VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHIC AND MULTIMEDIA

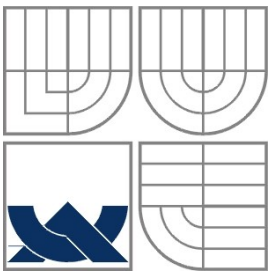
OVLÁDÁNÍ POČÍTAČE POHLEDEM

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

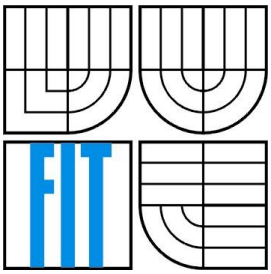
AUTOR PRÁCE
AUTHOR

MILOŠ CAHA

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHIC AND MULTIMEDIA

OVLÁDÁNÍ POČÍTAČE POHLEDEM

VIDEO-BASED HUMAN-COMPUTER INTERFACE

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE — MILOŠ CAHA
AUTHOR

VEDOUCÍ PRÁCE — ING. MICHAL ŠPANĚL
SUPERVISOR

BRNO 2008

Zadání bakalářské práce/6306/2007/xcaham01

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2007/2008

Zadání bakalářské práce

Řešitel: **Caha Miloš**

Obor: Informační technologie

Téma: **Ovládání počítače pohledem**

Kategorie: Počítačová grafika

Pokyny:

1. Prostudujte základy zpracování obrazu. Zaměřte se zejména na problematiku detekce lidských gest a směru pohledu.
2. Prostudujte dostupné materiály na téma ovládání počítače gesty, zaměřte se zejména na detekci směru pohledu.
3. Vyberte vhodné metody a navrhněte velmi jednoduché rozhraní pro ovládání počítače pohledem.
4. Experimentujte s vaší implementací a případně navrhněte vlastní modifikace metod.
5. Porovnejte dosažené výsledky a diskutujte možnosti budoucího vývoje.
6. Vytvořte stručný plakát prezentující vaši bakalářskou práci, její cíle a výsledky.

Literatura:

- Dle pokynů vedoucího.

Při obhajobě semestrální části projektu je požadováno:

- Splnění prvních tří bodů zadaní.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Španěl Michal, Ing.**, UPGM FIT VUT

Datum zadání: 1. listopadu 2007

Datum odevzdání: 14. května 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. Pan

Jméno a příjmení: **Miloš Caha**
Id studenta: 79078
Bytem: Citonice 11, 671 01 Citonice
Narozen: 30. 06. 1986, Znojmo
(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
   bakalářská práce

Název VŠKP: Ovládání počítače pohledem
Vedoucí/školitel VŠKP: Španěl Michal, Ing.
Ústav: Ústav počítačové grafiky a multimédií
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:

tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☐ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

....

Nabyvatel Autor

Abstrakt

Bakalářská práce se zabývá metodami detekce směru pohledu a jejich následným využitím pro ovládání počítače. Shrnuje a popisuje nejpoužívanější metody pro jednotlivé fáze zmíněné detekce, zejména použití detektoru obličeje a konvolučních filtrů pro následné vyhledání významných bodů v obličeji. Práce je zaměřena na návrh a popis implementace aplikace demonstrující jednu z metod bezkontaktního ovládání počítače.

Klíčová slova

pohled, bezkontaktní ovládání počítače, detekce směru pohledu, detekce obličeje, významné body v obličeji, Gáborovy filtry, diskrétní 2D konvoluce

Abstract

A bachelor thesis deals with methods about a detection of direction of look and with theirs following applications for PCs control. The thesis summarises and describes the most widely used methods for individual phases of the mentioned detection. Especially, use of a face detector and of convolution filters for searching of significant points in the face is used. The work is concentrated on a design and a description of an implementation of the application which demonstrates the method of contactless PCs control.

Keywords

gaze, contactless PCs control, detection of direction of look, face detection, significant points in the face, Gabor filters, 2D discrete convolution

Citace

Miloš Caha: Ovládání počítače pohledem, bakalářská práce, Brno, FIT VUT v Brně, 2008

OVLÁDÁNÍ POČÍTAČE POHLEDEM

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Michala Španěla. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Miloš Caha
12. května 2008

Poděkování

Rád bych poděkoval vedoucímu bakalářské práce Ing. Michalu Španělovi za podmětné rady a odborné vedení při zpracování této práce.

Obsah

1 Úvod

V současné době jsou počítače ovládány především pomocí klávesnice a myši, pomineme-li další specializované periferie jako jsou herní ovladače, je to téměř jediný způsob, jak s počítačem pracovat. Navíc jak klávesnice tak i myš jsou uzpůsobeny pro ovládání za pomocí rukou, proto se nabízí hledání dalších způsobů jak ovládat počítač. Zejména pro osoby s těžkým ochrnutím je vhodné bezkontaktní ovládání pomocí řeči nebo pohledu. Tato práce je zaměřena právě na zmíněné ovládání pomocí pohledu.

Cílem mé práce je vyzkoušet možnosti tohoto způsobu ovládání počítače a zhodnotit realizovatelnost jednotlivých metod určení směru pohledu použitelných v této oblasti. Dále pak na jednoduché aplikaci vyzkoušet možnosti a použitelnost ovládání počítače pohledem, například pro posun textu při jeho čtení.

Samotná práce je složena z několika kapitol, přičemž druhá a třetí se zabývá teoretickou problematikou celé práce. Dále je zde rozebráno několik různých metod používaných při detekci směru pohledu a na základě jejich vlastností je vybrána jedna, kterou se zabývám ve zbývající části práce.

Kapitola 4 popisuje Návrh rozhraní pro ovládání počítače pohledem. Podstatná část této kapitoly je věnována hledání vhodných filtrů pro detekci jednotlivých bodů v obličeji.

Kapitola 5 je zaměřena na implementaci celého rozhraní. Je zde popsán postup celého výpočtu směru pohledu od vytvoření pomocných dat, jako jsou jednotlivé filtry, přes použití detektoru obličeje a aplikaci filtrů, až po konečný výpočet poměru vzdáleností mezi oblastmi v obličeji, který určuje směr pohledu.

V 6. kapitole jsou popsány omezení zvolené metody a její úspěšnost v jednotlivých situacích. Na základě těchto informací si pak může uživatel udělat vlastní představu o použitelnosti tohoto způsobu ovládání počítače.

Závěrečná kapitola se věnuje zhodnocení výsledků práce a jsou zde popsány i možnosti dalšího vývoje.

2 Základy zpracování obrazu

V této kapitole se pokusím vysvětlit některé pojmy ohledně barevných modelů, které jsou v práci použity. Dále pak vysvětlím základní informace o operacích, které jsou prováděny se vstupními snímky, jako je konvoluce pro aplikaci filtrů nebo eliminace nežádoucích efektů v obraze.

2.1 Barevné modely a jejich převody

Základním prvkem, který bude v této práci zpracováván, je rastrový snímek. Ten je složen z množiny elementárních bodů tzv. pixelů pravidelně uspořádaných do dvourozměrné matice. Každý z těchto bodů nese informace o jednotlivých částech celého obrazu, ať už jde o hodnotu jasu, barvy, nebo průhlednosti.

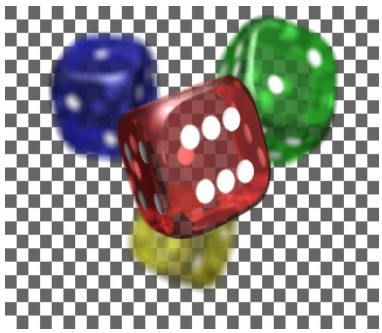
Jednotlivé barevné modely se od sebe výrazně liší, ať už kvalitou podání výsledného obrázku nebo množstvím dat, které o něm uchovávají. Právě tato informace je důležitá, protože čím více dat obsahují jednotlivé pixely tím jsou také náročnější výpočty prováděné s tímto obrázkem. Právě proto je vhodné zvážit, který model je pro určitý druh výpočtu nejvhodnější.

2.1.1 Barevné modely použité v této práci

2.1.1.1 Barevný

Jde o obrázek obsahující všechny důležité informace o obraze. Pro další zpracování se v této práci příliš nehodí, protože by zbytečně prodlužoval dobu výpočtu. Jediné části výpočtu kdy je použit, jsou vstup a informační výstup. Přičemž jsou zde použity dva modely a to RGB [1], který je složen ze tří základních barev červené, zelené a modré a RGBA, ten přidává navíc tzv. alfa kanál, určující průhlednost jednotlivých bodů.

Obr. 2.1: Obrázek s použitím palety RGB



Obr. 2.2: Ukázka použití alfa kanálu k nastavení průhlednosti objektů

Model RGBA je v práci téměř nepoužit, veškeré barevné obrázky, které jsou použity k výpočtu, jsou ve formátu RGB. Jediným využitím tohoto modelu je ukázka výstupu na formuláři, protože knihovna Qt standardně používá 32 bitový RGBA model.

2.1.1.2 Obrázek ve stupních šedi

Právě tento barevný model [2] je v celém programu využíván nejvíce. Jeho výhodou je, že minimalizuje objem dat potřebný k zapsání informací o hodnotě každého bodu, ale obrázek pořád dostatečně vypovídá o původním obsahu. Díky tomu se například výrazně urychlí konvoluce, která aplikuje jednotlivé filtry. Stejně tak detektory obličeje se za normálních okolností trénují za pomocí obrázků v odstínech šedé. Pro převod z barevného obrázku ve formátu RGB je možné použít vzorec :

$$Y = 0{,}3 \cdot R + 0{,}59 \cdot G + 0{,}11 \cdot B \tag{2.1}$$

Nelineárnost kódování je dána vlastnostmi lidského oka, které má různou citlivost na jednotlivé barvy, nejvyšší je na zelenou, naopak nejnižší je na modrou.

Obr. 2.3: Ukázka obrázku 2.1 po převedení na obrázek v odstínech šedi

2.1.1.3 Monochromatický

V tomto typu obrázku obsahují jednotlivé pixely nejjednodušší možnou informaci o barvě. Jejich hodnota je buď bílá(1) nebo černá(0). I přesto, že se zdá, že je tento barevný model téměř nepoužitelný, je v mé práci používán poměrně často a to zejména k úpravě výstupu jednotlivých filtrů, kde je možné, díky nastavenému prahu, určit, jaká intenzita odezvy filtru je už akceptovatelná.

Právě pomocí tohoto prahu, tedy funkcí nazývanou prahování, se vytváří monochromatický obraz. Vstupem této funkce je nejčastěji obrázek v odstínech šedé a její tvar je:

$$f(c)=\begin{Bmatrix}0\rightarrow c<treshold\\1\rightarrow c>treshold\end{Bmatrix} \tag{2.2}$$

Obr. 2.4: Ukázka obrázku 2.1 po převedení na monochromatický obrázek

2.2 Teorie obrazových filtrů

Vzhledem k tomu, že převážná část práce se zabývá nalezením určitých významných nebo jinak důležitých bodů v obličeji a to zejména pomocí různých filtrů, chtěl bych v této kapitole naznačit k čemu a jak je možné tyto filtry použít.

2.2.1 2D konvoluce

Spojitá konvoluce [7, 8] je jednou z matematických operací zpracovávající dvě funkce, značí se hvězdičkou a je definována vztahem:

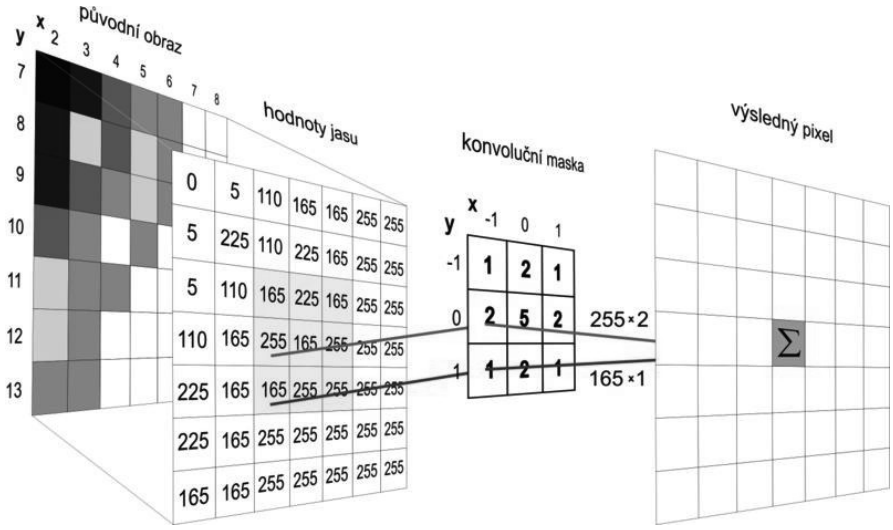
$$f(x)*h(x)=\int_{-\infty}^{\infty} f(x-\alpha)h(\alpha)d\,\alpha \tag{2.3}$$

Kde *f* je původní funkce a *h(α)* je konvoluční jádro. Vzhledem k tomu, že při změně tohoto jádra zůstává vždy zbytek rovnice stejný, používá se k definici pouze konvoluční jádro. V počítačové

grafice se používá pro zpracování obrazu diskrétní konvoluce a navíc v dvojrozměrném tvaru, výsledná rovnice pro dvojrozměrnou diskrétní konvoluci má pak tvar:

$$f(x,y)*h(x,y)=\sum_{i=-k}^{k}\sum_{j=-k}^{k} f(x-i,y-j)\cdot h(i,j) \tag{2.4}$$

Postup výpočtu diskrétní 2D konvoluce je pak následovný: Budeme uvažovat, že jádrem konvoluce je matice o rozměrech 3x3 a je naplněna hodnotami 1/9. Tuto matici přiložíme přes původní data z obrázku, jednotlivé položky nad sebou vynásobíme (bod na souřadnicích [1,1] z původního obrázku s bodem na souřadnicích [1,1] z konvolučního jádra), výsledky těchto součinů sečteme a uložíme na pozici, která je pod prostředním bodem konvolučního jádra. Výsledný efekt je takový, že jsme sečetli hodnoty pixelů ve čtverci 3x3 a jejich součet vydělili 9ti, tím jsme získali průměrnou hodnotu. Jestliže tento postup aplikujeme na celý obrázek. Bude každý pixel obsahovat hodnotu průměru sebe a okolních 8mi pixelů. Výsledkem je tak rozmazaný obrázek.



Obr. 2.5: Ukázka principu diskrétní dvourozměrné konvoluce

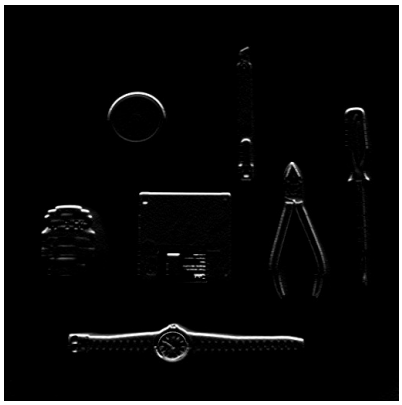
Samozřejmě je možné použít i jiný typ konvolučního jádra např. matice 5x5 naplněná hodnotami 1/25 způsobí taktéž rozmazání obrázku, ale ve větší míře. Obecně lze také říci, že čím větší matici použijeme tím je filtr méně náchylný na případný šum v obraze, samozřejmě to nemusí platit vždy.

Matici také můžeme naplnit jinými hodnotami, například nebudeme obrázek rozmazávat, ale zaostřovat, dále je možné vytvořit konvoluční jádra pro vyhledávání svislých nebo vodorovných hran. Následující obrázky ukazují, jak je možné pomocí jednoduché matice vyhledávat vodorovné hrany .

1	1	1
0	0	0
-1	-1	-1

Obr. 2.6: Matice pro detekci hran

Obr. 2.7: Vstupní obrázek detekce hran

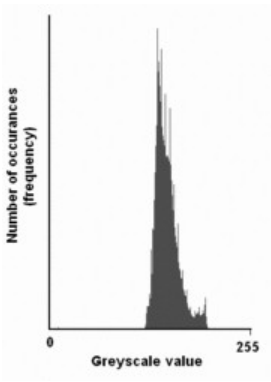


Obr. 2.8: Výstup po detekci hran

2.3 Vyrovnání osvětlení

K vyrovnání osvětlení, nejčastěji z důvodu přeexponování, nebo podexponování se používá tzv. ekvalizace histogramu [14]. Histogram je sloupový graf znázorňující množství jednotlivých barevných složek v obrázku. Následující obrázek je typickou ukázkou špatného kontrastu a navíc je mírně přeexponovaný:

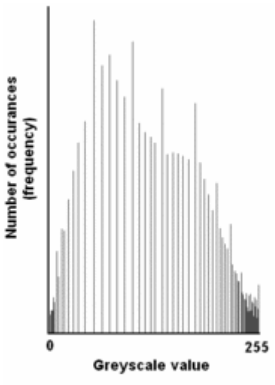
Obr. 2.9: Obrázek se špatným kontrastem



Obr. 2.10: Histogram k obrázku 2.9

To, že má obrázek nízký kontrast je patrné i z histogramu, který je úzký. Úpravu je možné provést pomocí zmíněné ekvalizace, která „roztáhne“ histogram přes celý rozsah barevné škály. V našem případě jde o odstíny šedé s intenzitou 0 až 255 .

Obr. 2.12: Obrázek 2.9 upravený pomocí ekvalizace histogramu



Obr. 2.11: Histogram k obrázku 2.12

3 Metody detekce směru pohledu

Základním předpokladem pro ovládání počítače je určení směru pohledu. Tuto problematiku je možné rozdělit na několik metod.

První z metod zjišťuje pouze směr pohledu v závislosti na postavení zorniček očí. K tomu je vhodné mít kameru umístěnou v blízkosti oka a to pokud možno tak, aby její pozice nebyla závislá na pozici nebo natočení hlavy. Nejčastěji jde o miniaturní kameru připevněnou na obroučkách speciálně upravených brýlí.

Další možností je, že budeme pouze sledovat pohyb hlavy a pohyb zorniček zanedbáme. Tato metoda je nejméně náročná na hardwarové vybavení. K určení směru pohledu stačí obyčejná webová kamera s rozlišením od 640x480. Takováto kamera už je součástí téměř všech novějších počítačů. Další výhodou tohoto řešení je možnost sledování dění na obrazovce během ovládací akce. Např. v případě ovládání myši pomocí pohybu oka je nutné dostatečně vychýlit oko ze středové polohy. To samozřejmě vede k tomu, že se osoba ovládající počítač dívá buď na okraj monitoru nebo dokonce až za jeho hranici. Proto je nutné po chvíli vrátit oko do středové polohy a zkontrolovat pozici kurzoru, pak následně provést případnou korekci posunu dalším vychýlením oka. Naopak při pohybu celé hlavy je možné očima neustále sledovat dění na obrazovce. A jakmile kurzor (nebo jiný ovládací prvek) dosáhne požadované pozice, je možné ovládací akci ukončit.

Dále je možné zkombinovat obě předchozí možnosti a to tak, že použijeme kameru s vysokým rozlišením a budeme zjišťovat směr natočení hlavy a také pozici zorniček. Zkombinováním těchto dvou informací můžeme zjistit směr pohledu bez nutnosti umístění kamery na hlavu. Na rozdíl od předchozí metody tato metoda umožňuje zjištění skutečného směru pohledu, kdežto předešlá metoda zjišťuje pouze zdánlivý směr, kterým by se osoba dívala v případě, že by šlo o strnulý pohled kolmý na rovinu obličeje.

Metod pro určení směru pohledu je celá řada, předchozí tři metody pracovaly na principu sledování osoby. Existuje, ale i obrácený způsob a to takový, že na hlavě uživatele je upevněna kamera. Ta ale tentokrát nezabírá oko ani obličej, ale je natočená na opačnou stranu, tedy směrem pohledu. Poté se určuje pohyb hlavy pomocí pohybu obrazu ve videosekvenci. Přičemž pro zjištění směru pohybu ve videosekvenci je možné opět použít řadu různých metod, takže by se tento způsob detekce směru pohledu dal dělit dále. To už ale přesahuje rozsah této kapitoly.

Ovládání počítače pohledem je většinou používáno pro handicapované občany, kteří nejsou schopni pohybovat končetinami, v takovém případě je nejvhodnější první nebo třetí metoda, zde totiž nejsou na obsluhu programu kladeny takové požadavky ohledně rozsahu pohybu. Jedinou částí těla, kterou pak musí být uživatel schopen pohnout, jsou oči. Na druhou stranu druhá metoda mi připadá

uživatelsky pohodlnější a co se týče implementace, není tak náročná na přesnost detekce jednotlivých bodů nebo oblastí. To byl také důvod, proč jsem se rozhodl pro realizaci tohoto způsobu ovládání počítače. Poslední metodu s obráceným postupem jsem uvedl spíše pro zajímavost, i když její realizace by měla patřit k těm nejjednodušším. Obzvláště to platí v případě, kdy by byla detekce pohybu zjednodušená prostředím např. tmavá místnost, kde v záběru kamery je pouze několik extrémně jasných bodů, které navíc nemění svoji vzájemnou pozici.

3.1 Detekce obličeje

V následujících kapitolách se budu také věnovat detekci významných bodů v obličeji. K tomu aby se nám podařilo najít tyto body je vhodné nejdříve omezit oblast ve které tyto body budeme hledat. Jako nejideálnější se zde jeví detekce obličeje, díky níž omezíme prohledávanou oblast prakticky na minimum.

Metod detekce obličeje je celá řada od porovnávání barevného spektra, přes detekci pomocí filtrů nebo detektorů až po vyhledávání pomocí informací z termokamery. Já se zde budu zabývat pouze detekcí pomocí natrénovaného detektoru.

Jde o metodu která je relativně rychlá a kvalitní. Základem úspěchu je kvalitní natrénování detektoru. V našem případě jde o detektor založený na algoritmu AdaBoost, který je natrénovaný pomocí tzv. Haarových příznaků. Trénování takovéhoto detektoru probíhá v několika fázích v první fázi se detektor učí na pozitivním vzorku dat. Dále se pokračuje učením na negativním vzorku. Celý proces je ukončen testem na dalších dvou vzorcích opět jde o pozitivní a negativní vzorek. Výsledkem je pak detektor, jehož vstupem jsou výřezy vstupního snímku o velikosti obrázků, které byly použity pro učení, a výstupem je informace jestli se v daném výřezu vyskytuje obličej nebo ne.

Jelikož je tato práce zaměřená na ovládání počítače pohledem a tedy i na detekci směru pohledu. Využil jsem už natrénovaný detektor obličeje a v práci jsem se zaměřil spíše na detekci samotného pohledu. Stejně tak detailnější popis funkce algoritmu AdaBoost nebo Haarových příznaků přesahuje rámec této kapitoly více viz [3, 4].

3.2 Detekce významných bodů v obličeji

Pro výpočet směru pohledu jsou zapotřebí nějaká vstupní data, nejlépe jednoznačné body, na základě jejichž vzájemné pozice je možné určit, kam se osoba dívá. Zpočátku mi připadaly jako ideální body středu očí a špičky nosu. Po dalších úvahách a testování se ukázalo, že tyto tři body nejsou dostačující, protože se do poměrů vzdáleností výrazně promítala vzdálenost osoby od kamery. Proto

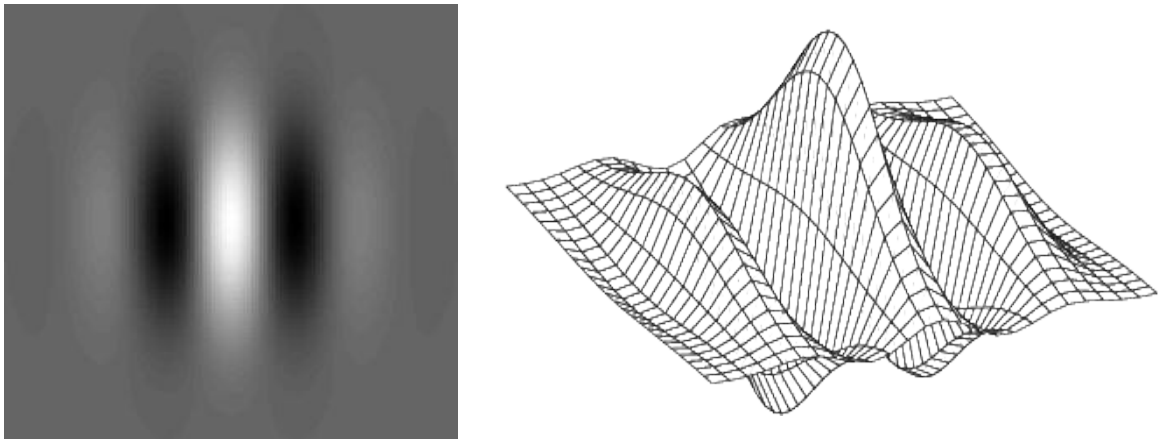
jsem přidal do výpočtů další bod a to střed úst. Díky tomu je možné vypočítat vzdálenost oči – ústa a nos – ústa. Poměrem těchto vzdáleností získáme informaci o tom, jestli se uživatel dívá nahoru, dolu nebo přímo před sebe. Protože veškeré výpočty probíhají pomocí poměrů, není výsledek závislý na vzdálenosti uživatele od kamery, ale na úhlu mezi kolmicí na objektiv kamery a kolmicí na rovinu plochy obličeje.

Další body, které by se daly použít pro určení směru pohledu, jsou obočí, brada případně další body po obvodu obličeje. Pro dostatečně kvalitní určení směru pohledu a zároveň i pro účely této práce bohatě postačí čtyři výše zmíněné body.

3.2.1 Gáborovy filtry

Použití Gáborových filtrů [5] je jednou z možností jak detekovat významné body v obličeji jako jsou oči, ústa nebo nos. Jejich využití je mnohem širší, používají se například při zpracování otisků prstů nebo pro detekci obrysů objektů v obraze.

V této kapitole bych se chtěl věnovat jejich matematickému základu a jejich vlastnostem v závislosti na parametrech filtru. Dále také doporučeným mezím a výchozím hodnotám těchto parametrů.



Obr. 3.1: Ukázka Gáborova filtru

Na výše uvedených obrázcích je ukázka vzhledu Gáborova filtru. Obrázek vlevo znázorňuje hodnoty funkce převedené do barevného spektra, kde nejjasnější bílá barva odpovídá maximální hodnotě funkce. V základním tvaru rovnice je to hodnota 1, tato hodnota odpovídá nejvyššímu bodu v grafu na druhém obrázku.

Tento filtr se skládá ze dvou funkcí a to z Gaussovy křivky, která má jako vstup hodnoty os x a y, je to ta část, která vytváří jednotlivé "kopcovité" oblasti funkce. Druhou funkcí je kosinus, jehož vstupem jsou pouze hodnoty z osy x. Tato funkce způsobuje "rozkmitání" Gaussovy křivky.

Spojením těchto dvou funkcí dostaneme výsledný vzorec. Aby bylo možné filtr natočit o určitý úhel, je základní rovnice ještě doplněna o dvě další, které přepočítávají hodnoty os *x* a *y*.
Gáborův filtr je tedy definován těmito vzorci:

$$g(x,y)=\exp\left(\frac{-x'^2+\gamma^2 y'^2}{2\sigma^2}\right)\cdot\cos\left(2\pi\frac{x'}{\lambda}+\Psi\right) \tag{3.1}$$

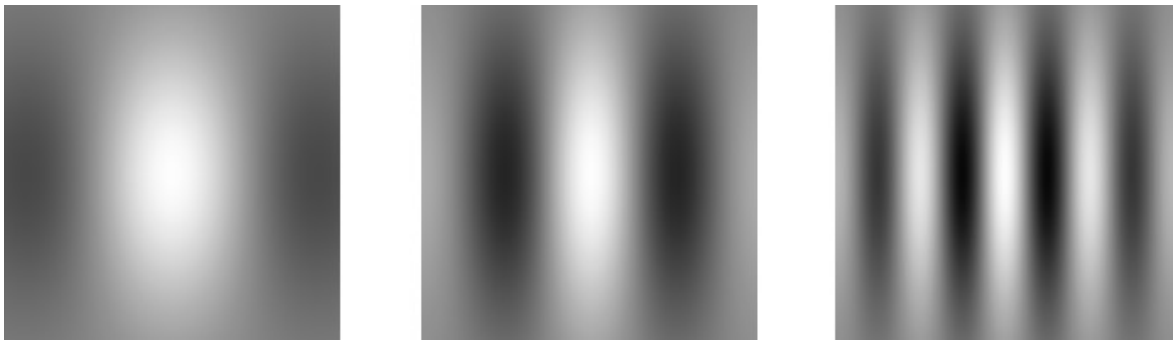
$$x'=x\cdot\cos(\theta)+y\cdot\sin(\theta) \tag{3.2}$$

$$y'=-x\cdot\sin(\theta)+y\cdot\cos(\theta) \tag{3.3}$$

Aby funkce nebyla závislá na velikosti konvolučního jádra, budeme ji používat tak, že osy *x* a *y* mají rozsah<-1,1>, funkce pak taktéž nabývá hodnot v rozsahu <-1,1>. Tento předpoklad nám později umožní přepočítat daný filtr na jakoukoli velikost konvoluční matice bez dalších úprav vlastností funkce.

3.2.1.1 λ(lambda) – Vlnová délka

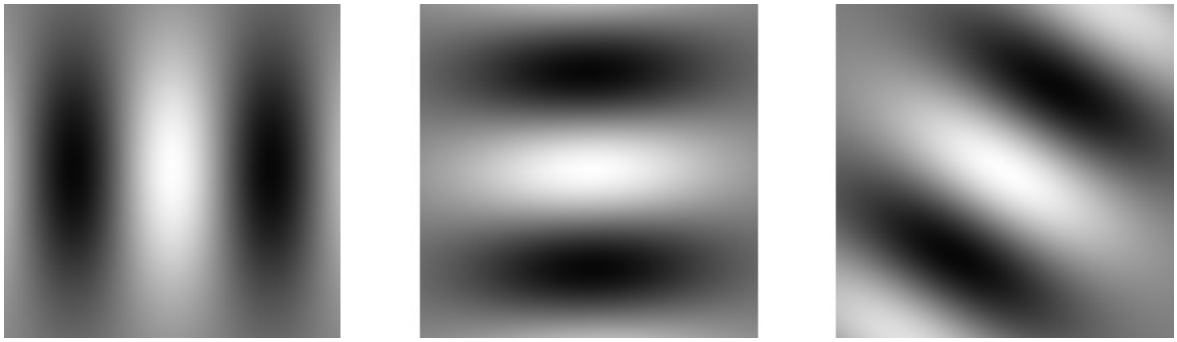
Určuje délku periody funkce kosinus. V našem případě jde o kladné desetinné číslo v rozsahu <0, 4>, přičemž mezi hodnotami 2 a 4 už funkce kosinus ve vytváření filtru ztrácí roli a filtr se začíná blížit tvaru Gaussovy křivky. Při nastavování tohoto parametru je zapotřebí brát ohled na velikost konvolučního jádra, pokud by byl rozměr jádra příliš malý vzhledem k zvolené vlnové délce, mohlo by dojít k velikému zkreslení filtru, což by mělo značný vliv na jeho funkčnost. Například při velikosti jádra 10x10 a vlnové délce 0.2 je počet vln tak vysoký, že není možné do takto malého jádra filtr zaznamenat tak, aby v něm byla obsažena informace o všech vlnách.



Obr. 3.2: Gáborův filtr v závislosti na parametru λ

3.2.1.2 θ(theta) – orientace filtru

Kromě hodnot os *x* a *y* je to jediný parametr rovnice (3.2) a (3.3), určuje natočení filtru ve směru proti pohybu hodinových ručiček. Natočení v rozsahu 0 – 360° odpovídá rozsah hodnot <0 – 2π>. Na následujícím obrázku jsou ukázky filtru natočeného pomocí hodnot 0, π/2, a 1.

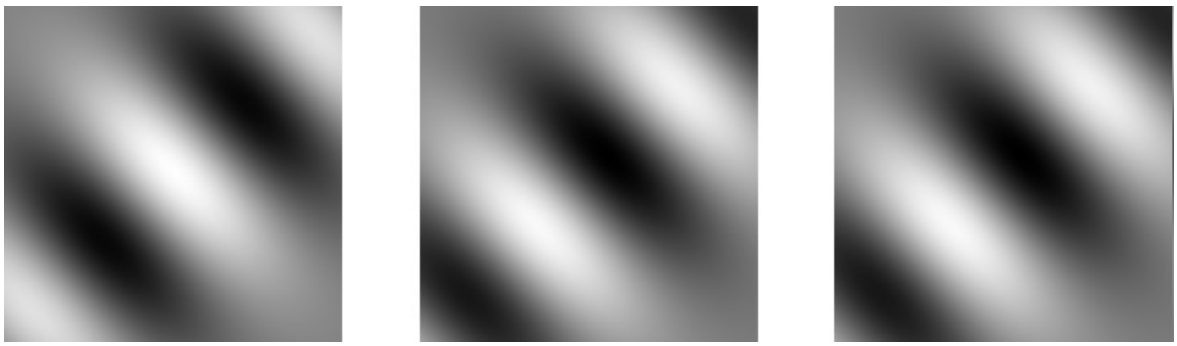


Obr. 3.3: Gáborův filtr v závislosti na parametru θ

3.2.1.3 ψ(psi) nebo φ(fi) – fázový posun

Určuje fázový posun funkce, ovlivňuje tedy kosinovou část rovnice. Fázový posun se nastavuje v rozsahu -180° - +180° respektive 0° - 360°. Za předpokladu, že filtr je v určitém směru souměrný, je možné tento rozsah omezit na 0°-180° a případný větší fázový posun kompenzovat změněním rotace filtru o +π nebo – π. Podobně je možné omezit rotaci s tím, že nebudeme omezovat fázový posun. Tento postup je výhodný zejména tehdy, pokud hledáme vhodný filtr pomocí nastavování všech kombinací parametrů, pak nám umožní zredukovat počet procházených filtrů na polovinu (přesněji se vyřadí duplicitní filtry). Fázovému posunu v rozsahu 0° - 180° pak odpovídají hodnoty parametrů <0 - π>.

Na obrázku 3.4 je zleva: filtr s rotací 1,25 a fázovým posunem 0, další filtr má rotaci 1,25π a fázový posun 1,25π, poslední filtr ukazuje, že je možné docílit stejného výsledku jako v předchozím případě, i když fázový posun nepřesáhne hodnotu π. Má rotaci 0,25π a fázový posun 0,75π.

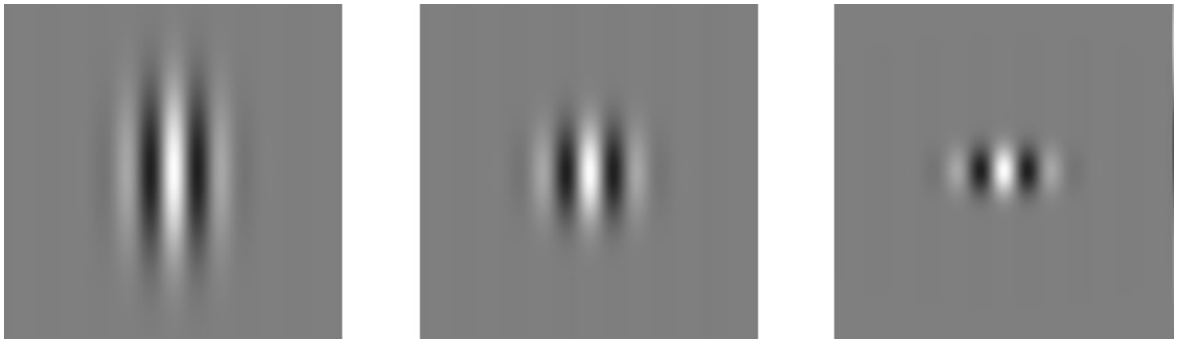


Obr. 3.4: Gáborův filtr v závislosti na parametru ψ

3.2.1.4 γ(gama) – prostorový poměr

Jde o parametr Gaussovy funkce. Jeho hodnota určuje eliptičnost celého filtru. Při hodnotě 1 se jedná o kružnici, hodnota menší než 1 způsobí protáhnutí pruhů funkce, naopak hodnota větší než jedna způsobí protažení v opačném směru. Implicitní hodnota je 0,5.

Na obrázku 3.5 je ukázka filtru s prostorovým poměrem 0,5; 1 a 2.

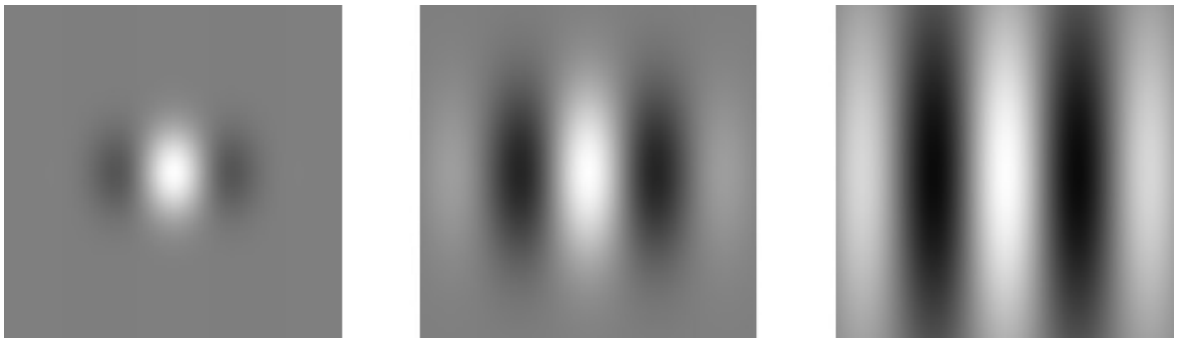


Obr. 3.5: Gáborův filtr v závislosti na parametru γ

3.2.1.5 σ(sigma) – velikost

Stejně jako γ je to parametr Gaussovy funkce, na rozdíl od předchozího parametru má ale vliv na oba rozměry Gaussovy funkce. Tzn. neurčuje její eliptičnost, ale její velikost. Implicitní hodnotu tohoto parametru nelze určit, protože závisí na vlnové délce (λ) a prostorovém poměru (γ). Někdy se proto místo tohoto parametru používá šířka pásma (*b*) [6], díky které se pak σ dopočítá na základě vlnové délky a prostorového poměru pomocí následujícího vzorce:

$$b = \log_2 \frac{\frac{\sigma}{\lambda}\pi + \sqrt{\frac{ln2}{2}}}{\frac{\sigma}{\lambda}\pi - \sqrt{\frac{ln2}{2}}} \qquad \frac{\sigma}{\lambda} = \frac{1}{\pi}\sqrt{\frac{ln2}{2}} \cdot \frac{2^b + 1}{2^b - 1} \tag{3.4}$$



Obr. 3.6: Gáborův filtr v závislosti na parametru σ

3.2.2 Další metody detekce bodů v obličeji

- Velice účinnou metodou je použití detektoru natrénovaného na jednotlivé části obličeje, které potřebujeme hledat. V takovém případě stojí za zvážení, jestli je nutné navíc používat detektor obličeje, protože výstup takovéhoto pro jednotlivé body by mohl být natolik kvalitní, že bodů mimo obličej by bylo naprosté minimum. Co se týče náročnosti na implementaci a výslednou kvalitu výstupu, měl by být srovnatelný s Gáborovými filtry, to ale pouze v případě, že Gáborovy filtry budou použity pouze na oblast obličeje nikoliv na celý snímek z kamery. Jelikož jsem se rozhodl, že budu využívat detektoru obličeje a nechtěl jsem, aby celý program byl založen pouze na detektorech, rozhodl jsem, že dalším stupněm detekce budou Gáborovy filtry, které byly popsány v předchozí kapitole.

- Dále je možné využít barevného spektra obličeje, kde kůže má ve všech oblastech přibližně stejnou barvu. Na rozdíl od očí, kde se vyskytuje černá a bílá barva nebo úst, kde je barva červená. V takovém případě se určí jednotlivé barevné podmnožiny pro každou z hledaných oblastí z celého barevného rozsahu RGB a navíc se jednotlivým barvám z podmnožin přiřadí hodnota, která reprezentuje pravděpodobnost použití konkrétní barvy v konkrétní oblasti. Při detekci se pak prochází celý obrázek a jednotlivým pixelům se přiřazují hodnoty získané z dříve nadefinovaných podmnožin. Výsledkem je pak matice nebo obrázek ve stupních šedi (případně více matic či obrázků podle toho kolik druhů oblastí hledáme), kde na každém bodu je hodnota reprezentující pravděpodobnost, že se na daném místě vyskytuje hledaný významný bod obličeje. Tuto matici je dále možné zpracovávat buď tak, že se bude brát v potaz několik bodů s nejvyšší hodnotou nebo se provede prahování, a pak se pracuje s celými oblastmi bodů.

  Tuto metodu jsem nepovažoval za vhodnou z důvodu veliké paměťové náročnosti. Je zde totiž zapotřebí mít v paměti uloženy informace o pravděpodobnosti výskytu jednotlivých barev. Při použití barevného modelu RGB, kde každá z barev je reprezentována 8mi bity a hodnota pravděpodobnosti bude uložena ve formátu double (32b) je výsledná velikost množiny 64 MB. A za předpokladu, že těchto množin bude zapotřebí více (minimálně budou zapotřebí 3), je množství potřebné paměti značně větší než u ostatních metod. Nehledě na to, že oproti metodám, kde se používají filtry, je zapotřebí barevný obrázek.

  Když nebudeme brát v potaz paměťové nároky, které jsou pro dnešní počítače únosné, zůstává zde ještě pořád minimálně jeden problém a to náročnost na kvalitu vstupu. Protože metoda je založená na porovnávání barev, je nutné, aby kamera tyto barvy v průběhu záznamu nijak nezkreslovala a navíc, aby se také nijak výrazně neměnilo osvětlení. Což

dnešní webové kamery neumožňují a bylo by potřeba využívat jako zdroje jinou velmi kvalitní kameru a to by tuto práci posouvalo do trochu jiné oblasti.

- Další možností, která je samostatně nepoužitelná, je sledování optického toku. Tuto metodu je zapotřebí zkombinovat s některou z předchozích. Jde o postup, který slouží spíše k urychlení detekce než k ní samotné. Např. budeme body detekovat na každém třetím snímku a na zbylých dvou snímcích body dopočítáme právě pomocí optického toku. Vzhledem k tomu, že výpočet optického toku je mnohem rychlejší než celá detekce bodů, dojte téměř k trojnásobnému zrychlení.

4 Návrh rozhraní pro ovládání počítače

Celý problém návrhu rozhraní jsem si rozdělil na dvě části. V první jsem si vytvářel podmínky pro hlavní část práce. Šlo převážně o hledání různých filtrů, ať už pro detekci jednotlivých významných bodů v obraze nebo pro úpravy obrazu tak, abych eliminoval nežádoucí zkreslení obrazu, jako je například nerovnoměrné osvětlení nebo šum v obraze. Z výsledků této části práce jsem měl určitou představu o tom, jak silné jsou jednotlivé metody a do jaké míry je bude zapotřebí podpořit jinými metodami nebo postupy, aby výstup jádra programu byl použitelný pro volání jednotlivých akcí pro ovládání počítače.

V druhé části jsem se věnoval samotnému problému detekce směru pohledu, protože právě tato detekce je klíčovým bodem celé práce. Zde už šlo čistě jenom o to vytvořit prostředí, které by umělo využít informací, které jsem získal v první části vývoje. V této části už se nalezené filtry pouze použily, hlavním problémem bylo spíše zpracování jejich výstupu, kde bylo zapotřebí vyřadit špatně nalezené oblasti. Dalším krokem je pak vypočtení směru pohledu z nalezených bodů. Tato informace je pak předána grafickému rozhraní, které podle ní provádí určitou akci, v tomto případě jde o posun (scrollování) textu.

4.1 Vyrovnání bočního osvětlení

Ekvalizace histogramu je vhodná, hlavně pokud je celý obrázek nasvícen stejně, já jsem potřeboval i možnost vyrovnání bočního osvětlení. Pro úpravu osvětlení jsem pak zvolil tento postup:

Vytvořil jsem si kopii původního obrázku ve stupních šedi, tuto kopii jsem velmi silně rozmazal. Výsledkem byl obrázek (spíše mapa), který znázorňoval, která místa v obrázku jsou světlá a která tmavá.

Na základě těchto dat jsem pak upravil hodnoty jednotlivých pixelů v původním obrázku. A to tak, že ty body, jimž odpovídala v jasové mapě hodnota vyšší než polovina maximálního použitelného jasu (z 255ti je to 127), jsem patřičně ztmavil, naopak tmavší body jsem zesvětlil.

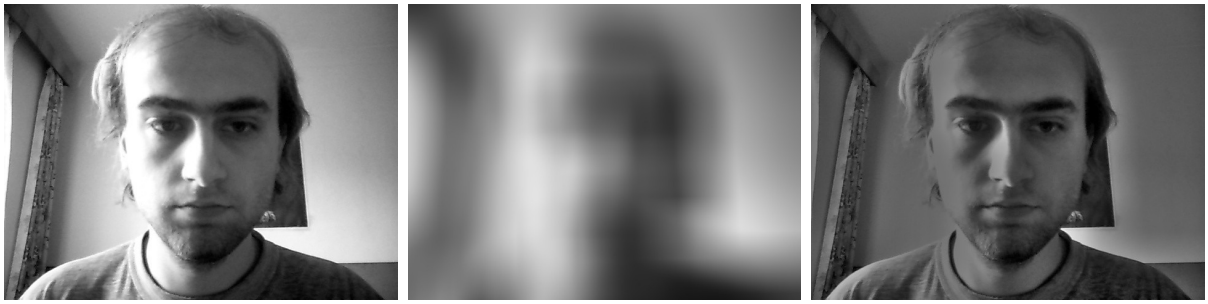
Původně jsem pro tento účel používal lineární funkci tzn. čím byl bod tmavší tím více byl zesvětlen a naopak. Tento postup se ukázal jako nepoužitelný, protože extrémně zvyšoval hladinu šumu v tmavých oblastech. Zvýrazněny nebyly jenom drobné šumové body, ale celé plochy i o rozměrech v řádech desítek pixelů.

Místo lineární funkce pro změnu jasu jsem nakonec zvolil funkci logaritmickou (s jistými úpravami)Ta má tu výhodu, že redukuje provedenou změnu jasu v místech, kde je rozdíl příliš veliký. Například při zesvětlování extrémně tmavých míst se neprovede přičtení tak vysokého čísla, aby se

výrazně zvýšila hladina šumu. Jedinou výraznou nevýhodou tohoto řešení je, že snižuje kontrast výsledného obrázku. Použitá rovnice má tvar :

$$o(x, y) = i(x, y) * \log(1 + \frac{127}{m(x, y)}) \quad (4.1)$$

kde *o* je výstupní obrázek, *i* je vstupní obrázek, *m* je maska jasu vypočtená rozmazáním vstupního obrázku, *x* a *y* jsou souřadnice .



Obr. 4.1 Ukázka práce funkce pro normalizaci osvětlení zleva vstupní obrázek s bočním osvětlením, obrázek ukazující jas v jednotlivých částech snímku, upravený obrázek

Tato funkce byla nakonec nastavena tak, aby byla standardně vypnutá, protože konvoluční jádro pro rozmazání obrázku má velikost 101x101 a výpočet s takovýmto jádrem je poměrně náročný .

4.2 Použití detektoru pro nalezení obličeje

Abych snížil nároky na jednotlivé filtry, rozhodl jsem se použít detektor objektů natrénovaný na vyhledávání obličejů. Tento postup sníží nároky na filtry hned dvakrát. Zaprvé se zmenší počet bodů, které použitý filtr nalezne (hledá na menší ploše), navíc na menším obrázku probíhá výrazně rychleji celý výpočet filtru.

Těchto filtrů se nabízela celá řada, já jsem se rozhodl pro už hotový detektor z knihovny OpenCV. Jeho kvalita je na vysoké úrovni a časová náročnost při správném nastavení je srovnatelná nebo nižší než při použití filtrů přes celý obraz .

4.3 Vyhledání vhodných filtrů pro detekci

Podmínkou správné funkce mého programu, který je z velké části založen na používání obrazových filtrů, je právě jejich nalezení. Jelikož se jedná o konvoluční filtry, které budou použity při výpočtu pro ovládání počítače, potřeboval jsem najít takové filtry, které by se daly realizovat v aplikaci běžící v reálném čase. Když nebudu brát v úvahu různé optimalizace výpočtu konvoluce, je jediným způsobem jak aplikaci filtrů urychlit zmenšení jejich konvolučního jádra, což právě u Gáborových

filtrů není jednoduchá záležitost. Na rozdíl od jiných filtrů např. pro rozmazávání jsou Gáborovy filtry použitelné až při použití podstatně větších matic (většinou jde o velikosti větší než 15x15).

Standardně se také pro detekci jednotlivých oblastí nepoužívá pouze jeden filtr, ale sada několika filtrů (asi 5 až 10). I přesto jsem se snažil najít vždy jeden filtr pro danou oblast, který by byl použitelný, protože s každým dalším filtrem zbytečně narůstá doba ke zpracování obrazu. Je třeba si totiž uvědomit, že i když se spokojíme 5ti snímky za vteřinu, je čas, který máme k dispozici na výpočet, pouhých 200ms. Samozřejmě není možné nalézt takový filtr, který by byl naprosto univerzální pro všechny osoby a navíc nedetekoval žádné oblasti navíc.

Nakonec jsem se spokojil vždy s jedním filtrem, který má velmi malé množství nadbytečných oblastí s tím, že ty správné body se budou vypočítávat pomocí vzájemné pozice tak, aby byla reálná. Navíc v případě, že by i tento postup nebyl schopný dostatečně přesně rozpoznat, které oblasti jsou ty správné, je možné počet filtrů pro konkrétní hledané body zvýšit až na tři. Pokud budou filtry zvoleny tak, aby se doplňovali a nikoli aby každý další filtr potvrdil všechny oblasti předchozího, jen nepatrně pozměnil jejich velikost, tak je tento počet dostačující. Navíc i při tomto zvýšení počtu filtrů není prodloužení výpočtu tak vysoké.

4.3.1 Výběr vhodné databáze pro nalezení filtrů

K tomu, abych mohl najít dostatečně kvalitní a funkční filtry pro detekci jednotlivých částí obličeje, jsem si potřeboval opatřit poměrně velikou databázi obličejů, na které by bylo možné tyto filtry testovat. Samozřejmě k tomuto účelu nelze použít jakoukoli databázi, je zapotřebí, aby databáze obsahovala jednak samotné obrázky, ale také záznamy s pozicemi a velikostí jednotlivých bodů. Dále je zapotřebí, aby obrázky, které bude databáze obsahovat, měly alespoň přibližně stejnou velikost, jakou bude mít snímek z kamery, kterou se bude nahrávat ovládací video. Další podmínkou je pak, aby obličeje byly dostatečně rozmanité tzn. aby měly různý tvar, barvu pleti, barvu a styl účesu, vousy apod. Dále by bylo vhodné, aby se v databázi vyskytovaly snímky od téže osoby s různým natočením hlavy.

Na internetu je těchto databází k dispozici celá řada, i přesto bylo nalezení vyhovující sady obrázků poměrně velikým problémem. Většina databází, které jsou k dispozici, je totiž vytvořená pro trénování detektorů obličeje a tudíž neobsahují žádné informace o pozici očí, nosu a úst. Naopak databáze pro trénování detekcí oblastí jako jsou oči, sice tyto informace obsahují, ale neobsahují obličeje natočené do různých směrů. Nakonec se jako nejvhodnější jevila databáze IMM Face Database od společnosti Active Appearance Models [9]. Zpočátku jsem se obával toho, že tato databáze bude příliš malá, protože obsahuje pouze 240 obrázků a pro trénování detektorů se používají sady 1 - 5ti tisíc obrázků. Ukázalo se, že pro nalezení vhodných filtrů je tato databáze dostačující.

Dále jsem uvažoval i o použití sady obrázků, které by neobsahovaly žádný obličej, ale nějaké složitější pozadí například místnosti s různým nábytkem. Pomocí této sady jsem chtěl zjišťovat odezvu na nesprávné objekty. Nakonec jsem od tohoto záměru upustil. Protože pro nalezení optimálních filtrů by to mělo spíše negativní důsledky. Navíc jsem se rozhodl pro detekci obličeje a následnou detekci bodů v obličeji, takže by se vliv vlastností filtrů na jiné objekty téměř neprojevil.

4.3.2 Zjištění kvality filtru

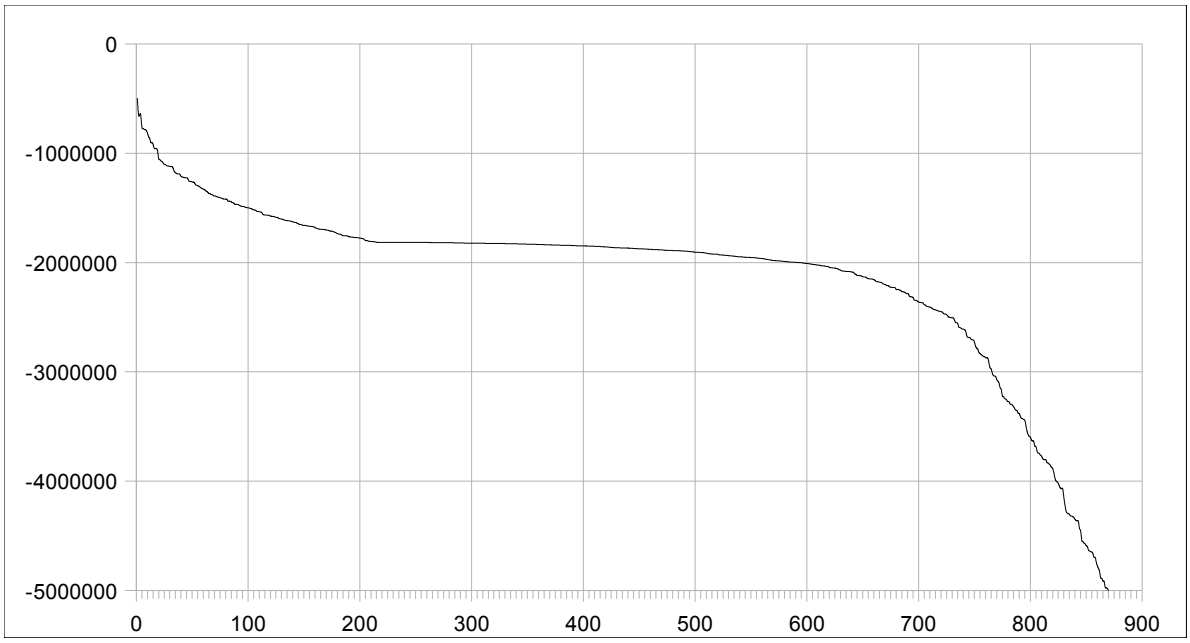
Aby bylo možné zjistit, který filtr je či není účinný, případně do jaké míry je daný filtr použitelný, musel jsem si vytvořit funkci, která porovná výstup konvoluce s požadovanými hodnotami z databáze a to tak, že výsledný výstup má obsahovat co nejvíce bodů v oblasti, kde se nachází hledaná část obličeje např. oči, ale také co nejméně bodů v ostatních oblastech. Přičemž to, že se některý z bodů nalézá v požadované oblasti, je několikanásobně důležitější, než to, že se mimo tuto oblast nachází málo bodů.

Podmínkou opravdu dobré detekce této oblasti i při různém natočení hlavy je, aby se v okolí výsledné oblasti pokud možno nevyskytoval žádný další bod (nebo shluk bodů). Toho jsem docílil za pomocí jakéhosi "hodnocení" které dostávají jednotlivé filtry a to tak, že za každý bod, který je uvnitř hledané oblasti se hodnocení zvedne o hodnotu 10, naopak za místa uvnitř hledané oblasti, ve kterých neleží žádný bod, se hodnocení sníží o -8. Dále body ležící v těsném pásmu kolem hledané oblasti taktéž sníží hodnocení o -8. To, že se v místě mimo detekovanou oblast nevyskytuje žádný bod, hodnocení nijak neovlivňuje, naopak všechny ostatní body, které leží mimo hledanou oblast, snižují hodnocení o -0,5. Hodnoty, které ovlivňují hodnocení filtru, byly zjištěny experimentálně a nejsou pro nalezení vodných parametrů nijak závazné, je velmi pravděpodobné, že lze nalézt kombinaci hodnot, která bude dosahovat lepších výsledků. Na druhou stranu je nutné si uvědomit, že hledání těchto filtrů probíhá na určitém omezeném vzorku a osoba, která bude nakonec snímána kamerou, může mít výrazně odlišné vlastnosti obličeje. Proto je vhodné parametry filtru pro danou osobu vždy mírně poupravit.

4.3.3 Samotné vyhledání filtrů

Vyhledávání nejvhodnějších filtrů pro detekci očí, nosu a úst jsem si rozdělil do jednotlivých částí a to podle toho, k detekci jaké části obličeje byl používán. Protože tyto filtry jsou značně veliké a výpočet konvoluce je také náročný, bylo nutné omezit počet procházených filtrů na co nejnižší číslo. Jako dostačujíc z hlediska doby zpracování a rozsahu hodnot se nakonec ukázalo množství přibližně 1000 filtrů. Tohoto množství je možné dosáhnout patřičným omezením rozsahů jednotlivých parametrů. Zejména určité natočení a fázové posuny nemají ve vyhledávání účinného filtru naprosto

žádný význam. Abych tyto rozsahy našel, spustil jsem vyhledávací skript pouze na jednom vzorovém obličeji, kde jsem zjistil vlastnosti jednotlivých parametrů Gáborovy funkce. Z těchto vlastností jsem posléze odvodil rozsah a krok jednotlivých parametrů. Následně byly tyto hodnoty použity do cyklů pro nalezení vhodného filtru za použití všech obrázků.



Obr. 4.2: Graf úspěšnosti jednotlivých filtrů

Z výše uvedeného grafu je patrné, že z 900 filtrů je významně kvalitních asi 20 – 50 dalších asi 700 filtrů má přibližně stejné vlastnosti, jejich kvalita je ale podstatně nižší. Ostatních přibližně 200 filtrů je nepoužitelných. Jde o kombinace parametrů, které vytvoří filtr, jehož výstupem nejsou téměř žádné body nebo naopak je výstupem téměř celá plocha snímku.

Tento výsledek, zatím nelze považovat za použitelný, jde pouze o hrubé roztřídění filtrů za pomocí poměrně jednoduchého programu. Abych našel nejvhodnější filtr, bylo zapotřebí ručně projít asi 20 – 30 nejlepších filtrů a otestovat je na videosekvenci. Zde jsem zjišťoval stabilitu filtru na různém stupni osvětlení, dále na změny vzdálenosti pozorovaného objektu od kamery a také na natočení. Zejména jsem se zaměřil na počet bodů v okolí hledané oblasti. Také tvar a vzdálenost jednotlivých nalezených oblastí byla důležitá, protože oblasti, které jsou příliš široké a navíc uprostřed zploštělé, jsou náchylné k tomu, že se při větším úhlu otočení hlavy rozdělí na více oblastí. Tento jev je pak velice problematické odhalit při následném hledání správných bodů a dochází ke špatné detekci směru pohledu a to vede k neočekávanému chování. Další problém je v oblastech, které jsou velmi široké a zároveň jsou velmi blízko u sebe. Zde naopak místo rozdělení dochází ke spojení dvou oblastí do jedné. Tento problém je nejkritičtější u očí, kde se hledají dvě oblasti ve stejné

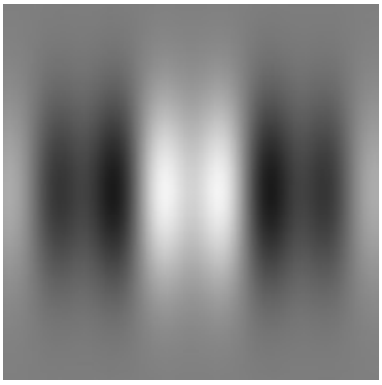
rovině u určitého rozmezí vzdáleností. V případě, že se tyto dvě oblasti spojí do jedné, není možné nalézt pozici očí a celý snímek je tím prohlášen za neplatný. Nebo naopak dojde k detekci obočí. Které má podobné vlastnosti jako oči, což má za následek změnu poměru vzdáleností mezi oči - ústa a ústa - nos.

4.3.4 Použité úpravy Gáborových filtrů

Jelikož jsem se při detekci některých částí obličeje (zejména u nosu) potýkal s problémy, jako je spojování jednotlivých nalezených oblastí ve veliké celky, byl jsem nucen, najít řešení, které tento problém eliminuje. Částečně se mi to podařilo díky úpravě rovnice (3.1) pro výpočet Gáborova filtru.

$$g(x, y) = \exp\left(\frac{-x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \cdot \left(\cos\left(2\pi \frac{x'}{\lambda} + \Psi\right) - \frac{1}{3} \cdot \cos\left(6\pi \frac{x'}{\lambda} + \Psi\right)\right) \tag{4.2}$$

Výsledkem je Gáborova funkce, ve které jsou jednotlivé vlny zdvojeny, vlnová délka však zůstává nezměněna, to má za důsledek strmější náběh a sestup jednotlivých vln. To v konečném důsledku zlepší chování detektoru asi o 5 až 10 %.



Obr. 4.3: Upravený Gáborův filtr

4.4 Výpočet směru pohledu z nalezených bodů

Jak již bylo zmíněno v kapitole 3, rozhodl jsem se pro metodu detekce směru pohledu, která využívá změny náklonu nebo natočení hlavy. Pokud omezíme pohyb hlavy na směr nahoru a dolu, je možné určit směr pohledu z poměru vzdáleností mezi ústa-oči a ústa-nos případně z poměru ústa-nos a nos-oči. Pro další použití je vhodnější první z nich, protože výstupem jsou čísla v rozsahu <0 - 1> kdežto

výstupem druhého je rozsah <0 - ∞>. V nastavení programu je pak jednodušší pracovat s rozsahem hodnot, který je omezen z obou stran konečnou hodnotou.

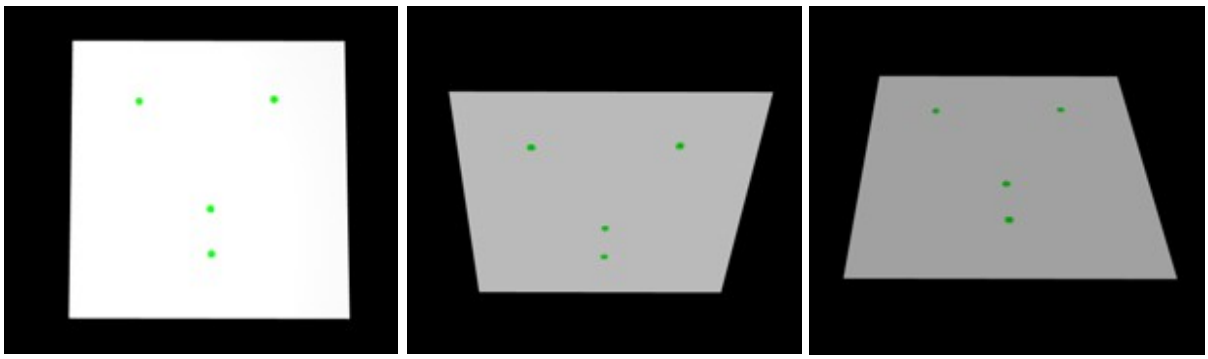
4.4.1 Perspektivní projekce

Jednou z možností jak zobrazit trojrozměrné těleso na dvojrozměrné ploše je promítání neboli projekce. Existují dva základní druhy projekcí: paralelní a perspektivní. V této práci budu využívat základů perspektivní projekce. Právě tento druh projekce je nejvíce podobný lidskému pohledu a záznamu kamery. Jestliže se budeme bavit o záznamu kamery, je pro nás nejpodstatnější jednobodová perspektivní projekce.
Základními rysy perspektivní projekce jsou:

- vzdálenější předměty jsou menší než předměty blíže k pozorovateli
- projekce nezachovává rovnoběžnost hran

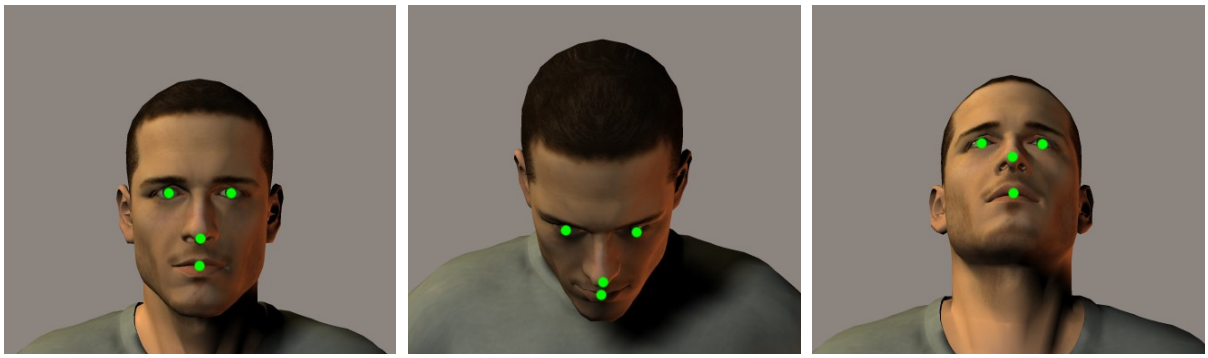
Právě to, že perspektivní projekce deformuje obraz, lze využít k výpočtu směru pohledu. V případě, že budeme zpracovávat 4 základní body a to oči, nos a ústa, vznikne nám poměrně jednoduchý příklad perspektivní projekce. Výsledný efekt při pohybu hlavy směrem do záklonu nebo předklonu je takový, že se na průmětně pohybuje nos buď směrem k očím nebo při pohybu hlavy dolu směrem k ústům.

Pokud budeme uvažovat rovinu s těmito body, je rozdíl v pozicích bodů téměř neznatelný tak, jak je to patrné z následujících obrázků.



Obr. 4.4: Změna pozice bodů v perspektivní projekci

Jakmile tento postup aplikujeme na obličej, kde můžeme využít vlastnosti, že nos je oproti očím a ústům vystouplý. Tak tento fakt způsobí, že je změna poměrů v závislosti na naklonění hlavy výrazně znatelnější. Následující obrázky to názorně ukazují .



Obr. 4.5: Ukázka změny vzájemné pozice bodů v závislosti na náklonu hlavy

Protože jsem se omezil pouze na záklon, předklon nebo stav, kdy je pohled směřován dopředu, mohl jsem si dovolit určovat směr prostým porovnáním poměru, který už zde byl zmiňován (nos - ústa / oči - ústa) s přednastaveným prahem. Přesněji jde o dva prahy- jeden pro pohyb dolů a druhý pro pohyb nahoru. Tyto prahy ale nelze obecně stanovit, záleží totiž na tvaru obličeje konkrétní osoby.

Podobné vlastnosti pak lze také vypozorovat u natáčení hlavy do stran. V takovém případě dochází k přesouvání nosu pod jedno nebo druhé oko. Tady je vhodné navíc detekovat místo jednoho bodu na ústech body dva (koutky úst). Tyto dva body pak umožní kvalitnější určení pozice v horizontálním směru. Nezbytnou podmínkou této části je kvalitní detekce špičky nosu nikoliv nosu jako takového. Právě tato detekce bývá často velice problematická, protože nos má na rozdíl od očí, obočí nebo úst mnohem menší barevný rozdíl oproti okolním částem obličeje.

Tento postup je dodatečně možné rozšířit i o určení míry naklonění hlavy, což je například při ovládání pomocí posunu kursoru myši možné využít k změně rychlosti pohybu. Možnosti se nabízejí dvě a to buď zvětšení počtu prahů nebo výpočet úhlu náklonu přímo z poměru vzdáleností. První metoda je vhodná v případě, že počet rychlostí, kterými se kursor může pohybovat, je omezený a nízký, např. 5. Druhá metoda umožňuje plynulou změnu rychlosti (počet stupňů rychlosti pohybu je tedy neomezen), na druhou stranu je zapotřebí převést výstupní hodnoty poměru vzdáleností na funkci, jejíž průběh bude buď lineární nebo alespoň souměrný podle bodu, který určuje nulovou rychlost.

5 Implementace

5.1 Použité knihovny

Jelikož je program vytvořen v jazyce C++ a ten jako takový nedisponuje prostředky pro snímání a zpracování obrazu, nebo funkcemi pro vytvoření kvalitního grafického uživatelského rozhraní, Rozhodl jsem se pro použití dalších knihoven a to OpenCV a Qt.

5.1.1 Knihovna OpenCV

OpenCV (open computer vision library) [10, 11, 12] je knihovna pro počítačové vidění, původně byla vyvíjena firmou Intel. Obsahuje více než 500 funkcí pro zpracování obrazu a vytvoření velmi jednoduchého pracovního prostředí (zobrazení výstupního obrázku, posuvníky pro nastavení, apod.). Vzhledem k množství funkcí a jejich druhu je pro tuto práci velmi vhodná. Z této knihovny jsou použity zejména funkce pro zachycení videa z kamery, následně funkce pro zpracování zachyceného snímku, jako je změna velikosti, nebo úprava barev či ekvalizace histogramu. Dalším důležitým prvkem, který byl z této knihovny použit, je detektor obličejů. Detektor, který knihovna obsahuje, je velmi kvalitní a pro potřeby práce je naprosto dostačující. V neposlední řadě byly použity funkce pro diskrétní 2D konvoluci, pomocí které se nanášely filtry.

5.1.2 Knihovna Qt

Jelikož mým úkolem bylo i vytvořit jednoduché rozhraní, ve kterém lze demonstrovat ovládání počítače pohledem, potřeboval jsem knihovnu, která by mi umožnila vytvoření vyhovujícího prostředí, k čemuž se knihovna OpneCV příliš nehodila. Proto jsem použil další knihovnu a to knihovnu Qt ve verzi 4. Qt je jedna z nejpopulárnějších knihoven pro vytváření grafického uživatelského rozhraní. Samozřejmostí je také multiplatformnost. V mé práci je knihovna použita pro vytvoření prostředí, ve kterém je možné číst text, který lze posouvat (scrollovat) pomocí změny směru pohledu. Dalším přínosem této knihovny je, že dokáže vytvořit vícevláknovou aplikaci, díky čemuž je možné velmi náročnou část programu starající se o výpočty spojené s detekcí směru pohledu přesunout do samostatného vlákna. Tím je umožněno používání ostatních částí programu bez ohledu na to, jak moc je počítač zaneprázdněn výpočty potřebnými pro posun textu.[13]

5.2 Charakteristika a popis funkce programu

Celý program je rozdělen do dvou hlavních částí: Jádro, které se stará o všechny výpočty a GUI, které je zde jako prostředník výstupu mezi jádrem a uživatelem. Obě části běží v samostatných vláknech, takže se nijak neovlivňují, hlavně situacích, kdy jádro obsadí procesor a GUI se tak stává neovladatelné.

5.2.1 Jádro programu

Jádro je bezesporu nejpodstatnější částí této práce. Z větší části je postaveno na knihovně OpenCV a jeho naprostá většina je v souboru CoreThread. Po spuštění běhu vlákna se nejprve připraví všechna potřebná data pro zpracovávání snímků. Jde hlavně o načtení nastavení jednotlivých filtrů, dále informace potřebné k běhu detektoru, především kaskáda klasifikátorů. V případě, že jsou všechna potřebná data připravena, pokračuje program do hlavní části s detekcí.
Detekce pak probíhá v několika základních krocích:

- V prvním kroku je načten snímek z kamery, ten je poté předán detektoru obličeje, který vrací obrázek ořezaný pouze na velikost obličeje. Přitom se bere v úvahu posledních 5 záznamů o pozici a velikosti obličeje. V případě, že detektor nalezne více obličejů, jako výstupní použije ten, který je nejblíže předchozímu a má nejmenší rozdíly ve velikosti. Abych upřesnil předchozí informace, výstupem detektoru není přesná pozice posledního nalezeného obličeje, ale jde o průměr posledních 5ti pozic, přičemž ta, která se hledala v posledním cyklu má nejvyšší váhu. Výsledná pozice je pak dána vztahem:

$$actual = \frac{1 \cdot last[0] + 0{,}5 \cdot last[1] + 0{,}35 \cdot last[2] + 0{,}25 \cdot last[3] + 0{,}15 \cdot last[4]}{2{,}25} \qquad (5.1)$$

  kde *actual* je pozice, kterou vrací detektor prostřednictvím ořezaného obrázku, *last[0]* je pozice nalezená v poslední detekci *last[1,]* je pozice v předposlední detekci atd. Stejný postup je realizován při výpočtu velikosti výsledného obrázku s tím rozdílem, že vstupem nejsou poslední pozice, ale velikosti.

- Výstupní obrázek z detektoru se pak převede na obrázek ve stupních šedi. Na tento obrázek je pak aplikována sada tří až devíti filtrů. Výstup každého z filtrů projde prahováním podle vlastního prahu. Tyto výstupy jsou pak sečteny podle jednotlivých kategorií pro očí nos a ústa.

Tím vzniknou tři obrázky, každý pro detekci jiné části obličeje. Jejich obsahem jsou vždy informace sloučené ze tří výstupů filtrů. Vzhledem k tomu, že výstupy filtrů byly prahovány, než došlo k jejich sečtení, obsahuje tento obrázek pouze čtyři úrovně šedi. A to pro oblasti, které nebyly výstupem ani jednoho z filtrů nebo právě jednoho, dvou nebo všech tří filtrů.

- Poté co je výstup z filtrů a prahovacích funkcí hotov, je potřeba najít v něm oblasti s vysokou odezvou na filtry. Jediným způsobem, který mě napadl je postupně procházet celý obrázek, dokud nenajdu bod s hodnotou větší, než je práh pro akceptování oblasti. Od tohoto bodu se pak spustí řádkovo-semínkové vyplňování oblasti, během kterého se určí rozměry, střed a velikost oblasti. Jelikož je tento postup časově náročný, nekontroluji každý z pixelů, ale při průchodů daným řádkem se čte hodnota každého druhého bodu, stejně tak se nečtou všechny řádky, ale čte se každý druhý. Pixely, které se tímto postupem zanedbají, jsou nepodstatné, protože i kdyby zde byla oblast s vysokou odezvou, byla by natolik malá, že by byla díky své velikosti při dalším postupu označena za oblast šumu nebo jiných rušivých elementů.

5.2.1.1 Nalezení očí

Máme-li seznamy jednotlivých oblastí ze všech tří výstupů, musíme z těchto seznamů vybrat pouze ty body, které dohromady tvoří logickou stavbu obličeje. Například postavení bodů, kde nos nad očima je nesmyslné. Nejdříve začneme hledáním očí, díky použitému detektoru je oblast výskytu očí poměrně dobře omezena. Jako vhodný výřez jsem zvolil obdélník, který je od horního okraje vzdálen 1/5 výšky obrázku a zasahuje až do jeho poloviny. Na obou krajích je pak oříznut o jednu dvacetinu. Následující obrázek tuto oblast názorně ukazuje:

Obr. 5.1: Oblast pro vyhledání očí

Samotný výřez pro dostatečné zredukování počtu bodů nestačí, takže dalším krokem je vyřazení bodů, které netvoří vodorovný pár, navíc je ještě podmínka rozšířena o minimální požadovanou vzdálenost, protože při větším záklonu hlavy docházelo k detekci nosních dírek místo očí. Tato množina bodů už je dostatečně malá, aby se z ní daly vybrat právě dva body reprezentující oči. Tento výběr se provádí až nakonec po nalezení nosu, tak aby oči byly co nejníže a každé bylo na jinou stranu od nosu.

5.2.1.2 Nalezení úst

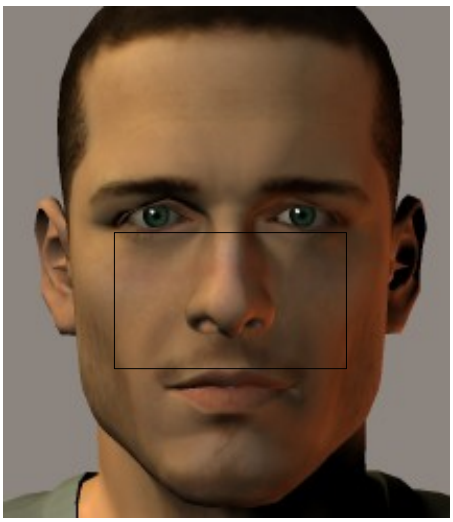
Dalším krokem je nalezení úst, ta se hledají podobně jako oči pomocí zredukování počtu nalezených bodů na základě pozice. Tato oblast je vymezena pouze vertikálně a to tak, že je vzdálen 3/5 výšky obrázku od horního okraje a 1/20 od spodního okraje. Dále musí mít hledaná oblast úst 3x větší šířku než výšku, což vyplývá z tvaru úst a tvaru oblasti, která je výstupem po použití Gáborových filtrů. Následující obrázek ukazuje oblast pro vyhledávání úst :

Obr. 5.2: Oblast pro vyhledání úst

Posunutí spodní hranice o 1/20 jsem provedl, protože bez ní docházelo k detekci ramen místo úst. Jejich šířka totiž vyhovuje předešlým podmínkám a navíc je širší než ústa, takže by měla v další části kódu přednost. Právě tato nejširší oblast se poté prohlásí za ústa.

5.2.1.3 Nalezení nosu

Jako poslední se vyhledává nos a to z důvodu, že jeho nalezení je nejproblematičtější. Oblast pro hledání tentokrát není vytyčená napevno, ale za pomocí očí a úst. Je omezena nejnižším, nejlevějším a nejpravějším bodem očí, a ústy. V tomto prostoru už pak stačí najít největší spojitou oblast.



Obr. 5.3: Oblast pro vyhledání nosu

Po nalezení všech těchto oblastí se podobným způsobem jako u detekce obličeje vypočítá průměrná hodnota pozice posledních 5ti detekcí, podle identického vzorce (5.1). Poté se pošle signál (zprostředkovaný pomocí knihovny Qt) o úspěšném nalezení všech bodů do GUI, zároveň se jiným signálem posílá směr pohledu vypočtený pomocí následujícího vzorce:

$$out = \frac{mouth - \frac{(Leye + Reye)}{2}}{mouth - nose} \tag{5.2}$$

Všechny výše uvedené hodnoty, zvláště pak meze jednotlivých oblastí, byly určeny experimentálně, tak aby vykazovaly přijatelnou chybovost v co největším množství případů. Doporučuji je brát pouze jako ukázkovou hodnotu, je totiž velice pravděpodobné, že existuje i jiná kombinace hodnot, vykazující lepší výsledky.

5.2.2 Grafické uživatelské rozhraní

Celé GUI je založeno na knihovně Qt. Ve skutečnosti jde jen o demonstrační ukázku zpracování výstupu jádra. Jelikož jsem chtěl, aby byl program v praxi použitelný, nevytvářel jsem jako výstup žádné animace. Jde o velice jednoduché okno, kde převážnou část zabírá objekt textBrowser, a vedle něj je velmi jednoduchým způsobem znázorněna signalizace o stavu programu. Jedinými vstupy GUI jsou tyto 3 signály od jádra:

- úspěšnost detekce obličeje
- nalezení všech bodů
- informace o naklonění hlavy

Výstup (jestli je možné toto výstupem nazvat) je reakce posuvníku v objektu textBrowser na pohyb hlavy. Při pohledu dolů se začne pohybovat posuvník taktéž dolu, při pohybu hlavy nahoru je postup obrácený.

Další funkcí GUI je, že umožňuje pohodlné nastavení celého programu, ať už jde o parametry jednotlivých filtrů a další vlastnosti s nimi spojené, jako je velikost nebo práh nebo ostatní nastavení, jako meze pro jednotlivé směry pohybu, rychlost scrollování textu apod .

6 Testování výsledků práce

Celé testování kvality jednotlivých částí detekce probíhalo výhradně na reálném videu, nešlo o žádné modelové situace. Jednoduše řečeno, testovala se úspěšnost při normálním používání programu, samozřejmě se při testech objevily i extrémní situace, ať už náklonu hlavy nebo různého nestandardního osvětlení (převážně silné boční, nebo oslňující objektiv kamery).

6.1 Úspěšnost detektoru obličeje

Úspěšnost detektoru, byla v průběhu celé práce na velmi vysoké úrovni obzvláště ve spojení s algoritmem, který byl popsán v kapitole 5.2.1, jehož funkcí je vyhodnotit, který z potenciálně nalezených obličejů patří uživateli. Navíc v případě, že se tento algoritmus splete, není tato chyba, díky průměrování posledních 5ti detekcí tak, závažná. Na druhou stranu to s sebou nese riziko, že detektor nebude schopen reagovat na příliš rychlý pohyb hlavy a ve výsledném výřezu se tak neobjeví celý obličej.

Za normálních okolností, během testování programu nevykazoval detektor téměř žádnou chybovost. Jeho úspěšnost byla vyšší než 90 %. Nutno podotknout, že v záběru kamery byla pouze jedna osoba, případné negativní detekce se tedy projevovaly pouze zřídka na stěnách nebo nábytku. Tyto chyby ale byly odstraněny již zmíněným algoritmem.

Při testování extrémních situací zvládal detektor s úspěšností alespoň 40 – 50 % i extrémně tmavou místnost (osvětlena převážně LCD monitorem) nebo při silném bočním osvětlení. Naopak situace, které nezvládal, byly při oslnění objektivu kamery v důsledku silného zdroje světla za uživatelem (jediným případem, kdy byl schopen reagovat, byl přímý pohled do kamery, při sebemenším pohybu hlavy selhal).

6.2 Úspěšnost použitých filtrů

Úspěšnost použitých filtrů byla nižší než u detektoru, při optimálních podmínkách se pohybovala mezi 75 – 80 %. Při nevhodných podmínkách je tento rozdíl ještě propastnější. Při extrémně tmavém prostředí je tato metoda téměř nepoužitelná a při bočním osvětlení sice body detekuje, ale s výraznou chybovostí.

Naopak v situaci, kdy je objektiv kamery oslněn a detektor obličeje byl téměř nepoužitelný, vykazovaly filtry téměř shodnou účinnost s ideálními podmínkami a navíc byla podstatně kratší doba výpočtu jednotlivých snímků, což vedlo téměř k zdvojnásobení snímků zpracovaných za jednu

vteřinu (fps). Je to zapříčiněno zejména zmenšením počtu nalezených bodů v obraze, které bylo nutné zpracovat.

6.3 Použitelný rozsah pohybu

Meze rozsahu pohybu jsou především závislé na typu a kvalitě osvětlení, dále na použité kameře. Většina převážně levných webových kamer má totiž problémy s vyvážením bílé.

Budeme-li brát jako modelový příklad místnost, v níž není výrazné osvětlení z jednoho směru, zároveň intenzita osvětlení je srovnatelná s osvětlením kanceláře tzn. minimálně 250 – 300 luxů. Pak je použitelný rozsah dostatečně velký, aby umožnil detekci při náklonu hlavy přibližně o 30° nahoru i dolů. Tomuto rozsahu odpovídají hodnoty poměru vzdáleností 0,18 až 0,6. Za tímto rozsahem (občas i v blízkosti tohoto rozsahu) začíná být detekce nestabilní, což je signalizováno zčervenáním jednoho z terčíku signalizujících úspěšnost detekce obličeje a jednotlivých významných bodů.

7 Závěr

Práce je zaměřená na nalezení významných bodů v obličeji, na základě jejichž vzájemné pozice je možné určit směr pohledu a ten dále využívat k ovládání počítače. Pomocí popsaných postupů se mi podařilo vytvořit ukázkový program, který při dodržení určitých omezení vyplývajících z kapitoly 6.3 umožňuje jednoduché ovládání tohoto ukázkového programu.

Jádro programu bylo vytvořeno tak, aby ho bylo možné použít i pro jiné účely než pro ovládání posunu textu. Jeho výstupem je desetinné číslo od nuly do jedné a na tom, jak bude použito, už nezáleží.

Jelikož vývoj ovládání počítače nebo jiných zařízení obecně je v současné době v začátcích, jsou možnosti dalšího vývoje značné. Zejména jde o zlepšení přesnosti a spolehlivosti, tak aby pokud možno nebyla závislá na typu a kvalitě osvětlení. Dále zrychlení výpočtů a upravení filtrů tak, aby byly co nejuniverzálnější.

Dalším výrazným rozšířením by mohlo být doplnění detekce pohybu zorniček, nebo pozice očních víček, aby bylo možné určit mrknutí oka, které by bylo použitelné například pro klikání.

Literatura

[1] Wikipedie: The Free Encyclopedia: RGB color model
http://en.wikipedia.org/wiki/RGB (2008)

[2] Wikipedie: The Free Encyclopedia: Grayscale
http://en.wikipedia.org/wiki/Greyscale (2008)

[3] Matas Jiri, Šochman Jan: AdaBoost, ČVUT Praha 2004
http://www.robots.ox.ac.uk/~az/lectures/cv/adaboost_matas.pdf (2008)

[4] Wikipedie: The Free Encyclopedia: Haar Wavelet
http://en.wikipedia.org/wiki/Haar_wavelet (2008)

[5] Wikipedie: The Free Encyclopedia: Gabor Filter
http://en.wikipedia.org/wiki/Gabor_filter (2008)

[6] Petkov N., Wieling M.B: *Gabor filter for image processing and computer vision*, University of Groningen, 2006
http://matlabserver.cs.rug.nl/edgedetectionweb/web/edgedetection_params.html (2008)

[7] Wikipedie: The Free Encyclopedia: konvoluce
http://cs.wikipedia.org/wiki/Konvoluce (2008)

[8] Davidson Michael W., Abramowitz Mortimer: *Convolution Kernel Mask Operation,*
Olympus America Inc. And The Florida State University 1998 - 2008
http://micro.magnet.fsu.edu/primer/java/digitalimaging/processing/kernelmaskoperation (2008)

[9] Active Appearance Models: *IMM Face Database*
http://www2.imm.dtu.dk/~aam (2008)

[10] Open Source Computer Vision Library: ReferenceManual, Intel Corporation 1999 - 2001
http://www.cs.unc.edu/Research/stc/FAQs/OpenCV/OpenCVReferenceManual.pdf

[11] OpenCV Reference Manuals, Cranfield Campus / ESTIA Campus, 2007/08
http://public.cranfield.ac.uk/soe/c5354/teaching/dip/opencv/manual (2008)

[12] Gady Adam: Introduction to programming with OpenCV, Illinois Institute of Technology, 2006
http://www.cs.iit.edu/~agam/cs512/lect-notes/opencv-intro/index.html (2008)

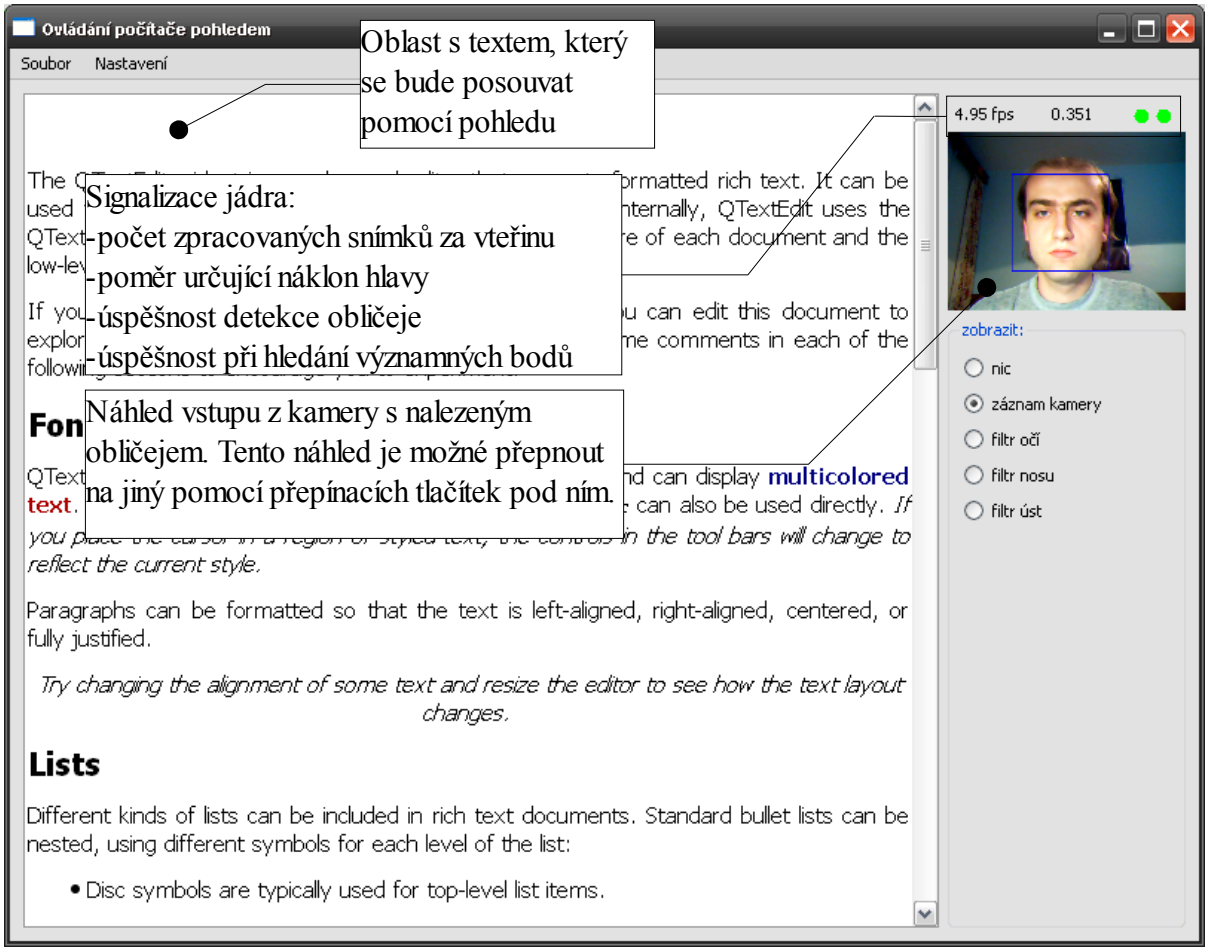
[13] Trolltech: Qt Reference Documentation (Open Source Edition)
http://doc.trolltech.com/4.3 (2008)

[14] Wikipedie: The Free Encyclopedia: Histogram equalization
http://en.wikipedia.org/wiki/Histogram_equalization (2008)

Seznam příloh

Příloha A - Manuál k programu

Hlavní okno programu:



Menu:

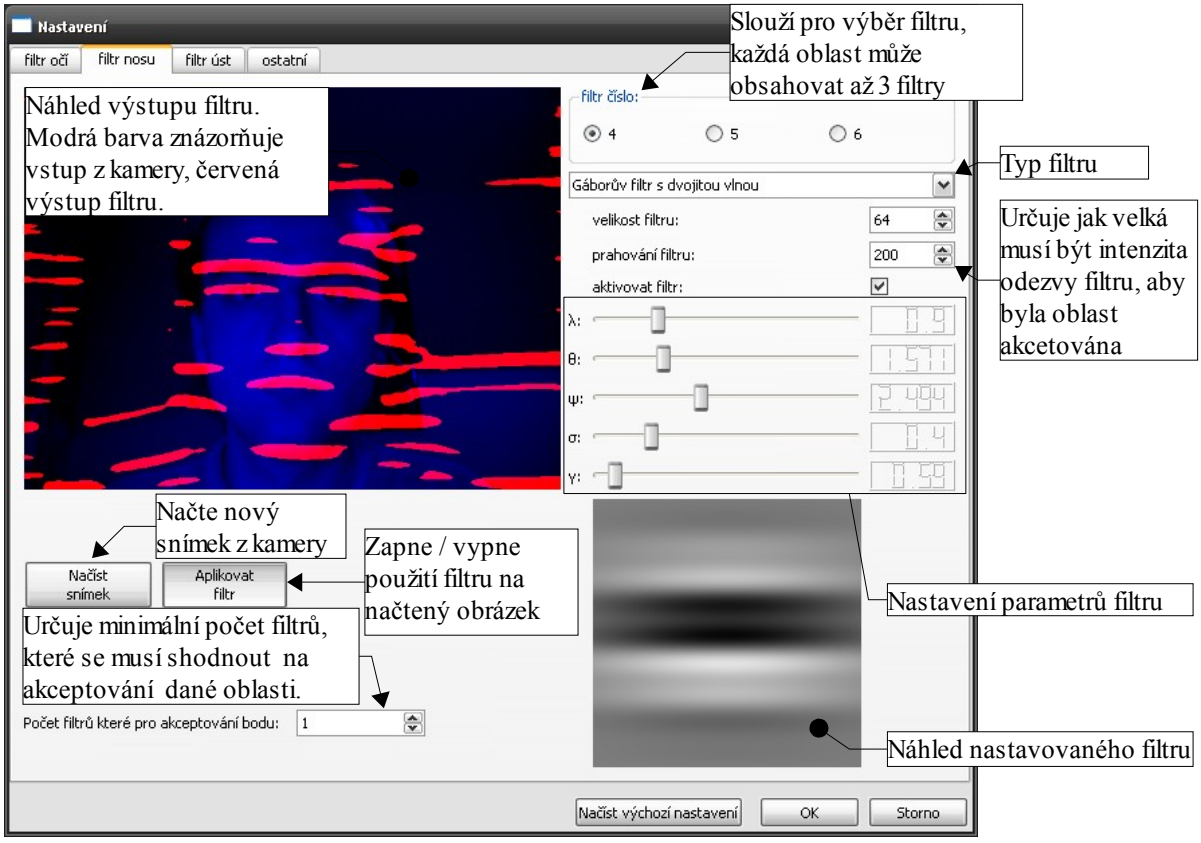
Soubor

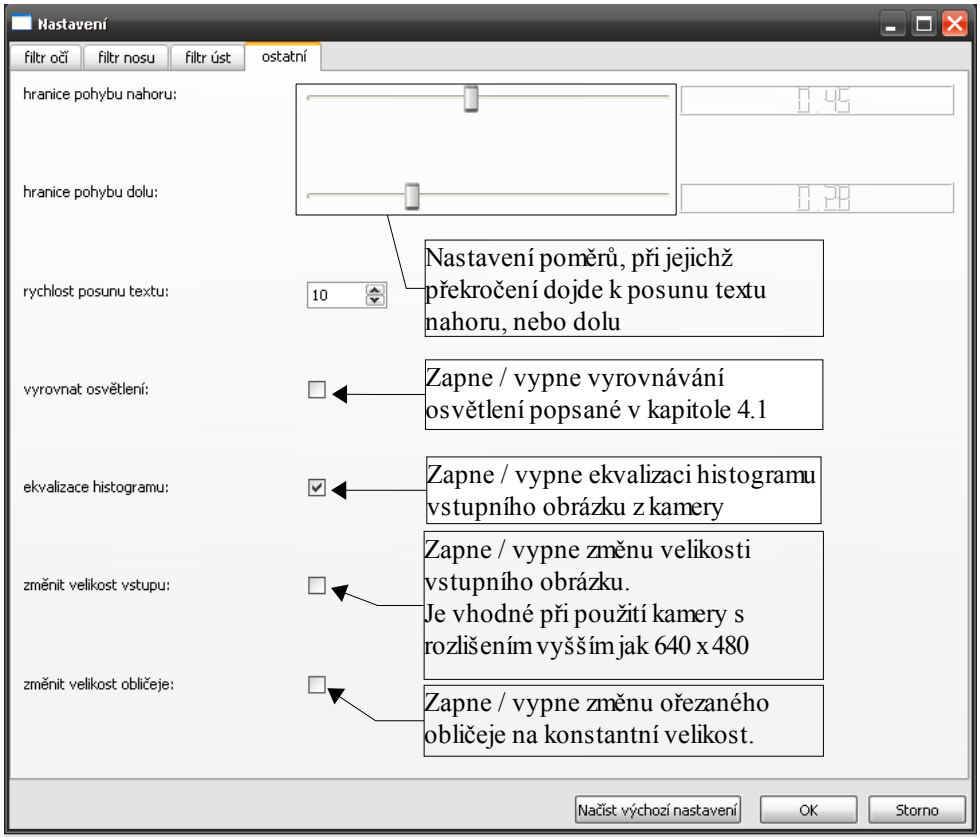
Otevřít	Otevře textový soubor pro prohlížení
Konec	Ukončí program

Nastavení

Načíst	Načte soubor s nastavením programu
Uložit	Uloží nastavení programu do souboru
Nastavení	Umožní nastavit program (parametry filtrů, meze pohybu, úprava osvětlení ...)
Jazyk	Změní jazyk programu

Nastavení:


Nastavení
filtr očí
filtr nosu
filtr úst
ostatní
Slouží pro výběr filtru, každá oblast může obsahovat až 3 filtry
Náhled výstupu filtru. Modrá barva znázorňuje vstup z kamery, červená výstup filtru.
filtr číslo:
4
5
6
Typ filtru
Gáborův filtr s dvojitou vlnou
velikost filtru:
64
prahování filtru:
200
aktivovat filtr:
Určuje jak velká musí být intenzita odezvy filtru, aby byla oblast akcetována
λ:
θ:
ψ:
σ:
γ:
Načte nový snímek z kamery
Načíst snímek
Aplikovat filtr
Zapne / vypne použití filtru na načtený obrázek
Nastavení parametrů filtru
Určuje minimální počet filtrů, které se musí shodnout na akceptování dané oblasti.
Počet filtrů které pro akceptování bodu:
1
Náhled nastavovaného filtru
Načíst výchozí nastavení
OK
Storno


Nastavení
filtr očí
filtr nosu
filtr úst
ostatní
hranice pohybu nahoru:
hranice pohybu dolu:
Nastavení poměrů, při jejichž překročení dojde k posunu textu nahoru, nebo dolu
rychlost posunu textu:
10
vyrovnat osvětlení:
Zapne / vypne vyrovnávání osvětlení popsané v kapitole 4.1
ekvalizace histogramu:
Zapne / vypne ekvalizaci histogramu vstupního obrázku z kamery
změnit velikost vstupu:
Zapne / vypne změnu velikosti vstupního obrázku. Je vhodné při použití kamery s rozlišením vyšším jak 640 x 480
změnit velikost obličeje:
Zapne / vypne změnu ořezaného obličeje na konstantní velikost.
Načíst výchozí nastavení
OK
Storno