf{r}|^2$ a dokud model konverguje.

2.4 Další metody

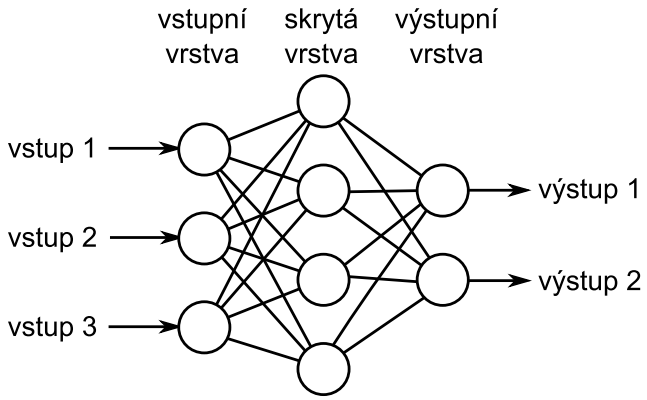
Mimo výše popsané metody existuje značný počet dalších, méně používaných přístupů, jejichž stručný přehled následuje.

Statistické metody

Statistické metody využívají statistického modelování vzorů a tříd objektů, kde každá třída je obvykle popsána vektorem číselných hodnot reprezentující vybrané vlastnosti (tzv. vektor příznaků). Příznaky mohou být libovolné – nemusí se jednat přímo o hodnoty pixelů obrazu, ale např. o odezvy různých filtrů, histogramy, apod. Detekce objektu pak spočívá v určení jeho vektoru příznaků a následného zařazení do nejpodobnější třídy. Jedná se tedy o problém určení nejbližšího shluku v prostoru příznaků, který je n-rozměrný, kde n je počet příznaků ve vektoru. V případě, že rozdělení do tříd je dáno učitelem, hovoříme o klasifikaci, v opačném případě o shlukování [8].

Neuronové sítě

Detekční metody používající neuronové sítě [24] jsou v podstatě také statistické klasifikační (shlukovací) metody, ale k dělení prostoru příznaku do shluků využívají tzv. hyperplochy[6]. Každou hyperplochu reprezentuje jeden umělý neuron[7], což je základní stavební jednotka neuronové sítě. Učení sítě spočívá v hledání optimálních vstupních vah jednotlivých neuronů. Váhy pak určují parametry dělicích hyperploch. Topologie neuronových sítí se mohou značně lišit. Na obrázku 2.8 je možno vidět ilustraci jednoduché neuronové sítě.



Obrázek 2.8: Jednoduchá neuronová síť s jednou skrytou vrstvou. Hrany označují propojení neuronů, ke každé hraně se váže váha, která je výsledkem procesu učení.

Metody pracující shora dolů

Tento typ metod nejprve v obraze vyhledává kandidátní pozice objektů pomocí rychlých testů na vysoké úrovni. Následně jsou kandidátní pozice potvrzeny nebo vyřazeny podle testu detailů. Například při detekci obličeje [28] se napřed naleznou oblasti s barvou kůže (potenciální tvář), ve kterých se poté hledají oči, ústa, nos, okraj obličeje a jejich vzájemné pozice.

[6] Hyperplocha n-rozměrného prostoru je jeho podprostor dimenze $n-1$, který jej dělí na dvě poloviny (jedná se o zobecnění plochy). Např. přímka je hyperplocha 2-rozměrného prostoru, rovina je hyperplocha 3-rozměrného prostoru apod. Obdobně u vícerozměrných prostorů, avšak bez geometrické interpretace.

[7] Podobně jako buněčný neuron má několik vážených vstupů, jejichž hodnoty kombinuje do jediné výstupní hodnoty.

Kapitola 3

Kalibrace scény

Tato kapitola pojednává o problematice kalibrace scény. Neklade si však za cíl být vyčerpávajícím zdrojem informací o této oblasti. Z důvodu rozsahu se omezuje pouze na témata relevantní vzhledem k zaměření práce.

Geometrii scény je možné odhadnout z obrazu vícero způsoby. Nejpouzívanějšími metodami jsou následující:

- Využití fotogrammetrie – odhad třetího rozměru podle posunutí identických význačných bodů ze 2 obrazů stejné scény, ovšem pořízených z různých míst vzájemně mírně posunutých [35].
- Odhad geometrie scény na základě odhadu typu povrchů [25]. Metoda je založena na úvaze, že určité typy povrchů mají obvykle předvídatelnou orientaci v prostoru vzhledem ke kameře. Např. asfalt bude pravděpodobně vodorovný, větvoví svislé apod.
- Odvození perspektivy z odhadu úběžnic a úběžníků[1] navzájem kolmých směrů [35]. Tato metoda vyžaduje přítomnost rovných kolmých hran v obraze, např. hran budov.
- Odhad třetího rozměru podle rozdílů dvou následujících obrazů ve videosekvenci [35]. Zde se také využívá fotogrammetrie.
- Výpočet pravděpodobné geometrie scény na základě ručně označených objektů známé velikosti v obraze [3].
- Odhad scény pomocí „vlastních“ obrazů[2] [26] – ze vstupního obrazu jsou vypočteny pomocné obrazy, každý z nich zobrazuje jinou informaci, např. hrany, předpokládanou hloubku prostoru, překryv objektů apod. Tyto obrazy tvoří společně kontextovou informaci, na základě které jsou klasifikovány povrchy, detekovány objekty, odhadovány jejich překryvy a tak podobně.

3.1 Stereosnímky

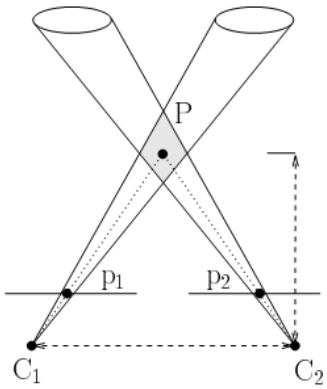
Výpočet parametrů scény z několika různých obrazů, které ji zachycují, je dobře prozkoumaná oblast počítačového vidění a metody z této oblasti (dále jen „stereometody“) jsou hojně využívány i v jiných odvětvích informačních technologií [30]. Stereometody vyžadují

[1] Pro definici úběžníků a úběžnic viz sekci 3.2 na straně 17.
[2] V anglickém originále tzv. „intrinsic images“.

znalost vnitřních parametrů kamery a alespoň jejich relativní pozici při záběru jednotlivých snímků. Výstupem stereometod je obvykle hloubková mapa, která pro některé nebo všechny pixely určuje, jak daleko leží objekt promítnutý na příslušný pixel. Podle [42] lze stereometody rozdělit na základě úrovně detailu hloubkové mapy do dvou základních tříd: Na metody produkující řídké hloubkové mapy a na metody produkující husté mapy. Množství aplikací vyžaduje husté mapy, proto je větší pozornost soustředěna na druhou třídu.

Metody pro tvorbu hustých hloubkových map lze rozdělit na lokální a globální. Lokální stereometody určují hloubku každého pixelu nezávisle na ostatních, typicky s využitím analýzy intenzit pixelů v určitém obdélníkovém okně. Pomocí různých statistických přístupů porovnávají korespondující okna mezi stereosnímky a obvykle jsou založeny na výpočtu korelace. Globální metody se snaží najít hloubkovou mapu pomocí minimalizace účelové funkce (nazývané též energie). Zpravidla využívají některou z iterativních optimalizačních technik, jako např. simulované žíhání [31].

Jak už bylo naznačeno, cílem stereometod je rekonstrukce trojrozměrného prostoru z jeho dvojrozměrné reprezentace. Problém rekonstrukce třetí dimenze z více obrazů je v zásadě korespondenční problém: Je-li v jednom z obrazů určen bod, je nutno korespondující body v ostatních obrazech. Jakmile jsou korespondence nalezeny, určení hloubky je již pouze geometrickým problémem. Na obrázku 3.1 je možno vidět, jak je určena hloubka bodu P na základě dvou obrazů pořízených kamerami C_1 a C_2. Průměty bodu P jsou označeny p_1, resp. p_2. Pozice průmětu p_1 je neurčitá, bod P tedy může ležet kdekoliv v levém kuželu. Podobně pro průmět p_2. Pokud jsou p_1 a p_2 korespondující body, leží bod P v průsečíku obou kuželů (šedá oblast). Je zřejmé, že pro danou hloubku je oddálením kamer C_1, C_2 docílena vyšší přesnost odhadu (osy kuželů budou svírat větší úhel, tedy šedá oblast označující možný výskyt bodu P se zmenší). Tato skutečnost vede ke konfliktu požadavků kladených na pozici kamer: Jejich menší vzdálenost přispívá ke snadnějšímu hledání korespondencí, ale vede k méně přesným výsledkům; větší vzdálenost kamer naopak implikuje přesnější výsledky za cenu složitého hledání korespondujících bodů.



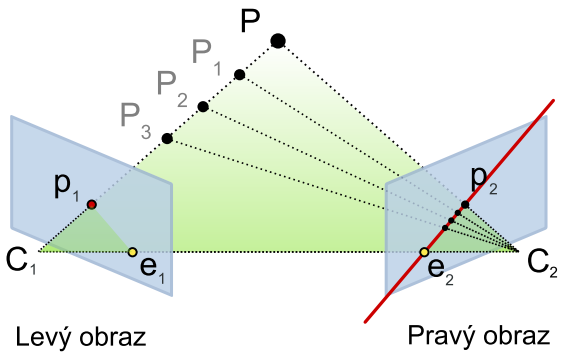
Obrázek 3.1: Určení hloubky bodu P pomocí jeho průmětu p_1 zachyceného kamerou C_1 a průmětu p_2 zachyceného kamerou C_2. Převzato z [42] a upraveno.

Hledání korespondencí mezi obrazy je obvykle prováděno ve dvou krocích: Nejdříve jsou v jednom obraze nalezeny význačné body (pomocí detektorů zájmových bodů, zmíněných v sekci 2.2 na straně 8). Poté jsou korespondující body hledány v dalších obrazech. S využitím epipolárního omezení není nutné hledat korespondence v celém obraze, ale pouze na tzv.

epipolárách (viz dále), což snižuje složitost metody.

Nechť je dána stereosestava dvou kamer, viz obrázek 3.2. Optické středy kamer jsou označeny C_1 a C_2. Bod P v trojrozměrném prostoru je promítán do průmětů p_1, p_2 v obou obrazových rovinách. Tyto průměty jsou korespondující pár bodů v obrazech. Leží-li bod p_1 v levé obrazové rovině, jeho korespondující bod v pravé obrazové rovině musí ležet na přímce zvané epipolára bodu p_1 (vyznačena červeně). Jelikož bod p_1 může být obraz jakéholiv bodu na přímce C_1P, je epipolára bodu p_1 projekcí přímky C_1P do optického středu druhé kamery C_2. Všechny epipoláry dané obrazové roviny se protínají v jednom bodě e_1 resp. e_2 nazývaným epipól, který je obrazem optického středu druhé kamery.

Pokud optický střed C_1 leží v obrazové rovině pravé kamery, epipól e_2 leží v nekonečnu a epipoláry v pravé obrazové rovině jsou rovnoběžné. Ve zvláštním případě mohou oba epipóly ležet v nekonečnu – tato situace nastává, pokud spojnice kamer C_1C_2 je kolmá k optickým osám obou kamer a ty jsou navzájem rovnoběžné. Epipoláry jsou pak rovnoběžné v obou obrazech. Každá dvojice obrazů může být transformována tak, aby epipoláry byly rovnoběžné a měly horizontální směr v obou obrazech, což je výhodné pro hledání korespondencí. Zmíněná transformace se nazývá rektifikace stereosnímků [20] a z důvodu rozsahu práce zde není popsána.



Obrázek 3.2: Epipolární geometrie. Bod v trojrozměrném prostoru P má průměty p_1, p_2 v obrazových rovinách kamer s optickými středy C_1, C_2. Epipolára bodu p_1 je vyznačena červenou čarou. e_1, e_2 jsou epipóly. Autor Arne Nordmann, převzato z [47] a upraveno.

V případě rovnoběžných optických os obou kamer je výpočet souřadnic bodů v trojrozměrném prostoru relativně jednoduchý. Pro každou dvojici obrazových bodů $P_L(x_L, y_L)$ a $P_R(x_R, y_R)$ je možné spočíst jejich disparitu $d = x_L - x_R$. Z podobnosti trojúhelníků vyplývá, že prostorové souřadnice bodu $P(x, y, z)$ mohou být vypočteny následovně:

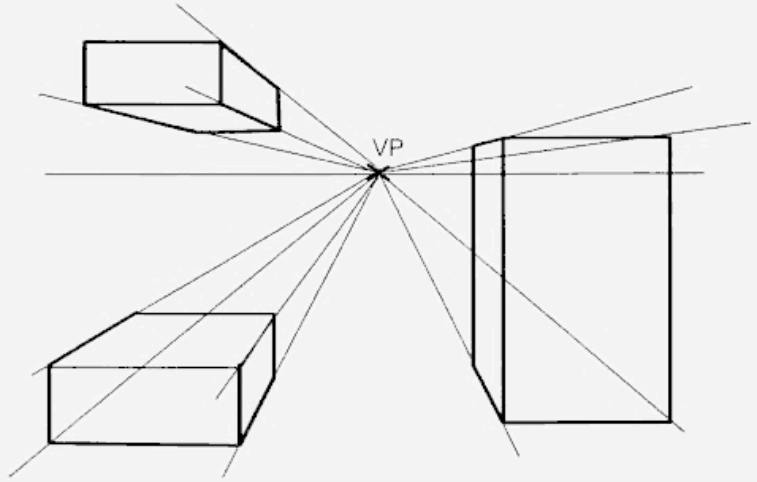
$$x = \frac{b}{d}x_L, \quad y = \frac{b}{d}y_L, \quad z = \frac{b}{d}f, \tag{3.1}$$

kde b je vzdálenost optických středů kamer a f je ohnisková vzdálenost kamery [16].

Případ, kdy optické osy kamer nejsou rovnoběžné, vyžaduje obecnější přístup, protože v mnocha případech nemusí existovat řešení uzavřeného tvaru. Z optických středů kamer jsou přes obrazové body vrhány paprsky zpět do scény. Jako bod v prostoru je určen takový bod, jehož suma vzdáleností od všech zpětně vržených paprsků je minimální.

3.2 Afinní geometrie

Pokud není k dispozici více snímků téže scény, je nutné extrahovat informaci o třetím rozměru pouze z jediného obrazu. Jedním z přístupů spadajících do této kategorie je kalibrace scény na základě určení úběžníků a úběžnic. Úběžník je myšlený bod, do kterého se při perspektivní projekci sbíhají všechny rovnoběžné přímky, které nejsou kolmé k rovině obrazu [17]. Úběžnice referenční roviny je přímka vzniklá průnikem obrazové roviny a roviny, která prochází optickým středem kamery a je rovnoběžná s referenční rovinou [15]. Ilustrace viz obrázky 3.3 a 3.4.



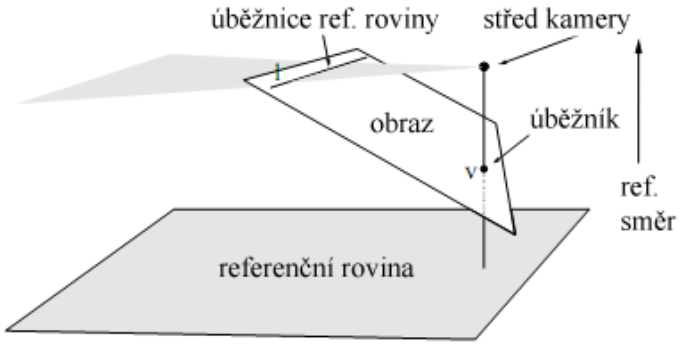
Obrázek 3.3: Ilustrace úběžníku (označen jako VP) a úběžnice plochy, na které leží spodní dva hranoly (vodorovná přímka procházající úběžníkem). Převzato z [29].

Geometrická primitiva popsaná v předchozím odstavci mají některé zajímavé vlastnosti patrné z obrázku 3.4: Úběžnice rozděluje body v prostoru tak, že všechny body promítnuté na uběžnici jsou ve stejné vzdálenosti od referenční roviny jako optický střed kamery; pokud průmět bodu leží „nad“ úběžnicí, je bod dále od roviny, pokud leží „pod“ úběžnicí, je blíže rovině. Dva body ležící v různých rovinách, které jsou obě rovnoběžné s referenční rovinou, korespondují, pokud je jejich spojnice rovnoběžná s referenčním směrem (nejčastěji svislice). To také znamená, že obraz jejich spojnice prochází úběžníkem referenčního směru. Např. pokud je v obraze zobrazena stojící osoba a referenční směr je svislý, vrchol hlavy a bod mezi patami osoby jsou korespondující body.

Pro určení úběžníků a úběžnic je v obraze zřejmě nutné nalézt přímky, které jsou v reálu rovnoběžné, případně leží v jedné rovině. Tento předpoklad splňují zejména fotografie městských oblastí, ve kterých se vyskytuje množství staveb a pravidelných tvarů. Znalost vnitřních ani vnějších parametrů kamery není při použití popisovaného přístupu třeba [15].

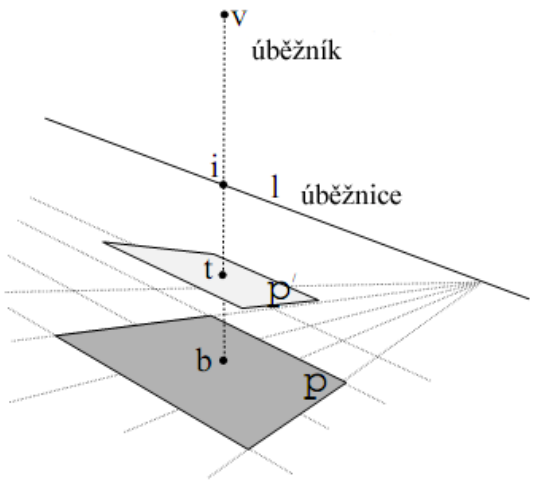
Pokud je třeba změřit vzdálenost dvou rovnoběžných rovin, je nutné znát dva korespondující body v těchto rovinách. Např. v obrázku 3.5 jsou takové korespondující body označeny t a b. Čtveřice bodů v, i, t a b definuje tzv. dvojpoměr[3], který udává poměr

[3]Pro 4 kolinární body A, B, C, D je dvojpoměr (nebo též „anharmonický poměr“) definován jako $(A, B; C, D) = \frac{|AC| \cdot |BD|}{|AD| \cdot |BC|}$, kde $|AB|$ označuje vzdálenost bodů A a B [13].



Obrázek 3.4: Úběžnice l referenční roviny je průsečnicí obrazové roviny s rovinou, která prochází optickým středem kamery a je rovnoběžná s referenční rovinou. Úběžník v je je průsečíkem obrazové roviny s přímkou procházející středem kamery, která je rovnoběžná s referenčním směrem. Převzato z [14] a upraveno.

mezi vzdáleností rovin obsahujících body t a b a vzdáleností optického středu kamery od roviny p (nebo od roviny p' v závislosti na uspořádání bodů definujících dvojpoměr). Absolutní vzdálenost rovin je možné z dvojpoměru určit, jakmile je známa vzdálenost kamery od roviny p. Většinou je ale praktičtější určit vzdálenost rovin pomocí odhadu obrazové velikosti nějakého dalšího objektu, jehož reálná velikost je známa. Je také možné určit vzdálenost kterýchkoliv dvou rovin na základě znalosti vzdálenosti jiné dvojice rovin. To je způsobeno faktem, že úběžnice je obrazem bodů v nekončenu všech těchto rovnoběžných rovin.

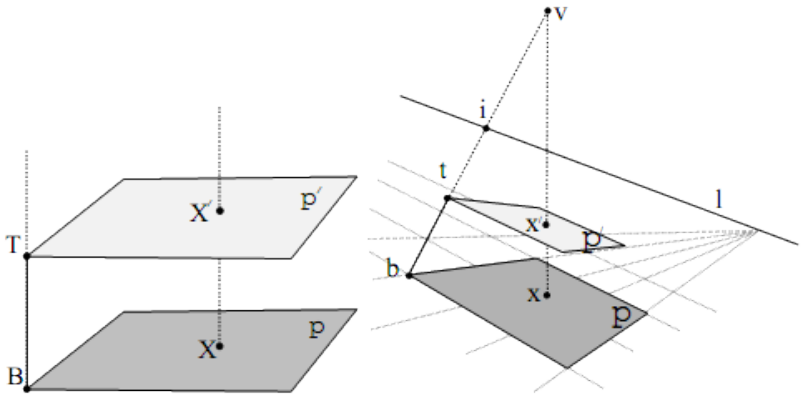


Obrázek 3.5: Bod b v rovině p a bod t v rovině p' korespondují, protože leží na jedné přímce s úběžníkem v. Tyto tři body v, t, b spolu s bodem i, což je průsečík jejich spojnice s úběžnicí, definují dvojpoměr. Převzato z [14] a upraveno.

Pokud je referenční rovina p afinně zkalibrována (tzn. její úběžnice je známa), je možné z obrazu spočíst poměrnou délku rovnoběžných úseček, které leží v této rovině. Právě tak

je možné určit poměr obsahů plošných útvarů této roviny. Jelikož je úběžnice referenční roviny sdílena také všemi ostatními rovinami rovnoběžnými s rovinou referenční, afinní měření mohou být provedena ve kterékoliv z těchto rovin. I přes to ale nelze přímo porovnat obsahy plošných útvarů ležících v různých rovinách. Pokud je však jeden z útvarů ve scéně promítnut paralelní projekcí z jedné roviny na druhou, stává se v obraze porovnatelným s druhým z útvarů. To proto, že paralelní projekce mezi rovnoběžnými rovinami nemění afinní vlastnosti a oba útvary nyní leží ve stejné rovině.

Zobrazení v prostoru scény mezi rovnoběžnými rovinami indukuje další zobrazení v obrazovém prostoru mezi obrazy bodů ležících v těchto rovinách. Indukované zobrazení je nazýváno „planární homologie“ [39], což je rovinná projektivní transformace s pěti stupni volnosti, která se vyznačuje množinou pevných bodů v přímce tvořících tzv. „osu“ a dalším výrazným pevným bodem mimo osu, zvaným „vrchol“. Planární homologie přirozeně vznikají v obraze, který perspektivně zobrazuje dvě rovnoběžné roviny, jak je ukázáno na obrázku 3.6. V modelovém případě je úběžnice osou homologie a úběžník jejím vrcholem a fixují čtyři z pěti stupňů volnosti homologie. Zbývající stupeň volnosti je jednoznačně určen kterýmkoliv párem obrazových bodů, které korespondují mezi rovinami (body b a t v obrázku 3.6).



Obrázek 3.6: (vlevo) Bod X v rovině p je paralelní projekcí zobrazen na bod X' v rovině p'. (vpravo) Zobrazení mezi rovinami v obrazovém prostoru je homologie s vrcholem v a osou l. Korespondence $b \to t$ určuje zbývající pátý stupeň volnosti homologie. Převzato z [14] a upraveno.

Důsledkem předchozího je možnost porovnat měření provedená v různých rovinách při použití homologie v referenčním směru mezi rovinami. Zejména je možné vyčíslit poměr velikostí rovnoběžných úseček v různých rovinách nebo poměr obsahů plošných útvarů v různých rovinách. Je také možné postupovat tak, že jsou všechny body ze všech rovin pomocí homologie zobrazeny do referenční roviny a poté jsou všechna měření prováděna v ní.

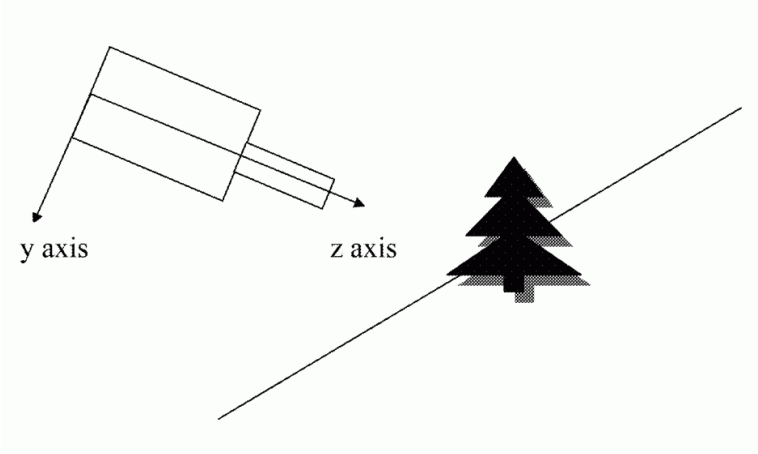
Jak již bylo popsáno dříve v této sekci, vzdálenost dvou rovnoběžných rovin je možné spočíst z dvojpoměru a vzdálenosti optického středu kamery od referenční roviny. Opačně je také možné spočítat vzdálenost kamery od určité roviny za předpokladu znalosti nějaké referenční vzdálenosti ve scéně. Z obrázku 3.4 také vyplývá, že pozice kamery vzhledem k referenční rovině je dána zpětnou projekcí úběžníku na tuto rovinu. Zmíněná zpětná

projekce je homografie mapující obrazovou rovinu do referenční roviny a naopak. Ačkoliv je volba souřadného systému scény libovolná, vždy unikátně definuje homografii a tím i pozici kamery ve scéně [15].

3.3 Ručně označené objekty známé velikosti

Protože reálný svět je trojrozměrný a jeho podoba zachycená na snímku pouze dvourozměrná, neexistuje jednoznačné zobrazení z prostoru snímku do reálného prostoru [3]. Tato nejednoznačnost se zpravidla odstraňuje zavedením předpokladu, že objekty scény spočívají na zemi nebo se nacházejí v nějaké známé výšce nad zemí. Země nebo též povrch se obvykle modeluje jako rovina, ve výjimečných případech jako soustava několika rovinných segmentů. Určení geometrie scény pak spočívá v odhadu parametrů roviny povrchu (jak je rovina orientována) a odhadu inklinace kamery (odklon od vodorovné roviny, čili pod jakým úhlem kamera snímá rovinu povrchu). Předpokládá se, že jsou známy parametry kamery jako její ohnisková vzdálenost a rozměry snímacího čipu.

Pro další výklad je nutné zavést souřadné systémy popisující trojrozměrný prostor scény i dvojrozměrný prostor obrazu. Souřadný systém scény má počátek přímo v kameře, osa x směřuje rovnoběžně s řádky obrazu zleva doprava, osa y kolmo na řádky obrazu shora dolů a osa z je totožná s osou optické soustavy směrem od kamery do scény. Situaci ilustruje obrázek 3.7. Pokud je vypuštěna osa z, vznikne dvojrozměrný systém obrazového prostoru (jeho počátek leží uprostřed obrazu).



Obrázek 3.7: Souřadný systém scény. Osa x vystupuje kolmo z roviny obrázku ke čtenáři. Převzato z [3].

Jednotkový vektor směřující vertikálně nahoru proti gravitační síle je označen jako u a normálový vektor roviny povrchu je označen jako a. Vektory u a a jsou hledanými parametry scény. Pro jednoduchost lze předpokládat nulový náklon kamery, kdy jsou řádky obrazu vodorovné, což vede nulové složce vektoru $u_x = 0$. Za této podmínky je počet neznámých roven čtyřem – tři složky vektoru a a inklinace α, která je definována v rovnici 3.2.

$$u = [0 \quad -\cos\alpha \quad -\sin\alpha]^T. \tag{3.2}$$

Zobecnění pro situaci s nenulovým náklonem kamery pouze způsobí přidání jedné neznámé [3].

Předpokládá se, že ohnisková vzdálenost kamery f a šířka W a výška H snímacího čipu jsou známé. Pro obraz o šířce $W_{px} \times H_{px}$ pixelů a bod p v prostoru scény na souřadnicích $[p_x\ p_y\ p_z]^T$ platí, že se zobrazí na obrazové souřadnice

$$X = A \cdot \frac{p_x}{p_z} \tag{3.3}$$

$$Y = B \cdot \frac{p_y}{p_z}, \tag{3.4}$$

kde

$$A = f \cdot \frac{W_{px}}{W} \tag{3.5}$$

$$B = f \cdot \frac{H_{px}}{H}. \tag{3.6}$$

Pro odhad parametrů scény je nutné v obraze změřit výšku několika objektů známé velikosti. Pokud jsou objekty dostatečně malé, každé z měření je možné interpretovat jako odhad parciální derivace Y-ové souřadnice bodu ve výšce h nad povrchem vzhledem k výšce h. Derivace $\partial Y/\partial h$ je odhadována v nulové výšce $h = 0$. Každé měření vede na jednu skalární rovnici, takže počet měření nutných k odhadu parametrů scény je 4 (kromě singulárních případů).

N měření $\partial Y/\partial h$ je možné reprezentovat N-rozměrným vektorem

$$\xi = M(\alpha)a, \tag{3.7}$$

kde $M(\alpha)$ je matice o velikosti $N \times 3$. Její k-tý řádek se spočte jako

$$(Y_k \sin\alpha - B\cos\alpha)\,[X_k/A \quad Y_k/B \quad 1], \tag{3.8}$$

kde (X_k, Y_k) jsou obrazové souřadnice spodního bodu změřeného objektu. Rovnice 3.7 a 3.8 plně definují výše popsaný model měření [3].

Odhad roviny povrchu a se v závislosti na inklinaci kamery α spočte jako

$$a = \left(M(\alpha)^T M(\alpha)\right)^{-1} M(\alpha)^T \xi \tag{3.9}$$

a jeho střední kvadratická chyba jako

$$J(\alpha) = \xi^T \left(I\ - M(\alpha)\left(M(\alpha)^T M(\alpha)\right)^{-1} M(\alpha)^T\right) \xi. \tag{3.10}$$

Odhad spočívá v nalezení inklinace α s minimální chybou $J(\alpha)$ podle rovnice 3.10 (například prohledáváním tabulkových hodnot) a následným vypočtením roviny povrchu a podle rovnice 3.9.

V praktických případech mohou nastat situace, kdy je možné využít apriorní informaci k minimalizaci chyby při měření nebo ke zmenšení počtu měření. Takové případy jsou například:

- Rovina povrchu je vertikální: Vektor a je kolmý k vektoru u. Nutno provést 3 měření.
- Rovina povrchu je rovnoběžná s osou x: $a_x = 0$. Nutno provést 3 měření.
- Rovina povrchu je horizontální a kamera je v neznámé výšce H nad ní: $a = -u/H$, $H = ?$. Nutno provést 2 měření.
- Rovina povrchu je horizontální a kamera je ve známé výšce H nad ní: $a = -u/H$. Nutno provést 1 měření.

V [3] je ukázáno, že velký vliv na přesnost odhadu parametrů scény má volba měřených objektů, resp. jejich lokace ve scéně. Obecně se doporučuje měřit objekty ležící blízko kamery (většinou tedy ve spodní části obrazu), protože přírůstek výšky objektů v závislosti na přírůstku y-ové souřadnice v obraze je u takových objektů malý, tím pádem i chyba měření je malá. Jako výjimečný případ je ukázána situace, kdy rovina povrchu je rovnoběžná s osou x a všechny měřené body se vyskytují na dvou řádcích obrazu. V tomto případě není odhad parametrů vůbec možný.

Kapitola 4

Analýza a návrh

4.1 Informační spleť

Moderní komunikační technologie umožnily generování obsahu v masovém měřítku a je stále složitější, ne-li nemožné, se v takovém množství informací zorientovat a vybrat ty podstatné. Jedná se o podobný fenomén jako byla před lety expanze webu, která si vyžádala vznik webových vyhledávačů. Rapidní rozvoj webových a dalších internetových služeb však stále ještě pokračuje a stejně tak bude pokračovat rozvoj komunikačních technologií. Bude tedy narůstat i objem generovaných dat a jejich pouhé indexování nemusí být dostatečným nástrojem pro jejich kvalitní analýzu. Vývoj naznačuje, že bude třeba vyvinout a nasadit inteligentní a komplexní techniky pro analýzu generovaného obsahu.

Nutnost orientovat se v informacích je zřejmá; ten, kdo disponuje určitou informací, může mít velkou výhodu oproti ostatním. Při současném růstu objemu dostupných informací však přestává být možné zachytit všechny důležité zprávy a naopak se vyhnout těm nepodstatným. Tento stav tak může vyústit ve snížení důležitosti kanálů, které jsou přehlceny informacemi bez možností orientovat se v nich. Jinými slovy: Subjekty, které uchovávají či zpřístupňují velké množství informací, budou potřebovat nástroje pro snadné a inteligentní vyhledávání v těchto informacích. Pokud takové nástroje nebudou mít či je nenabídnou zákazníkovi, mohou se dostat do konkurenční nevýhody s fatálními důsledky.

Má se za to, že existující konvenční vyhledávací mechanismy začnou být postupně nahrazovány mechanismy sémantickými. Pojem sémantické vyhledávání zatím není přesně definován, ale povětšinou je chápán jako vyhledávání, které nepracuje na základě toho, „co uživatel zadal“, ale na základě toho, „co zadaný dotaz znamená“ nebo ještě lépe „co uživatel chtěl hledat“. Druhá z interpretací je ve světě informačních technologií značně vágní či spíše nemyslitelná a v dnešní době tak spadá do oblasti vědecké fikce. I přes to neutuchají snahy o vývoj systémů využívajících sémantiky zpracovávaných dat[1].

Vzhledem k dostupnosti přenosných snímacích zařízení jako jsou fotoaparáty v osobních komunikátorech má dnes generovaný obsah výrazně multimediální povahu. Jak již bylo naznačeno, obraz může poskytnout o zachycené skutečnosti výrazně více informací než textový popis. Z tohoto důvodu je žádoucí existence metod, které by byly schopny z obrazu extrahovat co nejvíce relevantních informací. Jedná se především o informace vysokoúrovňové, které mají vysoký vypovídací potenciál o zobrazené scéně. Mohou to být např. typy přítomných objektů, jejich vzájemné vztahy a z toho vyplývající interpretace celé scény.

[1] Například projekt Wolfram Alpha http://www.wolframalpha.com/, který je spíše znalostním vyhledávačem, či pokus Googlu o porovnání objektů stejné třídy Google Squared http://www.google.com/squared.

4.2 Dostupné metody

Pokusíme-li si představit systém získávající informace z obrazu inteligentním způsobem, zřejmě shledáme, že bude nutné, aby takový systém flexibilně extrahoval různé druhy informací podle širšího kontextu. Jedním z oněch druhů budou pravděpodobně i metrické vlastnosti scény a objektů v ní obsažených. Zmíněný systém se bude nejprve snažit zjistit, o jaký typ scény jde, respektive jaké objekty se v ní nalézají. Na základě toho bude zjišťovat jejich základní parametry a vztahy, které poslouží ke zpřesnění představy o zobrazené scéně. Poté dojde k extrakci informace o vybraných relevantních objektech v závislosti na typu scény a výsledkem bude poměrně jasná představa o situaci v obraze. Jak je patrné, nedílnou součástí výše popsaného procesu je odhad parametrů objektů scény.

K úspěšnému odhadu parametrů je nutné scénu nejprve zkalibrovat, tedy odhadnout její geometrii. Jelikož musí být odhad proveden pouze na základě jediného snímku scény, není možné použít algoritmy využívající fotogrammetrie – tedy odhad třetího rozměru z dvojice stereosnímků nebo ze dvou po sobě následujících snímků videosekvence natočené pohybující se kamerou. Při kalibraci scény z jediného snímku (a obecně při odvozování parametrů 3D prostoru z parametrů 2D prostoru) je nutné poskytnout kalibrační metodě nějakou apriorní informaci o scéně a tím odstranit nejednoznačnost zobrazení 2D $\rightarrow$ 3D. Toutou informací může být např. velikost některých objektů ve scéně, jejich vzdálenost či orientace.

Nabízí se použít metodu založenou na odhadu typu povrchů, ale tato metoda vyžaduje přítomnost několika různých povrchů ve scéně s různou předpokládanou orientací v prostoru, na což nelze vždy spoléhat. Tato metoda by tedy mohla být použita pouze jako doplňková. Postup založený na odvození perspektivy z odhadu úběžníků navzájem kolmých směrů nelze použít ze stejného důvodu jako předchozí, protože vyžaduje přítomnost výrazných a pokud možno kolmých hran v obraze, např. na budovách. Mohlo by se tedy také jednat pouze o doplňkovou metodu. Odhad scény pomocí vlastních obrazů (viz stranu 14) je potenciálně vhodným kandidátem vzhledem ke své komplexnosti a využití kontextové informace, ovšem tento aspekt je zároveň i negativem s přihlédnutím k náročnosti implementace. Poměrně přímočarým způsobem kalibrace scény je její odhad na základě ručně označených objektů známé velikosti v obraze. Jeho využití v automaticky pracujícím systému sice diskvalifikuje nutnost manuálního označování objektů, ale tento nedostatek by mohl být jednoduše odstraněn tím, že by byly objekty známé velikosti detekovány automaticky. Rovněž by tato metoda musela být zbavena závislosti na vnitřních parametrech kamery, protože je nutné zpracovávat i obrazy bez jakýchkoliv metadat (např. EXIF[2]).

Oblast detekce objektů v obraze je poměrně dobře prozkoumána, jak je ukázáno v kapitole 2. Existují různé přístupy k této problematice, z nichž každý je vhodný pro detekci jiného druhu objektů. Metody pracující shora dolů se hodí pro detekci pouze úzké skupiny objektů, jelikož pro každý typ objektů je nutné vyvinout speciální metodu – nejprve se musí najít kandidátní oblasti podle specifických vlasností hledaných objektů, poté se musí verifikovat detaily objektů. Metody založené na modelu jako template matching či model fitting často vyžadují poměrně přesnou počáteční aproximaci pozice objektu, ze které iterují ke skutečné pozici. Samostatně jsou tedy v kontextu této práce nepoužitelné, mohly by být využity pouze pro velice přesnou lokalizaci objektů, které byly předem hrubě lokalizovány pomocí jiných metod. Metody založené na vzhledu, jako např. klasifikátory skenující celý obraz, se zdají být vhodnou volbou. To je dáno jejich relativní nezávislostí na třídě detekovaných objektů (klasifikátor může být natrénován pro detekci téměř libovolné třídy

[2]EXIF – zkratka z anglického Exchangeable Image File Format. Formát metadat používaný u obrazových souborů [46].

objektů, jejichž variabilita není příliš vysoká) a nižší výpočetní náročností. Vhodné jsou rovněž metody pracující zezdola nahoru. Jsou schopné vyhledávat široké spektrum objektů jak pevných, tak také těch s měnícím se tvarem.

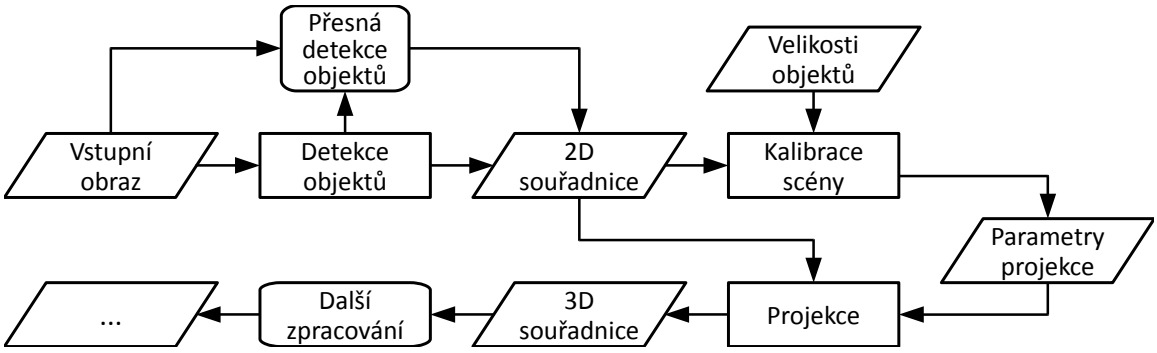
4.3 Návrh řešení

Vzhledem ke skutečnostem uvedeným v sekci 4.2 bylo rozhodnuto, že nejednoznačnost interpretace 3D prostoru z 2D snímku bude odstraněna přidáním apriorní znalosti o velikosti objektů ve scéně. To znamená, že v obraze musí být detekovány a lokalizovány objekty předem známé velikosti. K jejich detekci bude použit již hotový softwarový detekční framework využívající klasifikátory trénované metodou AdaBoost (dále jen framework „Abon“), který je vyvíjen na FIT VUT. Přednosti tohoto řešení jsou známé a mnoha aplikacemi ověřené vlastnosti, jako je dobrá škálovatelnost, rychlost a především úspěšnost detekcí. Nevýhodou je především nutnost existence velkého množství anotovaných trénovacích příkladů pro každou třídu rozpoznávaných objektů, nicméně tento aspekt je společný všem statistickým detekčním metodám, které využívají strojové učení s učitelem.

Výhledově by objekty mohly být detekovány také některou z metod pracujících zezdola nahoru, např. metodou popsanou v [34] a to kvůli schopnosti detekovat objekty měnícího se tvaru (osoby, zvířata apod). Za úvahu také stojí nasazení některých metod založených na modelu, protože jsou schopny detekovat pozici objektu se subpixelovou přesností, což by mohlo mít kladný vliv na přesnost výsledků.

Žádná z metod kalibrace scény zmíněných v předchozí sekci nevyhovuje všem požadavkům kladeným touto prací. Bude tak nutné modifikovat některou z existujících metod pracujících s jediným snímkem scény nebo vytvořit metodu novou. Řešením může být převedení souřadnic scény i obrazu do homogenních souřadnic a poté vypočítat projekční matici popisující zobrazení z prostoru scény do prostoru obrazu. Pokud se tak stane, bude projekce jednoznačně definována a bude možné určit souřadnice objektů v prostoru scény.

Jakmile budou 3D souřadnice objektů známy, může být určena např. topologie objektů, vzdálenosti mezi nimi, obsahy ploch jimi vymezené a další. Obecně mohou prostorové souřadnice objektů sloužit jako vstup pro další úlohy a algoritmy pro extrakci informace z obrazu. Dataflow diagram navrženého systému je znázorněn obrázku 4.1.



Obrázek 4.1: Dataflow diagram navrženého systému. Ve vstupním obraze jsou detekovány předdefinované objekty, volitelně mohou být detekce zpřesněny. S apriorní znalostí velikosti objektů je možné určit parametry projekce, jejíž aplikací na 2D souřadnice detekovaných objektů jsou získány jejich 3D souřadnice. Tyto pak mohou být dále využity.

Kapitola 5

Implementace a výsledky

V této kapitole je popsán software implementovaný v jazyce C++, který je schopen automaticky kalibrovat scénu na základě detekovaných objektů známé velikosti.

5.1 Model projekce

Kalibrace spočívá v určení roviny scény a v určení souřadnic detekovaných objektů v trojrozměrném prostoru scény. Tyto souřadnice jsou vypočítány na základě známých velikostí skutečných objektů a velikostí a pozic jejich obrazů v dvojrozměrném prostoru snímku. Vzhledem k předpokladu, že v drtivé většině případů budou zpracovávány snímky pořízené běžnými spotřebními fotopřístroji (videokamery, kompaktní fotoaparáty, kamery v mobilních zařízeních), je uvažováno zobrazení pomocí perspektivní projekce. Pokud by některý ze snímků byl pořízen pomocí paralelní projekce, je jednoduché přejít od perspektivní k paralelní projekci pouhým posunutím středu promítání do nekonečna.

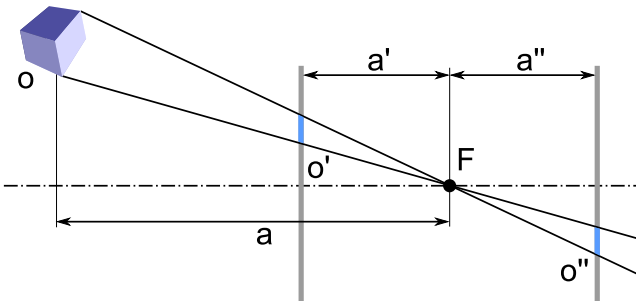
Při perspektivní projekci se všechny paprsky sbíhají v jediném bodě – optickém středu kamery či ohnisku. Tyto paprsky dopadají na stínítko a vytvářejí na něm obraz scény. Podle konstrukce objektivu může stínítko ležet buď před ohniskem nebo za ohniskem, viz obrázek 5.1. Z hlediska perspektivní projekce a kalibrace scény na tom ovšem nezáleží, protože obrazy o' a o'' objektu o jsou pouze převrácené a poměr jejich velikostí odpovídá poměru vzdáleností a' a a'' stínítek od ohniska F podle rovnice 5.1:

$$\frac{o}{a} = \frac{o'}{a'} = \frac{o''}{a''}. \tag{5.1}$$

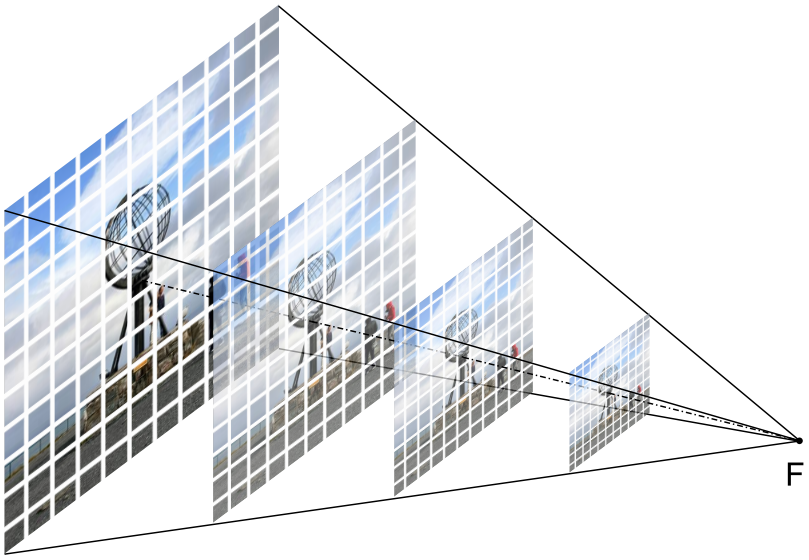
Je tedy jednoduché převést jeden případ na druhý a obráceně.

Je zřejmé, že velikost obrazu závisí při konstantní pozici objektu vzhledem ke snímací soustavě pouze na vzdálenosti stínítka od ohniska. Pokud je stínítko vnímáno jako snímací čip, který je složen z jednotlivých pixelů, stává se velikost obrazu relativní veličinou vzhledem k velikosti snímacího čipu či k rozměru jednotlivých pixelů. Stínítko nechť je dále vnímáno jako dvojrozměrná pravoúhlá pravidelná mřížka, kde každé pole mřížky představuje pixel výsledného obrazu.

Na základě výše uvedených předpokladů je možné dojít k závěru, že stejnou scénu pevně umístěnou vzhledem ke snímací soustavě je možné zachytit nekonečně mnoha způsoby – pokaždé na jiné stínítko ležící v jiné vzdálenosti od ohniska s jinou velikostí pole mřížky (pixelu). Rozlišení stínítka je vždy stejné. Vzdálenost stínítka od ohniska a velikost pixelu jsou však svázány v konstatním poměru, jak je ukázáno na obrázku 5.2.



Obrázek 5.1: Středové promítání. Objekt o ve vzdálenosti a od středu promítání (ohniska) F je zobrazen na dvě stínítka vyobrazená šedými svislicemi. Na prvním z nich, ležícím ve vzdálenosti a' před středem promítání, vzniká vzpřímený obraz o'. Na druhém, ležícím ve vzdálenosti a'' za středem promítání, vzniká převrácený obraz o''. Vzdálenosti a velikosti objektů a obrazů jsou závislé podle rovnice 5.1.



Obrázek 5.2: Stejná scéna zachycená na několika stínítkách se stejným poměrem vzdálenosti od ohniska F a velikosti pixelu.

Pokud je úkolem zjistit parametry projekce na základě daného obrazu a apriorních informací o objektech ve scéně, je nutné jednu ze zmíněných proměnných (vzdálenost stínítka od ohniska, velikost pixelu) zafixovat a druhou dopočítat tak, aby projekce vyhovovala zadání. Otevírají se tím dvě základní možnosti, jak zobrazení modelovat:

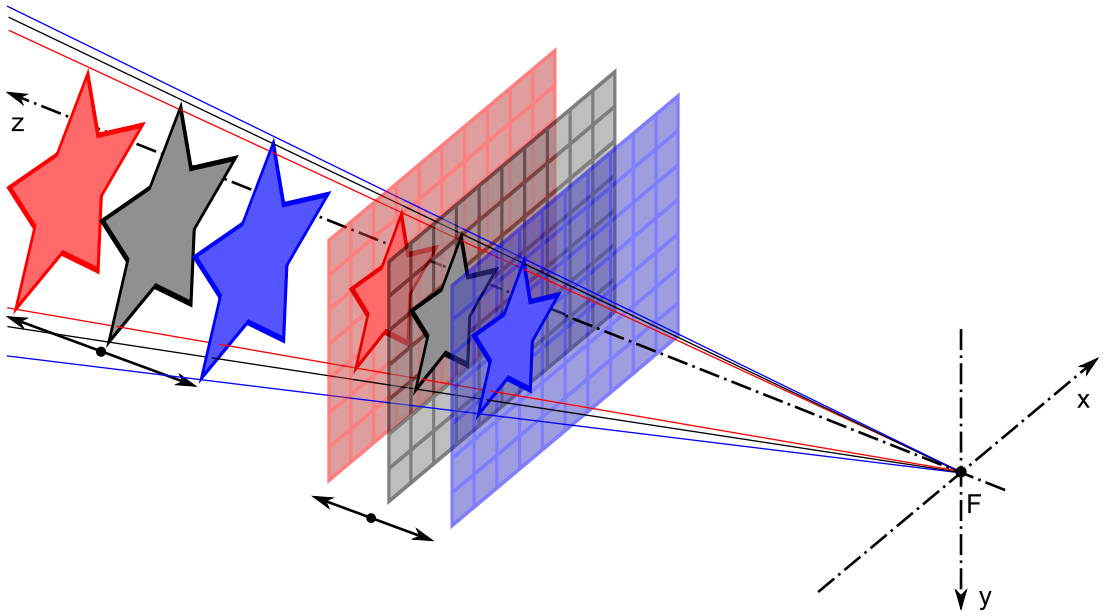
1. Pracovat s konstantní vzdáleností stínítka od ohniska, měnit velikost stínítka a tím i velikost jednotlivých pixelů. Intuitivně si lze tento model představit jako obdélník elastické tkaniny posazený na optickou osu v určité vzdálenosti od ohniska, přičemž tkanina je natahována do požadované velikosti. Se změnou velikosti tkaniny se mění i velikost ok v osnově. Paprsky vržené z ohniska jednotlivými oky (pixely) pak s optickou osou svírají různé úhly v závislosti na natažení tkaniny.

2. Pracovat s konstantní velikostí stínítka, měnit jeho vzdálenost od ohniska. Lze si představit drátěné síto pevné velikosti, které je možné posouvat po optické ose. Paprsky vržené z ohniska jednotlivými oky budou opět měnit úhel sevřený s optickou osou v závislosti na vzdálenosti stínítka od ohniska.

Oba modely jsou vzájemně převoditelné a ekvivalentní. Pro účely této práce byl však vybrán druhý z modelů a to zejména proto, že se autorovi zdál intuitivnější a lépe uchopitelný.

Pro další výklad je vhodné popsat souřadný systém scény a obrazu. Souřadný systém scény je kartézský pravotočivý a pro jednoduchost je pevně spojen s kamerou. Jeho počátek leží v optickém středu kamery (v ohnisku). Osa x míří vodorovně vpravo, osa y svisle dolů a osa z je totožná s optickou osou a míří směrem do scény. Souřadný systém obrazu je rovněž kartézský a jeho počátek leží ve středu obrazu. Osa x, stejně jako v systému scény, vede horizontálně vpravo a osa y rovněž míří svisle dolů. Přechod ze souřadného systému scény do systému obrazu je proto možno provést velmi jednoduše – pouhým zanedbáním z-ové souřadnice. Je zřejmé, že levý horní roh obrazu o rozměrech $X \times Y$ leží na souřadnicích $\left[-\frac{X}{2}; -\frac{Y}{2}\right]$ a pravý horní roh na souřadnicích $\left[\frac{X}{2}; -\frac{Y}{2}\right]$. Levý dolní roh má analogicky souřadnice $\left[-\frac{X}{2}; \frac{Y}{2}\right]$ a pravý dolní roh $\left[\frac{X}{2}; \frac{Y}{2}\right]$. Souřadný systém je ilustrován na obrázku 5.3

Výše popsaný model projektivního zobrazení byl zaveden zejména proto, aby bylo možné určit pozici skutečných objektů v prostoru scény na základě jejich obrazů zachycených ve snímku. Použití modelu je demonstrováno na obrázku 5.3. Situaci si lze představit tak, že po obrysu obrazu objektu jsou z ohniska vrhány paprsky. Tyto paprsky vytvoří kuželovitý svazek, ve kterém se musí nacházet zobrazený objekt. Ten leží v takové vzdálenosti, kde jej svazek paprsků přesně obepíná.

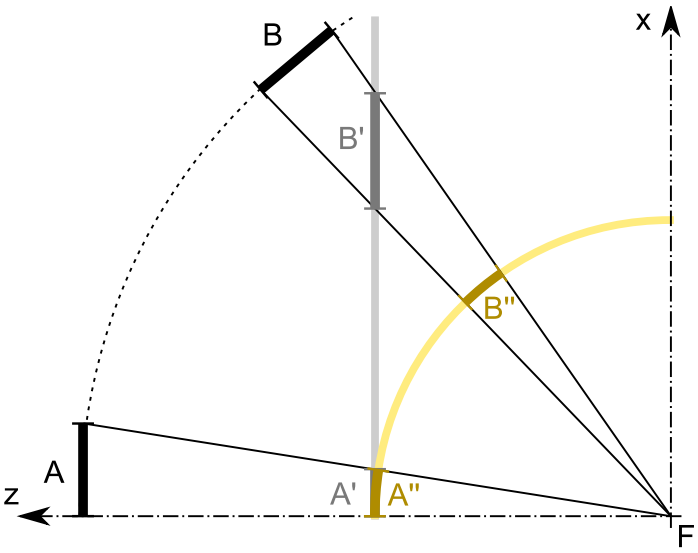


Obrázek 5.3: Souřadný systém a aplikace modelu projekce. Z ohniska F jsou vrhány paprsky po obrysu obrazu objektu. Poloha stínítka na ose z je pohyblivá. Při přiblížení stínítka (modře) svírají paprsky větší úhel a objekt dané velikosti musí ležet blíže. Při oddálení stínítka (červeně) svírají paprsky menší úhel a objekt o stejné velikosti musí ležet dále.

Při přesném výpočtu souřadnic objektů je každý objekt reprezentován jedním bodem, přičemž nezáleží na tom, o který bod objektu se jedná. U všech objektů v obraze by se však mělo jednat o stejný bod, např. o těžiště či o levý spodní přední roh, aby byla souřadnicová reprezentace objektů konzistentí. Tento zvolený bod bude dále nazýván „reprezentantem“ bodu a bude označován R.

Informace o jednotlivých detekovaných objektech, které jsou výstupem použitého detekčního frameworku Abon[1], budou dále nazývány jen „detekce“. Detekce obsahují údaje o obrazu objektu ve snímku, jako jeho polohu, rozměr, natočení v rovině snímku či sílu odezvy použitého klasifikátoru. Neobsahují však informaci o natočení objektu mimo rovinu snímku. To znamená, že není zcela jasné, jaká je orientace objektu vzhledem ke kameře. Vzhledem k tomu, že udávaná velikost detekce je v podstatě velikost obrazu obalové koule objektu, mohou být všechny detekce považovány za obrazy objektů natočených ke kameře čelně.

Jak je demonstrováno na obrázku 5.4, rovinný obraz O' objektu O natočeného čelně ke kameře musí být přepočítán na kulový obraz O''. Teprve tento kulový obraz může být použit při výpočtu vzdálenosti objektu O. Tato skutečnost je důsledkem jevu nastávajícího při použití výše popsané projekce: Obrazy objektů stejné velikosti ve stejné vzdálenosti od středu zobrazení jsou při promítnutí na kouli také stejně velké. Při promítnutí na rovinu toto neplatí a obrazy stejně velké nejsou.



Obrázek 5.4: Projekce na rovinu a na kouli. Rovinný obraz A' i kulový obraz A'' objektu A jsou téměř stejně velké, neboť objekt A leží blízko optické osy z. Objekt B má stejnou velikost jako objekt A a leží ve stejné vzdálenosti od ohniska F, avšak dále od optické osy. Jeho rovinný obraz B' je dokonce větší než objekt samotný, i když leží blíže středu projekce. Jeho kulový obraz B'' má stejnou velikost jako kulový obraz A'' objektu A.

Pro určení souřadnic reprezentanta R libovolného objektu je nejprve nutné znát měřítko m zobrazení objektu. Měřítko je z důvodů uvedených v předcházejícím odstavci vypočteno jako podíl skutečné výšky objektu h a výšky h'' jeho kulového obrazu:

[1] Abon – detekční framework využívající dvoutřídní klasifikátory trénované metodou AdaBoost. Viz http://www.fit.vutbr.cz/research/prod/index.php?id=43&notitle=1.

$$m = \frac{h}{h''} \tag{5.2}$$

Výška h'' kulového obrazu je spočtena z y-ových souřadnic horního a spodního okraje detekce y'_t, y'_b a ze vzdálenosti k stínítka od ohniska podle rovnice

$$h'' = 2\pi k \cdot \frac{\left|\arctan\frac{y'_t}{k} - \arctan\frac{y'_b}{k}\right|}{2\pi} = k \cdot \left|\arctan\frac{y'_t}{k} - \arctan\frac{y'_b}{k}\right|. \tag{5.3}$$

Následně je vytvořen vektor

$$\vec{v'} = \overrightarrow{FR'} \tag{5.4}$$

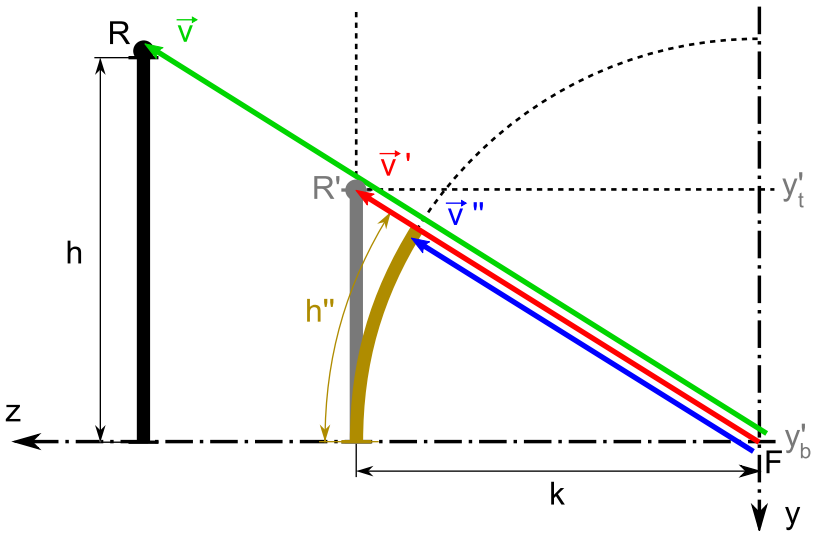
s počátkem v ohnisku F a koncovým bodem R', což je obraz reprezentanta R. Tento vektor je normalizován na délku k (vzdálenost stínítka od ohniska) a následně vynásoben měřítkem objektu m:

$$\vec{v} = \frac{k \cdot \vec{v'}}{\|\vec{v'}\|} m. \tag{5.5}$$

Výsledný vektor $\vec{v}$ má počátek v ohnisku F a jeho koncovým bodem je reprezentant objektu R. Objekt tedy leží na souřadnicích daných reprezentantem:

$$R = F + \vec{v}. \tag{5.6}$$

Postup je ilustrován na obrázku 5.5.



Obrázek 5.5: Výpočet prostorových souřadnic reprezentanta R. Vektor $\vec{v'}$ (červená) spojující ohnisko F s rovinným obrazem (šedá) reprezentanta R' je nejprve normalizován na velikost k, čímž vznikne vektor $\vec{v''}$ (modrá). Tento vektor je pak vynásoben měřítkem objektu m. Výsledkem je vektor $\vec{v}$ (zelená), jehož koncový bod je shodný s reprezentantem objektu R. Podrobněji viz algoritmus 5.1.

Algoritmus 5.1 Výpočet souřadnic reprezentanta objektu z obrazových souřadnic detekce a skutečné velikosti objektu. Geometrická interpretace viz obrázek 5.5.

1: **Vstup**: Souřadnice ohniska F, souřadnice rovinného obrazu objektu R', výška objektu h, y-ové souřadnice horního a spodního okraje detekce y'_t, y'_b, vzdálenost stínítka od ohniska k.
2: Vyčísli výšku kulového obrazu $h'' = k \cdot \left| \arctan \frac{y'_t}{k} - \arctan \frac{y'_b}{k} \right|$.
3: Spočítej měřítko objektu $m = \frac{h}{h''}$.
4: Zkonstruuj vektor $\vec{v'} = \overrightarrow{FR'}$.
5: Normalizuj $\vec{v'}$ na délku k: $\vec{v''} = \frac{k \cdot \vec{v'}}{\|\vec{v'}\|}$.
6: Zkonstruuj vektor $\vec{v} = m \cdot \vec{v''}$.
7: **Výstup**: Souřadnice reprezentanta $R = F + \vec{v}$.

5.2 Chyba řešení

Pro libovolnou vzdálenost k stínítka od ohniska je možné pro každý zobrazený objekt nalézt jeho vzdálenost, která splňuje podmínku „obepínajících paprsků“ uvedenou na straně 28 a ilustrovanou v obrázku 5.3. Aby bylo možné odhadnout správnou vzdálenost stínítka, která co nejlépe aproximuje skutečnou projekci, je nutné přidat další apriori znalost o scéně. Touto znalostí je předpoklad, že všechny detekované objekty spočívají na zemi.

U každého objektu je tedy nutné určit bod, který se dotýká země. Tento bod se stane reprezentantem objektu. V reálu samozřejmě existují i objekty, které zpravidla přímo na zemi neleží. Pokud mají i tyto objekty být využity ke kalibraci scény, je nutné znát jejich předpokládanou výšku nad zemí a podle toho pak určit tzv. „patu“, která se stane jejich reprezentantem. Pata objektu je myšlený bod, který se nachází na zemi a má k objektu nějaký předem definovaný vztah. Je-li např. za objekt považována dopravní značka, která bývá obvykle umístěna na stojanu známé výšky, je její patou myšlený bod, kde se stojan dotýká země. Jelikož není možné jednoduše určit svislý směr ve scéně, je nutné paty objektů určit pouze na základě detekovaných obrazů objektů. Nevýhodou je, že se tím zvětšuje chyba pozice zenesená do výpočtu při detekci objektu.

Má-li objekt o výšce h, šířce w a předpokládané výšce nad povrchem e obraz o výšce h' a šířce w', který je detekován na pozici O' s natočením α v rovině obrazu, je obraz P' jeho paty definován jako

$$P' = O' + \vec{d'}, \tag{5.7}$$

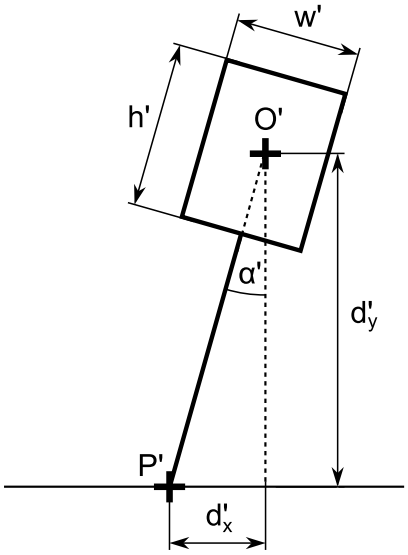
kde vektor $\vec{d'}$ má složky $\left[d'_x; d'_y\right]$, které jsou vyčísleny následovně:

$$d'_x = \sin\alpha \cdot e \cdot \frac{w'}{w} \tag{5.8}$$

$$d'_y = \cos\alpha \cdot e \cdot \frac{h'}{h} \tag{5.9}$$

Geometrický význam výpočtu viz obrázek 5.6.

Je-li možné vypočíst množinu bodů, které leží na zemi, je možné definovat i parametry geometrického modelu povrchu scény. Drtivá většina metod pro kalibraci scény, z nichž některé byly popsány v kapitole 3 (jako např. metody publikované v [3, 35]), uvažuje povrch



Obrázek 5.6: Výpočet souřadnic obrazu P' paty objektu. Obraz paty objektu je vzhledem k pozici O', kde byl detekován obraz objektu s natočením α', posunut o vektor $\vec{d'} = [d'_x; d'_y]$. Jeho složky jsou vypočteny podle rovnic 5.8 a 5.9.

scény jako jediný rovinný segment, výjimečně soustavu několika rovinných segmentů. Zřejmou výhodou a zároveň nevýhodou této aproximace je její jednoduchost. Téměř s jistotou však lze říci, že složitější aproximace by nepřinesla lepší modelovací schopnosti s adekvátní výpočetní složitostí. V této práci je proto povrch scény modelován jako rovina.

Úkolem je tedy určit parametry roviny na základě shluku bodů, které by v této rovině měly ležet. Zřejmě je k tomu zapotřebí alespoň tří bodů, jinak rovina nebude plně definována. Při vyšším počtu bodů je naopak vysoce pravděpodobné, že těmito body nebude možné jakoukoliv rovinu proložit. To je však s ohledem na nepřesnost vstupních dat očekáváno a je tedy žádoucí určit parametry roviny tak, aby chyba daná vzdálenostmi bodů od vypočtené roviny byla minimální. Pro tuto úlohu byla vybrána metoda PCA[2], která provede rozklad kovarianční matice souřadnic bodů na vlastní čísla a vlastní vektory. Vlastní vektor s odpovídajícím největším vlastním číslem určuje směr, ve kterém má shluk bodů největší rozptyl. Obdobně vlastní vektor s druhým největším vlastním číslem určuje směr, ve kterém má shluk bodů největší rozptyl, je-li zanedbán rozptyl ve směru daném prvním vektorem. Jinými slovy, první dva vlastní vektory leží v rovině, ve které má shluk bodů největší rozptyl, tzn. předpokládanou rovinu scény. Vlastní číslo odpovídající zbylému vlastnímu vektoru (který je kolmý k nalezené rovině scény), udává rozptyl shluku bodů v tomto směru, tedy součet čtverců vzdáleností bodů od nalezené roviny. Toto číslo udává chybu řešení.

Při výpočtu chyby řešení pro danou vzdálenost k stínítka od ohniska je nejprve nutné vypočítat prostorové souřadnice všech detekovaných objektů pomocí metody popsané pomocí algoritmu 5.1. Od každé ze souřadnic všech objektů je poté odečtena jejich střední hodnota, aby bylo odstraněno vychýlení datového souboru souřadnic. Následně je zkonstruována matice $\mathbf{M}$ o rozměrech $n \times 3$, kde n je počet detekovaných objektů:

[2]Zkratka z anglického „Principal Component Analysis“, česky Analýza hlavních komponent.

$$\mathbf{M} = \begin{bmatrix} x_0 & x_1 & \cdots & x_{n-1} \\ y_0 & y_1 & \cdots & y_{n-1} \\ z_0 & z_1 & \cdots & z_{n-1} \end{bmatrix} \tag{5.10}$$

Každý sloupec matice tedy obsahuje souřadnice jednoho z objektů, každý její řádek obsahuje stejné komponenty všech souřadnic. Nyní je možné spočítat kovarianční matici $\mathbf{\Sigma}$ matice $\mathbf{M}$:

$$\mathbf{\Sigma} = \mathbf{M}\mathbf{M}^T \tag{5.11}$$

Kovarianční matice $\mathbf{\Sigma}$ je symetrická matice o rozměru 3×3 a je vstupem pro PCA. Byla využita implementace PCA z knihovny OpenCV[3] voláním funkce `cvEigenVV`. Výstupem této funkce pro uvedenou kovarianční matici $\mathbf{\Sigma}$ je vektor $\mathbf{l}$ tří vlastních čísel seřazených od největšího po nejmenší a matice $\mathbf{E}$ tří odpovídajících vlastních vektorů, přičemž vektory jsou uloženy v jejích řádcích. Nejmenší vlastní číslo $\mathbf{l}_2$ je hledanou chybou řešení. Celý postup je zapsán pomocí algoritmu 5.2.

Algoritmus 5.2 Výpočet chyby řešení.

1: **Vstup**: Vzdálenost k stínítka od ohniska, množina detekcí D.
2: **Inicializace**: Střední hodnota souřadnic $avg = 0$, matice 3D souřadnic $\mathbf{M} = ()$.
3: **pro každé** $d \in D$ **proveď**
4: Spočti 3D souřadnice R_d objektu z jeho detekce d podle algoritmu 5.1.
5: $avg \leftarrow avg + R_d$
6: **konec** (pro každé)
7: $avg \leftarrow avg/|D|$
8: **pro každé** $d \in D$ **proveď**
9: Odečti od souřadnic jejich střední hodnotu $R_d \leftarrow R_d - avg$.
10: Přidej matici $\mathbf{M}$ jeden sloupec a naplň jej hodnotami souřadnic R_d.
11: **konec** (pro každé)
12: Spočítej kovarianční matici $\mathbf{\Sigma} = \mathbf{M}\mathbf{M}^T$.
13: Pomocí PCA vypočítej vektor vlastních čísel $\mathbf{l}$ a vektor vlastních vektorů $\mathbf{e}$: $(\mathbf{l}, \mathbf{e}) = \mathrm{PCA}(\mathbf{\Sigma})$.
14: Seřaď vlastní čísla ve vektoru $\mathbf{l}$ od největšího po nejmenší.
15: **Výstup**: Chyba řešení jako nejmenší vlastní číslo $err = \mathbf{l}_2$.

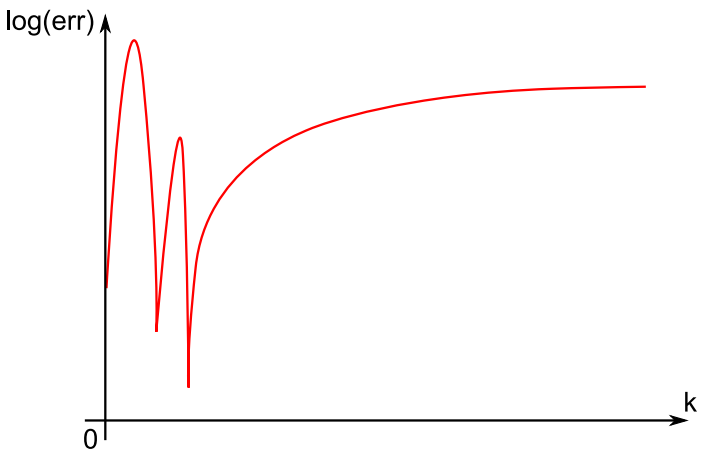
5.3 Hledání nejvhodnější projekce

Se zavedenou metrikou chyby daného řešení je již možné hledat nejvhodnější projekci, tedy projekci s minimální chybou v závislosti na hlavním parametru projekce, vzdálenosti k stínítka od ohniska. Jelikož není k dispozici analytické řešení problému, není ani možné využít analytických metod pro určování extrémů funkce.

Experimentálně bylo zjištěno, že průběh chyby v závislosti na parametru řešení k je průběhem nezáporné funkce, jejíž hodnota se pro velká k asymptoticky blíží určité hodnotě závislé na vstupních parametrech problému. Důležitější ovšem je fakt, že pokud má problém řešení, objevují se ve funkci lokální minima a maxima, zejména pro malé hodnoty k. Maxima jsou hladká, naopak minima jsou v drtivě většině případů ostrá, takže derivace

[3] OpenCV – zkratka z anglického Open Source Computer Vision. Knihovna funkcí pro počítačové vidění. Viz http://opencv.willowgarage.com/wiki/.

chybové funkce by v těchto bodech byla nespojitá. Charakteristický průběh chybové funkce je znázorněn na obrázku 5.7.



Obrázek 5.7: Charakteristický průběh chybové funkce. Na vodorovné ose jsou vyneseny hodnoty parametru projekce k, na svislé ose je vynesena hodnota chyby err v logaritmickém měřítku.

Hodnoty chybové funkce lze teoreticky zjišťovat na intervalu $(0; \infty)$, z praktických důvodů je ale zřejmě nutné omezit tento interval na $(0; maxK\rangle$, kde $maxK$ je experimentálně zjištěná maximální hodnota k, pro kterou má ještě smysl vyšetřovat chybu řešení.

Nejjednodušším přístupem k hledání nejvhodnější projekce by bylo vypočítat hodnoty chyby pro k měnící se s konstantním krokem s na celém intervalu $(0; maxK\rangle$ – tedy pro hodnoty $s, 2s, 3s$ atd. a vybrat z nich tu nejmenší; ta by pak byla hledaným minimem chybové funkce. Problémem je ovšem nastavení optimální hodnoty kroku s. Při malém kroku by totiž docházelo k plýtvání výpočetního času, při velkém kroku hrozí nepřesné určení hodnoty minima či dokonce jeho přeskočení.

Na základě výše popsané úvahy byla navržena následující metoda hledání optimální projekce: Nejprve je nastavená počáteční hodnota $kMin$ parametru k. Poté probíhá v cyklu hledání globálního minima na intervalu $\langle kMin; kMax\rangle$. Při každé iteraci cyklu je nejprve nalezeno nejbližší lokální maximum větší než aktuální hodnota k. Hledání maxim probíhá prohledáváním prostoru parametru k s exponenciálně se zvětšujícím krokem. Jakmile je maximum nalezeno, probíhá hledání nejbližšího minima následujícího po maximu, opět s exponenciálně se zvětšujícím krokem. Tento způsob hledání minima přirozeně vede k jeho nepřesné lokalizaci. Proto následuje přesné určení hodnoty minima opakovaným sekvenčním vyhledáváním na zmenšujícím se intervalu se zmenšujícím se krokem. Hodnota minima je uložena a je porovnána s hodnotou dosud nejmenšího minima. Pokud je hodnota aktuálního minima menší, přepíše hodnotu dosud nejmenšího nalezeného minima. Celý cyklus se opakuje. Jakmile je výše popsaným způsobem prohledán celý interval $\langle kMin; kMax\rangle$, je známa poloha a hodnota globálního minima. Poloha tohoto minima určuje parametr k nejvhodnější projekce a jeho hodnota udává chybu této projekce. Celý postup je podrobně popsán v algoritmu 5.3. Podprocedura hledání lokálního maxima s exponenciálně se zvětšujícím krokem je popsána v podproceduře 5.1, podprocedura obdobného hledání lokálního minima v podproceduře 5.2 a podprocedura pro zpřesnění hodnoty nalezeného lokálního minima je uvedena v podproceduře 5.3.

Algoritmus 5.3 Hledání nejvhodnější projekce.

1: **Vstup**: Interval prohledávání $\langle kMin; kMax\rangle$, výchozí velikost kroku $sInit$.
2: **Inicializace**:
$k = kMin$, krok $s = sInit$,
maximum $max = 0$, hodnota maxima $maxErr = 0$,
minimum $min = 0$, hodnota minima $minErr = \infty$,
globální minimum $minGlob = 0$, hodnota globálního minima $minGlobErr = \infty$.
3: **opakuj**
4: Najdi nejbližší lokální maximum s exponenciálně se zvětšujícím krokem podle podprocedury 5.1: $(max, maxErr) = \text{najdiMaximum}(k)$.
5: $k \leftarrow max$
6: Najdi nejbližší lokální minimum s exponenciálně se zvětšujícím krokem podle podprocedury 5.2: $(min, minErr, s) = \text{najdiMinimum}(k)$.
7: Zpřesni polohu nalezeného minima opakovaným sekvenčním prohledáváním se zmenšujícím se krokem podle podprocedury 5.3:
$(min, minErr) \leftarrow \text{zpresniMinimum}(min, s)$.
8: $k \leftarrow min$
9: **pokud** $minErr < minGlobErr$ **pak**
10: $minGlobErr = minErr$
11: $minGlob = min$
12: **konec** (pokud)
13: **dokud** $k < kMax$
14: **Výstup**: Parametr nejvhodnější projekce $minGlob$.

Podprocedura 5.1 Hledání nejbližšího lokálního maxima s exponenciálně se zvětšujícím krokem. Tato podprocedura je použita v algoritmu 5.3.

1: **Vstup**: Interval prohledávání $\langle kStart; kStop\rangle$, výchozí velikost kroku $sInit$, faktor zvětšování kroku $sMulFactor$.
2: **Inicializace**: $k = kStart$, krok $s = sInit$, dosud nalezené maximum $max = kStart$, maximální chyba $maxErr = 0$.
3: **opakuj**
4: Vyčísli chybu pro aktuální k podle algoritmu 5.2: $err = \text{spoctiChybu}(k)$.
5: **pokud** $err < maxErr$ **pak**
6: Pokud chybová funkce začala klesat, vrať aktuální hodnotu chyby a skonči: $maxErr \leftarrow err$.
7: **skonči**
8: **jinak**
9: $maxErr \leftarrow err$
10: $max \leftarrow k$
11: Zvětši k o krok s: $k \leftarrow k + s$.
12: Exponenciálně zvyš velikost kroku: $s \leftarrow sMulFactor \cdot s$.
13: **konec** (pokud)
14: **dokud** $k < kStop$
15: **Výstup**: Nejbližší lokální maximum a jeho hodnota $(max, maxErr)$.

Podprocedura 5.2 Hledání nejbližšího lokálního minima s exponenciálně se zvětšujícím krokem. Tato podprocedura je použita v algoritmu 5.3.

Vstup: Interval prohledávání $\langle kStart; kStop \rangle$, výchozí velikost kroku $sInit$, faktor zvětšování kroku $sMulFactor$.
Inicializace: $k = kStart$, krok $s = sInit$, dosud nalezené minimum $min = kStart$, minimální chyba $minErr = \infty$.
opakuj
  Vyčísli chybu pro aktuální k podle algoritmu 5.2: $err = \text{spoctiChybu}(k)$.
  pokud $err > minErr$ **pak**
    Pokud chybová funkce začala stoupat, vrať aktuální hodnotu chyby a skonči: $minErr \leftarrow err$.
    skonči
  jinak
    $minErr \leftarrow err$
    $min \leftarrow k$
    Zvětši k o krok s: $k \leftarrow k + s$.
    Exponenciálně zvyš velikost kroku: $s \leftarrow sMulFactor \cdot s$.
  konec (pokud)
dokud $k < kStop$
Výstup: Nejbližší lokální minimum, jeho hodnota a aktuální velikost kroku $(min, minErr, s)$.

Podprocedura 5.3 Zpřesnění hodnoty nalezeného lokálního minima opakovaným sekvenčním prohledáváním se zmenšujícím se krokem. Tato podprocedura je použita v algoritmu 5.3.

Vstup: Přibližné minimum $minApprox$, výchozí velikost kroku $sInit$, faktor zmenšování kroku $sDivFactor$, minimální velikost kroku $sMin$.
Inicializace: Krok $s = sInit$, dosud nalezené minimum $min = minApprox$.
pokud $s > sMin$ **proveď**
  Nastav levý okraj prohledávaného intervalu: $left = minApprox - (2 \cdot sInit)$.
  Nastav pravý okraj prohledávaného intervalu: $right = minApprox$.
  Započni prohledávání zleva: $k = left$.
  Zjemni krok: $s \leftarrow s/sDivFactor$.
  Minimální chyba $minErr = \infty$.
  opakuj
    Vyčísli chybu pro aktuální k podle algoritmu 5.2: $err = \text{spoctiChybu}(k)$.
    pokud $err >= minErr$ **pak**
      skonči vnitřní cyklus
    jinak
      $minErr \leftarrow err$
      $min \leftarrow k$
      Zvětši k o krok s: $k \leftarrow k + s$.
    konec (pokud)
  dokud $k > right$
zopakuj
Výstup: Přesné lokální minimum a jeho hodnota $(min, minErr)$.

5.4 Metoda kalibrace

Jak bylo naznačeno v sekci 5.2, pro každý typ detekovaného objektu je nutné znát jeho předpokládané rozměry a také předpokládanou výšku nad povrchem scény. Seznam typů objektů spolu s uvedenými atributy je jedním ze vstupů celého kalibračního systému. V sekci 5.1 bylo uvedeno, že pro detekce objektů v obraze bude použit detekční framework Abon. Systém je však připraven na možnost použití více druhů detektorů, proto má každý typ objektu ve svých parametrech definován také způsob, jak má být detekován – čili jaký detektor a s jakými parametry má být pro detekci objektů daného typu použit.

Již byly popsány všechny dílčí metody, které jsou použity v procesu kalibrace scény. Je tedy možné přistoupit k samotnému procesu kalibrace z jediného snímku. Nejprve proběhnou detekce objektů všech předdefinovaných typů, přičemž všechny detekované objekty jsou ukládány do jednoho kontejneru. Pokud kontejner obsahuje méně jak tři detekce, není možné scénu kalibrovat a běh systému končí s chybou. Pokud byly nalezeny tři a více detekcí, kalibrace pokračuje dále.

V případě, že je ve snímku detekováno větší množství objektů, je možné některé z detekcí vyřadit z dalšího zpracování a to buď z výkonostních důvodů nebo kvůli nedůvěryhodnosti některých z nich. Například detektor Abon je schopen poskytnout informaci o síle odezvy použitého klasifikátoru, která by měla korelovat s důvěryhodností detekce. V některých případech pak může být výhodné vynechat z dalšího zpracování právě tyto detekce, které do procesu kalibrace mohou potenciálně vnášet šum či jiným způsobem snižovat stabilitu kalibrační metody. Otázka výběru vhodných detekcí pro další zpracování zatím není plně zodpovězena a bude předmětem dalšího zkoumání.

Po volitelném vyřazení některých detekcí je hledána nejvhodnější projekce, tak jak bylo popsáno v algoritmu 5.3. Po jejím nalezení je určena rovina scény a ke všem detekovaným objektům jsou dopočítány prostorové souřadnice. Tím je hlavní činnost systému ukončena a extrahované informace mohou být použity dalším programem pro exrakci vysokoúrovňových dat z obrazu. Eventuálně může být prostor zkalibrované scény schematicky vizualizován či mohou být vypočteny vzdálenosti mezi jednotlivými objekty. Celý proces kalibrace scény je stručně popsán v algoritmu 5.4.

Určení roviny scény je poměrně jednoduché, pokud jsou k dispozici výstupy PCA použité při vyčíslování chyby řešení (viz algoritmus 5.2, kroky 1 – 13). Jedním z možných způsobů, jak může být rovina definována, je pomocí bodu, který v ní leží a jejího normálového vektoru. Bodem ležícím v rovině je v tomto případě myšlený bod, jehož souřadnice jsou rovny střední hodnotě souřadnic všech detekovaných objektů scény (opět viz algoritmus 5.2). Normálovým vektorem roviny je pak vlastní vektor, jemuž odpovídá nejmenší vlastní číslo. Vlastní vektory jsou k sobě po dvojicích navzájem kolmé. Jetliže vektory s největšími dvěma odpovídajícími vlastními čísly leží v rovině scény, třetí z vektorů je nutně k rovině scény kolmý. Tento vektor je normálovým vektorem roviny.

PCA je poměrně citlivá na vstupní data ve smyslu orientace vypočtených vlastních vektorů. Nezřídka se stává, že při drobné změně vstupního shluku bodů se změni orientace vypočtených vlastních vektorů, tzn. také orientace normálového vektoru. Proto je nutné kontrolovat, zda normálový vektor míří nad nebo pod rovinu a dodržovat tuto orientaci mezi jednotlivými výpočty.

Algoritmus 5.4 Kalibrace scény z jediného obrazu.

1: **Vstup**: Snímek scény im, seznam typů detekovatelných objektů $objTypes$.
2: **Inicializace**: Kontejner všech detekovaných objektů $allDetections = \emptyset$.
3: **pro každé** typ objektu $objType \in objTypes$ **proveď**
4: Spusť detektor: $detections =$ detekuj$(im, objType)$.
5: Přidej detekce do kontejneru: $allDetections \leftarrow allDetections \cup detections$.
6: **konec** (pro každé)
7: **pokud** $|allDetections| < 3$ **pak**
8: Příliš málo detekcí, není možné kalibrovat scénu: **konec**.
9: **jinak**
10: Volitelně filtruj kontejner $allDetections$ a ponechej v něm alespoň 3 detekce (podle libovolného kritéria).
11: Najdi parametr nejvhodnější projekce podle algoritmu 5.3:
$k =$ nejvhodnejsiProjekce$(allDetections)$.
12: Urči rovinu povrchu pl scény na základě parametru nejvhodnější projekce k.
13: Spočti 3D souřadnice všech detekovaných objektů.
14: **konec** (pokud)
15: **Výstup**: Parametr nejvhodnější projekce k, rovina povrchu scény pl, 3D souřadnice objektů.

5.5 Výsledky

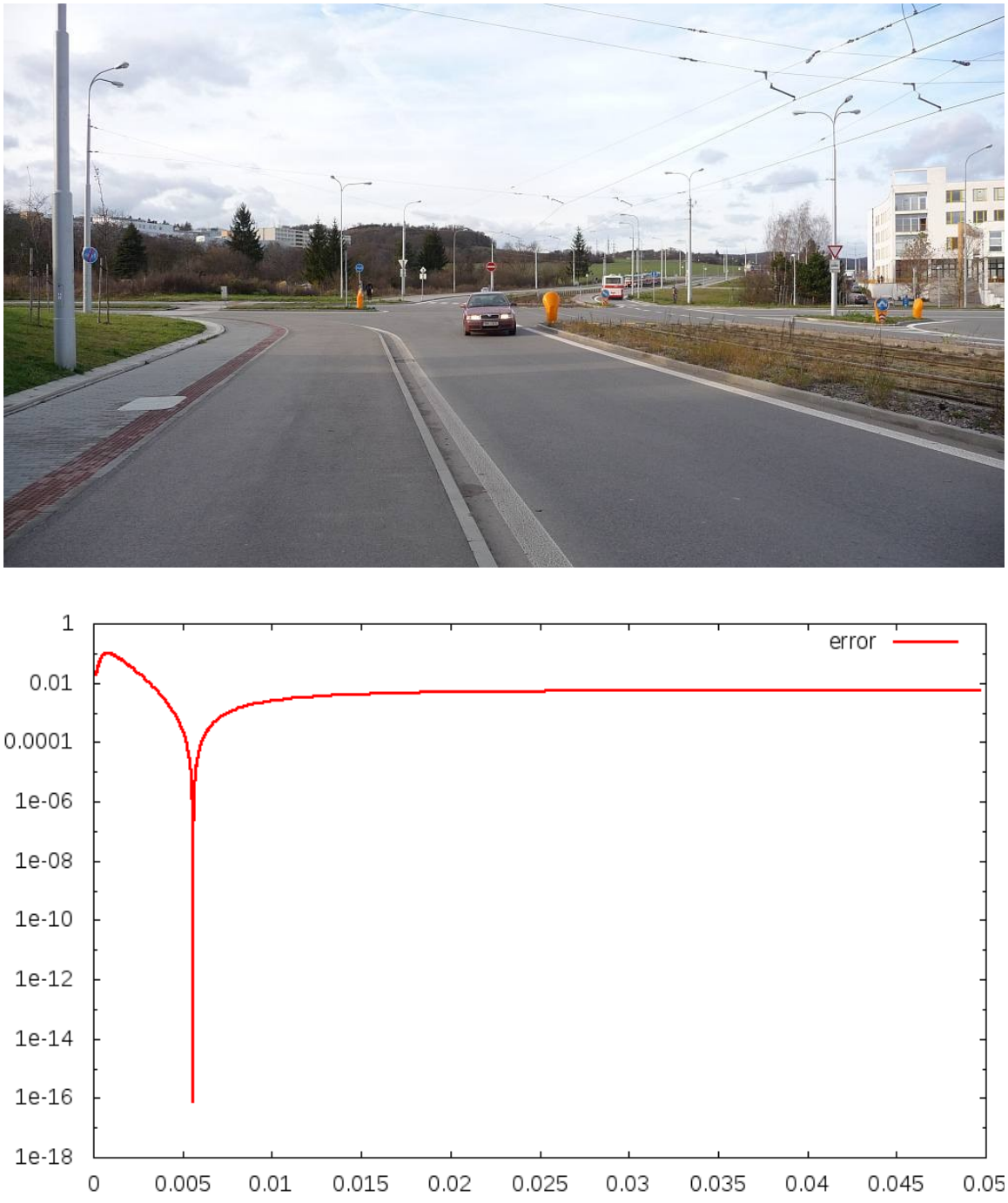
Navržený systém byl implementován jako program v jazyce C++ s využitím knihovny OpenCV a detekčního frameworku Abon. Při rozhodování, jaké typy objektů budou vhodné k detekci, byly vybrány dopravní značky. Tento typ objektů se vyskytuje ve velkém množství a je tak relativně snadné pořídit fotografie, na kterých je zachyceno více dopravních značek zároveň. Dalším důvodem pro volbu zmíněného typu objektů je fakt, že byl k dispozici jejich hotový natrénovaný detektor. Důležité také je, že dopravní značky mají neměnnou velikost[4].

Byly tedy pořízeny fotografie scén s dopravními značkami, na kterých byl systém testován. Vždy, pokud byla scéna rovinná, obsahoval graf chybové funkce jedno či více minim a řešení, neboli nejvhodnější projekce scény do snímku, tak bylo nalezeno (viz např. obrázky 5.8 a 5.9). Kvalitu nalezeného řešení vyčísluje chybová funkce popsaná v sekci 5.2. Její hodnota udává součet čtverců vzdáleností bodů popisujících rovinu od nalezené roviny v metrech. Chyba nalezených řešení se pohybuje zpravidla kolem hodnot řádu 10^{-12} – 10^{-17}. Pokud scéna zachycená na fotografii rovinná nebyla, chybová funkce byla minim prostá a řešení nemohlo být nalezeno (viz např. obrázek 5.10).

Doba běhu implementovaného programu přeloženého překladačem GNU C++ se zapnutou optimalizací (přepínač -O3) se na stolním počítači s 2 GHz procesorem pohybuje kolem 80 – 150 ms. Program spotřebuje cca 1 – 5 MB operační paměti. Měření byla provedena na snímcích scén se 4 – 9 detekovanými objekty, ovšem jak bylo zjištěno, doba běhu programu i jeho paměťová náročnost závisí spíše na průběhu chybové funkce než na počtu detekovaných objektů. Uvedená doba běhu ani paměťové nároky nezahrnují čas a zdroje spotřebované detektorem objektů.

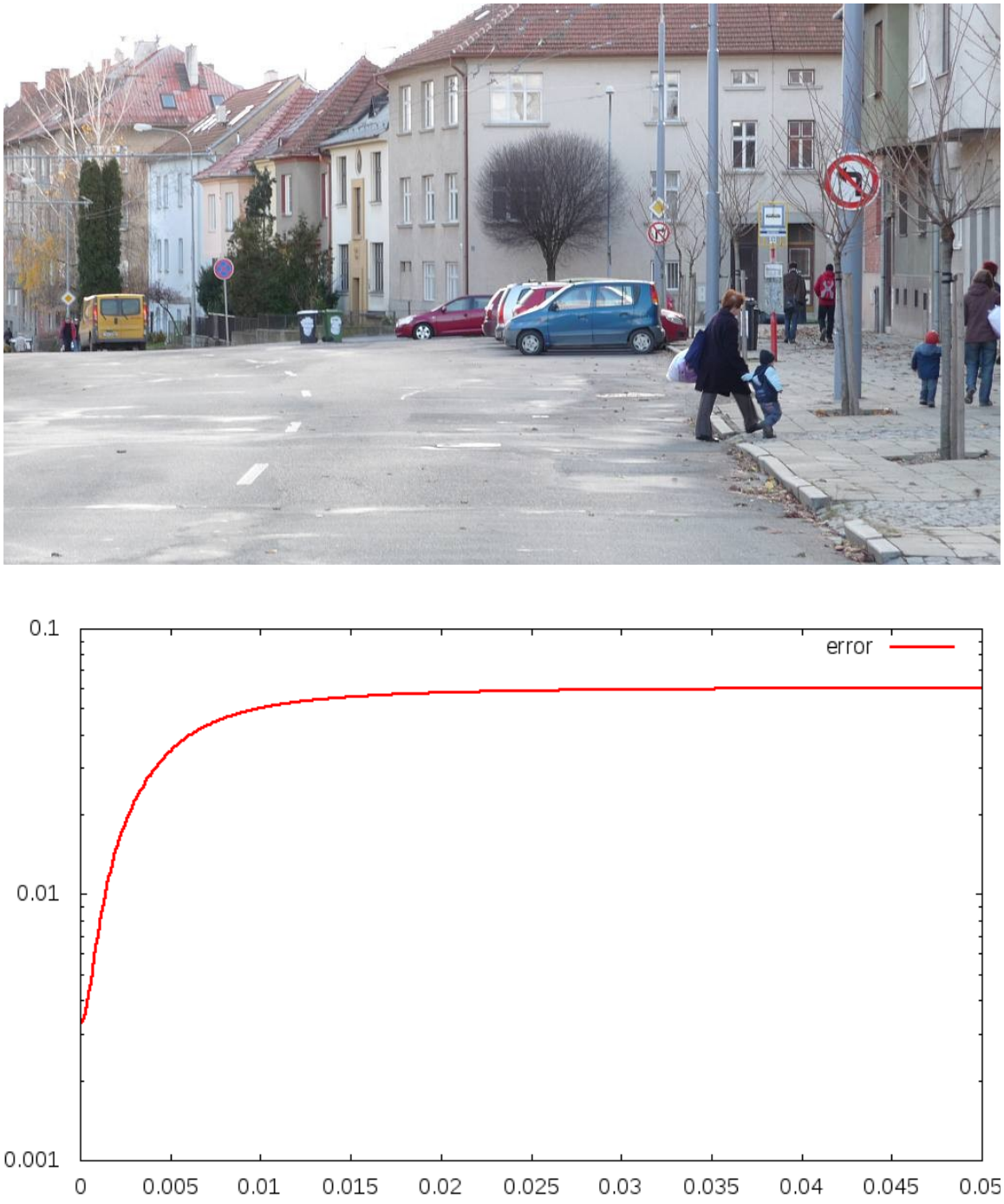
Demonstrační snímky scén spolu s výsledky jsou k vidění na následujících stranách.

[4] V České republice se dopravní značky mohou vyskytovat ve třech různých velikostech, avšak velká většina z nich má velikost stejnou – prostřední ze tří možných.



Obrázek 5.8: (nahoře) Scéna se čtyřmi detekovanými dopravními značkami. (dole) Závislost chyby projekce na jejím parametru k. Scéna je rovinná, tudíž průběh chyby má ostré lokální minimum v bodě $k = 0,0056$. Hodnota chyby v minimu je $6,52 \cdot 10^{-17}$.

Obrázek 5.9: (nahoře) Scéna se čtyřmi detekovanými dopravními značkami. (dole) Závislost chyby projekce na jejím parametru k. Scéna je rovinná, tudíž průběh chyby obsahuje ostrá lokální minima. Globální minium je v bodě $k = 0,0031$. Hodnota chyby v minimu je $2,86 \cdot 10^{-16}$.



Obrázek 5.10: (nahoře) Scéna se čtyřmi detekovanými dopravními značkami. (dole) Závislost chyby projekce na jejím parametru k. Tato scéna není rovinná (povrch se v zadní části scény svažuje), tudíž průběh chyby nemá žádná minima a řešení nebylo nalezeno.

Kapitola 6

Závěr

Cílem této práce bylo vybrat vhodnou metodu pro měření rozměrů a dalších parametrů objektů z obrazů, navrhnout architekturu systému pro implementaci vybrané metody a implementovat ji. Byla navržena nová metoda kalibrace scény na základě automatické detekce objektů známé velikosti a tato metoda byla implementována v jazyce C++, cíl práce byl tedy splněn.

Klíčem k proniknutí do problematiky kalibrace scény a měření rozměrů v obraze bylo studium dostupné literatury na toto téma. Metody popsané v uvedených pramenech ovšem nebyly vhodné k implementaci pro účely této práce. To je vysvětleno v kapitole 4. Proto byla navržena nová metoda kalibrace scény. Architektura systému, který implementuje tuto metodu, je popsána v kapitole 5. Funkčnost metody byla demonstrována na sadě příkladů. Systém byl implementován v jazyce C++, jeho zdrojové kódy jsou k dispozici v příloze A.

Účelem práce bylo vybrat či navrhnout metodu pro kalibraci scény a pro odhad parametrů objektů, což se podařilo. Žádná z metod publikovaných v dostupné literatuře nesplňovala požadavky této práce, proto byla navržena zcela nová netriviální metoda, která umožňuje automatickou kalibraci scény na základě detekce předdefinovaných objektů známé velikosti. Metoda rovněž implementuje výpočet prostorových souřadnic detekovaných objektů a určení roviny povrchu scény. Programová implementace zvládá kalibraci v řádu desítek až stovek milisekund při spotřebě paměti v jednotkách MB.

Implementovaný software bude po drobných úpravách rozhraní začleněn do systému pro inteligentní extrakci vysokoúrovňové informace z obrazu a bude využit v rámci projektu WeKnowIt. V budoucnu by bylo vhodné významně rozšířit množinu detekovaných objektů, aby se snížila závislost fungování celého systému na přítomnosti úzké skupiny objektů v obraze. Přínosné by také bylo vyvinout funkci pro výběr nejlepších detekcí, která by odfiltrovala nedůvěryhodné či jinak nevhodné detekce. To by mohlo přispět k celkové vyšší stabilitě kalibrační metody. Za úvahu stojí i možnost explicitní práce s neurčitostí v rámci celého procesu odhadu parametrů objektů.

Literatura

[1] AGARWAL, S.; AWAN, A.; ROTH, D.: Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2004: s. 1475–1490.

[2] ALSABTI, K.; RANKA, S.; SINGH, V.: An efficient k-means clustering algorithm. In *Proceedings of IPPS/SPDP Workshop on High Performance Data Mining*, 1998.

[3] AVITZOUR, D.: Novel scene calibration procedure for video surveillance systems. *IEEE Transactions on Aerospace and Electronic Systems*, ročník 40, č. 3, 2004: s. 1105–1110.

[4] BAJCSY, R.; SOLINA, F.; GUPTA, A.: Segmentation versus object representation – are they separable? In *Analysis and interpretation of range images*, Springer-Verlag New York, Inc., 1989, str. 223.

[5] BALLARD, D.: Generalizing the Hough transform to detect arbitrary shapes. In *Readings in computer vision: issues, problems, principles, and paradigms*, Morgan Kaufmann Publishers Inc., 1987, s. 714–725.

[6] BAY, H.; ESS, A.; TUYTELAARS, T.; aj.: Speeded-Up Robust Features (SURF). *Computer vision and image understanding*, ročník 110, č. 3, 2008: s. 346–359.

[7] BELONGIE, S.; MALIK, J.; PUZICHA, J.: Shape Matching and Object Recognition Using Shape Contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, ročník 24, č. 4, 2002: str. 522.

[8] BISHOP, C.: *Pattern Recognition and Machine Learning*. Springer, 2006, ISBN 0387310738, 738 s.

[9] BRIECHLE, K.; HANEBECK, U.: Template matching using fast normalized cross correlation. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, ročník 4387, 2001, s. 95–102.

[10] COMANICIU, D.; MEER, P.: Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, ročník 24, č. 5, 2002: s. 603–619.

[11] COOTES, T.; EDWARDS, G.; TAYLOR, C.: Active Appearance Models. *Lecture notes in computer science*, 1998: s. 484–498.

[12] COOTES, T.; TAYLOR, C.; COOPER, D.; aj.: Active shape models: their training and application. *Computer vision and image understanding(Print)*, ročník 61, č. 1, 1995: s. 38–59.

[13] COXETER, H.; GREITZER, S.: *Geometry Revisited*. Washington D.C., USA: The Mathematical Association of America, 1967, ISBN 0883856190.

[14] CRIMINISI, A.: *Accurate visual metrology from single and multiple uncalibrated images*. Springer Verlag, 2001.

[15] CRIMINISI, A.; REID, I.; ZISSERMAN, A.: Single View Metrology. *International Journal of Computer Vision*, ročník 40, č. 2, 2000: s. 123–148.

[16] DHOND, U.; AGGARWAL, J.: Structure from Stereo – A Review. *IEEE Transactions on Systems, Man and Cybernetics*, ročník 19, č. 6, 1989: s. 1489–1510.

[17] DIXON, R.: *Mathographics*. Mineola, NY, USA: Dover Publications, February 1991, ISBN 0486266397.

[18] FREUND, Y.; SCHAPIRE, R.: A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, ročník 55, č. 1, 1997: s. 119–139.

[19] FREUND, Y.; SCHAPIRE, R.: A Short Introduction to Boosting. *Journal of Japanese Society for Artificial Intelligence*, ročník 14, č. 5, 1999: s. 771–780.

[20] FUSIELLO, A.; TRUCCO, E.; VERRI, A.: A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, ročník 12, č. 1, 2000: s. 16–22.

[21] GOSE, E.; JOHNSONBAUGH, R.; JOST, S.: *Pattern recognition and image analysis*. Prentice Hall, 1996, ISBN 0132364158.

[22] HAAR, A.: Zur theorie der orthogonalen funktionensysteme. *Mathematische Annalen*, ročník 69, č. 3, 1910: s. 331–371.

[23] HARRIS, C.: A combined corner and edge detector. In *Proc. 4th Alvey Vision Conf*, ročník 1988, 1988.

[24] HAYKIN, S.: *Neural Networks: A Comprehensive Foundation*. IEEE Computer Society, druhé vydání, 1999, ISBN 0780334949, 700 s.

[25] HOIEM, D.; EFROS, A.; HEBERT, M.: Geometric Context from a Single Image. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, IEEE Computer Society, 2005, str. 661.

[26] HOIEM, D.; EFROS, A.; HEBERT, M.: Closing the loop in scene interpretation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, s. 1–8.

[27] HRADIŠ, M.: Detekce v obraze a AdaBoost. 2009, [online], [cit. 2010-01-02]. URL `<https://www.fit.vutbr.cz/study/courses/POV/private/lectures/pov_03_detekce_objektu_-_AdaBoost.pdf>`

[28] HSU, R.; ABDEL-MOTTALEB, M.; JAIN, A.: Face Detection in Color Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, ročník 24, č. 5, 2002: s. 696–706.

[29] HUDSON, P.: One Point Perspective. [online], [cit. 2010-01-03]. URL `<http://www.ider.herts.ac.uk/school/courseware/graphics/one_point_perspective.html>`

[30] KANG, S.; SZELISKI, R.; CHAI, J.: Handling occlusions in dense multi-view stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, ročník 1, 2001.

[31] KIRKPATRICK, S.: Optimization by simulated annealing: Quantitative studies. *Journal of Statistical Physics*, ročník 34, č. 5, 1984: s. 975–986.

[32] KUBÍNEK, J.: *Detekce objektů v obraze*. Diplomová práce, Fakulta informačních technologií, Vysoké učení technické v Brně, 2009.

[33] KURITA, T.: An efficient agglomerative clustering algorithm using a heap. *Pattern Recognition*, ročník 24, č. 3, 1991: s. 205–209.

[34] LEIBE, B.; LEONARDIS, A.; SCHIELE, B.: Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision*, ročník 77, č. 1, 2008: s. 259–289.

[35] LIEBOWITZ, D.; ZISSERMAN, A.: Combining scene and auto-calibration constraints. In *The Proceedings of the Seventh IEEE International Conference on Computer Vision*, ročník 1, 1999.

[36] LOWE, D.: Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, ročník 60, č. 2, 2004: s. 91–110.

[37] MIKOLAJCZYK, K.; SCHMID, C.: Scale & Affine Invariant Interest Point Detectors. *International Journal of Computer Vision*, ročník 60, č. 1, 2004: s. 63–86.

[38] MIKOLAJCZYK, K.; TUYTELAARS, T.; SCHMID, C.; aj.: A comparison of affine region detectors. *International Journal of Computer Vision*, ročník 65, č. 1, 2005: s. 43–72.

[39] SPRINGER, C.: *Geometry and Analysis of Projective Spaces*. Freeman, 1964, ISBN 0716704234.

[40] STEGMANN, M. B.; FISKER, R.; ERSBOLL, B. K.; aj.: Active Appearance Models: Theory and Cases. In *Proceedings of the 9th Danish Conference on Pattern Recognition and Image Analysis*, 2000, s. 49–57.

[41] SVOBODA, P.: *Vyhledávání osob ve fotografii*. Diplomová práce, Fakulta informačních technologií, Vysoké učení technické v Brně, 2009.

[42] SZELISKI, R.; ZABIH, R.: An Experimental Comparison of Stereo Algorithms. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, Springer-Verlag, 1999, str. 19.

[43] VECERA, S.; O'REILLY, R.: Figure-ground organization and object recognition processes: An interactive account. *Journal of Experimental Psychology: Human Perception and Performance*, ročník 24, č. 2, 1998: s. 441–462.

[44] VIOLA, P.; JONES, M.: Rapid object detection using a boosted cascade of simple features. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, ročník 1, 2001.

[45] VIOLA, P.; JONES, M.; TIEU, K.; aj.: Robust Real-time Object Detection. In *18th International Conference on Pattern Recognition (ICPR'06)*, ročník 2, s. 1102–1105.

[46] EXIF.org – EXIF and related sources. [online], [cit. 2010-01-04].
URL `<http://www.exif.org/>`

[47] Epipolar geometry. [online], [cit. 2010-05-09].
URL `<http://en.wikipedia.org/wiki/Epipolar_geometry>`

[48] Procrustes Analysis. [online], [cit. 2010-05-05].
URL `<http://en.wikipedia.org/wiki/Procrustes_analysis>`

Příloha A

Obsah CD

- Technická zpráva ve formátu PDF.
- Zdrojový text technické zprávy pro LaTeX.
- Zdrojové texty programu pro automatickou kalibraci scény na základě detekce objektů známé velikosti.