VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2020 Bc. Nikol Hunkařová

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV BIOMEDICÍNSKÉHO INŽENÝRSTVÍ
DEPARTMENT OF BIOMEDICAL ENGINEERING

SLEDOVÁNÍ POHYBU OČÍ POMOCÍ PLATFORMY RASPBERRY PI
EYE MOVEMENT TRACKING USING THE RASPBERRY PI PLATFORM

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Nikol Hunkařová**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Branislav Hesko**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Biomedicínské a ekologické inženýrství**

Ústav biomedicínského inženýrství

Studentka: Bc. Nikol Hunkařová ***ID:*** 186662
Ročník: 2 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Sledování pohybu očí pomocí platformy Raspberry Pi

POKYNY PRO VYPRACOVÁNÍ:

1) Nastudujte si problematiku sledování pohybu očí snímaných kamerou a proveďte v této oblasti literární rešerši. 2) Seznamte se s platformou Raspberry Pi a modulem PiCamera a navrhněte jednoduchý systém, který bude schopný plnit funkci sledování očí. 3) Implementujte načítání a předzpracování obrazů z kamery na platformě Raspberry Pi. 4) Vytvořte metodu, která bude automaticky detekovat směr pohledu. 5) Vhodným způsobem navrhněte a realizujte program, který bude vyhodnocovat změnu směru pohledu. 6) Navrhněte a vhodným způsobem otestujte funkčnost vámi vytvořeného programu a popište dosažené výsledky. K vytvoření aplikace využijte programovací jazyk Python a libovolnou knihovnu ke zpracování obrazů (OpenCV, scikit-image, PIL).

DOPORUČENÁ LITERATURA:

[1] DONGHENG, Li, D. WINFIELD and D.J. Parkhurst. Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches. 2005. 8p. DOI: 10.1109/CVPR.2005.531 ISSN 1063-6919.

[2] JAN, Jiří. Medical Image Processing, Reconstruction and Restoration: Concepts and Methods. Boca Raton: CRC Press, 2005, ISBN 0-8247-5849-8.

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 29.5.2020

Vedoucí práce: Ing. Branislav Hesko

prof. Ing. Ivo Provazník, Ph.D.
předseda oborové rady

ABSTRAKT

Tato diplomová práce se zabývá sledováním pohybu očí pomocí platformy Raspberry Pi. Teoretická část studuje problematiku anatomie oka, detekce a sledování očí. V praktické části je v programovacím jazyku Python navrhnut systém, který je schopen sledovat oči pomocí platformy Raspberry Pi a modulu RPi Camera. Pro načítání a předzpracování obrazů z kamery je použita knihovna OpenCV. Je vytvořen program, který po kalibraci detekuje směr pohledu a vyhodnocuje změny směru. Přesnost programu je testována třemi metodami určení vektoru směru pohybu a dvěma metodami sledování terče pro čtyři různá rozlišení výstupní obrazovky.

KLÍČOVÁ SLOVA

Sledování očí, Raspberry Pi, python, OpenCV, picamera

ABSTRACT

This master's thesis deals with eye movement tracking using the Raspberry Pi platform. The theoretical part describes eye anatomy, eye detection and eyetracking. A system in Python programming language was designed in the practical part. This algorithm is able to perform the eye tracking function using the Raspberry Pi platform and the RPi Camera module. The OpenCV library is used for loading and preprocessing images from the camera. A method that detects and evaluates the direction of view after a calibration is available. The accuracy of the program is tested on three vector methods and two target methods for four screen resolutions.

KEYWORDS

Eye tracking, Raspberry Pi, python, OpenCV, picamera

HUNKAŘOVÁ, Nikol. *Sledování pohybu očí pomocí platformy Raspberry Pi*. Brno, 2020, 82 s. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav biomedicínského inženýrství. Vedoucí práce: Ing. Branislav Hesko

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Sledování pohybu očí pomocí platformy Raspberry Pi“ jsem vypracovala samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autorka uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušila autorská práva třetích osob, zejména jsem nezasáhla nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědoma následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autorky

PODĚKOVÁNÍ

Děkuji vedoucímu semestrální práce Ing. Branislavu Heskovi za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Sledování očí má v dnešní době velký význam. Často se používá k analýze zájmu uživatele. Na co se člověk dívá? Co ignoruje? Jak oči reagují na různé podněty? Kde oči zaostřily? Na tyto otázky odpovídají různé typy analýz v eyetrackingu. Nejrozšířenějšími oblastmi využívající tyto analýzy jsou marketing, webdesign a psychologie. Zkoumáním různých typů očních pohybů a jejich trajektorií je možné efektivně analyzovat interesantnost média či vliv únavy na řidiče. Stále častěji se na trhu objevují programy, které využívají pohyby očí k ovládání. Může to být kurzor myši, invalidní vozík nebo zaměřování cíle vojenským pilotem. Zastoupení sledování očí v moderních přístrojích se stále vyvíjí, roste poptávka i nabídka, neboť tyto algoritmy velmi významně urychlují a usnadňují práci člověka.

Cílem diplomové práce je seznámit s problematikou detekce a sledování pohybů očí snímaných kamerou. Praktickou částí je návrh systému, který bude schopen plnit funkci sledování očí, implementace načítání a předzpracování obrazů z kamery. Vše bude provedeno pomocí platformy Raspberry Pi s modulem PiCamera, programovacím jazykem bude Python. Při použití modulu Raspberry Pi je velmi důležité myslet na výpočetní náročnost navrženého programu. Je zvolena kvalitní RPi kamera s velkým rozlišením, proto by složité zpracování videa mohlo být na Raspberry Pi příliš náročné. Rozlišení je možné za předpokladu velké výpočetní náročnosti snížit.

K detekci bude použita hlavně knihovna OpenCV, která obsahuje mnoho funkcí pro zpracování obrazu. Navržená metoda by měla co nejpřesněji detekovat zornice obou očí v obraze i při změně osvětlení scény. Předzpracování obrazu by mělo vylepšit a urychlit detekci. Bude navržena vhodná kalibrační metoda, která bude probíhat před každým měřením. Program bude automaticky detekovat směr pohledu a vhodným způsobem jej vyhodnocovat. Nakonec bude provedeno testování přesnosti vytvořeného eyetrackeru pomocí sledovacích terčů. Samozřejmostí budou heat mapy, které budou výstupem každého měření v podobě obrázku i NumPy pole.

1 Teoretická část

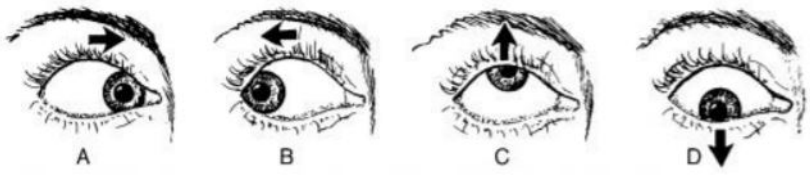
Teoretická část je věnována očním pohybům, metodám detekce obličeje, metodám detekce očí a teorií snímání pohybu očí.

1.1 Pohyby očí

Rozlišují se tři druhy postavení očí: primární, sekundární a terciální. Při vzpřímeném postoji s pohledem přímo před sebe oči zaujímají primární postavení. Pohled doprava a doleva se označuje jako sekundární postavení. Terciálním postavením je popisován pohled doprava nahoru nebo dolů a doleva nahoru nebo dolů. Oční koule jsou za fyziologických podmínek v primárním postavení paralelní. Toto postavení zajišťuje tonus okohybných svalů. Okohybné svaly jsou v neustálém pohybu při pohybu oční koule v různých směrech i při udržování primárního nebo jiného postavení. Pohyb oka je výsledkem změny tonu okohybných svalů [1].

Oční pohyby lze rozdělit na párové a nepárové. Párovými pohyby se myslí pohyby obou očí, nepárovými pohyby jsou míněny pohyby jednoho oka. Dále se pohyby oka dělí na volní (vědomé) a mimovolní (samovolné) [1, 2].

1.1.1 Nepárové pohyby



Obr. 1.1: Dukce; A – Addukce, B – Abdukce, C – Elevace, D – Deprese [1].

Základní pohyby jednoho oka jsou okolo hlavních os (vertikální, horizontální, sagitální). Tyto osy se protínají v bodě otáčení oka cca 13 mm za vrcholem rohovky. Pohyby se označují jako dukce. Rozlišuje se addukce, abdukce, elevace a deprese (viz obrázek 1.1) [1].

1.1.2 Párové pohyby

Každý sval má svého synergistu a antagonistu. Jejich souhra je esenciální při párových pohybech očí. Rozdělují se na konjugované a nekonjugované.

Konjugovanými se myslí pohyb očí stejným směrem (*verze*). Například při pohledu obou očí směrem doprava vede inervační impulz a následný stah k pravému vnějšímu a levému vnitřnímu přímému svalstvu. Mezi konjugované pohyby patří: sakadické, sledovací, fixační a reflexní pohyby [1].

Nekonjugované pohyby jsou pohyby očí opačným směrem (*vergence*). Zajišťují fixaci očí při pohybu pozorovaného objektu směrem od pozorovatele nebo k němu. Pohyby objektu od pozorovatele se označují jako divergence, pohyby objektu k pozorovateli jako konvergence. Vergenční systém kontroluje postavení zorných os oka tak, aby se předměty zobrazovaly na správných místech sítnic obou očí, což je jedním z hlavních předpokladů binokulárního vidění [1].

Sakadické pohyby

Rychlé pohyby očí dosahující maximální úhlové rychlosti 16 rad $\cdot$ s^{-1} se označují jako sakády. Tyto vysokofrekvenční změny jsou většinou reflexní, méně často volné. Sakády se objevují hlavně při čtení textu, kdy oko skáče z jednoho slova na druhé. Vyskytují se i u nevidomých. Doba mezi dvěma fixacemi je od 10 do 100 ms. K dalšímu cíli se oko dostane zhruba za 200 ms [1, 2].

Sledovací pohyby

Rychlost pohybu očí dosahuje jen 1,7 rad $\cdot$ s^{-1} za sekundu, proto se označují jako pomalé sledovací pohyby očí. Nejsou kontrolované naší vůlí, ale vyvolá je konkrétní zrakový podnět. Umožňují stabilní sledování pohybujících se předmětů nebo nepohybujících se předmětů, pokud se pohybuje hlava nebo tělo [1].

Jakmile rychlost pohybu předmětu dosahuje maximálně 0,52 rad $\cdot$ s^{-1} za sekundu, tak je předmět sledován přesně. Pokud se tato rychlost překročí, sledovací pohyby nestíhají předmět a objeví se trhané korekční sakády [3].

Fixační pohyby

Ani při usilovné fixaci není oko v klidu. Tyto pohyby však nejsou tak výrazné. Patří k nim hlavně drift, mikrosakády nebo tremor.

Drift jsou pomalé oční pohyby, kdy za 200 ms se osa vychýlí o 0,002 rad $\cdot$ s^{-1} a obraz promítnutý na sítnici se posune o 10 až 15 čípků. Obraz by se neměl dostat z centra žluté skvrny. Drift je asymetrickým pohybem. Význam je prozatím předmětem výzkumu [4].

Mikrosakády jsou nepravidelné a rychlé pohyby oka. Jejich amplituda se pohybuje okolo 0,0006 - 0,015 rad $\cdot$ s^{-1} v časovém rozmezí 10-20 ms. Mikrosakády jsou symetrické, ale absolutně neovlivnitelné vůlí. Vracejí zrakovou osu do základního postavení po vychýlení oka driftem [4].

Tremor jiným názvem oční třes má nejmenší amplitudu, je to 0,0001 - 0,0014 rad $\cdot$ s^{-1}. Naopak frekvence těchto pohybů jsou velké, pohybují se okolo 70-90 Hz, ale někdy přesahují i 100 Hz. Funkční význam zatím není prokázán [4].

Reflexní pohyby

Mezi reflexní pohyby patří optokinetický nystagmus a vestibulookulární reflex. Optokinetický nystagmus se skládá z pomalých očních pohybů v jednom směru, které jsou opakovaně přerušované rychlými očními pohyby sakadického typy ve směru opačném. Je to například pohled pasažéra hromadné dopravy ven z okna. Vestibulookulární reflex má za úkol udržet stálé postavení očí vzhledem k okolnímu prostředí i při sklonu nebo otočení hlavy či celého těla. Rotace hlavy v libovolném směru způsobí opačnou rotaci očí [1].

1.2 Detekce obličeje a očí

Před samotnou detekcí očí je někdy dobré detekovat nejdříve obličej. To platí pouze v případě, že se v obraze vyskytuje obličej celý. Pokud je vstupem oční prostor, lze jednoduše přejít na detekci očí. Pro detekci obličeje i očí existuje mnoho postupů a algoritmů, které se mohou navzájem překrývat.

1.2.1 Detekce obličeje založené na znalostech obličeje

Vychází se z typických vztahů mezi různými rysy a obličejem. Jako příklad lze uvést oči, které jsou v určité vzdálenosti od sebe, a jsou navzájem symetrické. Okolí oka bývá často tmavší než tváře a čelo. Velmi důležitá je přesná lokalizace a dobrý popis jednotlivých vztahů. Pokud se pravidla definují příliš volně, může dojít k detekci jiných struktur. Pokud se naopak pravidla definují příliš striktně, mohou být některé části obličeje přehlédnuty. Pro dobré výsledky musí být velmi vhodně zvolen kompromis [5].

Hierarchické metody

Hierarchické metody pracují se třemi úrovněmi. Nejprve se hledají struktury obličeje podle abstraktního modelu, jenž je složen z oblastí, které mají markantní rozdíly mezi stupni šedi v jednotlivých oblastech (vlasy, oblast očí, úst a nosu, vousy). Struktury, které odpovídají kritériím, jsou odeslány do další úrovně. Abstrakt modelu obličeje má malý rozměr (například 6 x 6 pixelů), proto se v originálním obraze využívá snižování rozlišení pomocí průměrování jasu. Ve druhé úrovni je prováděna lokální

ekvalizace histogramu a detekce hran. Výsledkem je nalezení přibližných obrysů obličeje v oblasti, kterou vymezí první úroveň. Třetí úroveň slouží pro úpravu výrazných obličejových struktur nalezených v druhé úrovni. Znovu se zde využívá znalostí rysů a obličeje. Struktury nosu se například hledají v oblasti mezi a pod očima. Oblasti s lokálními maximy se označují jako špičky nosu, oblasti s lokálními minimy se označují jako nosní dírky. Takto je prohledán celý obličej a všechny jeho struktury [5, 6, 7].

1.2.2 Detekce obličeje pomocí znalosti neměnných rysů

Vychází se ze znalosti, že některé obličejové rysy zůstávají i při natočení nebo změně osvětlení obličeje neměnné. Metody odvozují přítomnost obličeje z předešlé detekce obličejových rysů. Mezi tyto rysy patří barva obličeje, textura obličeje, výrazné struktury obličeje (oči, obočí, nos atd.). Pracuje se s geometrickými vztahy a jejich pravděpodobnostním výskytem. Metoda často dává velmi dobré výsledky, protože rysů, které se dají použít, je mnoho. Nejlepších výsledků se dosáhne kombinací co největšího počtu níže zmíněných postupů [6].

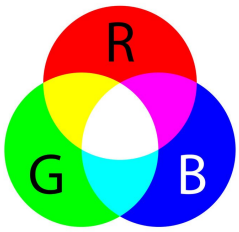
Obličejové rysy

Obličejové rysy nebo znaky hledáme podle znalosti anatomie. Z histogramu obrazu se určí práh, s jehož pomocí je obraz převeden na binární. Algoritmus se snaží najít oči v předzpracovaném binárním obraze. Dále podle očí hledá další rysy. Každému rysu se přiřadí vážená funkce podle důležitosti, která také slouží k výpočtu pravděpodobnostního výskytu obličeje. Tento předzpracovaný obraz je procházen směrem dolů. Je vyhledávána oční oblast, která většinou obsahuje velké množství hran, a oblast nad očima, která bývá bez hran. Pomocí poměru velikosti oční oblasti a oblasti nad očima lze vyhledat další rysy [8].

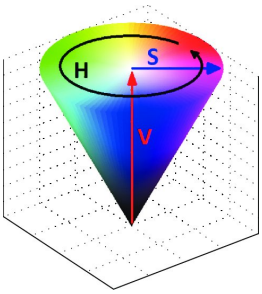
Barva obličeje

Rozdíly v barvě pleti jsou velmi často způsobeny rozdílnou intenzitou barev, proto lze obličej detekovat pomocí této proměnné. Místo tradičního RGB formátu se využívá formát HSV nebo YCbCr. V případě RGB formátu totiž obraz podléhá vlivu osvětlení scény. Někdy se může stát, že je obličej nadměrně osvětlen nebo některé části reflektují odražené světlo do okolí. Tím se myslí například odraz světla od rohovky. Na tuto problematiku se nejčastěji využívá YCbCr formát [9].

U RGB modelu se obraz skládá ze tří nezávislých obrazových rovin neboli primárních barev (červená, zelená, modrá). Barva jednoho pixelu v obraze je namíchána z těchto barev. Je důležité si uvědomit, že barvy, které vnímá oko, nejsou vytvořeny

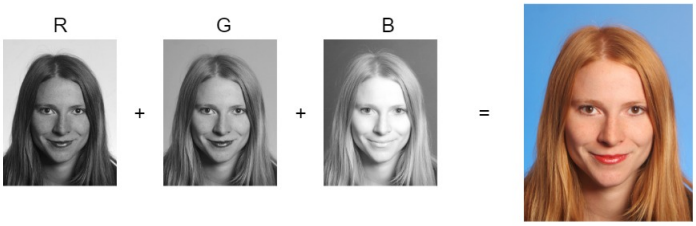


Obr. 1.2: RGB barevný model [13]



Obr. 1.3: HSV barevný model [22]

kombinací těchto tří barev, ale tak, jak je uvedeno na obrázku 1.2, ostatní odstíny vznikají různým poměrem každé hlavní složky. Nicméně RGB model je velmi přesný, úspěšný a hojně používaný v aplikacích pro zpracování obrazů. Barvy, které tvoří tento model, mohou být zpracovány samostatně ve stupních šedi podle intenzity dané barvy a poté rekombinovány do jednoho obrazu (obrázek 1.4) [11, 12].



Obr. 1.4: RGB složení obrazu [37]

HSV je cylindrické zobrazení souřadnicových bodů v RGB barevném zobrazení (obrázek 1.3). Písmena v názvu reprezentují odstín (hue), nasycení (saturation) a hodnotu (value). Odstín reprezentuje dominantní barvu, kterou pozorovatel vnímá. Saturace je množství bílého světla ovlivněné odstínem. Hodnota odpovídá chromatické představě intenzity, čím menší je hodnota, tím více je pixel vnímán černě, čím větší je hodnota, tím více pixel odpovídá dané barvě [22].

Hodnota jasu je u formátu YCbCr obsažena ve složce Y, barevnost ve složkách Cb a Cr. Složka Cb odpovídá červené barvě, Cr zastupuje barvu modrou. Převod mězi RGB a YCbCr lze provést na základě vztahu 1.1. Díky jasu, který je obsažen ve složce Y, lze jednoduše odstranit korneální reflexi, cože je odraz zdroje světla od rohovky. Tato skutečnost může velmi ovlivňovat detekci zornice. Proto bývá jakákoliv reflexe světla potlačena nebo odstraněna [23].

$$
\begin{aligned}
Y &= 0,299 * R + 0,587 * G + 0,114 * B \\
Cb &= -0,169 * R - 0,332 * G + 0,500 * B \\
Cr &= 0,500 * R + 0,419 * G - 0,081 * B
\end{aligned} \tag{1.1}
$$

1.2.3 Detekce obličeje pomocí porovnávání šablon

Tyto detekce jsou založeny na porovnání vzorů celého obličeje nebo jen jeho částí. Často je počítána korelace mezi vstupním obrazem a šablonami. Proces tvorby šablon je zdlouhavý a složitý. Novinkou v této oblasti jsou deformovatelné šablony kvůli časté změně pozice nebo překryvu hlavních komponent v obrazu [6, 16].

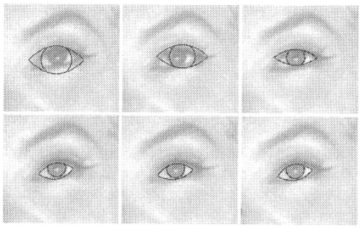
Nedeformovatelné šablony

Detekce je funkční pouze z předního pohledu. Obsahuje další podšablony pro struktury jako jsou oči, nos, ústa atd. Podšablony se slícují se vstupním obrazem a vypočte se vzájemná korelace. Pokud je výsledek vzájemné korelace dostatečný, je detekován obličej. U některých metod je aplikován opačný postup. Porovnávání šablon je nejčastěji zajištěno pomocí detekce hran v obraze [6].

Pro lokalizaci obličeje lze také využít siluet obličeje, které jsou získané redukcí dimenze dat pomocí analýzy hlavních komponent. Pomocí zobecněné Houghovy transformace je sada siluet využívána k lokalizaci tváře [5].

Deformovatelné šablony

Deformovatelné šablony odstraňují některé obtíže nedeformovatelných šablon. Jedná se především o úspěšnou detekci pouze z předního pohledu. Deformovatelné šablony by měly zajistit rozpoznání obličeje i při mírném natočení tváře. Šablona se posunuje po obraze a její tvar se může postupně přizpůsobovat obrazu pomocí zadaných parametrů, kterými jsou parametrická přímka a plocha šablony. Hodnoty parametrů se mění, až dokud je detekován obličej nebo dokud se nepřekročí nastavená mez. Při překročení meze algoritmus hledanou strukturu nenalezl. Příklad deformovatelné šablony pro detekci oka je uveden na obrázku 1.5 [5].



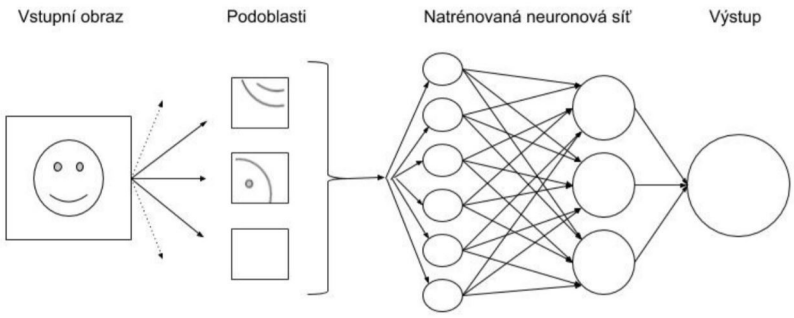
Obr. 1.5: Ukázka deformovatelné šablony [5]

1.2.4 Detekce obličeje založené na vzhledu

V tomto případě dochází k porovnání vstupního obrazu s trénovací množinou. V trénovací množině jsou uloženy variabilní vzory obličejů. Patří sem metody umělé inteligence například minimalizace diskriminační funkce, metody pravděpodobnostní a metody strojového učení využívající Haarovy příznaky [5].

Minimalizace diskriminační funkce

Minimalizace diskriminační funkce odstraňuje nevýhody nepředvídatelnosti vzhledu obličeje a okolí. Funkce je definována jako práh mezi třídou oblasti bez obličeje a třídou oblasti s obličejem. Diskriminační funkci lze získat pomocí mnohovrstvých neuronových sítí s nastavením vhodného procesu učení. Trénovací množina obsahuje velké množství obličejových a bez obličejových dat, která se porovnávají s histogramy podoblastí vstupních obrazů. Na obrázku 1.6 je znázorněn postup detekce [6, 17].



Obr. 1.6: Schéma postupu detekce pomocí diskriminační matice [17]

Pravděpodobnostní metody

Další metodou je realizace pravděpodobnostních metod, kde je každý obraz definovaný jako náhodná proměnná x. Dochází k definici pravděpodobnostní funkce

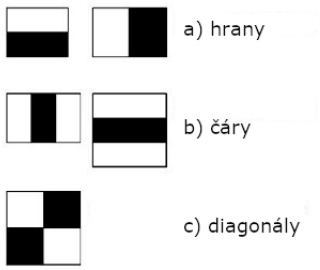
pro část, kde se obličej nachází a pro část, kde se obličej nenachází. K tomu se používá Bayesův vzorec (rovnice 1.2) a maximální pravděpodobnost [8].

$$\frac{P(x|obličej)}{P(x|bezobličeje)} > \frac{P(obličej)}{P(bezobličeje)} \tag{1.2}$$

Tvář je přítomna, pokud hodnota pravděpodobností na levé straně rovnice je vyšší než hodnota na pravé straně rovnice [5].

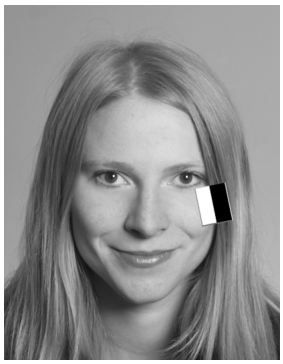
Haarovi příznaky

Metoda haarových příznaků je založena na strojovém učení, kdy je klasifikátor natrénován mnoha obrazy s obličejem. Tento algoritmus navrhl Paul Viola a Michael Jones v příspěvku [10] z roku 2001. Jedná se o metodu strojové učení s kaskádní funkcí natrénovanou na množině pozitivních a negativních obrazů. Knihovna OpenCV nabízí předem natrénované algoritmy této kaskády uspořádané do kategorií (obličej, končetiny atd.) v závislosti na obrázcích, na kterých jsou trénovány. Hlavní myšlenkou Haarových kaskád je extrakce příznaků z obrazu pomocí mnoha filtrů. Označují se jako Haarovy příznaky (viz obrázek 1.7) a posunují se po celém obraze [14].



Obr. 1.7: Filtry pro extrakci příznaků [14]

Pokud je použit hranový příznak *a) hrany* z obrázku 1.7 a aplikován na obrázek 1.8, výstupem je hodnota s velkou pravděpodobností výskytu hrany v tomto bodě, neboť se zde opravdu nachází hrana, kterou filtr kopíruje. Takových výpočtů zde probíhá mnoho, proto se proces zefektivnil pomocí sumačních map a další optimalizace, tím je umožněna filtrace libovolně velké oblasti. Cílem je snížit výpočty potřebné k zisku sumy intenzit pixelů v okně, vybrat pouze relevantní příznaky a vhodně optimalizovat algoritmus. Koncept Ensembilgovy metody kombinuje mnoho slabších algoritmů čímž vytváří jeden velmi silný. Kaskáda klasifikátorů nepoužívá všechny funkce na celý obraz, ale seskupuje funkce v obrazu do různých fází. Pokud okno neprojde prvním nejjednodušším algoritmem, tak je zahozeno a do dalších funkcí už nevstupuje. Správným výsledkem je okno, které projde všechny fáze [15].



Obr. 1.8: Ukázka detekce hrany [15]

1.2.5 Detekce očí

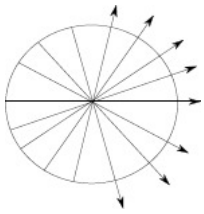
Detekce očí je dnes využívána v mnoha oblastech, protože díky ní lze získat informace o stavu člověka. U očí je významná a často detekovaná zornice i duhovka. Z rozměření lidského oka lze vyvodit mnoho závěrů. Měření rychlosti mrknutí může pomoci k zjištění únavy nebo dokonce odhalit Parkinsonovu chorobu. Snímky duhovky se používají při identifikaci osob nebo pro předpověď obstrukčního plicního nemocnění. U dysfunkce sakád může být prokázána souvislost s Huntingtonovou nebo Alzheimerovou chorobou [18].

Houghova trasnformace

Houghova transformace pro detekci kružnic je velmi časově i paměťově náročná, protože vyžaduje výpočet proměnných ve 3D parametrickém prostoru a to: poloměr kružnice r a souřadnice x a y. Takovou kružnici matematicky vyjadřuje rovnice 1.3 [19].

$$(x - x_{stred})^2 + (y - y_{stred})^2 = r^2 \tag{1.3}$$

Na druhé straně Houghova transformace pro detekci kruhů je schopná najít středy kruhů s neznámým radiem. Gradient obrazu je počítán hranovou detekcí s aproximací prvních derivací pomocí Sobelova filtru. Dále dochází ke kreslení úsečky ve směru gradientu z každého hranového bodu v obraze (obrázek 1.9). Kreslení úseček ve směru gradientu generuje projekci Houghova prostoru. Kruh s vyšším kontrastem, jako je například duhovka nebo zornice, přispívá svou gradientní silou počátečního bodu více k lokalizaci středu a nalezení kruhu. Jako příklad obrazu s detekcí Houghových kruhů je uveden obrázek 1.10. Kvalita výsledku úzce souvisí s kvalitou hranové detekce a předchozí znalosti o hledaných kruzích v obraze [19, 20].

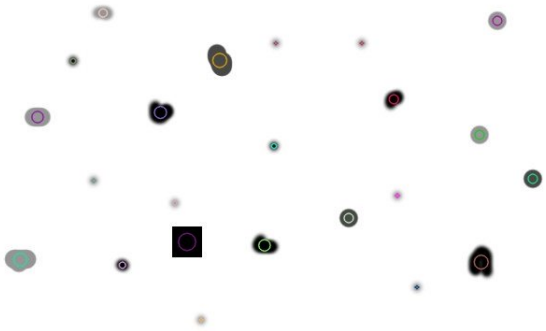


Obr. 1.9: Úsečky v Houghově prostoru [20]

Obr. 1.10: Příklad detekce kruhů pomocí Houghovi transformace [19]

Detekce shluku

Shluk je skupina sousedních pixelů, jenž sdílí společnou vlastnost, nejčastěji to bývá jas. Detekci různě velkých shluků ukazuje obrázek 1.11. Aby bylo možné shluk dobře detekovat, musí být obraz upraven tak, aby obsahoval pouze požadovaný shluk. Ostatní struktury musí být z obrazu odfiltrovány. To může být učiněno na základě morfologických operací a následného převodu šedotónového obrazu na binární obraz s určitým prahem [38].



Obr. 1.11: Ukázka detekce shluků [38]

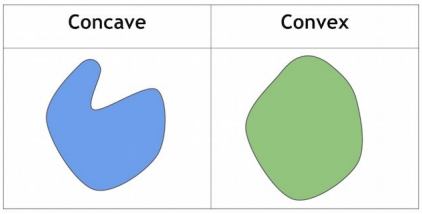
Algoritmus nejprve převede vstupní obraz na několik binárních obrazů na základě nastavení minimálního a maximálního prahu. V každém binárním obrazu jsou dále všechny bílé pixely seskupeny dohromady do shluků. Následuje určení středů shluků, poloměrů a ploch. Shluky nevyhovující zadaným parametrům jsou filtrovány [38] .

Nastavení parametrů je voleno s ohledem na detekci správného shluku, v tomto případě zornice nebo duhovky. Parametry, podle kterých lze filtrovat:

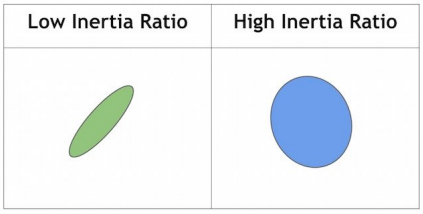
- barva,

- velikost,
- tvar,
- kulatost,
- konvexnost (obrázek 1.12),
- poměr setrvačnosti (obrázek 1.13) [38].

Funkce vrací klíčové body shluku, které jsou poté vykresleny do obrazu. Správná detekce je taková, kdy vykreslení klíčových bodů kopíruje obrys zornice [38].



Obr. 1.12: Konvexnost shluku [38]



Obr. 1.13: Setrvačnost shluku [38]

Starburst algoritmus

Tento algoritmus kombinuje přístupy založené na znalostech struktur obličeje, porovnávání šablon a deformace šablon. Pracuje s kompromisem mezi výkonem a přesností. Protože se většinou pracuje s obrazy pořízenými metodou tmavé zornice, je důležité odstranění odrazu světla od rohovky (korneální reflexe) ještě před detekcí zornice. Poté jsou detekovány hranové body zornice pomocí iterativních technik založených na vlastnostech obrazu. Následně je do podmnožiny detekovaných bodů fitována elipsa pomocí paradigma RANSAC. Elipsa se mění podle zvolených parametrů tak, aby se co nejvíce přizpůsobila detekovaným bodům [21].

Korneální reflexe odpovídá oblasti s nejvyšším jasem v okolí oka, takže může být odstraněna tak, že se obraz převode na binární s určitou hodnotou prahu. Práh se dá určit adaptivně, empiricky nebo manuálně. Korneální reflexe je odraz světla od rohovky, z toho plyne, že bude hledána v tmavé zornici [21].

1.3 Snímání pohybu očí

Sledování pohybu očí by se dalo nazvat jako určení místa pohledu nebo měření pohybu oka vzhledem k hlavě. Nejpoužívanější metodou je dnes videozáznam, ze kterého je extrahována informace o poloze oka [24].

1.3.1 Historie

Počátky sledování pohybu očí sahají až do konce 19. století. Louis Émile Javal v roce 1879 pozoroval pohyby očí pomocí zrcátka umístěného před čtenářem. Objevil, že pohyby oka jsou tvořeny sakádami a fixacemi (popsány v kapitole 1.1.2). Díky tomu, že experimenty pozoroval pouhým okem, mohl sledovat jen větší pohyby, které na sítnici odpovídají přibližně 0,2 cm. Proto byla metoda později zpřesněna zvětšením odraženého obrazu [25].

Další pokusy snímání pohybu očí na přelomu 19. a 20. století byly pomocí fotografování. Avšak metoda byla složitá a nepřesná. V roce 1898 přišel Delabarre s mechanickým přenášením pohybu oka využitím zapisovacího zařízení. Na povrch oka umístil tenkou čočku ze sádry, uprostřed čočky byl otvor, kterým procházel obraz. Čočka byla propojena se zapisovacím zařízením lanky. Metoda byla velmi invazivní, sádrová čočka navíc ovlivňovala přirozený pohyb oka. O vylepšení tohoto experimentu se zasloužili pánové Marx a Trendelenburgm, kteří k čočce přilepili malé zrcátko. Odraz světla od zrcátka byl zaznamenán na fotocitlivém papíře. V roce 1899 přišel Orschansky s využitím hliníkové čočky, která byla sice lepší než sádrová, ale stále byla velmi invazivní [25].

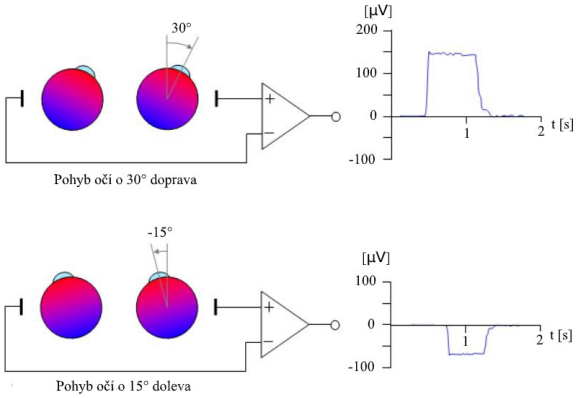
1.3.2 Současné techniky

V současnosti se pro snímání očí používají dvě metody: kontaktní a bezkontaktní. U kontaktních metod se na povrch oka připevní umělá elektroda, často to bývá speciálně upravená kontaktní čočka. Bezkontaktní metody využívají elektrických vlastností očí. Měření lze charakterizovat podle místa pohledu nebo podle pohybu oka vzhledem k hlavě [4].

1.3.3 Bezkontaktní metody

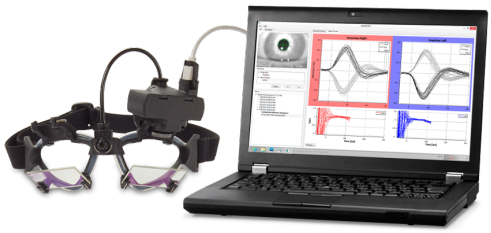
Patří sem elektrookulografie (EOG), videookulografie (VOG) a infračervená okulografie.

Elektrookulografie je založena na snímání napětí, které je tvořeno rozdílným potenciálem mezi rohovkou a sítnicí. Dosahuje amplitudy kolem 1 mV. Vektor elektrického pole se mění vzhledem k snímacím elektrodám podle pohybu očí. Rohovka má kladný náboj, sítnice nese náboj záporný. Elektrodami umístěnými na kůži v okolí očí se zaznamenává horizontální a vertikální složka pohybu. Pohybem očí doleva a doprava (viz obrázek 1.14) dochází ke změně elektrického potenciálu. Čím větší pohled doprava nebo doleva, tím větší amplituda elektrookulogramu. Nulový potenciál se objeví, když se oko dívá přímo vpřed, protože rohovka je přesně uprostřed pravé a levé elektrody [1].



Obr. 1.14: EOG [24]

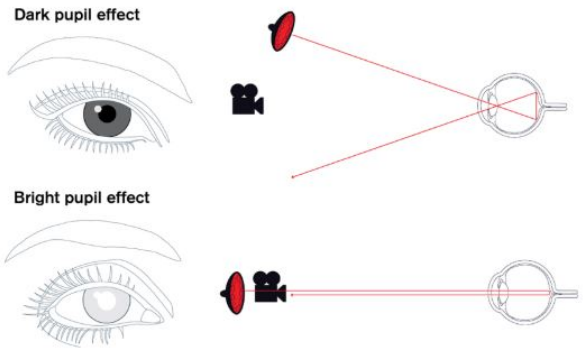
Videookulografie je metoda snímání pohybu očí pomocí digitální kamery. Kamera může být umístěna pod nebo nad monitorem nebo v brýlích v blízkosti oka. Často je využito infračerveného světla. K měření pohybu je zpracováván snímaný obraz, kamery rozpoznají a sledují střed zornice. Existuje několik způsobů snímání VOG. Kamera může být například umístěna přímo před jedním okem. Druhé oko pacient používá ke sledování objektů. Toto monokulární snímání není dokonale, jelikož neumožňuje hodnotit binokulární vidění a je limitováno jen na jedno oko. Preferovanější metodou je použití speciálních brýlí s dikrotickými filtry (obrázek 1.15). Brýle se chovají jako dvou vlnná zrcadla, které odrážejí infračervené světlo tak, že je pacient schopen normálního vidění. Počítačové algoritmy poté analyzují pozici každého oka zvlášť v závislosti na snímaný signál z obou kamer [26].



Obr. 1.15: Příklad snímání metody VOG dikrotickými brýlemi [25]

K detekci zornice se používají dvě nastavení osvětlení oka lišící se polohou zdroje vzhledem k oku. Jedná se o metody světlé a tmavé zornice. U metody světlé zornice je světelný zdroj umístěn na optické ose oka před rohovkou. Zdroj světla, snímač a oko přitom leží v jedné přímce. Infračervené světlo ze zdroje záření prochází přes oko, pokračuje na sítnici, kde dochází k odrazu. Odražené záření je detekováno.

V zornici je viditelný odraz světla, proto se jeví jako bílá. Uplatňuje se tu stejný fenomén, který způsobuje červené oči na některých fotkách. U metody tmavé zornice je světelný zdroj umístěn mimo optickou osu oka. Zornice se potom jeví černě, neboť skrz ni nedochází k odrazu paprsků. Záření se odráží mimo detektor, vše vysvětluje obrázek 1.16 [26].



Obr. 1.16: Světlá a tmavá zornice [26]

1.3.4 Kontaktní metody

Ke kontaktním metodám patří například magnetookulografie a elektroretinografie. Při magnetookulografii se používá cívka ke sledování pohybu, ta je složena z mnoha závitů velmi tenkého vodiče a umístěna v kontaktní čočce přímo na oku. V cívce se indukuje vlivem magnetického pole napětí, které se měří. Výhodou je velmi vysoká přesnost. Nevýhodou je invazivnost metody a vysoká cena čoček, které se musí často měnit [4].

Elektroretinografie je metoda, kdy se stejně jako u magnetookulografie používají kontaktní čočky pro měření. V tom to případě jsou v čočkách stříbrné či platinové drátky. Metoda umožňuje posouzení funkce sítnice a diagnostiku poruchy ještě před výskytem funkční vady, což je obrovskou výhodou. Nevýhodou je riziko poškození rohovky nebo zorného pole oka. Oko také musí být při měření lokálně umrtveno [4].

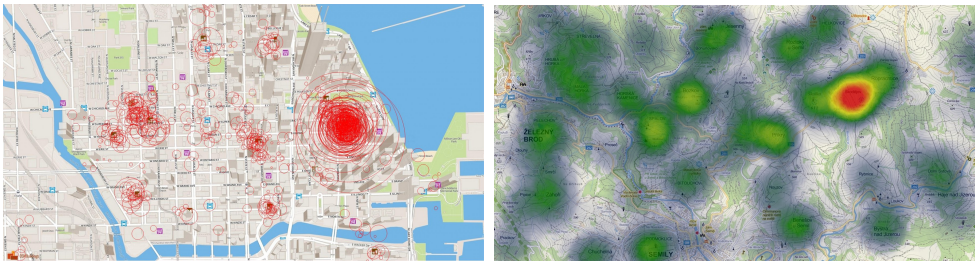
1.4 Eyetracking

Měření oční aktivity je nejjednodušší definicí eyetrackingu. Kam se lidské oko dívá, co ignoruje, kdy mrkne, jak reaguje měřený subjekt na různé podněty, to všechno

se zařazuje do eyetrackingu neboli sledování pohybu očí. Koncept je velmi jednoduchý, ale interpretace jsou docela složité. Níže je uveden stručný přehled technologie sledování očí a jejich aplikace.

1.4.1 Proces sledování očí v praxi

Data pro sledování očí jsou shromažďována pomocí dálkového eyetrackeru. I když v současné době existuje mnoho různých typů očních sledovačů, obvykle obsahují dvě stejné komponenty: zdroj světla a fotoaparát. Světelný zdroj je nasměrován k oku. Kamera sleduje odraz světelného zdroje společně s viditelnými očními rysy, jako je zornice nebo duhovka. Data jsou využita k extrapolaci rotace oka a určení směru pohledu. Eyetracker také dokáže sbírat další informace, jako je například frekvence mrknutí nebo změna průměru zornice [27].



Obr. 1.17: Heat mapy z eyetrackeru [24]

Existuje mnoho různých analýz očních dat. Nejběžnější je analýza vizuální informace z jednoho nebo obou očí přes rozhraní, kterým je nejčastěji obrazovka počítače. Obraz z každého oka je převeden do sady pixelových souřadnic. Odtud se na výstupu zkoumá přítomnost nebo absence bodů na všech souřadnicích obrazu. Tento typ rekonstrukce se využívá k určení objektů, na které je upoutána pozornost, jak rychle se oko pohybuje, jaký obsah je přehlížen a další analýzy související s pohledem. Ve vizualizaci jsou generovány heat mapy (obrázek 1.17), jenž znázorňují části obrazu, které uživatele zaujaly [27].

1.4.2 Oblasti využití

S pokročilou počítačovou technikou existuje celá řada disciplín využívající systémy pro sledování očí. Seznam oblastí využívajících různé metody sledování očí:

- Psychologie
- Laserová refrakční chirurgie
- Marketing
- Sportovní trénink

- Simulátory řízení vozidel
- Virtuální realita
- Detekce únavy
- Počítačové vidění
- Komunikace pro tělesně postižené
- Ovládání počítačů a chytrých zařízení
- Lékařský výzkum
- EEG
- Geoinformatika a mnoho dalších [24]

Lze řídit auto, číst časopis, surfovat po internetu, prohledávat uličky supermarketu, nakupovat, hrát videohry, absolvovat sportovní trénink, sledovat film, prohlížet obrázky na svém mobilním telefonu, ... Až na pár výjimek může být počítačově sledováno cokoliv. Člověk používá oči neustále, pochopení toho, jak fungují, je velkým klíčem k úspěchu v rozmanitých vědních disciplínách. Automobilový, lékařský i obranný průmysl používá sledování očních pohybů pro zajištění bezpečí. Marketingový, zábavný, obalový a internetový trh výrazně prospívá ve studiu zákazníka a jeho vizuálním chování (co na první pohled nejvíce zaujme). Speciální výzkum v této oblasti přinesl signifikantní průlom v psychologii i fyziologii. Každý den roste seznam aplikací, protože je eyetracking využíván stále více a více [27].

Komerční aplikace

Do velké kategorie komerčních aplikací patří zejména testování webů, reklam, inzercí, obalů a automobilového inženýrství. Komerčními systémy studují reakce subjektů na předložený cílový podnět. Eyetracker zaznamenává aktivitu očí, kam na cílový podnět se měřený subjekt v danou chvíli dívá. Pro zisk výsledků jsou data statisticky zpracována a převedena do podoby heat mapy (viz obrázek 1.17). Zkoumáním různých typů očních pohybů (kapitola 1.1) a jejich trajektorií lze vytvořit rozsáhlou analýzu efektivnosti daného média nebo produktu. K cílovým podnětům mohou patřit například mapy, webové stránky, časopisy, obrázky, videa, zprávy, filmy, noviny, bankomaty, památky atd [24].

Kromě analýzy vizuální pozornosti mohou být data z eyetrackeru zkoumána pro měření kognitivního stavu a pracovní zátěže subjektu [27].

Aplikace pro tělesně postižené osoby

Aplikace umožňují tělesně postiženým osobám mluvit, surfovat po internetu, komunikovat na sociálních sítích a vykonávat další aktivity jen za pomoci svých očí. Systémy počítají i s charakteristickými pohyby, jako jsou například následky mozkové obrny nebo jiných typů postižení. Fungují i pacientům, kteří nosí brýle [24].

1.4.3 Konkurenční eyetrackery

Eyetrackery se dělí do dvou hlavních kategorií: náhlavní a bezkontaktní. Do náhlavních se řadí různé druhy brýlí a helem. Nevýhodou těchto systémů je omezenost pohybu. Mezi takové patří například I4Control nebo Tobii Pro Glasses 2. Bezkontaktní systémy nejsou připevněny na tělo uživatele. Mohou obsahovat i více kamer pro snímání informace o natočení hlavy. Patří sem snímač Tobii Pro Spectrum.

I4Control byl vyvinut na katedře kybernetiky ČVUT v Praze. Skládá se z malé kamery, která je připevněna k obroučce brýlí, ty jsou k počítači připevněny USB portem. Kamera snímá jedno oko, detekuje zornici a počítá místo pohledu. Myš se pohybuje po obrazovce podle uživatelova pohledu. Je umožněno i klikání pomocí zavření snímaného oka na danou dobu. Brýle jsou využity zejména osobami s tělesným a psychických handicapem. Ukázka je na obrázku 1.18 [49].

Tobii Pro Glasses 2 od společnosti Tobii fungují na podobném principu. Na rozdíl od I4Control neslouží pro přímé ovládání počítače, ale pro pochopení lidského chování ve společnosti. Největší využití je v automobilovém průmyslu například při analýze sledování dopravních značek. Ukázka je na obrázku 1.19 [50].

Obr. 1.18: Brýle I4Control [49]

Obr. 1.19: Brýle Tobii Pro Glasses 2 [50]

Tobii Pro Spectrum je bezkontaktní systém sledování směru pohledu, který se skládá ze dvou kamer. Každá kamera natáčí s 600 FPS. Umožňuje detekovat směry pohledu i s natočením hlavy do hodnoty 30°. Před každým měřením je nutná kalibrace. Ukázka je na obrázku 1.20 [51].

Obr. 1.20: Bezkontatktní systém Tobii Pro Spectrum [51]

1.4.4 Požadavky eyetrackeru

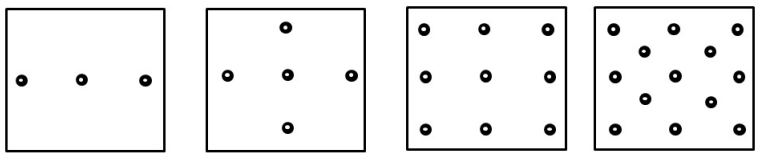
Požadavky na použitelnost zařízení shrnují řadu podmínek, které by mělo ideální zařízení splňovat:

- Nemá přímý kontakt s uživatelem
- Pracuje realtime
- Dokáže měřit všechny tři stupně volnosti – horizontální pohyb, vertikální pohyb a rotace oka
- Je snadno použitelné na velkou škálu aplikací
- Musí detekovat i jemné pohyby oka
- Správně snímá oči z nezakryté části hlavy nebo obličeje
- Je funkční i se změnou jasu ve scéně [28]

1.4.5 Realizace eyetrackeru

Nejprve je nutno sestavit detektor středu zornice oka, jejíž pohyb je sledován. Poté je nutné určit fixní referenční bod, se kterým je střed zornice porovnáván. Může to být střed oka, reflexe světelného zdroje, oční koutek nebo jiný manuálně zvolený bod. Pomocí těchto bodů lze vytvořit vektor s počátkem v referenčním bodě a koncem ve středu zornice. Vektor ukazuje směr vychýlení zornice oproti referenčnímu bodu, nikoliv směr pohledu oka. Z vektoru je pro snímání pohybu extrahována informace o velikosti a směru.

Vždy před použitím eyetrakeru musí dojít k jeho kalibraci. Kalibrace probíhá naměřením několika kalibračních bodů. Při pohledu do těchto bodů je uložena informace o velikosti a směru vektoru. Čím více kalibračních bodů je nastaveno, tím přesnější je následné určení pohledu. Poloha kalibračních bodů může být náhodná nebo pevně daná. Příklad rozmístění kalibračních bodů je na obrázku 1.21.



Obr. 1.21: Příklad rozmístění kalibračních bodů [46]

Všechny další body na obrazovce jsou počítány interpolací přes celou scénu. Velikost interpolovaného pole závisí na požadovaném rozlišení eyetrackeru. Dalším krokem je hledání nejpodobnějšího naměřeného vektoru v interpolovaném poli. Bod, jehož vektor má nejpodobnější velikost a směr s právě naměřeným vektorem, je označen jako bod pohledu. Přesnost eyetrackeru velmi záleží na správné kalibraci.

Pro dosažení velké přesnosti eyetrakeru je důležité velké rozlišení kamery, přesná detekce středu zornice, správná kalibrace, omezení pohybů hlavy nebo zapojení detekce pohybu hlavy a dostatečné rozlišení scény pro zobrazení výsledků.

2 Detekce očí

V této kapitole je popsáno použité zařízení Rasperry Pi a kamerový modul RPi Camera (H). Dále je popsána implementace detekce obličeje a očí. Zde jsou srovnány dvě metody: detekce pomocí Haarových příznaků a detekce pomocí Dlib s orientačními body obličeje. Kapitola pokračuje detekcí zornice a končí předzpracováním obrazu s potlačením reflexe světla, úpravou kontrastu a morfologickými transformacemi obrazu.

2.1 Příslušenství

Ke sledování očí je použit jednodeskový počítač Raspberry Pi 3 Model B s kamerovým modulem RPi Camera (H). Detekce očí probíhá v programovém jazyku Python 3.7 s implementovanou OpenCV knihovnou.

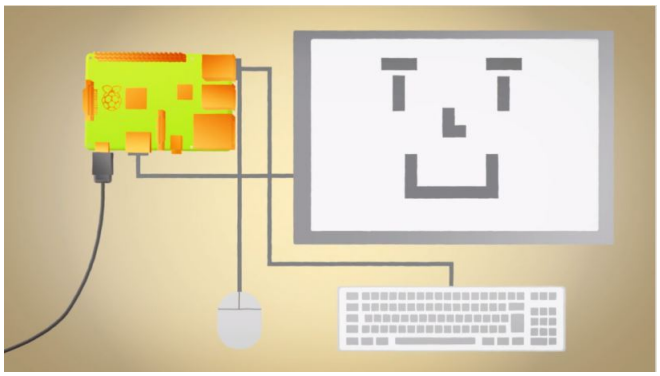
2.1.1 Raspberry Pi 3 Model B

Raspberry Pi je levný počítač o velikosti kreditní karty, který lze připojit k monitoru počítače nebo k televizoru pomocí HDMI výstupu. K ovládání postačí standartní klávesnice a myš s USB koncovkou. Napájení je řešeno síťovým adaptérem, nabíječkou k mobilu nebo power bankou s koncovkou micro USB. Připojení k internetu je možné pomocí standartního síťového UTP kabelu. Zapojení ukazuje obrázek 2.1. Prvotním cílem bylo umožnit lidem všech věkových skupin rozšiřovat své znalosti v oblasti práce na počítači a programování v jazycích Scratch a Python. Je schopen dělat vše, co lze očekávat od stolního počítače, od surfování po internetu a sledování videa ve vysokém rozlišení až po tvorbu tabulek a hraní her. Použití počítače Raspberry Pi bylo zvoleno s ohledem na jednoduché připojení kamerového modulu, kompatibilitu s programovým jazykem Python, jednoduchost aplikace knihoven pro zpracování obrazů a videí, cenu a celkovou dostupnost [29].

Velkou výhodou je schopnost interakce s okolím. Raspberry Pi lze najít v celé řadě projektů digitální tvorby od hudebních nástrojů a rodičovských detektorů až po meteorologické stanice a zpívající ptačí budky s infračervenými kamerami. Výrobci se snaží o celosvětové rozšíření hlavně mezi děti, aby se naučily programovat a porozuměly principům fungování počítačů [29].

2.1.2 RPi Camera (H)

S RPi kamerou lze natáčet videa a pořizovat fotografie. Připojení k Raspberry probíhá přes flex kabel. Kamera podporuje všechny Pi revize. Speciální úprava tzv. rybí



Obr. 2.1: Zapojení Raspberry Pi [29]

oko zajišťuje velmi široké zorné pole. Na pravé a levé straně kamery jsou umístěny infračervené LED, které podporují noční vidění. Obsahuje 5 Mpix OV5647 senzor s nastavitelnou vzdáleností zaostření. Tabulka 2.1 uvádí specifikace kamery [30].

Kamera byla vybrána s ohledem na dobré rozlišení, kompatibilitu s Raspberry Pi a jednoduchost použití.

Tab. 2.1: Specifikace kamery [30]

Velikost CCS čipu	¼ palce (0.635 cm)
Clona (F)	2,35
Ohnisková vzdálenost	3,15 mm
Úhel pohledu (úhlopříčka)	160° (běžné kamery obvykle 72°)
Nejlepší rozlišení	1080 p
Výstupní napětí	3,3 V
Velikost	25 mm x 24 mm

2.1.3 Nahrávání videa

Nahrávání videa pomocí RPi Camery lze realizovat dvěma postupy. Buď lze využít knihovnu Picamera nebo již zmiňovanou knihovnu OpenCV. Po testování obou metod, bylo rozhodnuto využití pouze knihovny OpenCV pro následnou detekci. Rozhodnutí bylo učiněno na základě zpoždění streamovaného videa. Při pouhém streamování videa pomocí knihovny Picamera bez jakékoliv detekce je viditelné zpoždění okolo jedné sekundy, u knihovny OpenCV je zpoždění i s aplikovanou detekcí signifikantně menší.

2.2 Detekce obličeje a očí

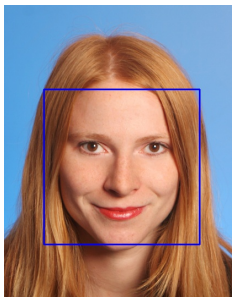
Před samotnou aplikací algoritmu pro sledování pohybu očí dochází k detekci částí, které lze sledovat. V tomto případě je hledán střed zornice, což je i střed duhovky, takže nezáleží, zda bude detekována duhovka nebo zornice. Nejdříve je detekován obličej a v něm jsou hledány oči. To proto, aby byla detekce očí z obrazu co nejpřesnější a nebyly detekovány falešné struktury podobné očím, které leží mimo obličej. Pro detekci obličeje byly testována dvě metody. První je pomocí Haarových příznaků, druhá je s využitím knihovny Dlib s Landmark příznaky. Obě metody jsou popsány níže.

2.2.1 Detekce oblasti očí pomocí Haarových příznaků

Detekce struktur pomocí Haarových příznaků je metoda strojového učení, více vysvětleno v kapitole 1.2.4. Metodou lze detekovat obličej, oči, nos, úsměv, dolní končetiny, apod. Algoritmus a jeho popis je rozdělen do dvou částí: detekce obličeje a detekce očí.

Detekce obličeje

V algoritmu je vybrána největší nalezená oblast obličeje, neboť v některých případech detekce, byly falešně detekovány struktury podobné obličeji. Toto se dělo nejčastěji ve videích se špatným osvětlením. Oblasti falešné detekce měly vždy menší plochu než oblasti správné detekce obličeje. Proto je vybrána vždy největší oblast (obrázek 2.2).



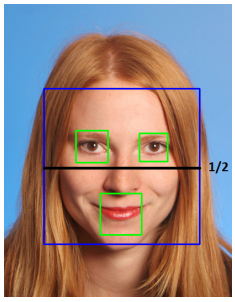
Obr. 2.2: Výsledek detekce obličeje po extrakci Haar příznaků s kaskádní klasifikací [37]

Detekce oblasti očí

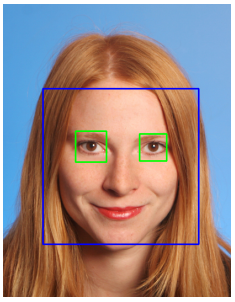
Detekce oblasti očí probíhá podobně jako detekce obličeje, pouze jsou použity jiné filtry a to z kategorie natrénované na očích. Do funkce vstupuje výstřižek detekova-

ného obličeje v předchozím kroku, protože oči se vyskytují jen v oblasti obličeji. Tím se zabrání falešným detekcím v oblastech mimo obličej. Výsledek ale stále není přijatelný, protože oči jsou touto metodou hůře detekovatelné. Často se vyskytují falešně detekované oblasti i na obličeji. Příčinou může být například to, že tvary očí jsou v populaci (trénované množině) rozdílnější než tvary obličeje. Bývá to často oblast nosu, obočí a úst. Proto je algoritmus dále upraven, aby byla detekce co nejlepší. Obrázek 2.3 zobrazuje všechny detekce po použití neupraveného algoritmu.

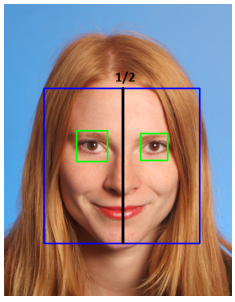
Algoritmus je déle upravován tak, aby správně detekoval pouze oči a aby dokázal rozlišit pravé a levé oko. Je využito znalosti, že oči se vždy nachází na horní polovině obličeje. Detekční okno je zmenšeno a oči jsou hledány pouze v horní polovině (obrázek 2.4). Při hledání očí v horní polovině obličeje se zabrání falešné detekci očí v dolní polovině obličeje. Vhodné je také přidat rozlišení pravého a levého oka. Nejjednodušší metoda vyplývá z anatomie. Provede se rozměření obrazu. Obraz je rozpůlen podle svislé osy (obrázek 2.5). Pravé oko je potom na pravé půlce obličeje, levé oko na levé půlce obličeje (obrázek 2.6).



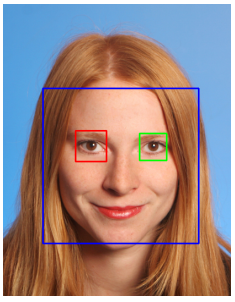
Obr. 2.3: Rozpůlení obrazu [37].



Obr. 2.4: Pravé a levé oko [37].

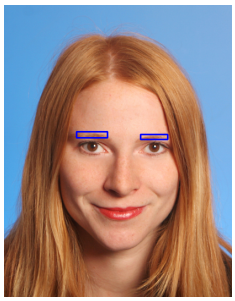


Obr. 2.5: Detekce obličeje a očí [37]

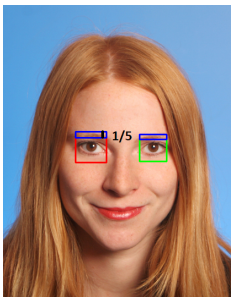


Obr. 2.6: Detekce obličeje a očí v horní polovině obličeje, rozlišení levého a pravého oka [37]

Pro optimální detekci je lepší, aby prohledávaná oblast obsahovala co nejméně jiných struktur a oblastí, které by mohly být zdrojem falešné detekce. Z obrázku 2.6



Obr. 2.7: Nalezené obočí [37]

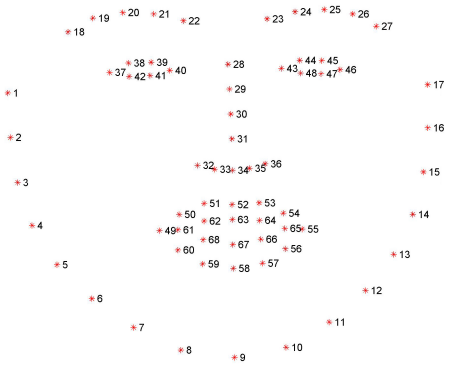


Obr. 2.8: Rozměření obrazu pro odstranění obočí [37]

je zřetelné, že v detekované oblasti oka (červený a zelený čtverec) se nachází kousek obočí. Ten lze také odstranit empiricky. Obočí většinou zasahuje do 1/5 detekovaného čtverce. Tato oblast je z okna odečtena (obrázek 2.7), výsledkem je obdélníková oblast pro přesnější detekci zornice (obrázek 2.8).

2.2.2 Detekce oblasti očí pomocí Dlib s orientačními body obličeje

Dlib je moderní sada nástrojů obsahující algoritmy strojového učení a nástroje pro vytváření komplexního softwaru. Mimo jiné obsahuje také funkci pro detekci obličeje a jeho struktur. Dokáže najít body odpovídající významným strukturám na obličeji. Detektor je schopen najít pozici všech 68 bodů v obličeji a tím mapovat lidskou tvář s velkou přesností. Mapa bodů, které dokáže detektor identifikovat, je zobrazena na obrázku 2.9.



Obr. 2.9: Mapa orientačních bodů obličeje [34]

Obr. 2.10: Příklady detekce pomocí Dlib s orientačními body tváře [43]

Detektor je navržen pomocí metody Histogram of oriented gradients (HOG) v kombinaci s lineárním klasifikátorem, image pyramid a posuvným oknem pro de-

tekci scény. Odhad pozice je zakomponován pomocí další Dlib implementace od Vahid Kazemi a Josephine Sullivan v příspěvku [33]. K trénování detektoru byl využit dataset tváří s orientačními body iBUG300-W z práce [34]. Příklad detekce na různých tvářích je na obrázku 2.10.

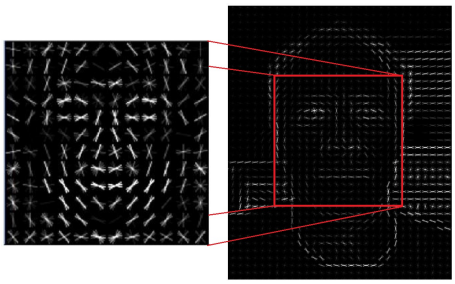
Obr. 2.11: Ukázka detekce obličeje a očí [37]

Souřadnice bodů, které náleží očím, lze jednoduše extrahovat z výstupu funkce *get_frontal_face_detector()*, která je součástí Dlib. Jsou to body 37 až 42 pro pravé oko a 43 až 48 pro levé oko. Pak lze jednoduše pomocí znalosti souřadnic těchto bodů vystřihnout oblast očí z obrazu. V této oblasti bude dále prováděna detekce zornice nebo duhovky. Protože je získaná oblast přesná a neobsahuje mnoho elementů, které by mohly být zdrojem falešné detekce, očekávají se dobré výsledky u dalších algoritmů. Detekci obličeje a očí je na obrázku 2.11.

Aplikace metody HOG na obličej

Metoda HOG počítá v pro každý pixel gradient tak, že porovnává intenzitu pixelu s pixely sousedními. Výstupní obraz obsahuje místo pixelů šipky, které jsou orientovány ve směru gradientního toku od světlých pixelů po tmavé pixely. Detekce obličeje a jeho struktur probíhá porovnáním natrénovaného HOG obrazu s aktuálním HOG obrazem, ve kterém je obraz detekován. Předloha projede jako posuvné okno celým obrazem v různých velikostech a hledá podobné struktury, ukázka na obrázku 2.12. V případě kladné odezvy je metoda schopna vypsat pozice obličeje a jeho struktur (obrázek 2.10) [44].

Metoda bohužel nedokáže detekovat zornici nebo duhovku neboť takové struktury jsou HOG metodou velmi špatně zachytitelné, proto je použita pouze na nalezení oblasti, kde budou dále struktury hledány.



Obr. 2.12: Ukázka detekce tváře z HOG obrazů [44, 45]

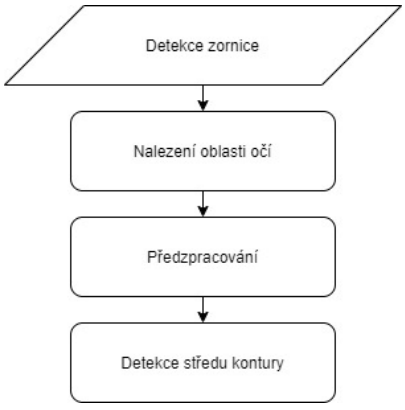
2.3 Detekce zornice

Z obličeje je selektována oblast očí, ve které dochází k detekci zornice. Byly testovány dvě metody: Detekce shluku a Houghova transformace pro detekci kruhů. Detekce shluku byla vybrána jako vhodnější metoda pro detekci zornice s ohledem na přesnost detekce a jednoduché nastavené parametrů shluku.

V kapitole 2.2.1 je popsána detekce oblasti očí pomocí Haarových příznaků a pomocí knihovny Dlib s orientačními body obličeje. Kvůli potřebě přesné detekce středu oka pro vykreslení vektoru v dalších krocích návrhu eyetrackeru je pro detekci očí používána pouze metoda s knihovnou Dlib. Metoda navíc umožní nalezení přesnější oblasti oka, proto je také vhodnější pro samotnou detekci zornice.

Ve výřezu oblasti oka je aplikováno nalezení kontur a jejich centroidu. Kontury jsou hledány v binárním obrazu, jenž je vytvořen s prahem, který se dá nastavit posuvníkem přímo v hlavním okně. Souřadnice nalezeného centroidu jsou souřadnicemi zornice. Obrázek 2.13 ukazuje detekci zornice v oblasti očí nalezené pomocí Dlib. Vývojový diagram je na obrázku 2.14.

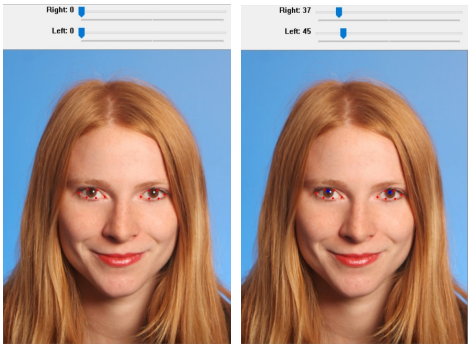
Obr. 2.13: Ukázka detekce zornice [37]



Obr. 2.14: Vývojový diagram detekce zornice

2.3.1 Posuvníky pro nastavení prahu pravého a levého oka

Převod šedotónového obrazu na binární je esenciální pro zobrazení kontur a detekci jejich centroidu. Práh je volen přímo v okně pomocí posuvníků (obrázek 2.15) pro pravé a levé oko zvlášť. To zajistí velmi přesnou detekci zornice i při nerovnoměrném osvětlení obličeje. Vliv prahu na pravé oko je demonstrován na obrázku 2.16, jedná se o binární obraz v HSV barevném formátu. Na tuto výseč je také aplikováno předzpracování, jenž je popsáno v kapitole 2.4. Bez předzpracování by zornice nebyla tak jednoznačná a střed nalezeného objektu by nemusel odpovídat středu zornice.

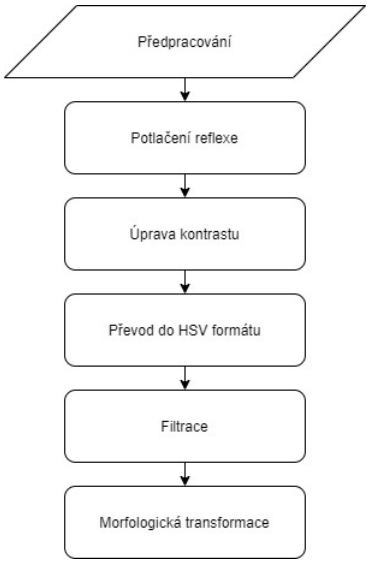


Obr. 2.15: Ukázka posuvníků pro změnu prahu [37]

Obr. 2.16: Vliv prahu pro pravé oko na binární obraz, vlevo práh 0, vpravo práh 37

2.4 Předzpracování obrazu

Předzpracování je realizováno před samotnou detekcí zornice v nejmenším nalezeném okně, vývojový diagram je na obrázku 2.17. Pro lepší detekci zornice je využita filtrace, úprava kontrastu a morfologické operace (erose, dilatace, morfologické otevření a uzavření). Metody jsou popsány níže. Binární obraz před předzpracováním a po předzpracování je na obrázku 2.18. Detekce nemusí velikostně odpovídat zornici nebo duhovce, protože v další části programu jsou využity pouze souřadnice středu nalezeného shluku.



Obr. 2.17: Vývojový diagram předzpracování obrazu

Obr. 2.18: Porovnání oblasti oka, binárního HSV obrazu a předzpracovaného obrazu

2.4.1 Reflexe světla

Korneální i jiné reflexe světla velmi výrazně ovlivňují spolehlivost a přesnost detekce zornice i duhovky. Proto je esenciální ji při detekci těchto struktur potlačit či odstranit. Reflexe se může vyskytnout u metody tmavé zornice, jenž popsáno v kapitole 1.3.3. A to v případě, že světelný zdroj přímo osvětluje obličej. Pro správnou detekci obličeje i očí se většinou světelný zdroj používá. Tvar reflexe je úměrný tvaru, velikosti a síle světelného zdroje. Úspěšnost potlačení reflexe je závislá na správné lokalizaci, segmentaci a také na intenzitě reflexe. Je vhodné odstranit tyto reflexe ještě před předzpracováním, protože později mohou být rozmazány filtry.

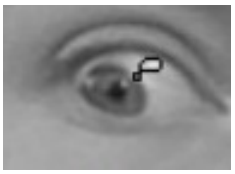
Nalezení reflexe

Reflexi zastupují vysoké hodnoty jasu. K nalezení je využit YCbCr barevný model, jenž je více vysvětlen v kapitole 1.2.2. Díky složce Y (jas) je velmi jednoduché najít oblasti s vysokými hodnotami jasu, které odpovídají reflexi. V algoritmu je nastaven práh, který nalezne oblasti s vysokými hodnotami jasu pro převod na binární obraz. Níže na obrázku 2.20 je znázorněna detekce na originálním obraze 2.19.

Potlačení reflexe

Pixely, které jsou označeny jako reflexe jsou potlačeny. Takové pixely lze jednoduše nahradit nulovou hodnotou šedi (černá barva) nebo střední hodnotou šedi (šedá

Obr. 2.19: Originální obraz



Obr. 2.20: Detekce reflexe

barva). To by ale mohlo způsobit negativně pozitivní detekci zornice v oblasti mimo zornici nebo pozitivně negativní detekci zornice v oblasti zornice. Proto se zvolil způsob nahrazení hodnoty každého reflexního pixelu hodnotou levého sousedního pixelu ve směru osy X. Pokud se reflexe nachází v okrajových hodnotách výřezu obrazu v ose X, nahradí se hodnota pixelu hodnotou horního sousedního pixelu ve směru osy Y. Tento postup je aplikován, i když se nepředpokládá, že by se reflexe nacházela v okrajových bodech. Hodnoty reflexe se tímto způsobem potlačí více v oblasti zornice a méně v oblasti mimo zornici, což je pro tento případ ideální.

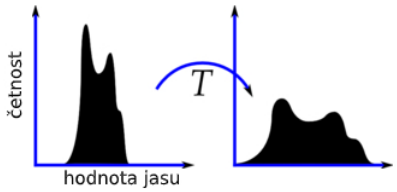
2.4.2 Úprava kontrastu

Úpravou kontrastu obrazu dochází ke zlepšení subjektivního dojmu a informačního využití. Pokud se zvolí správné postupy, tak dojde velmi výrazně ke zvýšení diagnostické výtěžnosti. Použít se dají lineární i nelineární transformace [31].

Gama korekce byla v této práci aplikována na obraz s potlačenou reflexí. Parametr gama je zvolen empiricky na 1,2.

Ekvalizace histogramu

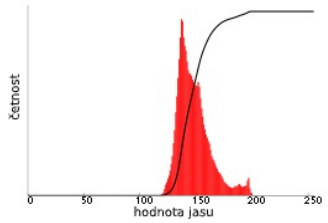
Vychází ze statistických metod zpracování obrazu. Histogram poskytuje základní informaci o úrovni jasu v obraze. Ekvalizace histogramu je metoda, která upravuje kontrast obrazu s pomocí histogramu. Metoda pomáhá k zisku lepší informace z obrazu díky zvyšování lokálního kontrastu v obraze, právě když jsou dvě stejné hodnoty jasu blízko u sebe. Díky těmto úpravám je dosaženo lepšího rozložení jednotlivých jasových hodnot v histogramu (obrázek 2.21) [35].



Obr. 2.21: Ukázka roztažení histogramu od 0 do 255 jasových hodnot [35]

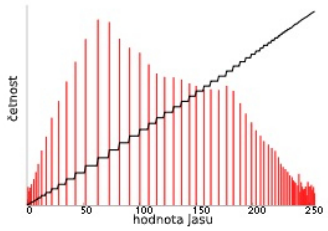
Na originálu (obrázek 2.22 a jeho histogram 2.23) s nerozloženými jasovými hodnotami přes celý interval (od 0 do 255), byla provedena ekvalizace histogramu. Výsledek po ekvalizaci znázorňuje obrázek 2.24 a obrázek 2.25.

Obr. 2.22: Originální obraz [36]



Obr. 2.23: Originální histogram[36]

Obr. 2.24: Obraz po ekvalizaci [36]

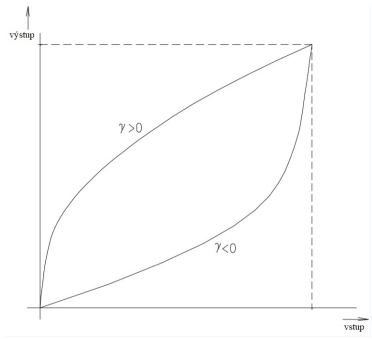


Obr. 2.25: Histogram po ekvalizaci [36]

Gama korekce

Pokud digitální kamera zachytí dvakrát větší počet fotonů, je výstupní signál dvojnásobně zesílen. Tak ale lidské oko nefunguje. Vztah zde není lineární, jako tomu je u kamery. Lidské oko nevidí dvojnásobný signál, ale pouze o zlomek jasnější, aby se oči během slunečného dne nepoškodily. Jsou navíc mnohem citlivější na změny tmavých tónu než světlých. Díky této vlastnosti je člověk schopen dobré orientace v tmavé místnosti. Proto se využívá gama korekce, která upravuje bitovou hloubku obrazu a rozloží barvy tak, aby z části kompenzovala tento rozdílný převod signálu v umělém a přírodním orgánu. Velmi jednoduše řečeno se stíny se zesvětlí, vysoké intenzity se utlumí. Gama korekce je dána jednoduchým vztahem 2.1, funkce je součástí knihovny OpenCV. Parametr gama lze měnit, průběh funkce gama při jeho změně lze vidět na obrázku 2.26. Příklad gama korekce lze vidět na obrázku 2.27 [39, 40].

$$I_{vystup} = I_{vstup}^{\gamma} \tag{2.1}$$



Obr. 2.26: Průběh gama korekce [31]

Obr. 2.27: Originál; gama = 0,5; gama = 1,2; gama = 2,2 [37]

2.4.3 Využití HSV barevného formátu

Obraz je převeden do HSV barevného formátu, neboť následné binární výsledky v HSV formátu se empiricky projevily jako přesnější než v RGB formátu. Formát HSV je více popsán v kapitole 1.2.2.

2.4.4 Filtrace

Dále je obraz filtrován mediánovým filtrem pro odstranění šumu typu sůl a pepř s maskou o velikosti 5x5 pixelů a bilaterálním filtrem s parametrem d = 9 a sigmou v hodnotě 75, jenž vyhlazuje obraz a současně zachovává hrany. Parametry a velikosti masek byly nastaveny empiricky.

Gaussův filtr

Je lineární filtr, který filtruje šum s náhodným rozdělením. Gaussův filtr je rozšířením průměrování v obraze pomocí Gaussova rozložení. Používá se konvoluční matice se zvýšenou vahou na středový bod nebo i jeho okolí. Nevýhodou Gaussova filtru je skutečnost, že rozmazává hrany. Proto se začal používat bilaterální filtr, který je

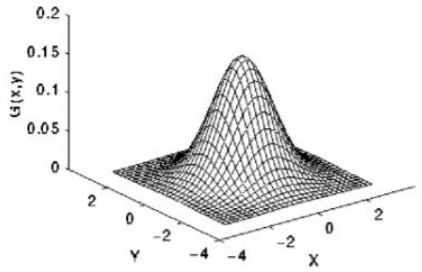
vysvětlen níže. 2D Gaussův filtr definován jako:

$$G_2(x,y) = \frac{1}{2\pi\sigma^2} * e^{-\frac{(x-\mu)^2+(y-\mu)^2}{2\sigma^2}}, \tag{2.2}$$

kde $[\mu, \mu]$ je střední hodnota a σ^2 je rozptyl. Konvoluční maska může mít tvar:

$$\frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}. \tag{2.3}$$

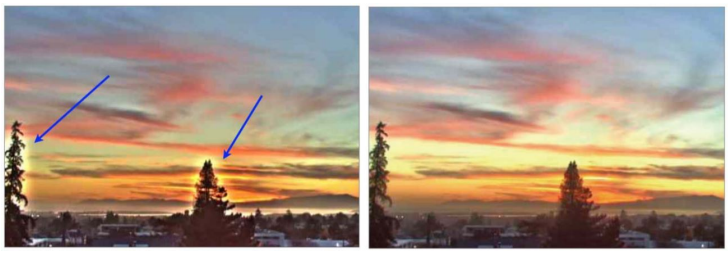
Celková světlost obrazu se nemění, neboť součet všech složek konvoluční matice vynásobených vahou dává výsledek 1. 2D Gaussova konvoluční maska je ilustrována níže (obrázek 2.28) [41].



Obr. 2.28: 2D Gaussova konvoluční maska [35]

Bilaterální filtr

Patří do skupiny lokálních filtrů, které vyhladí obraz a zároveň ponechají kontrast hran. To je dosaženo kombinací prostorové (poloha bodu) a jasové závislosti pixelů. Hodnoty prostorově blízké a jasově podobné jsou upřednostněny. Prostorová blízkost je interpretována dolní propustí. Spočítá se vážený průměr bodů v okolí, váha se vzdáleností klesá podle Gaussovi funkce nikoli lineárně. Tímto postupem je následována myšlenka velmi podobných bodů v blízkém okolí, proto je možné je průměrovat. S rostoucí vzdáleností však pravděpodobnost klesá, velmi tedy záleží na zvoleném okně. Kontrast hran se zachová díky začlenění váhovací funkce, která zohledňuje podobnost pixelů v okně. Výsledná hodnota je dána jako normalizovaná suma váhovaných hodnot pixelů v okolí. Váha je určena ze vzdálenostní a pravděpodobnostní funkce. Využívá se několika přístupů, jenž se při průměrování vyhnou hranám a rozmažou jen některé regiony. Jedním z hojně využívaných přístupů je anizotropní difúze, kde jsou zjišťovány lokální změny a jejich gradienty. Na základě toho se určí okno, ve kterém se bude průměrovat. Kvůli řešení parciálních derivací je největší nevýhodou tohoto filtru nízká efektivita [42].



Obr. 2.29: Gaussův filtr s vyznačeným halo efektem vlevo, bilaterální filtr vpravo [42]

Bilaterální filtr lze aplikovat na jas nebo na jednotlivé kanály. V této práci se filtr aplikuje pouze na jasovou složku obrazu kvůli výpočetní náročnosti. Výhodou bilaterálního filtru oproti Gaussovu filtru je, že bilaterální filtr zabraňuje rozmazání kontrastních hran a tvorbě tzv. halo efektu (obrázek 2.29).

2.4.5 Morfologické transformace

Mezi základní morfologické operace patří erose a dilatace. Použité morfologické operátory v této práci jsou lokální nelineární operátory využívající binární masky, které se aplikují na binární obraz. Na HSV obraz byla aplikována jednoduchá erose s jednou iterací, dilatace s jednou iterací, morfologické otevření a uzavření.

Erose

Erose zmenšuje objekty tak, že odstraňuje výběžky a malé izolované shluky pixelů. Může rozdělovat propojené objekty. Zmenšuje původní velikost obrazu. Pokud kryje maska H o referenčním bodu v x některý pixel objektu v f, pak se vloží jednička do výstupu na polohu x referenčního bodu H ve výstupním obrazu. Erose je definována vztahem 2.4:

$$Y = E_H(X) = \bigcap_{h \in H} X_{-h} \tag{2.4}$$

kde X-h je vstupní množina posunutá o -h, h je poziční vektor bodu z H vzhledem k referenčnímu bodu [31].

Dilatace

Dilatace naopak odstraňuje malé otvory a úzké zálivy. Zároveň zvětšuje původní velikost objektu a může dojít k propojení dvou sousedních objektů. Pokud kryje maska H o referenčním bodu v x některý pixel objektu v f, pak se vloží jednička do výstupu na polohu x referenčního bodu H ve výstupním obrazu. Dilatace je

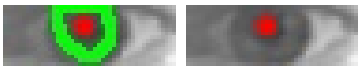
definována následujícím vztahem 2.5:

$$Y = D_H(X) = \bigcup_{h \in H} X_{-h}, \tag{2.5}$$

kde H_{-h} je vstupní množina posunutá o $-h$ [31].

2.4.6 Vykreslení kontur a nalezení jejich středu

Obraz je nyní předzpracován a připraven k hledání kontur a detekci jejich středu. Všechny potřebné funkce jsou obsaženy v knihovně OpenCV a Imutils. Knihovna Imutils spolupracuje s OpenCV tak, že dokáže najít a uložit potřebné parametry kontur. Detekce středu shluku je provedena nalezením centroidu shluku. Výsledek je na obrázku 2.30.



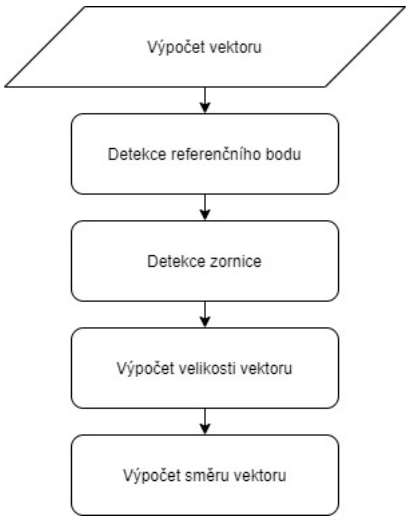
Obr. 2.30: Ukázka detekce kontur a jejich centroidu v oblasti oka

3 Realizace eyetrackeru

Kapitola slouží k popisu návrhu eyetrackeru. Nejprve je určen vektoru směru pohybu zornice. Následně je provedena kalibrace na devíti kalibračních bodech a vytvořena kalibrační mapa. Konečným krokem je hledání nejpodobnějšího právě naměřeného vektoru směru pohledu z očí a porovnávání s kalibrační mapou. Tak se najde souřadnice pohledu na obrazovku. V závěru kapitoly je vysvětleno ovládání programu a využití sledovacích terčů pro ověření přesnosti.

3.1 Vektor směru pohledu

Pro určení směru pohledu je důležité vybrat vhodný referenční bod (začátek vektoru). Byly testovány dvě metody určení referenčního bodu. Prvním je střed oblasti očí, druhým je bod, který odpovídá středu zornice při stisknutí příslušné klávesy. Obě metody mají světlé i stinné stránky. Druhým bodem (konec vektoru) je střed zornice, mezi ním a referenčním bodem je vykreslen a počítán vektor. Dochází k výpočtu velikosti vektoru u a směru vektoru v. Postup je znázorněn ve vývojovém diagramu 3.1.

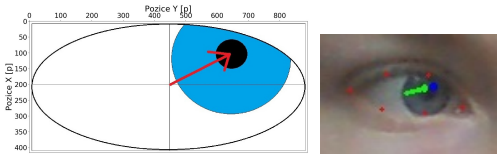


Obr. 3.1: Vývojový diagram vektoru směru pohybu

U prvního případu, kdy je referenční bod určen středem oblasti oka, je potlačen pohyb hlavy v rovině rovnoběžné s obrazovkou, který je zásadní při eyetrackingu. Rotační pohyb hlavy tuto metodu oslabí stejně jako závislost na pohybu očích víček, neboť oblast očí kopíruje oční víčka. Z toho plyne, že pokud měřený subjekt při měření pohybu očí pohne očními víčky jinak, než při kalibraci, bude do výsledku zanesena chyba přímo úměrná odlišnosti pohybu očních víček. U druhého případu,

kdy je referenční bod určen manuálně v celém obraze není potlačen pohyb hlavy. Jakýkoliv pohyb hlavy (rotační i rovnoběžný s obrazovkou) ve vztahu ke kameře nebo obrazovce velmi negativně ovlivňuje přesnost výsledku. Ale na druhou stranu tato metoda nezavádí žádnou chybu při pohybu očních víček, jako příklad lze uvést vykulení nebo přivření očí.

Pokud je hlava při měření i kalibraci velmi dobře fixována, je možné využít druhý způsob měření vektoru. Pokud hlava velmi dobře fixována není, je lepší využít první způsob měření vektoru. Ukázka vektoru je na obrázku 3.2. Celkově je přesnější první způsob, neboť nebyl vyvinut systém, který by fixoval hlavu tak, aby docházelo k přesnější detekci u manuálního určení referenčního bodu. Řešením je zapojení systému, který bude snímat pohyby hlavy nebo pevné ukotvení kamery na hlavu uživatele.



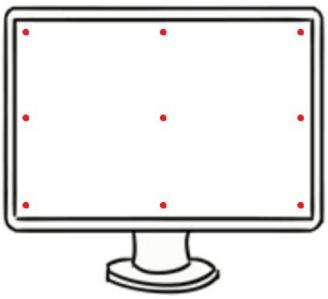
Obr. 3.2: Schéma vektoru a ukázka reálného vektoru

Velikost i směr vektoru je počítána pro obě oči a průměrována. Tím se zajistí stejný pohled pro obě oči při nedokonalé detekci některých struktur. Je potřeba poznamenat, že vektor neodpovídá přesně pohybu očí. Odpovídá pouze vzdálenosti střed oblasti oka až střed zornice. Proto například při pohledu očí dolů neukazuje dolů.

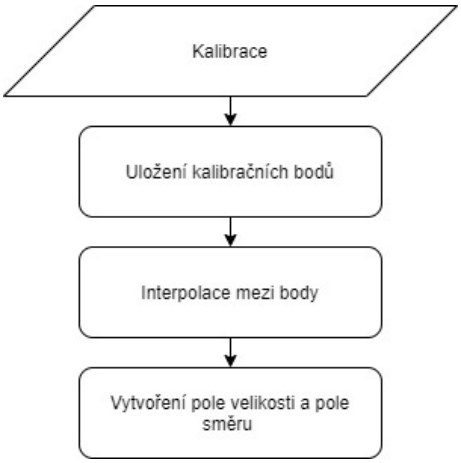
3.2 Kalibrace

V kalibrační fázi oko sleduje kalibrační body, parametry vektoru pro daný kalibrační bod se ukládají pro danou pozici ve scéně. Poté se interpoluje mezi kalibračními body, aby se spočítala daná hodnota vektoru pro každý pixel v obraze. Tím se sestaví kalibrační mapa velikosti a směru vektoru. Velikost kalibračního pole je dána tím, jak velké rozlišení výstupní scény uživatel požaduje a je nastavitelné v parametrech programu. Kalibrační fáze je nejdůležitější fází celého programu, pokud se eyetracker nakalibruje špatně, bude vykazovat chybné výsledky. Vždy se musí kalibrovat na obrazovce, která bude sloužit k eyetrackingu a zobrazení výsledků. Vývojový diagram kalibrace je na obrázku 3.4.

Kalibračních bodů je devět, jejich rozmístění lze vidět na obrázku 3.3. Uložení souřadnic je ovládáno postupným stisknutím číslic 1-9 na klávesnici.



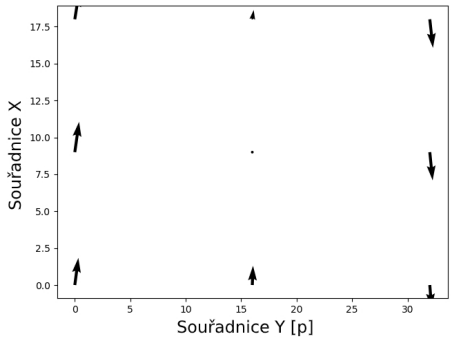
Obr. 3.3: Rozmístění kalibračních bodů



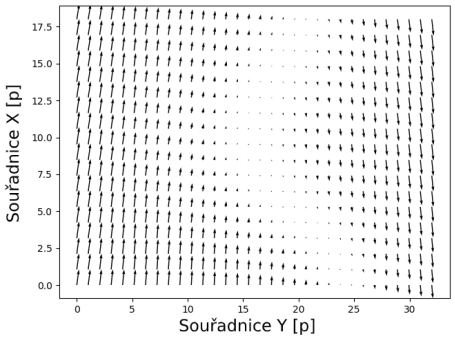
Obr. 3.4: Vývojový diagram kalibrace

3.2.1 Interpolace

Interpolace mezi kalibračními body je realizována interpolační funkcí z knihovny Scipy. Interpolace probíhá zvlášť pro velikost a směr vektoru. Nejprve jsou vytvořeny dvě prázdná pole o velikosti požadovaného výstupu. Poté jsou metodou bikubické interpolace počítány body mezi kalibračními body. Výstupem jsou dvě interpolovaná pole pro velikost a směr vektoru. Na obrázku 3.5 jsou zobrazeny vstupní vektory kalibračních bodů v daných souřadnicích, obrázek 3.6 znázorňuje interpolované pole, oba obrázky jsou s rozlišením 32x18 pixelů (velikost rozlišení je pouze ilustrativní).



Obr. 3.5: Vektory kalibračních bodů



Obr. 3.6: Interpolované pole vektorů

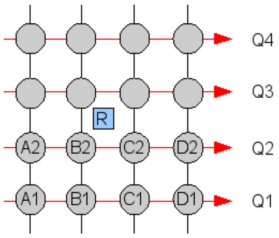
Bikubická interpolace

Bikubická interpolace je 2D variantou kubické interpolace. Jedná se o polynomiální interpolaci třetího stupně, výpočet je proveden pomocí kubického polynomu, znázorněn v rovnici 3.1. Snaží se docílit velké kontinuity jednotlivých pixelů a umožňuje

zapojení velkého počtu originálních bodů. Patří mezi výpočetně náročnější metody, ale dává v tomto případě velmi dobré výsledky [47].

$$y = ax^3 + bx^2 + cx + d \tag{3.1}$$

Interpolace se prokládá křivkou zadanou čtyřmi body. Nejprve se počítá s řádky, později se sloupci. Každý řádek je proložen polynomem a hodnotou hledaného bodu, výsledkem jsou čtyři body Q1 až Q4 (obrázek 3.7). Tyto řídící body slouží pro křivku v druhé dimenzi [48].



Obr. 3.7: Bikubická interpolace [48]

V jedné dimenzi se počítá váha pro čtyři pixely. Zjišťuje se, jak daleko je bod A vzdálený od interpolovaného bodu. Bod A leží mezi dvěma prostředními pixely, proto bude vzdálenost nabývat maximální hodnoty dva. Pokud vzdálenost odpovídá číslu mezi nulou a jedničkou, použije se pro výpočet váhy vzorec 3.2. Pokud bude vzdálenost mezi jedničkou a dvojkou, použije se vzorec 3.3 [47].

$$w = ((A+2)x - (A+3))x^2 + 1, \qquad pro \quad 0 < x < 1 \tag{3.2}$$

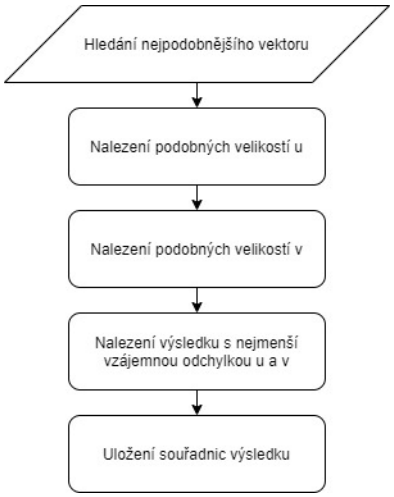
$$w = ((Ax + A)x + 8A)x - 4A, \qquad pro \quad 1 < x < 2 \tag{3.3}$$

3.3 Eyetracking

Kalibrační mapa, která je výstupem interpolace, je nejdůležitější součástí eyetrackingu. Bez ní není možné najít polohu pohledu ve scéně. Kalibrační mapy jsou k dispozici dvě, jedna pro velikost vektoru, druhá pro směr vektoru. Jelikož jsou hodnoty velikosti a směru vektoru poměrově odlišné, dochází k jejich normalizaci. Odečte se minimum a celé pole se vydělí maximální hodnotou, výsledkem je pole s hodnotami od nuly do jedné.

Následně je hledán takový vektor, který má co nejpodobnější velikost a směr. Souřadnice tohoto vektoru odpovídají souřadnicím pohledu. To je realizováno funkcí,

která hledá takové souřadnice, jejichž hodnoty jsou co nejvíce podobné hodnotám právě změřeného vektoru. Funkce nejprve prohledá velikost vektoru u a najde nejlepší možné řešení, které uloží. Pak prohledá směr vektoru v, nejlepší výsledky uloží. Nakonec prohledává uložené výsledky tak, aby byl minimální rozdíl v obou parametrech současně. Nalezené souřadnice se zaznačí do výstupního obrazu. Vývojový diagram na obrázku 3.8 vysvětluje postup. Výsledná detekce pohledu na obrazovku je pak tvořena červenými pixely. Pixely s větší intenzitou červené barvy značí delší pohled na toto místo. Vývojový diagram celého programu je uveden v příloze A.1.



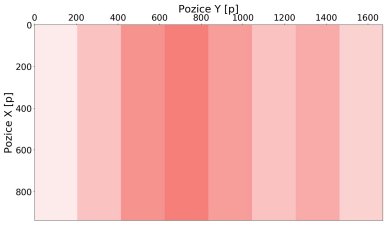
Obr. 3.8: Vývojový diagram nalezení nejpodobnějšího vektoru

3.4 Ovládání programu

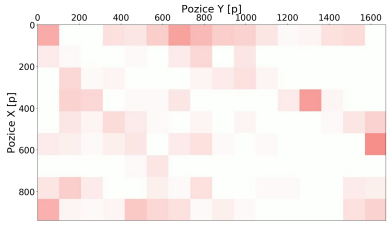
Program je ovládán v okně, klávesovými zkratkami a příkazovým řádkem. Ovládání je velmi intuitivní, ke zmáčknutí klávesové zkratky vždy vyzývá příkazová řádka nebo pokyn v okně. V první fázi se volí rozlišení výstupního okna, jenž je nepřímo úměrné velikosti pixelů.

Různá rozlišení znamenají různou velikost jednoho pixelu na obrazovce. Čím menší počet pixelů, tím větší je oblast jednoho pixelu na obrazovce. Větší rozlišení jsou více výpočetně náročné a metoda určení pohledu musí být dokonalá, protože vybírá z více pixelů. Menší rozšíření jsou rychlé a programy nemusí být velmi propracované, ale na druhou stranu je výsledkem pouze velká potenciální oblast pohledu. Například, kdyby bylo rozšíření 2x1 pixel, tak výsledkem programu je pouze informace, jestli se uživatel dívá na pravou nebo na levou polovinu obrazovky. Rozlišení obrazovky pro tento eyetracker bylo voleno hlavně s ohledem na výpočetní náročnost, neboť Raspberry Pi nezvládá příliš velké rozlišení. Při rozlišení 32x18 pixelů

bylo velmi obtížné naměřit data, proto se zvolila pouze rozlišení 16x9 pixelů, 11x4 pixelů, 9x2 pixelů a 8x1 pixelů. Ukázka výsledného rozmístění pohledu uživatel některých rozlišení je na obrázku 3.9 a 3.10. Červené pixely na obrázcích značí pohled uživatele na tento pixel. Čím více je pixel červený, tím více (déle) se na něj uživatel díval.



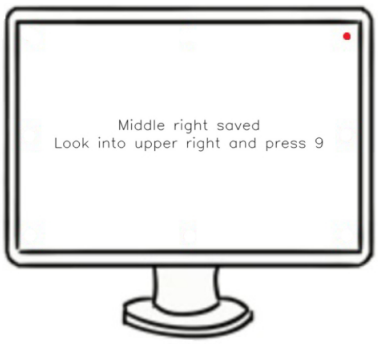
Obr. 3.9: Ukázka rozlišení 8x1 pixelů



Obr. 3.10: Ukázka rozlišení 16x9 pixelů

Ve druhé fázi se nastaví velikost prahů pro pravé a levé oko pro kvalitní detekci zornice, aby program mohl určit její střed (vysvětleno v kapitole 2.3.1). Dále se aktivuje zobrazení vektoru pomocí tlačítka "v". Pokud není nalezen žádný shluk, objeví se upozornění na opravení prahu.

Jakmile je zobrazen vektor, lze pomocí tlačítka "c"aktivovat kalibrační režim. Objeví se bílá obrazovka s červeným kalibračním bodem v levém dolním rohu. Jakmile uživatel sleduje kalibrační bod, zmáčkne klávesu "1", velikost a směr vektoru pohledu se uloží do těchto souřadnic. Kalibrační bod zmizí, objeví se nový bod, který je ovladatelný klávesou "2". Pokyny ke klávesám jsou zobrazeny na obrazovce i v příkazové řádce. Takto program projde všech devět kalibračních bodů obrazovky, po posledním kalibračním bodu se program potvrdí stisknutím enteru. K ovládání slouží klávesy "1"až "9"a "enter". Ukázka kalibrace je na obrázku 3.11.



Obr. 3.11: Ukázka kalibrace

Obr. 3.12: Výsledná detekce pohledu na obrazovce

Příkazová řádka vypíše hodnoty naměřených kalibračních bodů a program vytvoří kalibrační mapy interpolací. Po dokončení interpolace je možné aktivovat eyetracker stisknutím klávesy "e". Nyní lze vidět, které souřadnice uživatel na obrazovce sleduje. Uživatel vidí výslednou detekci (obrázek 3.12) pohledu téměř v reálném čase. Nalezená pozice pohledu je označena zbarvením pixelu na červenou barvu. Čím více pixelů padne do jednoho pixelu, tím tmavší je červená barva. Jakmile se překročí prahová hodnota v kanále pro červenou barvu v RGB obraze, vizuální informace už nepřibývá. Velikost červených pixelů závisí na velikosti zvoleného rozlišení. Eyetracker lze deaktivovat stisknutím klávesy "s".

Klávesami "n"a "m"je aktivován režim terče, který je blíže popsán v kapitole 3.4.1. Klávesa "n"je pro náhodný terč a klávesa "m"je pro animační terč. Pro vypnutí terčů a ukončení programu slouží klávesa "q".

Tab. 3.1: Klávesové zkratky k ovládání programu

Klávesa	**Akce**
v	aktivace vektoru
c	start kalibrace
1-9	zachycení kalibračních bodů
d	odstranění měřených kalibračních dat
e	start eyetrackeru
s	stop eyetrackeru
n	přepočet náhodného terče
m	start animačního terče
q	vypnutí eyetrackeru a uložení výsledků

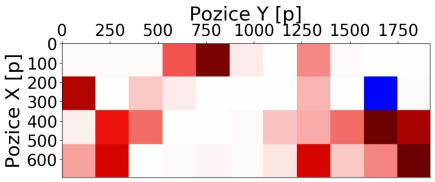
V programu lze stisknout také klávesu "d", která deaktivuje eyetracker, vymaže všechny kalibrované data a deaktivuje vektor. Záleží, ve kterém kroku se uživatel nachází. Program je připraven provést všechny kroky znovu bez nutnosti úplného vypnutí. Vypnutí a uložení výsledků je možné pomocí klávesy "q". Tímto způsobem lze také vykreslit heat mapu aktuálního měření, které trvá od stisknutí klávesy "e"(aktivace eyetrackeru) nebo zapnutí jednoho z terčů po stisknutí klávesy "q"(ukončení eyetrackeru). Pokud program pracoval s terči, uloží se i heat mapa daného terče pro okamžité porovnání výsledků.

Stisknutí klávesových zkratek je kontrolováno knihovnou OpenCV. V různých částech programu lze pomocí klávesových zkratek aktivovat jen některé funkce, aby nedocházelo k chybám a pádu programu. Seznam veškerých klávesových zkratek je v tabulce 3.1

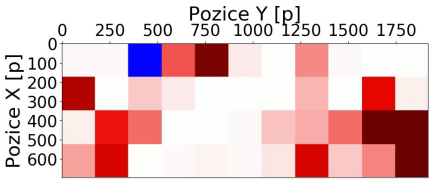
3.4.1 Sledovací terče

V programu lze po aktivaci vykreslování polohy pohledu aktivovat dva typy sledovacích terčů. Sledovací terč je v podobě modrého pixelu zobrazen na obrazovce. Využít lze animaci nebo náhodný terč. Oba terče jsou aktivovatelné pomocí stisknutí klávesy na klávesnici.

Náhodný terč je vhodný pro kontrolu nakalibrování a celkovou přesnost eyetrackeru. Na obrazovku se kreslí červené pixely znázorňující aktuální vypočtený pohled eyetrackerem a modré pixely, které odpovídají terči. Při každém stisknutí dané klávesy se souřadnice terče přepočítají. Uživatel se snaží sledovat terč a vidí, kam dopadají červené pixely, tím lze subjektivně vyhodnotit přesnost a rozhodnout se, zda je potřeba lepší kalibrace či nikoliv. Ukázka pro rozlišení 11x4 pixely je níže, nejdříve je modrý terč vpravo (obrázek 3.13), poté se přemístí vlevo a zůstane po něm červený pixel (obrázek 3.14), protože eyetracker správně našel pohled uživatele na tento terč.

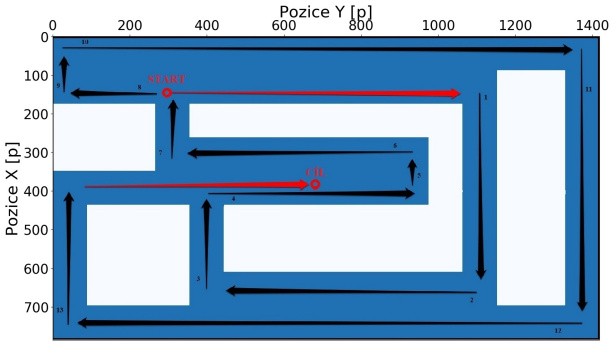


Obr. 3.13: Ukázka přemístění náhodného terče (modrý pixel) před



Obr. 3.14: Ukázka přemístění náhodného terče (modrý pixel) po

Animace terče je tvořena fixní dráhou po které se terč pohybuje s určitou rychlostí. Dráha je určena tak, aby terč projel všechny části obrazovky. Rychlost je vhodně nastavena vzhledem k rozlišení výstupu a lze ji jednoduše měnit. Dráha terče je na obrázku 3.15. Pro přehlednost a lepší soustředěnost na terč se při měření této metody nezobrazují červené pixely eyetrackeru.



Obr. 3.15: Ukázka dráhy terče v animaci

Výsledky obou metod se automaticky ukládají pro další analýzy. Je vykreslena heat mapa terčů a heat mapa eyetrackeru pro porovnání výsledků.

4 Testování přesnosti

V samotném programu lze testovat schopnost eyetrackeru a to nalezení přesné pozice pohledu uživatele. Je možné zvolit ze dvou testovacích režimů, kdy se po obrazovce pohybuje terč v podobě modrého pixelu. Velikost terče je vždy rovna jednomu pixelu a je nepřímo úměrná rozlišení obrazovky. První metoda je náhodný pohyb terče, druhá metoda je animační pohyb terče. Náhodný pohyb terče je spíše určen ke kontrole kalibrace eyetrackeru, aby bylo možné rychle zjistit, zda eyetracker funguje správně. Při každém stisknutí klávesy "n"je terč náhodně přepočítán. Tato metoda není vhodná k následné analýze přesnosti, neboť při každé rychlé změně terče chvíli trvá, než ho oči začnou sledovat. Poté by se zde vyskytovaly velké nepřesnosti včasech po přepnutí terče. Druhá metoda animace pohybu je spuštěna po stisknutí klávesy "m". Terč má vždy stejnou spojitou dráhu v každém měření, rychlost terče lze nastavovat v programu. U obou metod je ošetřen přepis pixelů terčem. Hodnoty pixelu, které se nacházely na stejném pixelu jako terč, jsou po přemístění terče zpět vloženy do obrazu. To stejné platí pro hodnoty pixelu, které padly přímo na terč. Výstupem je pole s hodnotami jednotlivých pixelů obrazovky, heat mapy, pole terče a pole vektoru. Pole terče a pole vektoru obsahuje vektor se souřadnicemi x a y, s velikostí vektoru u a směrem vektoru v. Všechna výstupní pole jsou použita ve skriptu s analýzou přesnosti eyetrackeru.

Výstupem analýzy přesnosti je znovu obrázek heat map, excelovská tabulka s výsledky a vektory veličin, které jsou dále využity k dalším analýzám. Excelovská tabulka obsahuje rozdíl vzdálenosti terče a eyetrackeru pro osu X a Y v pixelech, rozdíl velikosti a směru naměřeného vektoru a nalezeného vektoru, dále stejné analýzy pro normalizovanou velikost vektoru a normalizovaný směr vektoru. Jsou zde automaticky počítána procenta, průměr a maximální hodnota přesnosti pro souřadnice X a Y, velikost vektoru a směr vektoru. Všechny výstupní analýzy jsou ukládány do složky *results*.

4.1 Výsledky

V této části jsou shrnuty naměřené výsledky pomocí hodnocení přesnosti eyetrackeru. Vždy se porovnávají pixely terče versus pixely eyetrackeru nebo hodnoty z vektoru směru pohledu (naměřená velikost vektoru u a směr vektoru v versus nalezená velikost vektoru u a směr vektoru v). Hodnocení přesnosti eyetrackeru je testováno pro tři metody vektoru (značené 1, 2 a 3), dvě metody pohybu terče (značené M a R) a dvě fixace hlavy (značené F a N). Označení metod měření vektoru, terčů a fixací je vysvětleno v tabulce 4.1, takto jsou označeny i výsledky měření.

Tab. 4.1: Označení metod měření vektorů, terčů a fixací

Označení	**Popis metody**
1	Metoda měření vektoru z obou očí, kdy počátek vektoru odpovídá středu oblasti oka.
2	Metoda měření vektoru z obou očí, kdy počátek vektoru odpovídá souřadnicím zornice při aktivaci vektorového módu programu (k aktivaci vektorového módu je určeno tlačítko "v").
3	Metoda měření vektoru z jednoho oka, kdy počátek vektoru odpovídá souřadnicím středu oblasti očí.
M	Metoda animačního terče.
R	Metoda náhodného terče.
F	Metoda s fixací hlavy.
N	Metoda bez fixace hlavy.

První metodou (1) je určení vektoru pro obě oči s počátkem vektoru (referenčním bodem) ve středu oblasti oka, jenž je zjištěna pomocí knihovny Dlib. Druhá metoda (2) určí vektor pro obě oči tak, že počátek vektoru je určen manuálně jako pozice zornice při pohledu doprostřed obrazovky. Obě metody jsou blíže popsány v kapitole 3.1. Třetí metoda (3) je určení vektoru pouze z jednoho oka, počáteční bod vektoru je střed malého okna, ve kterém se oko při detekci nachází. Metoda je zde implementována s vidinou větší přesnosti eyetrackeru. Kamera je totiž blíže oku, z toho plyne větší rozlišení oka a následně i vektoru. Na druhou stranu je tato metoda ovlivněna nepřesnou detekcí zornice, protože u metody detekce obou zornic se vektor průměruje a chyba se zmenšuje.

Varianty terčů jsou popsány v kapitole 3.4.1. Písmeno R značí náhodný terč, písmeno M značí animaci terče.

Dvě varianty fixace jsou použity z důvodu velké závislosti přesnosti eyetrackeru na pohybu hlavy. Více byly popsány v kapitole 3.1. Písmenem F jsou značeny případy, kdy byla hlava fixována tak, že si uživatel při měření položil bradu na podložku a snažil se minimálně pohybovat po celou dobu měření. Písmenem N jsou značeny případy, kdy se hlava nefixovala vůbec, ale přesto se uživatel snažil minimálně pohybovat.

Bylo realizováno měření přesnosti eyetrackeru na různých rozlišeních výstupní obrazovky. Výsledky průměrných diferencí pro rozlišení 8x1 pixelů jsou uvedeny v tabulce 4.2, pro rozlišení 9x2 pixely v tabulce 4.3, pro rozlišení 11x4 pixely v tabulce 4.4 a pro rozlišení 16x9 pixelů v tabulce 4.5. Tabulky s výsledky obsahují hodnoty diference terče a eyetrackeru pro souřadnice X a Y v pixelech. které budou porovnány

pro jednotlivá rozlišení. Dále budou porovnány různé kombinace metod, rozlišení a jejich průměrné diference.

Tab. 4.2: Výsledky průměrných diferencí pro rozlišení 8x1 pixelů

Číslo měření	**Metoda**	**X[p]**	**Y[p]**
1	1MF	0	0,6
2	2MF	0	2,32
3	3MF	0	0,57
4	1MN	0	1,63
5	2MN	0	1,14
6	3MN	0	1,73
7	1RF	0	0,61
8	2RF	0	1,93
9	3RF	0	0,93
10	1RN	0	0,94
11	2RN	0	3
12	3RN	0	2,53

Z tabulky 4.2 lze vyčíst, že nejlepší výsledky pro rozlišení 8x1 pixelů jsou pro měření číslo 1 (metoda 1MF s průměrnou diferencí s hodnotou 0,6 pixelů v ose Y), číslo 3 (metoda 3MF s průměrnou diferencí 0,57 pixely v ose Y), číslo 7 (metoda 1RF s průměrnou diferencí 0,61 pixely v ose Y) a číslo 10 (metoda 1RN s průměrnou diferencí 0,94 pixely v ose Y). Naopak nejhorší výsledky vyšly u měření číslo 2 (metoda 2MF s průměrnou diferencí s hodnotou 2,32 pixelů v ose Y), číslo 11 (metoda 2RN s průměrnou diferencí 3 pixely v ose Y) a číslo 12 (metoda 3RN s průměrnou diferencí 2,53 pixely v ose Y). Sloupec s hodnotami diference X je pro každé měření nulový, neboť rozlišení 8x1 pixel má pouze jednu souřadnici Y v každém sloupci. Metoda 1 byla úspěšná pro fixovanou i nefixovanou hlavu. Metoda 3 byla úspěšná pro fixovanou hlavu. Metoda 2 nebyla úspěšná ani pro fixovanou ani pro nefixovanou hlavu. Nejlepší metodou je v tomto případě metoda 1 s fixací hlavy.

Z tabulky 4.3 pro rozlišení 9x2 pixely vychází výsledky následovně. Nejlepší výsledky jsou u měření číslo 1 (metoda 1MF s průměrnou diferencí s hodnotou 0 pixelů v ose X a hodnotou 1,03 pixelu v ose Y), číslo 3 (metoda 3MF s průměrnou diferencí s hodnotou 0,21 pixelu v ose X a hodnotou 1,14 pixelu v ose Y), číslo 4 (metoda 1MN s průměrnou diferencí s hodnotou 0,22 pixelu v ose X a hodnotou 1,95 pixelu v ose Y), číslo 7 (metoda 1RF s průměrnou diferencí s hodnotou 0 pixelů v ose X a hodnotou 0,68 pixelu v ose Y) a číslo 9 (metoda 3RF s průměrnou diferencí s hodnotou 0,05 pixelu v ose X a hodnotou 1,12 pixelu v ose Y). Nejhorší výsledky vyšly

Tab. 4.3: Výsledky průměrných diferencí pro rozlišení 9x2 pixelů

Číslo měření	**Metoda**	**X[p]**	**Y[p]**
1	1MF	0	1,03
2	2MF	0,48	2,48
3	3MF	0,21	1,14
4	1MN	0,22	0,95
5	2MN	0,44	1,81
6	3MN	0,23	1,38
7	1RF	0	0,68
8	2RF	0,48	1,47
9	3RF	0,05	1,12
10	1RN	0,34	1,4
11	2RN	0,41	1,36
12	3RN	0,68	3,42

u metody číslo 2 (metoda 2MF s průměrnou diferencí s hodnotou 0,48 pixelu v ose X a hodnotou 2,48 pixelu v ose Y), číslo 5 (metoda 2MN s průměrnou diferencí s hodnotou 0,44 pixelu v ose X a hodnotou 1,81 pixelu v ose Y) a číslo 12 (metoda 3RN s průměrnou diferencí s hodnotou 0,68 pixelu v ose X a hodnotou 3,42 pixelu v ose Y). To znamená úspěšnou metodu 1 více pro fixaci. Úspěšnou metodu 3 pro fixaci. Neúspěšnou metodu 2 pro fixaci i nefixaci a metodu 3 pro nefixaci. Zde se na rozdíl od rozlišení 8x1 pixelů zahrnují do hodnocení diference pro X i pro Y. Z toho plyne, že pro rozlišení 9x2 pixelů lze dosáhnout i přesnost 0 pixelů v ose X, nejlepší metodou je v tomto případě metoda 1 s fixací hlavy.

Tabulka 4.4 říká, že rozlišení 11x4 pixely má nejlepší výsledky pro měření číslo 1 (metoda 1MF s průměrnou diferencí s hodnotou 0,84 pixelu v ose X a hodnotou 1,17 pixelu v ose Y), číslo 4 (metoda 1MN s průměrnou diferencí s hodnotou 0,56 pixelu v ose X a hodnotou 1,34 pixelu v ose Y), číslo 7 (metoda 1RF s průměrnou diferencí s hodnotou 0,39 pixelu v ose X a hodnotou 1,42 pixelu v ose Y) a číslo 10 (metoda 1RN s průměrnou diferencí s hodnotou 0,74 pixelu v ose X a hodnotou 1,55 pixelu v ose Y). Nejhorší výsledky pro měření číslo 2 (metoda 2MF s průměrnou diferencí s hodnotou 1,75 pixelu v ose X a hodnotou 2,92 pixelu v ose Y), číslo 5 (metoda 2MN s průměrnou diferencí s hodnotou 1,49 pixelu v ose X a hodnotou 3 pixely v ose Y), číslo 6 (metoda 3MN s průměrnou diferencí s hodnotou 1,43 pixelu v ose X a hodnotou 3,51 pixelu v ose Y) a číslo 12 (metoda 3RN s průměrnou diferencí s hodnotou 3,59 pixelu v ose X a hodnotou 6,67 pixelu v ose Y). Z toho plyne, že metoda 1 excelovala pro obě metody fixace. Metody 2 a 3 naopak poho-

řely u nefixace. Nejlepší výsledky pro rozlišení 11x4 pixely lze dosáhnout metodou 1 s fixovanou i nefixovanou hlavou.

Tab. 4.4: Výsledky průměrných diferencí pro rozlišení 11x4 pixelů

Číslo měření	**Metoda**	**X[p]**	**Y[p]**
1	1MF	0,84	1,17
2	2MF	1,75	2,92
3	3MF	0,9	2,82
4	1MN	0,56	1,34
5	2MN	1,49	3
6	3MN	1,43	3,51
7	1RF	0,39	1,42
8	2RF	0,94	2,31
9	3RF	0,63	2,11
10	1RN	0,74	1,55
11	2RN	0,94	2,93
12	3RN	3,59	6,67

U tabulky 4.5 pro rozlišení 16x9 dává nejlepší výsledky měření číslo 2 (metoda 2MF s průměrnou diferencí s hodnotou 1,64 pixelu v ose X a hodnotou 1,62 pixelu v ose Y), číslo 4 (metoda 1MN s průměrnou diferencí s hodnotou 1,79 pixelu v ose X a hodnotou 1,29 pixelu v ose Y) a číslo 9 (metoda 3RF s průměrnou diferencí s hodnotou 0,65 pixelu v ose X a hodnotou 1,69 pixelu v ose Y). Špatné výsledky dávají měření číslo 5 (metoda 2MN s průměrnou diferencí s hodnotou 3,47 pixelu v ose X a hodnotou 3,77 pixelu v ose Y), číslo 6 (metoda 3MN s průměrnou diferencí s hodnotou 3,5 pixelu v ose X a hodnotou 5,65 pixelu v ose Y) a číslo 12 (metoda 3RN s průměrnou diferencí s hodnotou 4,04 pixelu v ose X a hodnotou 5,48 pixelu v ose Y). Metoda 1 je dobrá pro nefixovanou hlavu. Metody 2 a 3 daly dobré výsledky při fixaci hlavy, ale špatné výsledky při nefixaci hlavy. Nejlepší výsledky pro rozlišení 16x9 pixelů lze dosáhnout metodou 3 s fixovanou hlavou.

Podle výsledků výše lze říci, že nejlepší detekci na příkladech měření má metoda číslo 1 pro fixaci i nefixaci a metoda 3 pro fixaci. Špatné celkové výsledky dává metoda číslo 2 při fixaci i nefixaci a metoda číslo 3 při nefixaci. Lze si všimnout, že metoda 3 s fixací hlavy se s větším rozlišením stávala přesnější. Některé individuální špatné nebo dobré výsledky mohou být zkresleny nedokonalou kalibrací.

Tabulka 4.6 porovnává nejlepší výsledky měření z předešlých tabulek. Z tabulky lze také vidět, že přesnost eyetrackeru je přímo úměrná rozlišení. U malých rozlišení je přesnost větší než u větších rozlišení.

Tab. 4.5: Výsledky průměrných diferencí pro rozlišení 16x9 pixelů

Číslo měření	**Metoda**	**X[p]**	**Y[p]**
1	1MF	2,5	1,76
2	2MF	1,64	1,62
3	3MF	1,3	3,77
4	1MN	1,79	1,29
5	2MN	3,47	3,77
6	3MN	3,5	5,65
7	1RF	2,87	2,11
8	2RF	2,74	4,43
9	3RF	0,65	1,69
10	1RN	1,35	1,93
11	2RN	2,94	0,96
12	3RN	4,04	5,48

Tab. 4.6: Porovnání nejlepších výsledků pro různá rozlišení a metody měření

Rozlišení	**Metoda**	**X[p]**	**Y[p]**
8x1	1MF	0	0,6
8x1	3MF	0	0,57
9x2	1MF	0	1,03
9x2	1RF	0	0,68
11x4	1MF	0,84	1,17
11x4	1RF	0,39	1,42
16x9	2MF	1,64	1,62
16x9	3RF	0,65	1,69

Přesnější porovnání pomocí průměrování jednotlivých diferencí dá daleko lepší informaci o přesnosti metody, proto je dále znázorněno a diskutováno. Průměry každého měření jednotlivých metod (1, 2, 3), terčů (M, R) a fixací hlavy (F, N) porovnává tabulka 4.7. Při porovnání metod měření vektoru je jasné, že nejpřesnější metodou je metoda 1 s průměrnou diferencí 0,73 pixelů ve směru osy X a 1,28 pixelů ve směru osy Y. Přesnosti metod 2 a 3 jsou ve směru osy X srovnatelné, u osy Y vede metoda 2. U terčů jsou očekávaně lepší výsledky animačního terče (M), ve směru osy X to je 0,95 pixelů, ve směru osy Y 2,06 pixelů. U náhodného terče jsou hodnoty horší a to diference 0,98 pixelů pro osu X a 2,27 pixelů pro osu Y. Tyto výsledky jsou

očekávané z důvodu reakční doby člověka na změnu náhodného terče. Animační terč se pohybuje plynule, takže na něho reakční doba má mnohem menší vliv. Tabulka také porovnává celkovou přesnost osy X a Y pro všechna uvedená měření. Je patrné, že velikosti průměrných diferencí ve směru osy X je daleko menší než ve směru osy Y. Z toho plyne daleko větší přesnost ve směru osy X, která je v tomto případě 0,97 pixelů, ve směru osy Y je diference 2,17 pixelů. Jezdím Z důvodů je větší obor hodnot pro směr osy X a větší rozdíl mezi minimem a maximem ve směru vektoru než ve velikosti vektoru.

Tab. 4.7: Celkové výsledky průměrných diferencí pro každou metodu měření vektoru (1, 2, 3), terče (M, R) a fixaci hlavy (F, N)

Metoda	**X[p]**	**Y[p]**
1	0,73	1,28
2	1,1	2,43
3	1,08	2,79
M	0,95	2,06
R	0,98	2,27
F	0,76	1,86
N	1,17	2,48
Průměr	0,97	2,17

Porovnání fixace a nefixace hlavy jednotlivých metod měření vektoru ukazuje tabulka 4.8. Závěry jsou takové, že metoda 1 není velmi závislá na fixaci či nefixaci hlavy, neboť hodnoty diferencí si jsou velmi blízké. Pro fixaci hlavy je hodnota diference ve směru osy X 0,83 pixelů, ve směru osy Y 1,18 pixelů. Pro nefixovanou hlavu je hodnota diference ve směru osy X 0,63 pixelů, ve směru osy Y 1,38 pixelů. Metody 2 a 3 jsou naopak velmi závislé na fixaci hlavy, neboť výsledky pro fixaci a nefixaci jsou velké. Lepší výsledky dávají obě metody pro hlavu fixovanou. Pro metodu 2 to je pro fixovanou hlavu ve směru osy X 0,98 pixelů, ve směru osy Y 2,25 pixelů, pro nefixovanou hlavu ve směru osy X je výsledek diference 1,21 pixelů a pro osu Y 2,62 pixelů. U metody 3 je diference fixované hlavy pro osu X 0,47 pixelů. pro osu Y 1,77 pixelů a pro nefixovanou hlavu osa X 1,69 pixelů, osa Y 3,8 pixelů. Lepší metodou je metoda číslo 1 pro fixovanou i nefixovanou hlavu. Z metody 2 a 3 je přesnější metodou pro fixovanou hlavu metoda číslo 3, přesnější metodou pro nefixovanou hlavu metoda číslo 2. Všechny výsledky z této tabulky jsou očekávány.

Tab. 4.8: Průměry diferencí fixované (F) a nefixované hlavy (N) pro každou metodu měření vektoru

Metoda	**X[p]**	**Y[p]**
1F	0,83	1,18
1N	0,63	1,38
2F	0,98	2,25
2N	1,21	2,62
3F	0,47	1,77
3N	1,69	3,8

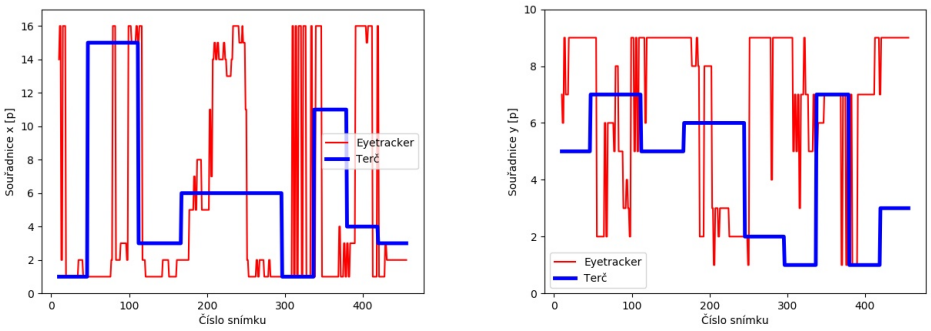
4.1.1 Diference jako křivka pro každou osu

Dále jsou porovnávány diference v pixelech X a Y na grafech vždy s jedním vykresleným signálem. Modrá křivka znázorňuje pozici terče, červená křivka pozici detekovaného směru pohledu na obrazovce. Na levém grafu je znázorněna osa X, na pravém grafu je osa Y. Grafy znázorňují posun terče a detekci směru pohybu podle dané osy. Lze z nich vyčíst přesnost detekce.

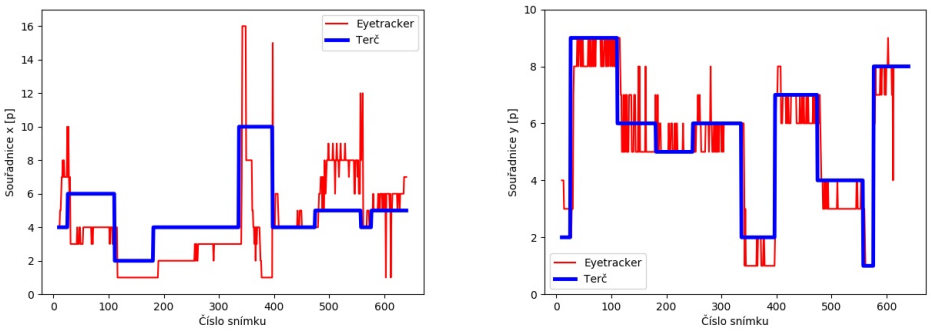
Na obrázku 4.1 a 4.2 lze vidět rozdíl výsledků v ose X a Y mezi fixovanou a nefixovanou hlavou pro metodu měření 3 čili pro jedno oko se začátkem vektoru uprostřed oblasti oka. Tato metoda je citlivá na pohyb hlavy, což měření potvrdilo.

Obrázky 4.3 a 4.4 porovnávají metodu 1 při rozlišení 16x9 pixelů s nefixovanou hlavou a rozlišení 8x1 pixelů s fixovanou hlavou. Lze říci, že fixace hlavy má nepatrně lepší vliv na diferenci pro metodu 1. Také zde lze vidět porovnání náhodného terče (obrázky 4.1 a 4.2) s animačním (obrázky 4.3 a 4.4).

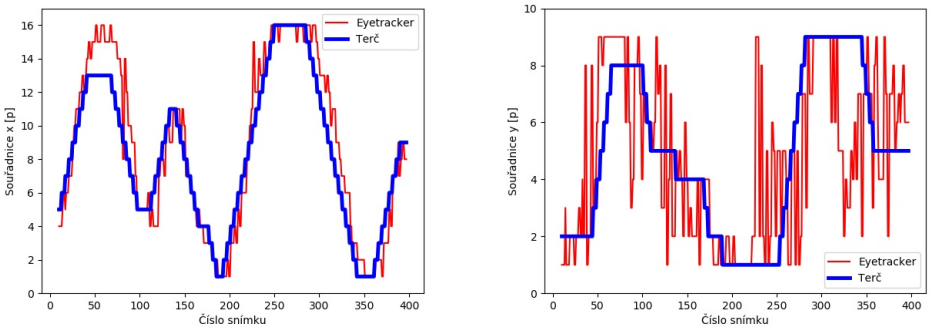
Kolísání signálu je viditelné na většině grafech. Pokud by se aplikoval buffer, tak by výsledek mohl být přesnější. Lze si také všimnout, že detekce v ose Y je výrazně horší než detekce v ose X. Důvod je ten, že detekce v ose Y je určena z větší části směrem vektoru v a detekce v ose X je určena z větší části velikostí vektoru u. Směr vektoru v má větší rozsah hodnot než velikost vektoru u.



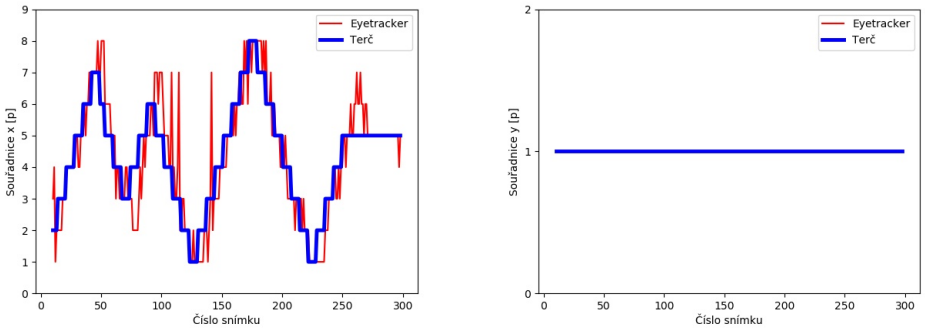
Obr. 4.1: Graf náhodného terče s nefixovanou hlavou pro metodu 3 při rozlišení 16x9 pixelů



Obr. 4.2: Graf náhodného terče s fixovanou hlavou pro metodu 3 při rozlišení 16x9 pixelů



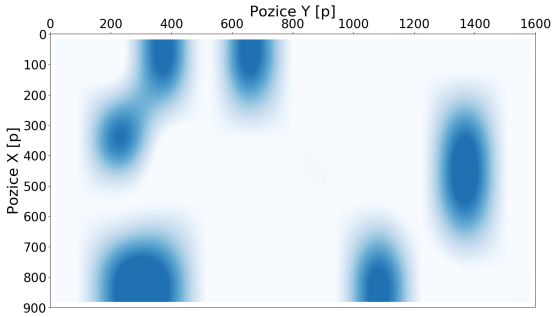
Obr. 4.3: Graf animačního terče s nefixovanou hlavou pro metodu 1 při rozlišení 16x9 pixelů



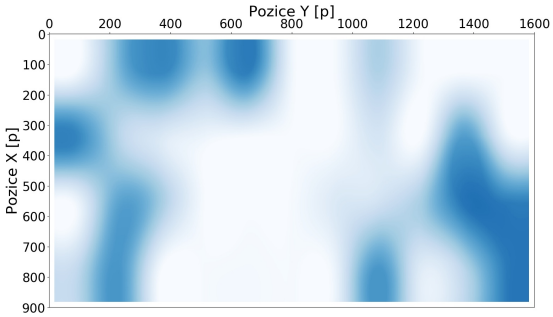
Obr. 4.4: Graf animačního terče s fixovanou hlavou pro metodu 1 při rozlišení 8x1 pixelů

4.1.2 Diference jako heat mapa

Dalším způsobem analýzy diference jsou heat mapy, jenž jsou automaticky vykresleny jako obrázky programem po každém měření. Reprezentují intenzitu pohledu na jedno místo obrazovky. To znamená, že se zvýrazní souřadnice, které uživatel pozoroval delší dobu. Při testování na eyetrackeru mají heat mapy větší význam pro náhodný teč, neboť ten se nachází na jednom místě delší časový okamžik. Proto je na heat mapě možné sledovat velké intenzity v této oblasti (obrázek 4.6). Modré pixely jsou místa pohledu, bílé pixely jsou místa, kam se uživatel nepodíval. Čím více tmavší modrá barva pixelu je, tím delší dobu uživatel pixel sledoval.



Obr. 4.5: Heat mapa terče pro rozlišení 11x4 pixelů



Obr. 4.6: Heat mapa eyetrackeru při testování náhodným terčem pro rozlišení 11x4 pixelů

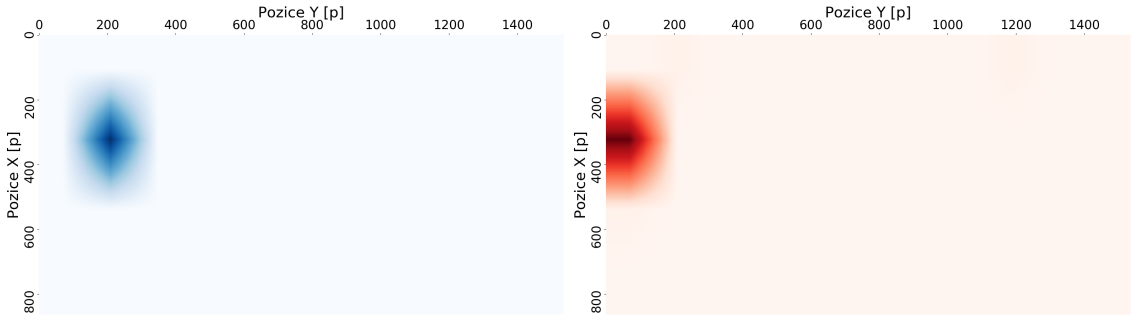
Porovnáním heat mapy terče a eyetrackeru lze vidět, jak je měření přesné. Vlevo je vždy heat mapa terče, vpravo je heat mapa eyetrackeru. To ilustrují obrázky 4.5 a 4.6 pro náhodný terč. Heat mapa terče je ideální případ heat mapy, kdy všechny pixely eyetrackeru padnou přesně do pixelů terče. Heat mapa eyetrackeru je reálný

případ při sledování terče. Čím více se heat mapa eyetrackeru podobá heat mapě terče, tím přesnější eyetracker je.

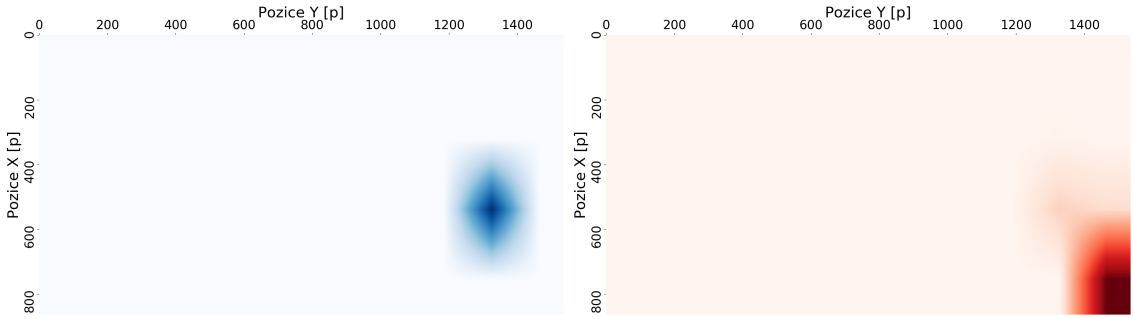
Porovnání přesnosti z heat mapy pro různé směry pohledů

Všechny následující obrázky jsou po sobě jdoucí snímky eyetrackeru a výstupního videa z jednoho měření pro rozlišení 11x4 pixely. Celkový počet snímků je v tomto případě 770. Výsledky pro toto porovnání byly upraveny tak, aby jedna heat mapa obsahovala pouze pohled na jeden nepohyblivý terč. Níže je ukázka výsledků nepohyblivého terče a jeho sledování pomocí eyetrackeru (obrázek 4.7 a 4.8), písmeno t znamená číslo snímku. Na obrázcích je vždy vlevo terč, vpravo výsledek detekce. Heat mapa terče (modrá barva) je ideální případ detekce eyetrackeru, čím tmavší barva, tím delší pohled do této oblasti. Heat mapa eyetrackeru (červená barva) je reálný případ eyetrackeru, kdy uživatel sleduje nepohyblivý terč na obrázku vlevo (modrá heat mapa terče). Zase platí, čím tmavší je barva, tím delší je pohled do těchto souřadnic. Rozmazání terče i eyetrackeru je dosaženo interpolací obrazu pomocí metody bilineární interpolace kvůli zvětšování obrazu. K získání informace z nepohyblivého terče a eyetrackeru se dosáhlo tak, že se rozdělilo měření náhodného terče do několika heat map. Vždy je ještě odečteno několik vzorků ze začátku a konce změny terče, protože se počítá s reakční dobou uživatele. Až na malé nepřesnosti eyetrackeru, které jsou pravděpodobně způsobeny pohybem hlavy při měření nebo kalibraci, nepřesným určením nejpodobnějšího vektoru nebo přeskakující detekcí, lze říci, že je toto měření úspěšné.

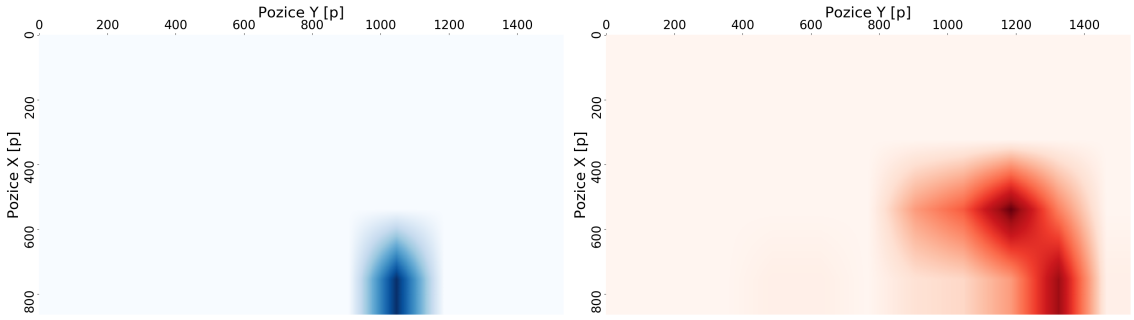
Využití eyetrackingu bylo popsáno v teoretické části v kapitole 1.4.2. Navržený program se dá využít například při testování pohledu řidiče automobilového vozidla nebo na analýzu zájmu webové stránky. Pohled lze zkoumat na základě vytvoření heat mapy přímo ve scéně. Heat mapa je na následujících obrázcích tvořena červenými shluky pixelů, čím tmavší je barva pixelu, tím déle uživatel tento pixel pozoroval. Obrázek 4.9 ukazuje pohled uživatele na webovou stránku `https://www.seznam.cz/` na vyhledávací okno, obrázek zpráv a obrázek novinek. Obrázek 4.10 ilustruje pohled uživatele na psa v pravém horním rohu obrazovky. Přesnost heat mapy není perfektní, ale shluky se blíží přesnému pohledu. Simulace analýzy pohledu řidiče automobilového vozidla je provedena na obrázcích 4.11 a 4.12. Na levém obrázku je znázorněn pohled řidiče na informativní dopravní značku, na levém obrázku je pohled před sebe na cestu a do zpětného zrcátka. V tomto případě je eyetracker přesnější než v případě webové stránky. Důvodem je lepší kalibrace.



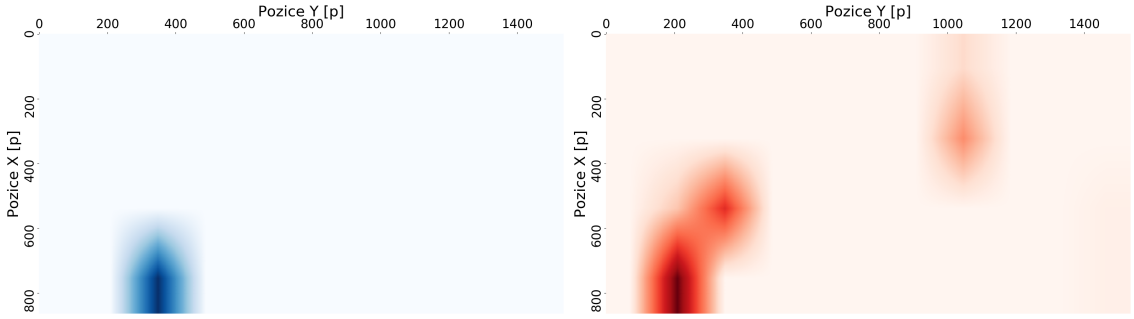
(a) t = 17



(b) t = 84

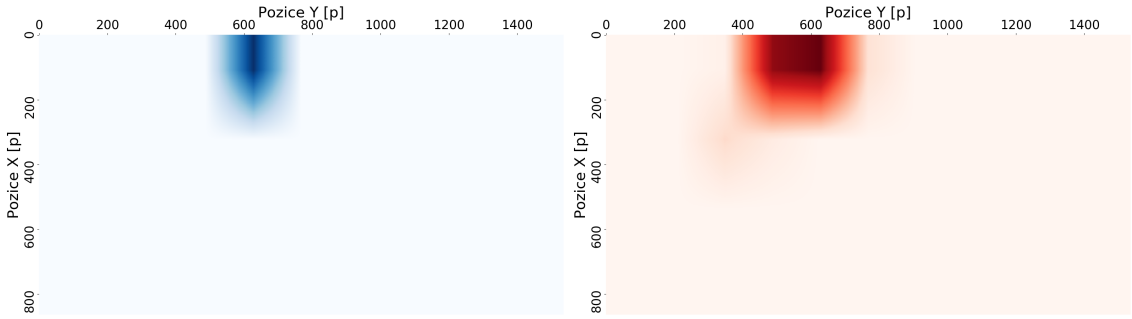


(c) t = 193

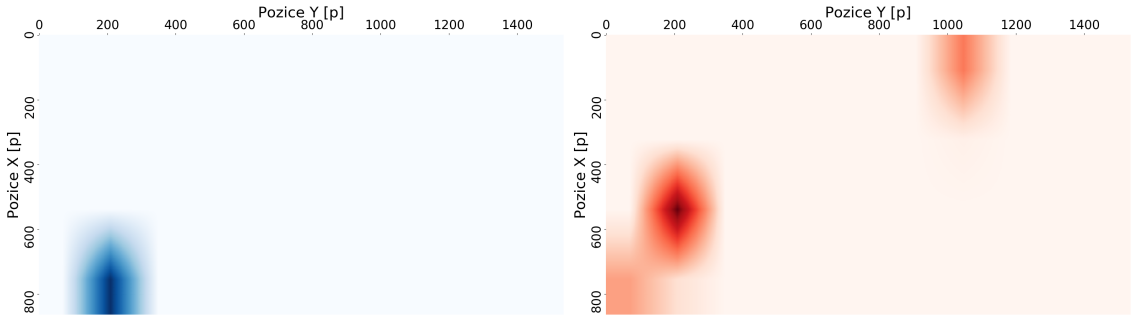


(d) t = 315

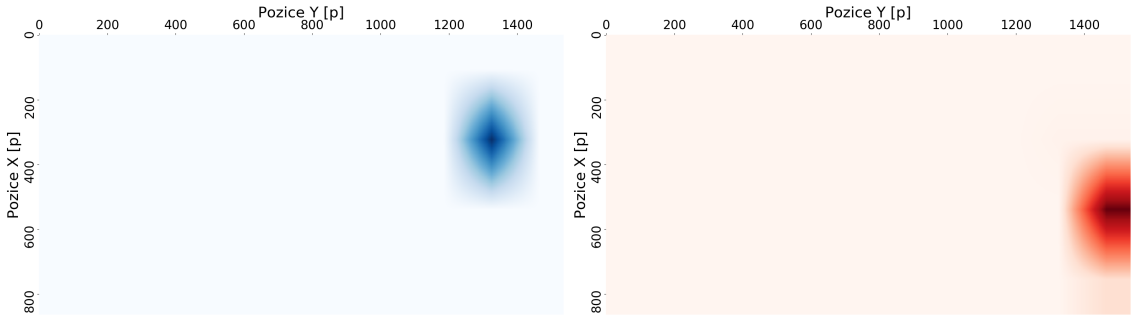
Obr. 4.7: Ideální heat mapa při sledování nepohyblivého terče vlevo, reálná heat mapa při sledování nepohyblivého terče vpravo, část 1



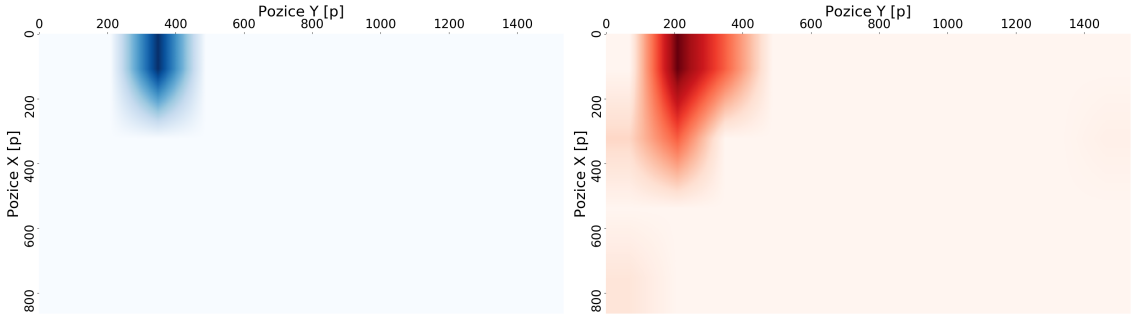
(a) t = 405



(b) t = 504

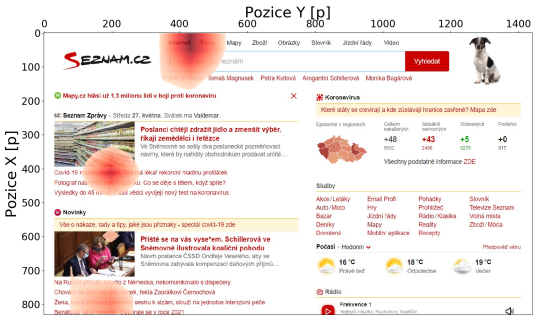


(c) t = 588

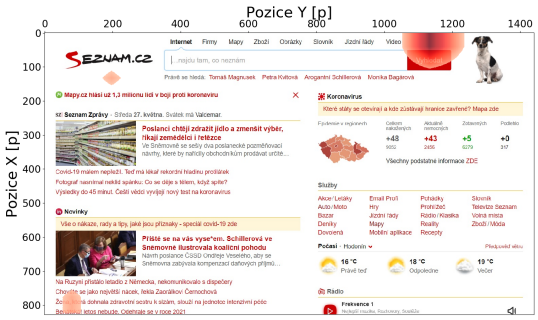


(d) t = 672

Obr. 4.8: Ideální heat mapa při sledování nepohyblivého terče vlevo, reálná heat mapa při sledování nepohyblivého terče vpravo, část 2

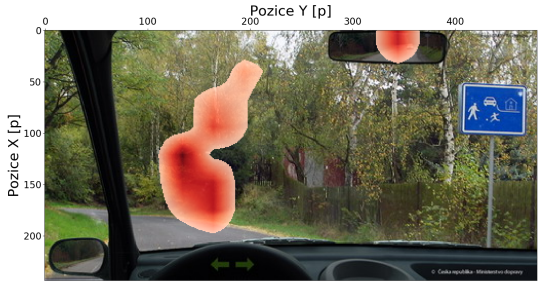


Obr. 4.9: Analýza webové stránky, pohled uživatele vyhledávací okno, obrázek zpráv a obrázek novinek [52]



Obr. 4.10: Analýza webové stránky, pohled uživatele na psa [52]

Obr. 4.11: Simulace analýzy pohledu řidiče vozidla, pohled na dopravní značku *Obytná zóna* [53]



Obr. 4.12: Simulace analýzy pohledu řidiče vozidla, pohled na cestu a do zpětného zrcátka [53]

4.2 Zhodnocení

Výše byly zobrazeny a diskutovány některé výsledky. Úspěšnost měření závisí zejména na správné kalibraci, volbě metody, z toho vyplývající fixaci či nefixaci hlavy, nebo na správném nastavení prahů pro detekci zornice. Po zpřesnění, jenž je dále možné (diskutováno níže), je program vhodný pro sledování pohybu očí ve všech běžně využívaných oblastech (kapitola 1.4.2). Dále jsou diskutovány nedokonalosti Raspberry Pi a možnosti vylepšení programu.

4.2.1 Nedokonalosti Raspberry Pi 3

Výpočetní náročnost algoritmu na videu z RPi kamery je velká, proto se rozlišení videa zmenšilo 4x. Z 1080 pixelů se tak stane 270 pixelů. Zmenšení je na úkor přesnosti eyetrackeru, ale bez zmenšení Raspberry program nezvládne. Ze stejného důvody bylo také potřeba upravit výstupní velikost obrazovky pro detekci pohledu. Z chování eyetrackeru lze říci, že je spíše vhodnější pro větší plochy. Kvůli výpo-

četní náročnosti se také nemohly zakomponovat některé zpřesňující postupy, které se obvykle v eyetrackerech používají.

Důležitá veličina při měření je také vzdálenost obličeje od kamery. Pokud obličej nebude viditelný celý, nebude nalezena ani oblast očí. Pokud bude naopak obličej příliš daleko od kamery, bude menší rozlišení obrazu a eyetracker bude nepřesný. Ideální vzdálenost pro měření je v případě metody 1 a metody 2 cca 30 cm. V případě metody 3 je tomu jinak, není zde totiž aplikována knihovna Dlib pro hledání obličeje a jeho oblastí, obličej proto nemusí být celý v záběru. Naopak je u této metody ideální, aby kamera byla co nejblíže k oku kvůli většímu rozlišení.

Možnost vylepšení

Větší přesnost by se dala zajistit výkonnější verzí Raspberry Pi, protože by se tolik nemuselo zmenšovat rozlišení kamery. Je zde také možnost většího přiblížení kamery k oku, pak bude rozlišení vektoru směru pohledu větší a metoda přesnější. Toho se snažilo dosáhnout použitím metody výpočtu vektoru číslo tři (měření jednoho oka). Metoda ale neměla příliš dobré výsledky, protože je velmi náročná na znehybnění hlavy. Při snímání celého obličeje se dají souřadnice počátku vektoru ukotvit na některou ze struktur obličeje, proto byla kamera umístěna do větší vzdálenosti a detekován obličej.

V eyetrackerech je také hojně využíván buffer, které potlačí rychlé přeskoky výsledného směru pohledu. Přeskoky jsou způsobeny změnou detekce středu zornice a okolí očí. Tak dokonalý detektor, který by při každém nepatrném pohybu určil vždy stejnou a správnou oblast oka nebo střed zornice je velmi složité navrhnout. Buffer funguje jako fronta, do které se ukládají souřadnice pohledu předchozích snímků. Z těchto souřadnice lze dále počítat medián, průměrovat, násobit různými vahami atd. Buffer nebyl implementován, neboť Raspberry už bylo velmi zatížené detekcí směru pohledu.

Mrknutí nebo zavření očí negativně ovlivňuje přesnost detekce pozice pohledu. Pro vylepšení eyetrackeru by také mohla být vhodná detekce mrknutí a následné odstranění výsledků v těchto detekovaných intervalech.

Závěr

Práce je rozdělena do čtyř hlavních kapitol: teoretická část, detekce očí, realizace eyetrackeru a testování přesnosti.

První kapitola nese název Teoretická část, je zde vypracována literární rešerše a to v první až čtvrté podkapitole. Nejprve jsou rozebrány pohyby očí a rozdíl mezi párovými a nepárovými pohyby. Dále je věnována pozornost detekci obličeje a očí, historii a současných metodách snímání pohybu očí. Na konci teoretické části je rozebrán eyetracking, jeho využití v aplikacích, konkurenční produkty, požadavky perfektního eyetrackeru a samotná realizace.

Použité příslušenství spolu s krátkou dokumentací a specifikací použitých platforem a modulů je uvedeno v další kapitole s názvem Detekce očí. Následuje vysvětlení algoritmů pro detekci obličeje, detekci očí a předzpracování obrazu. Detekce obličeje je realizována pomocí Haarových příznaků s kaskádní klasifikací pro obličej a pomocí knihovny Dlib s detekcí orientačních bodů obličeje. Dále je používána pouze metoda s knihovnou Dlib, neboť vykazuje lepších výsledků při detekci oblasti očí a dovoluje jednoduché nalezení středu očí. Předzpracování obrazu je realizováno po detekci oblasti očí před detekcí zornice. Práh pro převod na binární obraz je zadáván uživatelem pomocí posuvníku a lze jej měnit v průběhu detekce. Pro dobrou detekci i za nerovnoměrných jasových podmínek se práh nastavuje pro každé oko zvlášť. Jsou použity tyto metody předzpracování obrazu: odstranění reflexe, gama korekce, převod do HSV barevného formátu, mediánový filtr, bilaterální filtr, erose, dilatace, morfologické otevření a morfologické uzavření. Po předzpracování vstupuje obraz do funkce pro detekci zornice ze shluků, které vznikly ze zornice po předzpracování. Dochází k nalezení kontury zornice a jejího středu.

Kapitola třetí, realizace eyetrackeru, vysvětluje postup detekce směru pohledu. Na začátku je počítán vektor směru pohledu s počátkem ve středu oblasti očí a koncem ve středu zornice. Každý vektor charakterizuje velikost u a směr v. Před každým použitím eyetrackeru je velmi důležitá kalibrace, kdy oko sleduje devět kalibračních bodů rozmístěných po celé obrazovce, jejich rozmístění bylo uvedeno na obrázku 3.3. Vektory naměřené při pohledu na kalibrační bod se ukládají do kalibrační mapy na pozici kalibračního bodu. Kalibrační mapa je naplněna bikubickou interpolací mezi kalibračními body (obrázek 3.5 a 3.6). V kalibrační mapě se hledá vektor, který je co nejvíce podobný (velikostí a směrem) aktuálně naměřenému vektoru. Souřadnice nalezeného nejpodobnějšího vektoru jsou souřadnicemi směru pohledu na obrazovce. Kalibrace a kalibrační mapa jsou nejdůležitější částí eyetrackeru, pokud nebude kalibrace provedena správně, bude eyetracker velmi nepřesný. Při kalibraci i měření je také velmi důležitá fixace hlavy u algoritmů, které s pohybem hlavy nepočítají. K testování eyetrackeru jsou vytvořeny dva testovací terče, náhodný terč

a animační terč. Obě metody testovacích terčů lze aktivovat přímo v navrženém programu. Ovládání programu je umožněno klávesami na klávesnici, je velmi intuitivní s nápovědou v příkazovém řádku nebo přímo na obrazovce. Program je volně dostupný na `https://github.com/Nikolhun/Eyetracking.git`.

V poslední kapitole jsou testovány tři metody výpočtu vektoru, dvě metody terče a dvě metody fixace hlavy pro čtyři rozlišení výstupní scény. Výsledky byly diskutovány v kapitole 4.1. Nejlepší metodou měření vektoru je s ohledem na naměřené výsledky metoda číslo 1, která funguje dobře s fixovanou i nefixovanou hlavou. Nejpřesnější detekci je možné dosáhnout při rozlišení 8x1 pixelů, ale detekovaná oblast bude několikrát větší než při použití rozlišení 16x9 pixelů, které je méně přesné. Při měření jakékoliv metody i rozlišení s fixací hlavy jsou vždy lepší výsledky než bez fixace hlavy. Nechybí ani porovnání měření s terčem pomocí křivky pro každou osu a pomocí heat mapy. V závěru kapitoly je zhodnocení navrženého eyetrackeru s diskutovanými nedokonalostmi použitého příslušenství i návrhu metody.

Algoritmus pro detekci zornice nefunguje dokonale. Výskyt falešných detekcí není častý, problém ale nastává s občasnou detekcí více kontur, vynecháním detekce při pohledu do strany nebo zavřením očí. Příčin může být více, velká vzdálenost od kamery, změna osvětlení scény, špatné nastavení prahu, přivřené oči nebo malý detekovaný shluk, který se v předzpracování vyfiltruje. Nepřesná detekce má negativní vliv na přesnost eyetrackeru. Při hledání nejlepšího vektoru nelze najít přesně naměřený vektor, ale hledají se pouze dva nejbližší parametry. Pokud se při kalibraci uloží do místa kalibračního bodu špatný vektor, má to fatální následky na výslednou přesnost.

Literatura

[1] VAŠČÁKOVÁ, Viktória: *Očné svaly: Inervácia, fyziológia, funkcia.* Brno, 2018. Bakalářská práce. Masarykova univerzita. Vedoucí práce MUDr. Karolína Skorkovská, Ph.D.

[2] SYNEK, Svatopluk a Šárka SKORKOVSKÁ: *Fyziologie oka a vidění.* Praha: Grada, 2004.

[3] VESELÁ, Cindy: *Eyetracking při řešení dopravních situací.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2018. 62s. Vedoucí práce: Ing. Oto Janoušek, Ph.D.

[4] KŘÍSTEK, Jakub: *Detekce a analýza pohybu očí.* Brno, 2013. Bakalářská práce. Vysoké učení technické. Vedoucí práce Prof. Ing. IVO PROVAZNÍK, Ph.D.

[5] ROZHOŇOVÁ, Andrea: *Detekce a rozpoznání obličeje s využitím platformy Raspberry Pi.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav biomedicínského inženýrství, 2017. 56 s. Vedoucí práce Ing. Branislav Hesko

[6] YANG, Ming-Hsuan, David J. KRIEGMAN a Narendra AHUJA: *Detecting Faces in Images: A Survey. Transactions on pattern analysis ande machine intelligence.* 2002, 24(1), 34-58

[7] KONTROPOULOS, Constantine a Ioannis PITAS: *Rule-based Face Detection in Frontal Views.* Greece, 2016. Aristotle University of Thessaloniki.

[8] HJELMAS, Eric a Boon Kee LOW: *Face Detection: A Survey. Computer Vision and Image Understanding.* 2001, 83(3), 236-274. DOI: 10.1006/cviu.2001.0921. ISSN 10773142.

[9] KIM, Inseong, Joon Hyung SHIM a Jinkyu YANG: *Face detection.* 2016. Stanford University.

[10] VIOLA, Paul and Michael JONES: *Rapid Object Detection using a Boosted Cascade of Simple Features.* Accepted Conference on Computer Cision And Pattern Recognition 2001, [online]. Dostupné z URL: <`https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf`>.

[11] *Colour Image Processing: The RGB Model.* 2019. Colour Models. Department of Computer Science.

[12] KEIN Robert: *Understanding Color Models Used in Digital Image Processing* 2018. All about circuits.

[13] *RGB color model.* Academic, Hisour. Color frames used in image processing. 2017

[14] *Face Detection using Haar Cascades. In: OpenCV-Python Tutorials.* [online]. Dostupné z URL: <https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html>.

[15] ALTO, Valentina: *Face recognition with OpenCV: Haar Cascade.* [online]. A Medium Corporation, 2019

[16] MARQUÉS, Ion: *Face Recognition Algorithms.* Leioa, 2010. Universidad del País Vasco. Vedoucí práce Manuel Grana.

[17] AL-ALLAF, Omaima L. A: *Review of Face Detection Szstems Based Artifical Neural Networks Algorithms.* The International Journal of Multimedia and Its Applications. 2014, 1(6).

[18] RYNKIEWICZ, Filip, Piotr NAPIERALSKI a Marcin DASZUTA: *Pupil Detection Methods for Eye Tracking.* Journal of applied computer science. 2018, Lodz University of Technology, Institute of Information Technology. 201-211

[19] BAPAT, Krutika: *Hough Transform with OpenCV (C++/Python).* 2019. Learn OpenCV.

[20] ITO, Y., OHYAMA, W., WAKABAYASHI, T. a KIMURA, F.: *Detection of eyes by circular Hough transform and histogram of gradient.* 2012, IEEE, ISBN: 978-4-9906441-0-9

[21] DONGHENG, Li, D. WINFIELD and D.J: *Parkhurst. Starburst: A hybrid algorithm for video-based eye trackingcombining feature-based and model-based approaches.* 2005. 8p. DOI: 10.1109/CVPR.2005.531 ISSN 1063-6919.

[22] LIH-JEN Kau, TIEN-LIN Lee: *An Efficient and Self-Adapted Approach to the Sharpening of Color Images.* Department of Electronic Engineering and Graduate Institute of Computer and Communication Engineering. National Taipei University of Technology, Taiwan, 2013

[23] DIAS D.: *What is YCbCr ? (Color Spaces).* 2017. Research and Development Engineering at Synopsys. Inc. BSc Computer Engineering. medium.com [online]. Dostupné z URL: <https://medium.com/breaktheloop/what-is-ycbcr-964fde85eeb3>.

[24] VOJTÍŠEK, Jiří: *Systém pro sledování pohybu očí.* Brno, 2011. Diplomová práce. Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií. Ústav biomedicínského inženýrství. Vedoucí práce Vratislav Čmiel.

[25] SMOLÍK, Tomáš: *Automatická segmentace objektů s využitím dat získaných ze zařízení pro sledování pohybu očí.* Plzeň, 2016. Diplomová práce. Západočeská univerzita v Plzni. Vedoucí práce Bureš Lukáš, Ing.

[26] *Dark and bright pupil tracking. Tobii Pro.* [online]. [cit. 2019-11-25]. Dostupné z URL:
<https://www.tobiipro.com/learn-and-support/learn/eye-tracking-essentials/what-is-dark-and-bright-pupil-tracking/>.

[27] *Eye tracking: What Is Eye Tracking.* [online]. [cit. 2019-11-25]. Dostupné z URL:
<http://www.eyetracking.com/About-Us/What-Is-Eye-Tracking>.

[28] SCOTT, D. a J. FINLEY: *Visual search, eye movements and display units: human factors report.* Durham: University of Durham, 1993.

[29] *Raspberry Pi: What is a Raspberry Pi?* [online]. [cit. 2019-11-26]. Raspberry Pi Foundation Dostupné z URL:
<https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/>.

[30] *RPi Camera (H): Features. Waveshare.com* [online]. [cit. 2019-11-26]. Dostupné z URL:
<https://www.waveshare.com/rpi-camera-h.htm>.

[31] JAN, Jiří: *Medical image processing, reconstruction and restoration: concepts and methods.* Boca Raton: Taylor, 2006, 730 s. ISBN 08-247-5849-8.

[32] PIKORA, Jan: *Implementace grafických filtrů pro zpracování rastrového obrazu.* Brno, 2008. Bakalářská práce. Masarykova Univerzita Brno. Vedoucí práce Mgr. Tomáš Staudek, Ph.D.

[33] KAZEMI, V. a SULLIVAN J.: *One Millisecond Face Alignment with an Ensemble of Regression Trees.* 2014, Computer Vision and Pattern RecognitionAt. Ohio, USA. DOI: 10.13140/2.1.1212.2243

[34] C. SAGONAS, E. ANTONAKOS, G, TZIMIROPOULOS, S. ZAFEIRIOU, M. PANTIC: *300 faces In-the-wild challenge: Database and results.* Image and Vision Computing (IMAVIS), Special Issue on Facial Landmark Localisation "In-The-Wild". 2016.

[35] TSANG, Tat: *Image-Based Face Detection System.* Oxford Brookes University, 2009.

[36] SUDHAKAR, Shreenidhi: *Histogram Equalization: Contrast Enhancement. In: Towards Data Science.* [online]. 2007 [cit. 2019-12-26]. Dostupné z URL: <https://towardsdatascience.com/histogram-equalization-5d1013626e64>.

[37] SLÁMA: *Průkazové fotografie. In: Foto Sláma.* [online]. [cit. 2019-12-02]. A Medium Corporation, 2019 Dostupné z URL: <http://www.fotoslama.cz/cs/fotografie-prukazove>.

[38] MALLICK, Satya: *Blob Detection Using OpenCV (Python, C++).* [online]. [cit. 2019-12-21].

[39] MCKESSON, Jason L: *Learning Modern 3D Graphics Programming.* Arcsynthesis. org. 2012.

[40] KOLEK, Petr: *Detekce směru pohledu řidiče v obrazech.* Fakulta elektrotechniky a informatiky. Technická univerzita. Diplomová práce 2017

[41] PIKORA, Jan: *Implementace grafických filtrů pro zpracování rastrového obrazu.* Brno, 2008. Bakalářská práce. Masarykova Univerzita. Vedoucí práce Mgr. Tomáš Staudek, Ph.D. Dostupné z URL: <https://is.muni.cz/th/qr2pv/bakalarka.pdf>.

[42] NEJEZCHLEB, Ivan: *Tone-mapping pro HDR obrazy.* Fakulta informačních technologií. Ústav počítačové grafiky a multimédií. Vysoké učení technické v Brně. Diplomová práce 2009.

[43] A. ROSENBROCK: *Training a custom dlib shape predictor.* pyimagesearch.com - Dlib. 2019.

[44] RI CERD NG, KIAN MING LIM, CHIN POO LEE, SITI FATIMAH ABDUL RAZAK: *Surveillance system with motion and face detection using histograms of oriented gradients.* Indonesian Journal of Electrical Engineering and Computer Science. 2018.

[45] RYAN P.: *Facial Recognition — a visual step by step.* Use the HOG to find the faces. medium.com. 2019.

[46] *Psychwire Eye-tracking: Making and analysing eye-tracking experiments.* Calibration / Validation. WordPress.org. 2014

[47] FRANĚK P.: *Metody zvyšování rozlišení digitálních snímků.* Fakulta elektrotechniky a komunikačních technologií. Ústav biomedicínského inženýrství. Vysoké učení technické v Brně. Bakalářská práce 2011.

[48] HAVELKA J.: *Geometrické transformace obrazu* Fakulta informačních technologií. Ústav informačních systémů. Vysoké učení technické v Brně. Bakalářská práce 2008.

[49] FEJTOVÁ, M., et al. System I4Control: *Contactless control PC.* Intelligent Engineering Systems, 2006. INES'06. Proceedings. International Conference on. IEEE, 2006.

[50] *Tobii Pro Glasses 2.* Tobii Pro products. <https://www.tobiipro.com/product-listing/tobii-pro-glasses-2/>.

[51] *Tobii Pro Spectrum.* Tobii Pro products. <https://www.tobiipro.com/product-listing/tobii-pro-spectrum/>.

[52] *Webová stránka seznam.cz.* 2020. <https://www.seznam.cz/>.

[53] Ministerstvo dopravy České republiky: *Autoškola testy.* Otázka 06060299. <https://www.autoskola-testy.cz/prohlizeni_otazek.php?otazka=446-v_useku_za_touto_dopravni_znackou>.

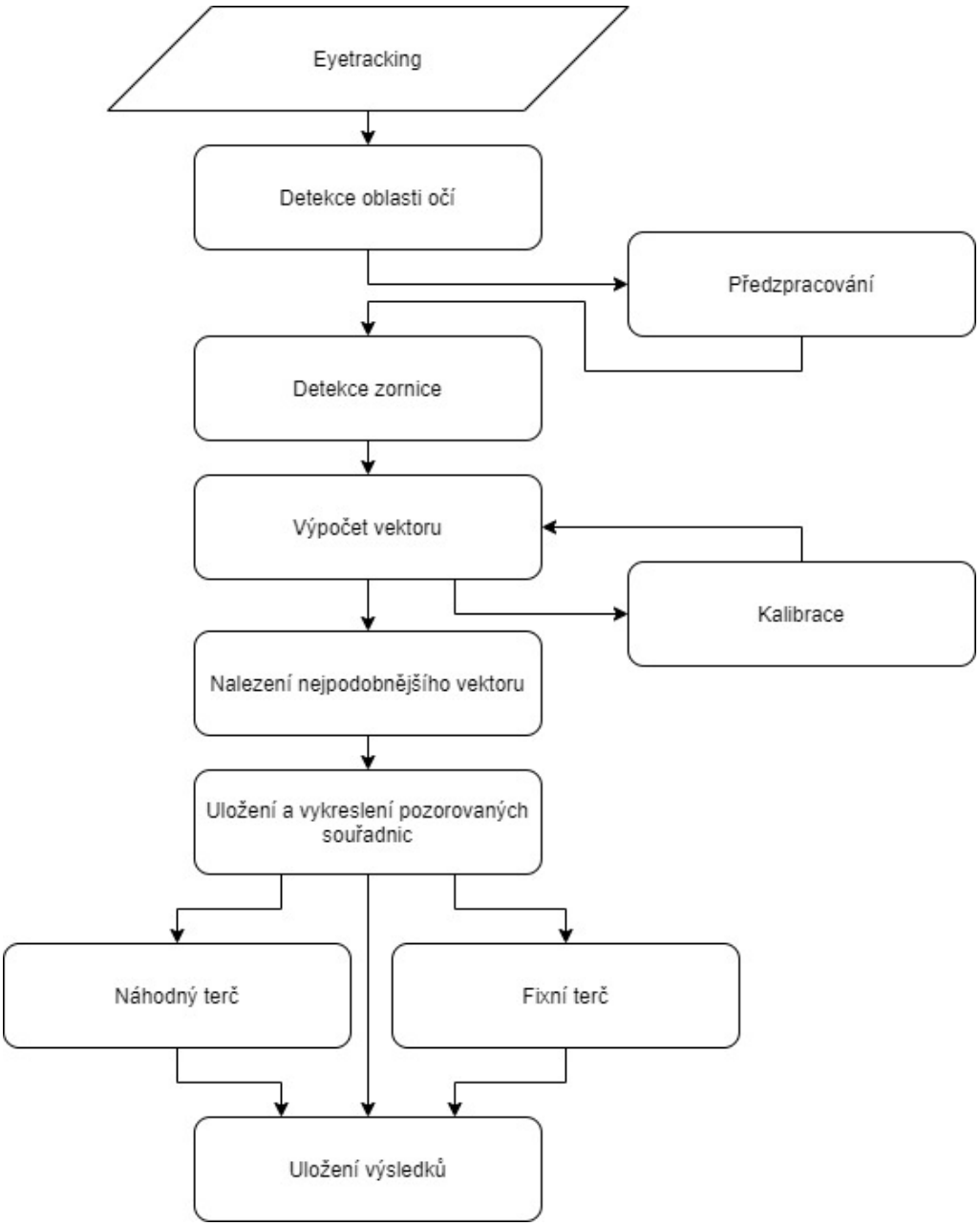
Seznam symbolů, veličin a zkratek

ms	milisekundy
mm	milimetry
EOG	elektrookulografie
VOG	elektrookulografie
LED	elektroluminiscenční dioda
EEG	elektroencefalogram
Hz	Hertz
D	dioptrie
RGB	Red, Green, Blue
HSV	Hue, Saturation, Value
YCbCr	Luminence, Chromatic Blue, Chromatic Red
USB	Universal Serial Bus
RANSAC	Random Sample Consensus
cm	centimetr
mV	milivolt
rad	radián
s	sekunda
W	Watt
p	pixel
Mpix	Megapixel
RPi	Raspberry Pi
OpenCV	Open Computer Vision
HDMI	High-Definition Multi-media Interface
UTP	Unshielded Twisted Pair

HOG	Histogram of oriented gradients
CVPR	Conference on Computer Vision and Pattern Recognition
FPS	Frames per second
ČVUT	České vysoké učení technické

Seznam příloh

A Vývojový diagram programu



Obr. A.1: Vývojový diagram celého programu

B Struktura programu

Program pro metodu 1 je spuštěn pomocí souboru *program_1.py*, metoda 2 je spuštěna programem *program_2.py*, metoda 3 je spuštěna programem *program_3.py*. Ostatní soubory s příponou *.py* jsou funkce. Ve složce *Dlib_landmarks* je soubor pro detekci landmarků z obličeje. Složka *results* slouží pro ukládání výsledků. V souboru *README.md* je blíže popsána struktura programu a program samotný.

```
Eyetracker
├── Dlib_landmarks ..... Dlib detektor landmarků v obličeji
│   └── shape_predictor_6_face_landmarks.dat
├── results ..... složka pro ukládání výsledků
├── _init_.py ..... složka pro ukládání výsledků
├── analyse_measured_data.py ..... script pro analýzu výsledků
├── calibration.py ..... kalibrace
├── reflection.py ..... odstranění reflexe
├── detect_pupil.py ..... detekce zornice
├── dlib_landmarks.py ..... detekce landmarků v obličeji
├── eyetracking.py ..... sledování očí
├── interpolate.py ..... interpolace mezi kalibračními body
├── make_target.py ..... tvorba terčů
├── program_1.py ..... program pro metodu 1
├── program_2.py ..... program pro metodu 2
├── program_3.py ..... program pro metodu 3
├── README.md ..... informace o programu
└── vector.py ..... výpočet vektoru
```