VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ZPRACOVÁNÍ OBRAZU V EMBEDDED SYSTÉMECH

IMAGE PROCESSING IN EMBEDDED SYSTEMS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **IGOR FRANK**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. PAVEL NAJMAN**
SUPERVISOR

BRNO 2018

Zadání bakalářské práce/19300/2017/xfrank12

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2017/2018

Zadání bakalářské práce

Řešitel: **Frank Igor**
Obor: Informační technologie
Téma: **Zpracování obrazu v embedded systémech**
Image Processing in Embedded Systems
Kategorie: Vestavěné systémy

Pokyny:

1. Seznamte se se základními metodami zpracování obrazu a možnostmi jejich implementace v embedded systémech.
2. Vyberte vhodný embedded systém a navrhněte vhodnou aplikaci, která vhodně využije zpracování obrazu ve vybraném embedded systému.
3. Navrhněte postup implementace vybrané aplikace a vyhodnoťte vlastnosti zvoleného postupu, zejména s ohledem na funkčnost, rychlost a spotřebu energie.
4. Implementujte vybranou aplikaci a demonstrujte vhodným způsobem její funkčnost.
5. Zhodnoťte dosažené výsledky a možnosti pokračování práce.

Literatura:

- Dle pokynů vedoucího

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

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

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

Vedoucí: **Najman Pavel, Ing.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 16. května 2018

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

L.S.

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Tato práce se zabývá tvorbou netradičního vstupního zařízení s využítím zpracování obrazu. Přesněji se jedná o klávesy na papíře snímané pomocí kamery. S pomocí těchto kláves může uživatel následně ovládat multimediální aplikace jednoduchými příkazy. Cílem bylo nejen vytvořit zmíněné zařízení, ale i získat nové poznatky z oblasti zpracování obrazu. Celá práce je tak věnována analýze několika existujících řešení, základní teorii, návrhu a samotné implementaci vstupního zařízení.

Abstract

This thesis deals with creation of unusual input device based on image processing. More specifically, it's input device where camera captures keys printed on paper. Those keys are then used by user to control multimedia applications. The goal of this thesis was not only creation of such device, but also gaining new knowledge about image processing. The whole work is then devoted to analysis of existing solutions, basic theory, design and device implementation itself.

Klíčová slova

zpracování obrazu, vestavěný systém, virtuální klávesnice, Raspberry Pi, multimediální klávesnice

Keywords

image processing, embedded system, virtual keyboard, Raspberry Pi, multimedia keyboard

Citace

FRANK, Igor. *Zpracování obrazu v embedded systémech.* Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Pavel Najman

Zpracování obrazu v embedded systémech

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Pavla Najmana. Další informace mi poskytl Prof. Dr. Ing. Pavel Zemčík. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Igor Frank
16. května 2018

Poděkování

Chtěl bych poděkovat svému vedoucímu panu Ing. Pavlu Najmanovi za rady při konzultaci. Dále bych chtěl poděkovat panu Prof. Dr. Ing. Pavelu Zemčíkovi především za pomoc s výběrem tématu a za rady při ranné tvorbě práce. Nakonec děkuji všem, kteří si moji práci přečetli a upozornili mě na její nedostatky.

Obsah

1 Úvod **4**

2 Analýza existujících řešení **6**
2.1 Existující netradiční vstupní zařízení 6
2.1.1 Kinect 6
2.1.2 Projekční klávesnice 8
2.1.3 FlowMouse a detekce gesta 9

3 Teorie **12**
3.1 Obraz 12
3.2 Zpracování obrazu 14
3.2.1 Ztenčování hran – algoritmus Zhang-Suen 15
3.2.2 Hledání přímky – Houghův algoritmus 16
3.3 Vestavěný systém 17
3.3.1 Arduino 18
3.3.2 Raspberry Pi 19
3.3.3 Intel Edison 21

4 Návrh systému **23**
4.1 Specifikace požadavků 23
4.2 Výběr platformy pro vstupní zařízení 23
4.3 Konstrukce a uchycení vestavného zařízení 25
4.4 Strana klientské aplikace 25
4.5 Strana vstupního zařízení 27
4.6 Návrh komunikace 27

5 Implementace **29**
5.1 Použité nástroje pro vývoj 29
5.2 Komunikace 31
5.3 Kalibrace 31
5.3.1 Ztenčování hran 31
5.3.2 Rozpoznávání čtverců 32
5.4 Detekce vybrání čtverce 33

6 Testování **35**

7 Závěr **37**

Literatura **38**

A Obsah přiloženého paměťového média **41**

Seznam obrázků

2.1 Novější verze Kinectu určená pro generaci konzole Xbox One. 7
2.2 Kinect for Windows Sensor Components and Specifications. 8
2.3 Projekční klávesnice Celluon Epic. 9
2.4 Uchycení a využití zařízení FlowMouse. 10

3.1 Porovnání modelů RGB a CMYK a znázornění aditivního a subtraktivního míchání barev. 14
3.2 Příklad skeletonizace na písmenu H, kde linie označuje kostru. 15
3.3 Obrazová reprezentace akumulátoru. 16
3.4 Arduino Mega 2560 18
3.5 Arduino shield rozšiřující schopnosti Arduina. 19
3.6 Výkonější varianta Raspberry Pi 3 B. 20
3.7 Kompaktnější varianta Raspberry Pi Zero W. 20
3.8 Výpočetní jednotka Intel Edison. 21
3.9 Rozšiřující deska Intel Edison mini breakout board poskytující vstupy a výstupy pro výpočetní jednotku. 22

4.1 Oficiální Raspberry Pi kamera v první verzi. 25
4.2 Celková konstrukce zařízení vytvořená pomocí stavebnice Merkur. 26
4.3 Uchycení Raspberry Pi na horní části samotné konstrukce. 26
4.4 Komunikační diagram znázorňující návrh komunikace mezi klientskou aplikací a vlákny běžícími ve vstupním zařízení. 28

5.1 Diagram systému. 29
5.2 Výsledek ztenčování provedený nad samotnou klávesnicí. 32
5.3 Linie které musí tvořit čtverce pro jejich správnou detekci. 32
5.4 Snímky jendotlivých čtverců při detekovaném výběru. 34

6.1 Jednotlivé výstupy mezikroků při kalibraci za nedostatku světla. 36

Kapitola 1

Úvod

Nejdříve je nutné poznamenat, že zadání této práce bylo velmi obecné. Na zadání „Zpracování obrazu na embedded systémech“ může být aplikováno několik řešení. Mohlo by se například jednat o jednoduchý přehrávač videa či stříhací program. Další možností je detekce a klasifikace. Bylo by možné věnovat se detekci objektů, pohybu nebo rozpoznávání textu a vytvořit tak například systém rozpoznávající státní poznávací značky. Další možností by bylo vytvořit zařízení sledující určitý prostor a na jeho základě řídit s ním související systém, například řízení vlakových modelů za pomoci kamer a senzorů.

Vytvořená práce se nejvíce inspirovala posledním zmíněným, zabývá se tedy vstupním zařízením k počítači založeném na zpracování obrazu. Přesněji se jedná o „virtuální“ klávesy na papíře, na které bude moci uživatel ukazovat, a ovládat tak multimediální aplikace na svém počítači. Celý systém se sestává ze dvou částí, klientské aplikace, která slouží především pro simulaci stisku kláves, a vestavěného systému, na kterém běží samotné zpracování obrazu, a který předává informace klientské aplikaci. Takovéto vstupní zařízení lze teoreticky využít v případech, kde není možné nebo chtěné používat klasické vstupní zařízení. Využití by zmíněné zařízení našlo například v případech, kdy chceme, aby měl uživatel možnost interagovat s počítačem, ale zároveň není chtěné poskytnout plnou kontrolu pomocí klávesnice a myši. Navíc přidáním každého netradičního zařízení je zvýšena atraktivita celého systému pro uživatele.

Přes zmíněné možnosti využití je zřejmé, že existují mnohem vhodnější řešení tohoto problému. Proto se tato práce nezaměřila primárně na tvorbu revolučního systému usnadňujícího práci s počítačem, nýbrž cílem bylo zaměřit se na zpracování obrazu a nabrat nové vědomosti. Účelem práce tak bylo prozkoumat možnosti zpracování obrazu vztahující se k popsanému zařízení a následně tyto nově nabité poznatky uplatnit.

První kapitola je věnována existujícím systémům podobného účelu. Kapitola je tak zaměřena na reálné řešení vstupních zařízení na bázi zpracování obrazu. Vybrána byla dvě zařízení a jedna technika využití kamery pro ovládání počítače. Každé zařízení bylo popsáno a je vysvětleno jeho využití v praxi.

Další kapitola se zaměřuje na teorii. Je zde vysvětleno nejen jak obraz vnímá lidské oko a počítač, ale i co znamená takový vestavěný systém, a které platformy takto lze využít. V neposlední řadě je popsáno, kde se může využít různých technik zpracování obrazu, společně s popisem několika důležitých algoritmů použitých při implementaci.

V následující kapitole je popsán návrh celého systému. Část kapitoly tvoří výběr vhodné platformy, kde je popsáno z jakých důvodů byla vybrána konkrétní platforma. Dále je popsáno grafické rozhraní klientské aplikace a návrh komunikace mezi koncovými body.

Předposlední kapitola představuje především implementaci vstupního zařízení. Celý systém byl rozdělen do několika bloků, které představují části systému. Tyto části jsou seřazeny tak jak jdou po logicky po sobě, a každá z nich je následně rozeberána. Popis je zaměřen především na jejich funkčnost a jsou zde představena případná omezení, která se na celý systém vytváří.

Závěr je věnován krátce testování. Je zde popsáno jakým způsobem bylo prováděno testování, a na které části systému bylo zaměřeno konkrétně. Dva poslední odstavce se věnují z testování vyplývající problematice se světlem a možností systému pracovat v reálném čase.

Kapitola 2

Analýza existujících řešení

Tato kapitola se zaměřuje na analýzu existujících řešení. Jsou prozkoumávána zařízení jako projekční klávesnice nebo Microsoft Kinect a část je zaměřena i na technologii FlowMouse. U každého řešení byl nastudován princip fungování, možnosti využití a klady a zápory dané technologie.

2.1 Existující netradiční vstupní zařízení

Existuje celá řada netradičních vstupních zařízení, a každé je vytvořené za jiným účelem. Nalezneme například ergonomicky navržené klávesnice a myši redukující zátěž na ruce při práci s počítačem. Dále bychom mohli narazit na zařízení zaměřené na hráče, tablety využívané při práci grafiky nebo například klávesnice pro nevidomé. Pokud bychom se zaměřili na vstupní zařízení založené na zpracování obrazu, nalezneme zde pro širokou veřejnost populární Kinect. Mezi zařízení nejpodobnější mému projektu bych následně zařadil projekční klávesnici nebo technologii FlowMouse.

2.1.1 Kinect

Kinect je zařízení sledující pohyb objektů a lidí ve třech rozměrech. Poprvé vyšel na trh v roce 2010 a to pro konzoli Xbox 360. Měl za úkol rozšířit konzoli i mezi uživatele, kteří běžně hry nehrají, a zvýšit tak atraktivnost celého systému přidáním možnosti ovládat hry pomocí pohybu. Později, společně s novou generací konzole Xbox, vyšla i nová verze senzoru (viz obr. 2.1), která nabídla především přesnější snímání gest a pohybu. V říjnu roku 2017 však Microsoft oznámil zastavení výroby poslední verze senzoru, což znamenalo konec Kinectu.[27] Nicméně v době psaní této práce je senzor stále v prodeji a lze jej koupit ve dvou verzích, pro konzoli Xbox One a PC. Verze se liší pouze vzhledem, přičemž funkčnost a hardwarová výbava je identická. [11]

Jak je znázorněno na obrázku 2.2, po hardwarové stránce Kinect obsahuje RGB kameru, senzor a vysílač infračerveného světla a mikrofonní pole. Ke stejnému hardwaru navíc původní Kinect pro Xbox 360 přidal i motor schopný natáčet celý senzor nahoru a dolů chybějící u novější verze senzoru. Kinect snímá barevný obraz pomocí RGB kamery, který je využíván k doplnění informací získaných z hloubkového obrazu, případně jej lze využít i jako prostou webkameru. Hloubkový obraz Kinect získává pomocí infračerveného senzoru a vysílače. Ten vysílá pravidelné infračervené paprsky, které se odráží od objektů v místnosti a senzor je zachytává. Hloubku obrazu pak vypočítává pomocí délky času, kterou paprsek urazil od vysílače k senzoru. Mikrofonní pole slouží nejenom k rozpoznávání hla-

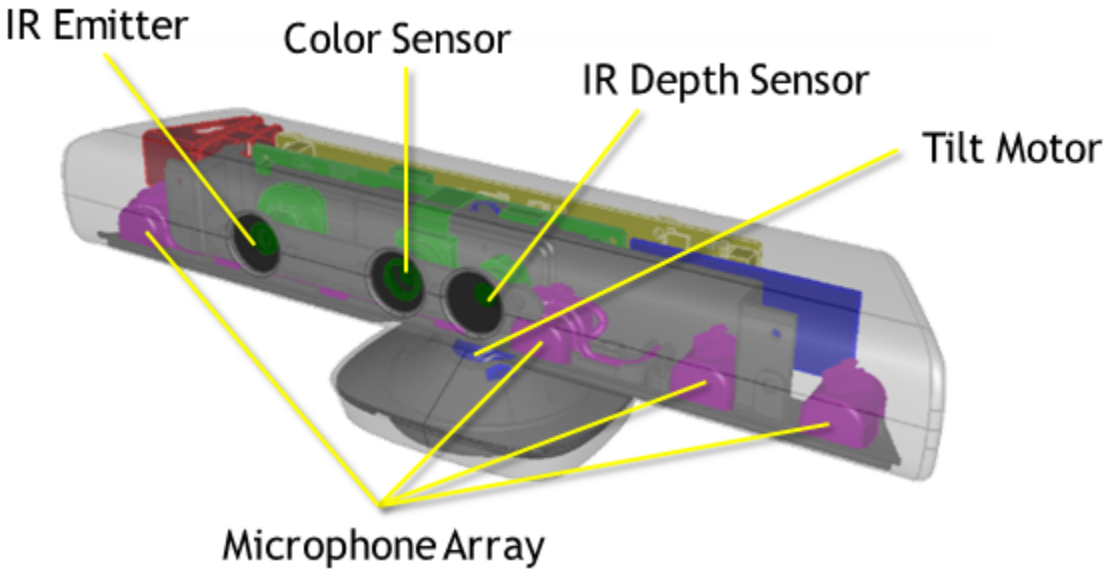
Obrázek 2.1: Novější verze Kinectu určená pro generaci konzole Xbox One.
Zdroj: https://www.xbox.com/cs-CZ/xbox-one/accessories/kinect

sových příkazů, ale pomocí více mikrofonů lze určit směr, ze kterého zvuk přichází a lépe odstranit šum v pozadí. [21]

Pro Kinect pro PC bylo vydáno několik SDK od třetích stran, ale i přímo oficiální SDK od společnosti Microsoft, pomocí nějž lze programově využít potencionál celého zařízení. Například programátor může využít schopnost Kinectu rozpoznávat až šest osob a u dvou z těchto šesti osob sledovat detailní pohyby bez toho, aniž by musel implementovat zpracování dat ze senzorů v Kinectu. Pokud je však potřeba, může programátor využít datového proudu přímo, a to z RGB kamery, infračerveného senzoru i mikrofonního pole.[20]

Poskytnuté API je určeno především pro vývoj aplikací na PC, a ačkoliv byl Kinect původně vydáván jako příslušenství pro herní konzoli, vznikají i aplikace mimo herní odvětví. Existují například systémy pro 3D skenování objektů využívající Kinect. Aplikaci pro 3D sken na svých stránkách nabízí i přímo společnost Microsoft. Po zprovoznění aplikace není zapotřebí dalších zařízení a pro naskenování stačí otáčet senzorem okolo objektu.[19] Dalším zajímavým projektem je vytvoření překladače pro znakovou řeč. S pomocí Kinectu aplikace sleduje pohyb rukou uživatele, rozpoznává jednotlivé znaky, a nakonec větu zobrazí na obrazovce a pomocí hlasového syntezátoru reprodukuje.[28]

Kinect je tedy velice zajímavé zařízení vhodné pro složitější zpracování obrazu. Není to však jednoúčelové řešení pro potřeby uživatele, nýbrž prostředek jak tyto potřeby řešit. Hodí se především na zpracování obrazu, kde je nutné nebo výhodné pracovat i s hloubkou obrazu. Pro běžné zpracování obrazu, jako tomu je například u této práce, bylo shledáno toto zařízení zbytečně komplexní a současně i nedostatečné. Primárním problémem je počet snímků zachycených za sekundu, kde původní Kinect pro Xbox 360 dosahuje 30 snímků za sekundu při rozlišení 640x480 a 10-15 snímků při rozlišení 1280x1024. [22]



Obrázek 2.2: Kinect for Windows Sensor Components and Specifications.
Zdroj: https://msdn.microsoft.com/en-us/library/jj131033.aspx

2.1.2 Projekční klávesnice

První zařízení podobné dnešním projekčním klávesnicím bylo vymyšleno v roce 1992 inženýry zaměstnanými ve společnosti IBM, kteří jej dali i patentovat. V patentu však šlo pouze o sledovaní uživatelových rukou a jejich pohybu v prostoru, nikde nedocházelo k projekci klávesnice na povrch. Tento typ interakce uživatele a počítače byl navrhnut, aby nabídnul nové a pokročilejší možnosti zadávaní dat a příkazů do počítače. Dalším důvodem byl trend minimalizace zařízení, a jak autoři popisují, klávesnice jako takové nelze dostatečně minimalizovat, aniž by nedošlo ke ztrátě funkčnosti. Zároveň zde autoři popisují i jiné možnosti řešení problému vstupních zařízení při minimalizaci. Jsou jimi hlasové příkazy nebo převod psaného textu do počítače, u kterých poukazují na náročnost na výkon při zpracování vstupních dat. U svého nápadu pak vyzdvihují nízkou pořizovací cenu, ale i určitou univerzálnost. Jako vstup lze brát nejen obyčejnou klávesnici, ale i jako numerickou klávesnici u notebooků, kde není dostupná. Další možností je využívat tento systém jako virtuální klavír, a stvořit tak počítač pro tvorbu hudby. Možností, jak celý systém využít, je mnoho. [17]

Dnešní projekční klávesnice pracují na dvou principech. U obou případů je však na podložku za pomocí laseru promítána klávesnice. V prvním případě je prostor klávesnice snímán kamerou, která detekuje interakci uživatele s klávesnicí. U druhé verze je těsně nad podložkou vysílán neviditelný infračervený paprsek, který pak snímá senzor, jenž rozpozná jeho přerušení a dokáže určit, kde byl paprsek přerušen. [12] Tento paprsek je přítomný i u prvního typu, místo senzoru je však klávesa rozpoznána pomocí kamery. Typ založený na infračerveném senzoru je ale mnohem dostupnější a častější. Jedna z nyní prodávaných klávesnic je například Epic od korejské firmy Celluon znázorněná na obrázku 2.3. Tato klávesnice se připojuje pomocí bezdrátové technologie Bluetooth a dokáže dokonce rozpoznávat jednoduchá gesta prováděná na podložce, na kterou je klávesnice promítána.

Díky technologii Bluetooth nejsou klávesnice limitovány na jeden typ zařízení. Přesto není pravděpodobné, že by se tato klávesnice využívala u notebooku nebo stolního počítače.

Obrázek 2.3: Projekční klávesnice Celluon Epic.
Zdroj: https://www.celluon.com/shop/epic/

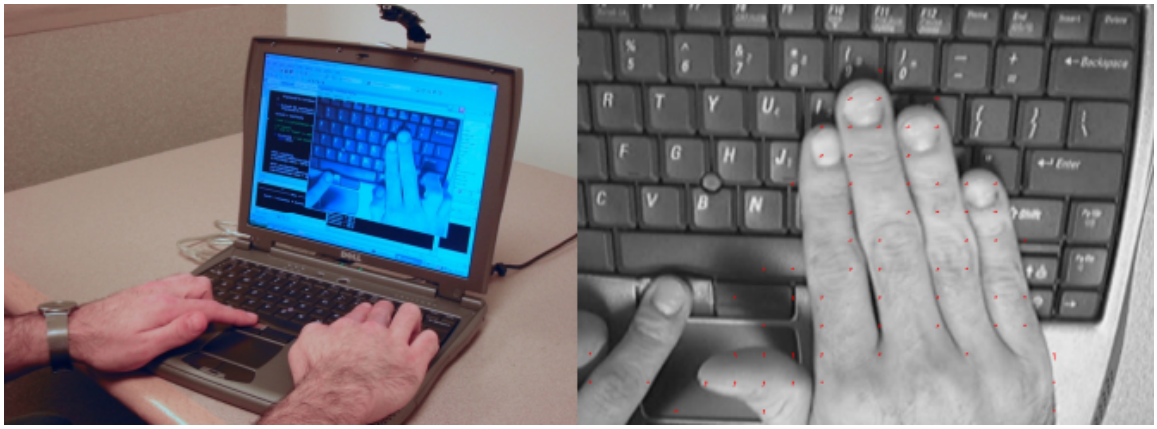
Jedná se spíše o produkt zaměřený na mobilní zařízení jako chytré telefony a tablety. Navíc projekční klávesnici nejspíše nevyužije každý uživatel. Pro běžného vlastníka mobilního zařízení bude rychlejší a pohodlnější využít pro napsání SMS zprávy nebo ktratšího emailu softwarovou klávesnici dostupnou přímo v zařízení. Projekční klávesnici by však mohl využít člověk, který cestuje a píše často delší texty. Ten pak potřebuje kompaktní zařízení, na kterém dokáže pohodlně psát. Problémem se však může stát okolní světlo či povrch, na který je klávesnice promítána. Na přímém světle nebo na lesklém povrchu se může stát, že laserová projekce bude špatně čitelná. Celkově jsou tedy projekční klávesnice kompaktní, přenosné, a přestože nejsou náhradou běžných klávesnic, svoje uživatele si najdou především díky atraktivitě a zmíněné kompaktnosti.

2.1.3 FlowMouse a detekce gesta

FlowMouse je polohovací zařízení založené na počítačovém vidění. Celý tento systém byl popsán autory Andrewem D. Wilsonem a Edwardem Cutrellem v článku vydaném v roce 2005 v knize Human-Computer Interaction – INTERACT 2005. [25] Tento systém nabízí ovládání počítačové myši pomocí ruky pohybující se nad klávesnicí. Autoři celý tento systém představují především pro notebooky, kde je kamera připevněna na horní část obrazovky a snímá prostor nad klávesnicí. Díky dotykovému pásku umístěnému v prostoru klávesnice lze FlowMouse jednoduše zapínat a vypínat. Autoři jako nejlepší místo pro umístění tohoto senzoru vybrali levé tlačítko myši na touchpadu. Chtěli především nalézt místo, které nemění chování uživatele při práci s počítačem a zároveň není dostupné pouze rukou používanou k ovládání kursoru. Samotnou ukázku použití lze vidět na obrázku 2.4 pocházejícího ze zmíněného článku. Na prvním obrázku lze vidět umístění kamery na notebooku. Na druhém obrázku je pak znázorněna samotná detekce uživatelovy ruky.

Aby docházelo k zachytávání pohybu, je tedy nutné, aby se uživatel druhou rukou dotýkal zmíněného senzoru. Během té doby jsou porovnávány vždy dva snímky, současný a předchozí. Každý snímek je pak rozdělen na několik segmentů, které jsou mezi sebou porovnávány. Autoři pracují s rozlišením 640x480 pixelů a obraz rozdělují na 20x15 segmentů. Pro

každý tento segment je pak vypočítán směr a velikost pohybu tak, že v každém segmentu je sečtena hodnota pixelů a tyto hodnoty se porovnávají. Odpovídající segmenty jsou pak ty, které mají nejmenší rozdíl vypočtených sum. Pomocí dalších výpočtů, které zahrnují mimo jiné korekci vertikálního pohybu a přepočet na správné rozlišení displeje, je získán offset pohybu myši.[25]



Obrázek 2.4: Uchycení a využití zařízení FlowMouse.
Na levé straně lze vidět umístění kamery a aktivaci pomocí senzoru. Na pravé straně je ukázána detekce ruky jako kursoru.
Zdroj: FlowMouse: A Computer Vision-Based Pointing and Gesture Input Device

V dalším článku jednoho z autorů (Andrew D. Wilson) [26] je nabízena možnost rozpoznávání gesta „štípnutí“, kdy se spojí palec a ukazováček do pomyslného oválu. Tento systém pak navrhují integrovat do výše zmíněného systému FlowMouse. Využitím zmíněného gesta je možnost zapínání a vypínání polohovacího zařízení namísto dotykového senzoru zavedeného na touchpad notebooku. S tímto vylepšením by tedy nebylo nutné zavádět senzor a jediná komponenta, do které by bylo nutno investovat, je prostá webkamera.

Algoritmus nejprve vytvoří binární obraz, ve kterém je detekována ruka uživatele. Binárního obrazu je dosaženo porovnáváním předem získaného obrazu pozadí se sejmutým obrazem v konkrétním čase. Pokud je pixel výrazně světlejší, než pixel v obrazu pozadí, je vyhodnocen jako bod obsahující část ruky. Pomocí zmíněhého postupu se dá zároveň velice snadno vyrušit ovlivnění pomocí stínů. Detekování děr v obrazu reprezentujících gesto pak probíhá za pomoci algoritmu flood fill. Pomocí tohoto algoritmu jsou všechny díry v obrazu ohodnoceny podle velikosti. Následně jsou z potencionálních gest vyřazena ta, která obsahují krajní pixely, čímž se snadno vyřadí nekompletní gesta na kraji obrazovky. Konečné detekování gest probíhá porovnáváním velikosti děr s prahem reprezentujícím minimální velikost gesta.

Při použití systému detekce gest tak lze obohatit polohovací zařízení FlowMouse o ovládání oběma rukama, což přináší nové a zajímavé způsoby, jak interagovat s počítačem. Autoři implementovali gesta pro práci s mapou využívající obě ruce. Uživatel může mapu posouvat, rotovat nebo přibližovat pomocí gest obou rukou. Systém tak nabízí velice zajímavý a přirozený způsob ovládání, který by mohl mít využití nejen u mapových softwarů, ale například by mohl nabídnout zajímavou možnost, jak si prohlížet 3D modely. Bohužel u tohoto provedení nastává několik problémů. Prvním a asi i nejočekávanějším problémem je světlo. Při nedostatku světla je samozřejmě nemožné pomocí kamery snímat, a je nutno použít přídavné osvětlení. Dalším problémem, který autoři zmiňují, je, že snímaný prostor

pro uživatele není viditelně vymezen. Proto může dojít k situaci, že se uživatel pokouší ovládat aplikaci mimo prostor snímaný kamerou. Poslední problém může nastat při vytváření binárního obrazu. Zde nastává potíž, pokud uživatel nepracuje nad dostatečně kontrastní podložkou. Může tedy nastat situace, kde při převodu do binárního souboru nebude detekována ruka uživatele, a ten nebude moci ovládat svůj počítač.[26]

Kapitola 3

Teorie

Pro implementaci a hlubší poznávání problematiky je důležité mít základní znalosti z oborů, kterých se tento projekt týká. Následující kapitola se proto zaměřuje na základní teorii. Popisuje se zde obraz z pohledu vnímání člověkem, ale i z pohledu uložení v počítači, možnosti zpracování obrazu a s ním související důležité algoritmy, které byly použity při implementaci a v neposlední řadě je vysvětlen pojem vestavěný systém, kde jsou mimo jiné uvedeny možné platformy pro využití jako vstupní zařízení.

V následujících podkapitolách bylo čerpáno z knihy Moderní počítačová grafika a skript k předmětu Základy počítačové grafiky pro podkapitolu Obraz. Dále byla nastudována opora předmětu Zpracování obrazu a kniha Zpracování obrazu a algoritmy v C# pro část o zpracování obrazu. Nakonec pro část popisující vestavěné systémy byla využita skripta předmětu Mikroprocesorové a vestavěné systémy a kniha Encyclopedia of Computer Science and Technology.

3.1 Obraz

Chceme-li se bavit o tom, co je to obraz nebo jak je reprezentován v počítači, je nutné nejdříve vysvětlit, jak funguje lidské vidění. Člověk dokáže vnímat svět okolo sebe díky odrazu světla od okolních objektů. Některé frekvence světla se od objektu odrazí a jiné jsou pohlceny, výsledné odražené světlo pak lidským okem vnímáme jako barvu předmětu. Viditelné světlo jako takové je součástí elektromagnetického spektra ve vlnové délce od přibližně 380 do 720 nm. Elektromagnetické záření mimo tento rozsah je pro lidské oko neviditelné, přesto se však využívá pro různé formy zobrazování. Záření s krátkou vlnovou délkou (okolo 0,1 nm) nesou dostatečnou energii pro prostup větším objemem hmoty. Nejznámějším využitím je asi rentgenové záření používané především v medicíně pro získávání obrazu vnitřních částí lidského těla, využívat se může ale i pro kontrolu vnitřní struktury výrobků a hledání jejich defektů. Naopak zaměříme-li se například na infračervené záření, které má větší vlnovou délku než viditelné světlo, zjistíme, že jej vyzařují tepelné objekty. Tento jev se proto dá využít pro detekci osob, zvířat či sálajících objektů ve tmě.

Prostor okolo nás tedy vnímáme lidským okem díky odrazu světla od okolních objektů. Nejdůležitějšími částmi lidského oka jsou rohovka, duhovka, čočka a sítnice. Světlo přicházející do oka nejprve prochází rohovkou, která má však u oka převážně ochranný charakter. Dále světlo projde duhovkou se schopností se stahovat a roztahovat, a tím ovládat množství světla vstupujícího do oka. Poslední částí oka, kterým projde světlo, je čočka. Čočka je průhledná a pružná vrstva buněk připojená ke svalům. Pomocí těchto svalů se čočka smršťuje

a natahuje, čímž mění svoji ohniskovou vzdálenost, a umožňuje zaostřovat na vzdálené či blízké předměty. Nakonec světlo dopadá na sítnici, kde se nacházejí fotoreceptory čípky a tyčinky, předávající signály pomocí zrakového nervu do mozku. Tyčinky jsou schopné rozpoznávat pouze odstíny šedi a jsou proto primárně určené pro neostré noční vidění. Jsou zároveň desetkrát citlivější než čípky a na sítnici jich nalezneme přibližně patnáctkrát více. Tyčinky jsou rozmístěné po celé sítnici a jejich největší koncentrace se nachází okolo slepé a žluté skvrny. Slepá skvrna je místo, kde se nervové zakončení sbíhá do zrakového nervu. Žlutá skvrna představuje oblast s nejvyšší koncentrací čípků. Každý čípek obsahuje jeden fotopigment zajišťující lidem barevné vidění. Existují 3 druhy fotopigmentů, červený, zelený a modrý, každý citlivý na jinou vlnovou délku světla. Důležité je nakonec zmínit, že jednotlivé fotopigmenty nejsou zastoupeny ve stejném poměru. Nejvíce je červených pigmentů (64 %), pak následují zelené pigmenty (32 %) a nakonec modré (2 %).

Počítač samozřejmě vnímá a zpracovává obraz úplně jinak než lidské oko. Pro definování obrazu matematicky můžeme použít následující vzorec definující spojitou funkci dvou proměnných.

$$z = f(x, y) \tag{3.1}$$

Zdroj: Moderní počítačová grafika

Jelikož obraz je zvláště v počítači prostorově omezen, můžeme zapsat funkci jako kartézský součin dvou spojitých intervalů. Obrazová funkce pak realizuje následující zobrazení:

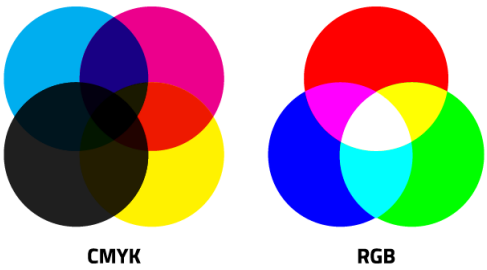
$$f : (< x_{min}, x_{max} > \times < y_{min}, y_{max} >) \to H \tag{3.2}$$

Zdroj: Moderní počítačová grafika

Hodnoty x a y reprezentují souřadnice v obraze, kde funkce f nabývá určité hodnoty z z oboru hodnot H. Hodnota může být jedna, reprezentující kupříkladu jas nebo barvu v monochromatickém obrazu, ale může se jednat i o n-tici reprezentující více informací o daném bodě, jako jsou třeba hodnoty jednotlivých barevných složek v barevném modelu RGB.

Jelikož běžné počítače nedokáží zpracovávat spojité signály, je obraz reprezentován v paměti počítače diskrétně, kde souřadnice x a y jsou celá čísla. Nejjednodušší reprezentací obrazu v počítači je tedy dvojrozměrné pole bodů, z nichž každý nese informaci podle typu obrazu. Každému takovému bodu v obrazu se pak říká pixel, což je zkratka z anglického dvousloví picture element. Nese-li pixel pouze jeden typ informace, vzniká nám monochromatický obraz. Pokud bychom chtěli mít obraz barevný, musíme vybrat barevný prostor a hloubku pro reprezentaci.

Barevný prostor je výčet složek, pomocí kterých můžeme poskládat výslednou barvu. Mezi základní barevné prostory bychom mohli zařadit například RGB(A), CMY(K), HSV/HLS nebo YUV. RGB je barevný prostor, ve kterém se kombinují dohromady intenzity barvy červené (Red), zelené (Green) a modré (Blue). U prostoru RGBA je pak přidán alfa kanál představující průhlednost celé barvy. V prostoru RGB(A) funguje tzv. aditivní míchání barev. Znaméná to, že čím více přidáváme barev, tím světlejší je výsledek. Pokud tedy smícháme všechny barvy v nejvyšší intenzitě, vznikne nám bílá barva. Prostor CMY(K) je subtraktivní, kde složením všech barev vznikne barva černá. Tento prostor pak pracuje s barvami tyrkysovou (Cyan), fialovou (Magenta) a žlutou (Yellow). Prostor CMY může být rozšířen o černou složku (blacK) pro dosáhnutí plně černé barvy. Rozdíl mezi RGB a CMYK a zároveň rozdíl mezi subtraktivním a aditivním mícháním barev lze vidět na obrázku 3.1. Prostory HSV a HLS jsou definovány bez použití základních barev. Prostor HSV



Obrázek 3.1: Porovnání modelů RGB a CMYK a znázornění aditivního a subtraktivního míchání barev.
Zdroj: https://trillioncreative.com

je reprezentován pomocí trojice barevný tón (Hue), sytost (Saturation) a jasová hodnota (Value), prostor HLS pak pomocí barevného tónu (Hue), světlosti (Lightness) a sytosti (Saturation). Tyto barevné modely jsou vhodné především pro tvůrce grafiky, jelikož poskytují přirozenější způsob výběru barvy. Barevný prostor YUV se využívá především pro přenos televizních signálů, kde se jedna složka využívá pro přenos jasové složky (Y) a ostatní dva signály přenáší informace o barevných složkách. Barevná hloubka pak představuje rozsah hodnot, kterými můžeme vyjádřit jednotlivé složky v barevném prostoru. U počítačů to znamená, kolika bity jsou jednotlivé složky reprezentovány.[29] [14]

3.2 Zpracování obrazu

Zpracování obrazu je způsob zpracovávání signálu, kde jsou na vstupu jako signál obrazová data. Při zpracování obrazu se vstupní obraz snažíme transformovat na jiný námi požadovaný obraz, nebo se z něj snažíme pro další práci získat příznaky charakterizující obraz původní. Tyto příznaky pak můžeme využít pro popis a zařazení obrazu či detekci tvarů a objektů.

Zpracování obrazu můžeme využít například pro úpravu pořízených digitálních fotografií. Můžeme používat různé filtry a metody pro redukci šumu, doostření obrazu, nebo jsme schopni provádět úpravy jasu a barevného tónu obrazu, možností je mnoho. Pokud jsme pořídili nějakým způsobem geometricky deformovanou fotku, můžeme využít geometrické transformace objektu a fotografii tímto způsobem vylepšit. Tato transformace však neslouží pouze k opravování špatně pořízených fotografií, ale lze díky ní tvořit „umělecké“ fotografie pomocí deformací, jako je efekt rybího oka. Fotíme-li panoramata, běžným postupem je vyfotit několik překrývajících se snímků, které nám poté dodatečný software, případně rovnou fotoaparát či mobilní telefon, spojí v jedinou panoramatickou fotografii. V průběhu spojování fotografií pak dochází ke zpracování obrazu ve formě automatického rozpoznávání překrývajících se částí, normalizace jasu a ořezu jednotlivých snímků. Další možností využití zpracování obrazu je použití technik pro zvýrazňování rysů v obraze. Pomocí těchto technik lze v obraze zvýraznit či naopak potlačit části obrazu jako jsou například hrany či vrcholy, části obrazu s určitou barvou nebo jasem, případně jinak definované části obrazu. Tyto zvýrazněné části se mohou dále používat například pro klasifikaci. Pro tyto potřeby může posloužit i segmentace obrazu, kdy je obraz rozdělen na několik spolu souvisejících částí. Tyto části mohou být používány kupříkladu pro hrubé rozdělování obrázků do kategorií podle obsahu. Velice zajímavou možností je pak ukládání informací do obrázku. Do

obrázku jako takového lze vložit vodoznak, ať už viditelný nebo skrytý. Viditelný vodoznak slouží jako znatelná autorská ochrana proti padělání, avšak pomocí neviditelného vodoznaku lze dokonce do obrázku zapsat zprávy tak, aby lidským okem nebylo poznat, že se v obrázku zpráva ukrývá. Této technice se říká steganografie a lze ji kombinovat i s šifrováním, aby byla zpráva nejen skrytá, ale i nečitelná, pokud by byla odhalena. [30] [10]

Většinu těchto metod lze na obrázek aplikovat pomocí softwaru jako je GIMP či Adobe Photoshop. Tyto programy jsou zaměřené především na tvorbu a úpravu fotografií pro domácí a profesionální umělecké použití. Získávání rysů z obrazu je většinou záležitost většího celku, kde dochází ke zpracování oněch rysů, klasifikaci a dalším procesům. Běžný uživatel tak nemá potřebu využívat těchto možností zpracování. Vývojáři samotné získávání rysů implementují vlastnoručně nebo využívají knihoven, jako je například OpenCV, které poskytují potřebné funkce.

V následujících dvou sekcích jsou popsány důležité algoritmy využité při implementaci vstupního zařízení. Jendá se o ztenčování hran pomocí algoritmu Zhang-Suen a hledání přímek pomocí Houghovy transformace.

3.2.1 Ztenčování hran – algoritmus Zhang-Suen

Pomocí algoritmů pro ztenčování dostáváme ze snímků kostry objektů, které jsou pouze jeden pixel široké. Tyto techniky pro získávání kostry jsou hojně využívány především při rozpoznávání znaků z obrazu v OCR aplikacích (viz obr. 3.2). Předzpracováním obrazu a vytvořením kostry tak dojde k odstranění velkého množství pixelů a další algoritmy pracují s mnohem jednoduššími modely, které však zachovávají většinu vlastností. [23]

Obrázek 3.2: Příklad skeletonizace na písmenu H, kde linie označuje kostru.

Jednou z možností pro dosažení kostry je využít algoritmu Zhang-Suen. Popis algoritmu vyšel poprvé v roce 1984 v článku [31] vydaném v časopise ACM a do dnešních dob je velice populárním algoritmem. Vstupem pro tento algoritmus je nejčastěji binární obraz po detekci hran, například po aplikování Sobelovy detekce hran. V principu je algoritmus velice jednoduchý. Jedná se o dvoufázový průchod obrazem, přičemž pro každý nalezený bílý pixel (pixel reprezentující ztenčovanou oblast snímku) je vyhodnoceno několik podmínek vztahujících se k okolním pixelům. Splňuje-li pixel veškeré podmínky, je pouze označen pro budoucí smazání a pokračuje se dalším pixelem. Všechny takto označené pixely jsou smazány až po průchodu celým obrazem, aby nedocházelo k ovlivňování jiných částí obrazu. Druhá fáze algoritmu je naprosto totožná, pouze s pozměněnými podmínkami. Pokud se v jedné z fází vymazal alespoň jeden pixel, nastává další iterace a celý dvoufázový průběh se opakuje. Algoritmus končí, nedojde-li k žádné změně v obraze.[31] [23]

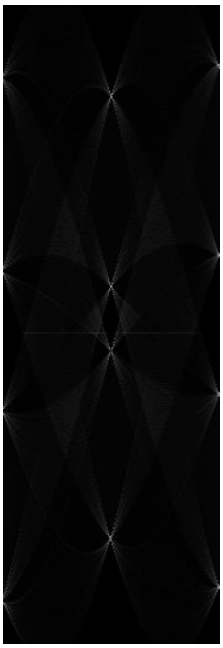
3.2.2 Hledání přímky – Houghův algoritmus

Pro správnou funkčnost zpracovávaného vstupního zařízení bylo nutné mít možnost nacházet přímky v obraze. Z tohoto důvodu byl zvolen Houghův algoritmus, kde principem je v jednoduchosti převod z kartézského souřadného systému do polárního a zase nazpět. Prvním krokem je vytvoření akumulátoru reprezentujícího prostor s polárním souřadným systémem také zvaným jako Houghův prostor. Akumulátor však musí mít specifické rozměry. Jedním rozměrem jsou stupně úhlů, kterými může být přímka natočena, a druhým je největší možná vzdálenost v obraze. Jako stupně úhlů stačí pouze rozsah 180°, jelikož dále přímka opakuje svoje natočení. Po přichystání akumulátoru nastává převod z kartézského systému do polárního. Postupně je procházen obraz a pro každý výskyt bílého pixelu, který představuje bod na přímce, je vypočítána následující rovnice:

$$r = x \cos \rho + y \sin \rho \tag{3.3}$$

Zdroj: Zpracování obrazu a algoritmy v C#

Rovnice je reprezentací přímky v polárním souřadném systému. Parametry x a y jsou známé, jsou to souřadnice aktuálního bodu v obraze. ρ následně reprezentuje sklon přímky. Jelikož bodem může procházet nekonečné množství přímek a pouze z bodu nezjistíme, která z přímek je naše hledaná, za ρ dosadíme postupně úhly 0°-180°. Výsledné r nám udává velikost kolmice od počátku k přímce. Vypočítané hodnoty využijeme následně v akumulátoru (viz obr. 3.3), kde inkrementujeme každou pozici odpovídající souřadnicím $[\rho,r]$. Pro každý bod v původním obraze je takto vytvořena sinusoida uvnitř akumulátoru, udávající všechny možné přímky procházející bodem. Máme-li dva body v obraze a tedy dvě sinusoidy v akumulátoru, průnik dvou sinusoid reprezentuje přímku procházející zmíněnými dvěma body v obraze. Parametry této přímky jsou souřadnice průniku $[\rho,r]$. [10] [23]



Obrázek 3.3: Obrazová reprezentace akumulátoru.
Čím bělější bod tím více sinusoid jim prošlo. Na obrázku lze pak vidět osm bodů reprezentujících osm přímek.

Po zapsání všech informací do akumulátoru přichází na řadu rozpoznávání průniků reprezentujících přímky v původním obraze. Postupně se tedy prochází celý akumulátor a jsou hledány veškeré body nad určitou hodnotou, neboli počet sinusoid, které bodem prošli. Vzít každý bod splňující určitý počet průniků sinusoid však není nejlepším řešením, proto v okolí takového bodu je navíc zjišťováno, zda se jedná o lokální maximum v rozšířeném Moorově okolí. Pokud se opravdu jedná o lokání maximum, pomocí následujících rovnic 3.4 a 3.5 vycházejících z rovnice 3.3 jsou vypočítány dva body reprezentující přímku.[10] [23]

$$x = \frac{r - y \sin \rho}{\cos \rho} \tag{3.4}$$

Zdroj: Odvozeno z rovnice 3.3

$$y = \frac{r - x \cos \rho}{\sin \rho} \tag{3.5}$$

Zdroj: Odvozeno z rovnice 3.3

Pro jednoduchost výpočtu krajní body přímky vždy leží na kraji obrazu, tudíž má jedna ze souřadnic hodnotu nula nebo hodnotu maximální šířky případně délky obrazu. Pro rozhodnutí, která ze souřadnic bude dopočítána, slouží úhel sklonu. Pro přímku se sklonem v intervalu $<0°;45°) \cup (135°;180°)$ je dopočítávána souřadnice y, jelikož tato přímka je nejblíže k tomu, co bychom popsali jako vodorovná přímka. V intervalu $<45°;135°>$ se jedná o „vertikální“ přímku a je dopočítávána souřadnice x. [10] [23]

3.3 Vestavěný systém

Vestavěný systém neboli embedded systém je jednoúčelový systém, který je součástí většího celku. Vestavěné systémy můžeme najít v autech, pračkách, žehličkách, ale i třeba v semaforech či prodejních automatech. Takový vestavěný systém má pak přesně danou funkci v rámci celého systému a její vykonávání není pro uživatele přímo viditelné. [15]

Vestavěné systémy typicky obsahují několik částí. Jádrem systému je mikrokontroler či mikroprocesor. Mikroprocesor je čip obsahující téměř vše potřebné pro vykonávání programu. Čip obsahuje obvykle aritmeticko-logickou jednotku (ALU), dekodér instrukcí, registry a řadič. Všechny tyto části řídí chod systému a starají se o komunikaci s periferiemi. Mikrokontroler je pak jednotka obsahující stejné části jako mikroprocesor, ale přidává ještě další komponenty jako paměť RAM, programovou paměť a obvody zajišťující vstupně výstupní operace. Z toho je zřejmé, že pokud vestavěný systém využívá mikroprocesor, musí mít k dispozici také nevolatilní paměť pro uložení kódu.

K těmto systémům jsou zpravidla připojeny další vstupní a výstupní komponenty poskytující možnost uživateli s daným systémem pracovat. Takovými vstupními komponenty mohou být například tlačítka, kamery či různé senzory kontrolující stav systému nebo okolí. Jako výstupní zařízení se u vestavěných systémů nacházejí například LED kontrolky, displeje, motory, reproduktory či topné spirály. Jelikož výstup ze senzorů je často analogový signál, je zároveň zapotřebí analogově digitální převodník, schopný konvertovat signál ze senzoru na pro mikrokontroler/mikroprocesor čitelný diskrétní signál.

Pro vytváření programů běžících ve vestavěných systémech existují specializované kompilátory a jazyky, jako je například jazyk FORTH. Při navrhování programů pro vestavěné systémy je pak obzvláště důležité dbát na robustnost a odolnost vůči chybám. Takovéto systémy musejí často nepřetržitě běžet a být neustále funkční bez jakýchkoliv vnějších zásahů. Je tedy nutné, aby se systémy dokázaly obnovit z chybových stavů do výchozího stabilního stavu. [13]

V následujících několika sekcích jsou příblíženy platformy, které jsou využitelné jako vestavěné systémy. Zkoumány byly tři zařízení, Arduino, Raspberry Pi a Intel Edison. U

každého zařízení je popsána motivace pro jeho vznik, hardwarové parametry a způsob vývoje pro danou platformu.

3.3.1 Arduino

Arduino je open-source hardwarová platforma, pomocí níž lze vytvářet zajímavé projekty. Jeho rodištěm je Ivrea Interaction Design Institute, kde vznikalo jako jednoduchá platforma pro studenty, kteří neměli zkušenosti s programováním ani elektronikou. Tento projekt tedy vznikal za podobným účelem, jako mnohem sofistikovanější FITkit na VUT Fakultě informačních technologií. Jakmile se Arduino dostalo mezi širší veřejnost, začala stejnojmenná firma vyrábět řadu různých modelů. Každý z těchto modelů je pak výkonnostně, rozměrově či jinak zaměřen na konkrétnější oblast. Najdeme zde zařízení zaměřené na 3D tiskárny, nositelnou elektroniku či IoT aplikace.

Obrázek 3.4: Arduino Mega 2560
Zdroj: https://store.arduino.cc

Nejrozšířenějšími se pak staly desky Arduino Uno a Arduino Mega 2560 (viz obr. 3.4) nabízející především více pinů a paměti. Základem obou jsou osmibitové mikrokontrolery od společnosti Atmel pracující na frekvenci 16 MHz. Dále jsou přítomny paměti SRAM a Flash ve velikosti 32 kB a 2 kB u zařízení Arduino Uno a 256 kB a 8 kB u Arduino Mega 2560. Zmíněná Arduina mají v aktivním režimu příkon přibližně 0,12 W bez dalších připojených zařízení či součástek. Jedná se však o aktivní režim a při běžném provozu budou hodnoty menší. Jediná konektivita, kterou tyto platformy nabízí, jsou vstupně výstupní analogové a digitální piny a sériová komunikace přes USB port. Přesto lze k Arduinu zapojit nejenom běžné elektronické součástky jako jsou LED, tlačítka, rezistory, senzory, případně větší zařízení jako kamery a displeje, ale i tzv. Arduino Shields (viz obr. 3.5). Tyto moduly se nasazují na přítomné piny Arduina a rozšiřují jeho funkčnost o možnosti jako jsou zápis na SD kartu, komunikaci přes Ethermet či WiFi, nebo sledování polohy pomocí modulu GPS.

Pro oživení samotného Arduina je poskytnut programovací jazyk Arduino. Ten je téměř totožný s programovacím jazykem C/C++, přičemž však nelze využít standardních knihoven. Arduino nicméně nabízí několik vlastních knihoven zaměřených především na práci s rozličnými moduly a několik specifických knihoven zaměřených přímo na konkrétní modely Arduina. Pro vývoj a programování samotného zařízení je poskytnuto vývojové prostředí Arduino IDE. Další možností je využít webového rozhraní, kde lze kód nejen psát, ale s po-

Obrázek 3.5: Arduino shield rozšiřující schopnosti Arduina.
Konkrétně se jedná o GSM shield schopný odesílat SMS, provádět hovory a připojit se na internet.
Zdroj: https://store.arduino.cc

mocí nainstalovaného pluginu i naprogramovat do zařízení. Nakonec se dá sáhnout i přímo po nástrojích od tvůrců mikrokontroleru Atmel. [2]

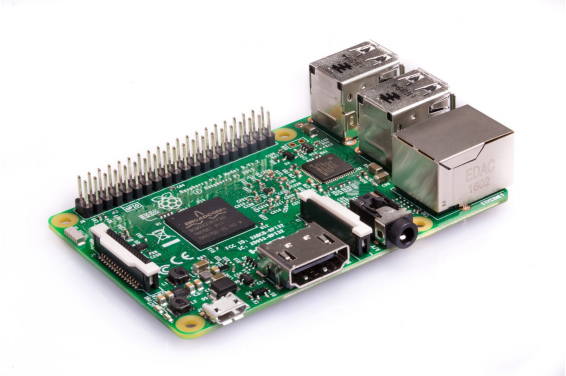
Celkově je Arduino ze všech tří platforem asi nejrozšířenější, přičemž jedním z důvodů rozšířenosti je cena. Ta se pohybuje u originálů přibližně mezi 600 – 1200 Kč. [6] Na trhu však díky otevřenosti platformy nalezneme několik legálních klonů. U těch je pak cena několikanásobně nižší a pohybuje se v řádech stovek korun podle dodávaného příslušenství. [3] Cena se dá dále snížit, koupíme-li si samostatné součástky a Arduino poskládáme sami. Dalším důvodem rozšířenosti je komunita pohybující se okolo celé platformy. Na oficiálních stránkách jsou prezentovány nejrůznější projekty pro inspiraci a na oficiálním fóru lze najít rady nejen ohledně výběru projektu, ale i programování a elektroniky. Zmíněné modely Adruino Uno a Mega pak hravě zvládnou ovládat většinu součástek a periferií, nicméně se nehodí na náročné výpočty, ať už z hlediska výkonu nebo paměti. Pro takové případy existuje model Arduino Due, který je výrazně výkonnější než předchozí dva modely, přesto ale takt 84 MHz, 512 kB Flash paměti a 96 kB SRAM nemusí stačit každému. Příkon tohoto modelu v aktivním režimu je okolo 0,45 W.

3.3.2 Raspberry Pi

Raspberry Pi by se dalo popsat jednoduše jako velice levný a malý počítač. Raspberry Pi přišlo na trh v roce 2012 a jeho účel byl velice podobný tomu u Arduina. Vše začalo, když si akademik Eben Upton začínal všímat, že se školy správně nevěnují informačním technologiím. Eben začal zjišťovat, že na většině škol se vyučuje především jak používat programy, základy fungování počítačů pak byly pro děti vycházející ze škol úplně neznámé. Zároveň vzdělávání tímto způsobem viděl jako důvod zvýšeného nezájmu o technické vysoké školy. Pokusil se tedy vytvořit levné zařízení, na kterém by dětem mohlo být jednoduše ukázáno, jak počítače fungují a podnítit tak jejich zájem věnovat se informačním technologiím. [18]

Charitativní organizace Raspberry Pi Foundation nabízí svoji platformu ve dvou modelových řadách Raspberry Pi a Raspberry Pi Zero. Raspberry Pi vyšlo již v několika generacích a revizích, přičemž poslední verzí v době tvorby této práce bylo Raspberry Pi 3 Model B (viz obr. 3.6). Tento model nabízí čtyřjádrový 64bitový ARM procesor Cortex-A53 schopný běžet na frekvenci 1,2 GHz. Paměťově nabídne 1 GB SDRAM a slot pro mikro SD kartu, na které běží instalovaný operační systém. Zmíněný výkon něco stojí a tak zmíněná

hardwarová výbava vytváří příkon zařízení okolo 3 W v závislosti na zátěži. Co se týče standardní konektivity, zmíněný model Raspberry Pi poskytuje Wifi, Bluetooth, RJ-45 zásuvku, čtyři USB porty, HDMI výstup a audio konektory pro reproduktory a mikrofon. Pro připojení kamery nebo displeje nalezneme CSI a DSI porty a součástky jako LED, tlačítka a jiné lze připojit pomocí poskytnutých GPIO pinů. Druhá zmíněná modelová řada Raspberry Pi Zero vychází ve dvou variantách, přičemž model s označením W (viz obr. 3.7) nabízí navíc konektivitu pomocí WiFi a Bluetooth. Model Zero je pak méně výkonnějším sourozencem plnohodnotného modelu Raspberry Pi. Nabízí tedy pouze jednojádrový procesor o taktu 1 GHz a 512 GB RAM. Spotřeba obou těchto zařízení se pak pohybuje okolo 0,9 W. Dále na desce nalezneme USB OTG, Mini HDMI a stejně jako u plnohodnotného modelu CSI port pro kameru a GPIO piny. Ztrátu konektivity a výkonosti však nahrazuje kompaktností. Model Zero a Zero W je přibližně o polovinu menší, a především je velice tenký, na výšku nemá více než několik milimetrů. [9]



Obrázek 3.6: Výkonější varianta Raspberry Pi 3 B.
Zdroj: https://www.raspberrypi.org

Obrázek 3.7: Kompaktnější varianta Raspberry Pi Zero W.
Zdroj: https://www.raspberrypi.org

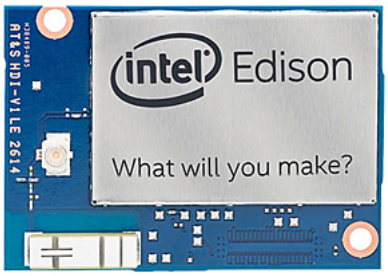
Jak bylo zmíněno, Raspberry Pi je kompaktní počítač, proto je počítáno s instalací operačního systému na SD kartu. Lze tedy použít téměř jakýkoliv operační systém podporující procesory architektury ARM. Na oficiálních stránkách je nabízen operační systém Raspbian. Raspbian je založený na Linuxové distribuci Debian a jedná se o jediný oficiálně podporovaný operační systém. Výrobcem jsou však poskytovány i odkazy na další operační systémy, jako jsou deriváty distribuce Ubuntu, Windows 10 zaměřený pro IoT zařízení nebo operační systémy sloužící jako multimediální centra. Díky těmto systémům není problém vytvářet programy v jakémkoliv jazyce, který daná distribuce nabízí, čímž celou platformu značně otevírá. [9]

Raspberry Pi je opět zařízení s obrovskou popularitou a komunitou podporující a inspirující nejen nováčky. Svoji popularitu získalo především kvůli své velikosti a výkonu, díky jimž jej lze využít pro paměťově, ale i výpočetně náročnější projekty. Svojí výbavou poskytne dostatečný výkon pro různé emulátory, webové servery nebo zpracování videa, zvuku či obrazu. Jako více uživatelským využitím může být multimediální centrum umožňující snadný přístup k multimediálním proudům na domácí síti. Za výkon se však platí, a tak se cena u standardního modelu pohybuje okolo 900 Kč. Pokud vývojáři postačuje méně výkonná varianta Zero, pohybuje se cena okolo 300 Kč. V této ceně však není zahrnuta SD karta a napájecí zdroj, a proto pro získání funkčního zařízení je nutné si ještě připlatit.

Raspberry Pi se přesto stalo velice levným počítačem, se kterým mohou nejenom děti, ale i dospělí experimentovat a prozkoumávat, jak výpočetní technika funguje.[9]

3.3.3 Intel Edison

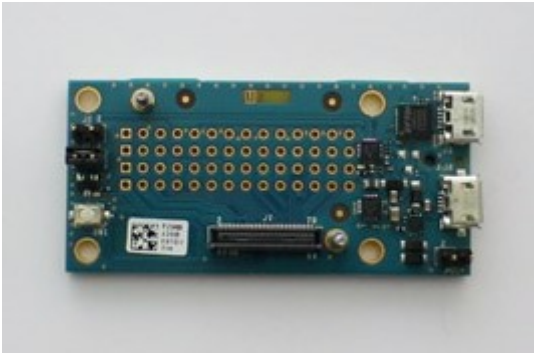
Intel Edison je velice zajímavý počin od společnosti Intel představující kompletní systém o velikosti lehce přesahující rozměry SD karty. Poprvé bylo zařízení odhaleno roku 2014 jako základ pro další projekty v oblasti nositelné elektroniky. Původně se mělo jednat o čip poháněný úsporným mikrokontrolerem řady Intel Quark, kde celé zařízení mělo mít stejnou velikost jako SD karta. [16] Později však bylo oznámeno, že bude obsahovat procesor z řady Intel Atom, a z toho důvodu bude celková velikost zařízení větší, než původně oznámili. Intel Edison byl na trhu přibližně tři roky, než Intel vydal oznámení o ukončení produkce zařízení společně s veškerým příslušenstvím. [7]



Obrázek 3.8: Výpočetní jednotka Intel Edison.
Zdroj: https://software.intel.com/

Tento miniaturní systém (viz obr. 3.8) nakonec nabídnul dvoujádrový procesor Intel Atom poskytující frekvenci 500 MHz a přidaný mikrokontroler Intel Quark s taktem 100 MHz. Pro operační systém a ukládání dat poskytuje 4 GB eMMC úložného prostoru a k tomu 1 GB LPDDR3 operační paměti. Intel Edison je poměrně úsporná platforma a tak příkon dosahuje pouze 0,6 W. Pro bezdrátovou komunikaci nabídne integrovaný WiFi a Bluetooth modul. Nakonec nabídne 40 GPIO pinů, které mohou být různě nakonfigurovány a sloužit tak jako například USB OTG, I2C sběrnice nebo rozhraní pro SD kartu. Je však nutné poznamenat, že se jedná pouze o čip nesoucí procesor, paměti a bezdrátové moduly. Pro používání je zamýšleno Intel Edison zasadit pomocí 70pinového konektoru do některé nabízené rozšiřující desky (angl. breakout board). Skrze takovou desku (viz obr. 3.9) je poskytováno čipu napájení a vstupní prostředky, jako je slot na SD kartu nebo USB konektor, v závislosti na použité desce. [5]

Oficiální operační systém pro Intel Edison je Yocto Linux obsahující základní vývojové nástroje. Žádné další operační systémy nejsou oficiálně podporovány, nicméně komunita neoficiálně nabízí například funkčně omezenou distribuci Ubuntu ve verzi 16.04 [4]. Jakožto linuxová distribuce však Yocto Linux poskytuje podporu pro několik programovacích jazyků a nástrojů, a v základu nabízí i vývojové prostředí Eclipse a Arduino IDE. Pro Mikrokontroler Intel Quark je poté přítomen operační systém reálného času. Tvorba programů pro mikrokontroler probíhá pomocí vývojového prostředí dodávaného s MCU SDK od společnosti Intel. Pomocí vývojového prostředí mohou vývojáři vytvářet programy pro mikrokontroler, operovat s volnými GPIO piny a využívat tak elektronické součástky. Na stránkách Intelu je detailně popsáno, jak pracovat nejen s mikrokontrolerem, ale i zařízením jako takovým. [5]



Obrázek 3.9: Rozšiřující deska Intel Edison mini breakout board poskytující vstupy a výstupy pro výpočetní jednotku.
Zdroj: https://software.intel.com/

Společnost Intel přišla s velice netradičním nápadem, a to vytvořit zařízení o velikosti SD karty, které má výpočetní možnosti mobilních zařízení. Pomocí takové „SD karty“ je možné pohánět nejen rozšiřující desky poskytované pro vývoj, ale teoreticky jakékoliv zařízení vytvořené se schopností využívat potenciál Intelu Edison. Bylo by tak možné, vytvořit vlastní desku pouze s USB portem, displejem nebo pouze s napájením, což dává možnost vzniku různorodých řešení nejenom nositelné elektroniky. Dokud byl produkt na trhu, pohybovala se jeho cena v přepočtu okolo 1250 Kč. [1] Nyní je však téměř vyprodaný a k dostání je především na aukčních serverech jako je Ebay. Zde se ceny pohybují od 2000 Kč za pouhý výpočetní modul až po 5000 Kč, za které dostaneme i rozšiřující desku, případně další příslušenství.[3]

Kapitola 4

Návrh systému

Před samotnou implementací obou koncových zařízení bylo nutné promyslet záležitosti, jako je výběr jazyka nebo platformy, na které poběží samotné rozpoznávání. Důležité bylo i popsat, jak bude uživatel se zařízením pracovat, k čemu jej bude používat a v jakých případech. Následující podkapitoly se tak věnují nejen těmto věcem, ale i návrhu uživatelského rozhraní pro klientskou aplikaci a nakonec je popsána komunikace mezi oběma koncovými body.

4.1 Specifikace požadavků

Pro uživatele bude vytvářené vstupní zařízení fungovat jako jednoduchý multimediální ovladač. Na svém stolním počítači či notebooku uživateli poběží aplikace na přehrávání hudby či videa a ty bude moci ovládat jednoduchými příkazy pomocí navrhovaného zařízení. Výběr dané akce následně proběhne položením prstu na vytištěné čtvrtce na papíře. Čtverce budou čtyři a budou reprezentovat tlačítka předchozí stopa, následující stopa, přehrát/pozastavit a zastavit. Komunikace mezi uživatelským počítačem a vstupním zařízením bude probíhat bezdrátově, proto bude muset uživatel využívat klientskou aplikaci zajišťující navázání spojení, komunikaci, a především simulaci stisknutých multimediálních kláves. Díky bezdrátové technologii uživatel nebude nucen pracovat se zařízením přímo u svého počítače, ale bude jej moci využít i ve větší vzdálenosti. Využití bezdrátové komunikace však vytváří požadavek mít používaný počítač i vstupní zařízení připojené k jedné síti.

Jak bylo však již zmíněno, celá tato práce je postavena především na tom vyzkoušet si tvorbu takového zařízení a prozkoumat možnosti zpracování obrazu. Z toho důvodu jako hlavní požadavek na celý tento projekt bylo nejen vytvoření funkčního modelu, ale hlavně získání nových poznatků a zkušeností nejen z okruhu zpracování obrazu, ale i programování samotného. Je tedy důležité poznamenat, že projekt neměl za cíl vytvořit nové zařízení schopné nahradit současné řešení na trhu jako jsou multimediální klávesnice a dálková ovládání. Z toho důvodu byl celý projekt vytvořen jako prototyp, a to především po vzhledové stánce. Vzhled má tak převážně funkční charakter a na design celého zařízení nebyl brán velký zřetel.

4.2 Výběr platformy pro vstupní zařízení

První problém, který vyvstal, byl výběr platformy, na které se budou odehrávat veškeré výpočty a procesy spojené se získáváním dat z kamery, jejich zpracování a poskytování vý-

sledků klientské aplikaci. Vybraná platforma musela být dostatečné výkonná pro zpracování obrazu, mít možnost připojit periferie jako je kamera a být rozumně dostupná i s ohledem na pořizovací cenu. Mezi zvažované platformy proto bylo zařazeno Arduino, Raspberry Pi a Intel Edison. Detailnější popis každého z těchto zařízení byl proveden v kapitole 3.3.1, 3.3.2 a 3.3.3.

Prvotním nápadem bylo využít Arduino, především kvůli jeho dostupnosti, ceně, spotřebě energie a rozsáhlé komunitě. Další plus bylo vlastnictví jednoho z modelů a s ním spojené i určité zkušenosti. Po delší úvaze však bylo dospěno k názoru, že se nejedná o nejvhodnější platformu pro tento projekt. Hlavním úskalím na této platformě při jejím využití by byla paměť pro uchovávání snímků. I když se sleví z požadavků na výkon a nebude docházet ke zpracování obrazu v reálném čase, je nutné uchovávat jeden či více snímků v paměti. Nekomprimovaný snímek o rozlišení 640x480 zabírá okolo 300 kB. Arduino Due nabízí pak nejvíce paměti SRAM a to 96 kB, což samozřejmě nestačí. Tento nedostatek by musel být řešen paměťovými moduly, a to nebylo považováno za optimální řešení.

Další možností byl Intel Edison, avšak toto zařízení bylo zavrhnuto hlavně kvůli ceně. Ta byla sice v době prodeje na trhu podobná jako u Raspberry Pi, avšak nyní, když není k sehnání běžně v obchodech, se cena vyšplhala na více jak dvojnásobek. Za tak vysokou cenu však nenabízí Intel Edison vyšší výkon, více paměti nebo jiné možnosti, které by poskytnuly výhodu při vývoji či provozu.

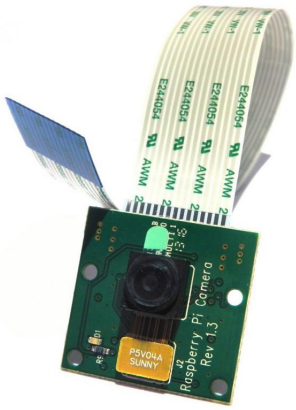
Ze zmíněných důvodů bylo rozhodnuto v konečném důsledku využit platformu Raspberry Pi. Ta nabízí vysoký výkon a dostatek paměti za rozumnou cenu. Jediným úskalím této platformy je spotřeba elektrické energie vůči ostatním zmiňovaným platformám, avšak pro běh aplikace v reálném čase byl potřeba dostatečný výkon, a tak byla tato vyšší spotřeba zanedbána. V důsledku tedy bylo zakoupeno zařízení Raspberry Pi 3 model B, poskytující společně s 32 GB SD kartou dostatek paměti a výkonu pro vývoj i běh vytvářeného projektu. Porovnání všech tří platforem z hlediska paměti a výkonu a příkonu je pak znázorněno v tabulce 4.1

Tabulka 4.1: Tabulka porovnávající paměť a výpočetní jednotky jednotlivých zvažovaných vestavěných systémů.

	Arduino Due	**Raspberry Pi 3 model B**	**Intel Edison**
CPU/MCU	MCU 32bit 84 MHz	CPU 4 jádra 64bit 1,2 GHz	CPU 2 jádra 64bit 500 MHz, MCU 32bit 100 MHz
Paměť	Flash: 512 kB, SRAM: 96 kB	RAM: 1 GB	RAM: 4GB
Příkon	0,45 W (Aktivní režim)	3 W	0,6 W

Po zvolení vhodné platformy přišel na řadu výběr kamery. Zvolena byla oficiální kamera pro Raspberry Pi (viz obr. 4.1), která je připojována pomocí CSI portu. Zmíněná kamera zvítězila nad běžnou webkamerou připojenou pomocí USB portu především kvůli kompatibilitě. Bylo zřejmé, že oficiální kamera bude fungovat s větší pravděpodobností než generická webkamera, a zároveň bude mít větší podporu. Předem byla ověřena existence

API pro jazyk C/C++ pro obsluhu CSI kamery. Oficiální kamera je dostupná ve dvou verzích, u každé lze navíc vybrat mezi modelem snímajícím viditelné světlo nebo infračervené záření. Mezi verzemi však není podstatný rozdíl, alespoň co se týče pořizování videa. Oba senzory zvládají stejná rozlišení při stejných snímcích za sekundu. Hlavní rozdíl nastává jen ve výrobci senzoru a v rozlišení zachytávaných statických snímků. [9] Z důvodu počítaného využití pro pořizování videosekvencí nebylo na rozdíly pohlíženo a byla zakoupena levnější první verze.



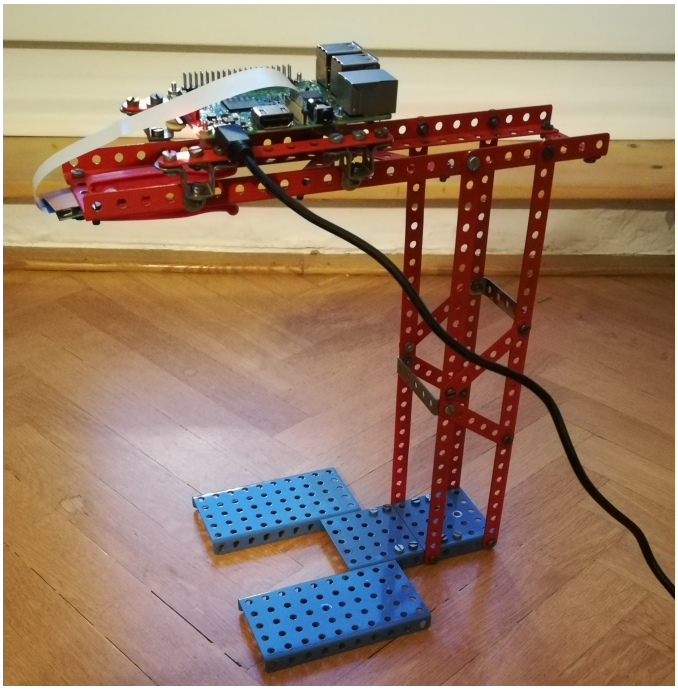
Obrázek 4.1: Oficiální Raspberry Pi kamera v první verzi.
Zdroj: https://uk.pi-supply.com

4.3 Konstrukce a uchycení vestavného zařízení

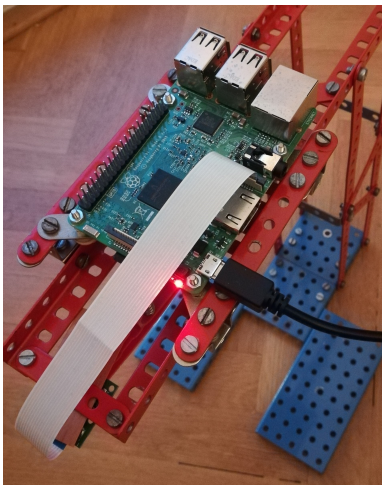
Následujícím řešeným úkolem bylo vytvoření konstrukce pro uchycení Raspberry Pi a kamery. První velice jednoduchá konstrukce byla vytvořena za pomoci kartónové krabice, knihy a kolíčku. Kamera držela pomocí kolíčku na knize, která ležela na krabici a tvořila jakési rameno. Popsané řešení bylo užíváno v začáteční fázi, kdy byly především prozkoumávány možnosti a prováděny různé experimenty. Po dosažení určité fáze postupu však bylo zapotřebí vytvořit stabilnější uchycení, jelikož nastávaly nechtěné a někdy i nezaregistrované pády nebo posuny kamery. Díky těmto posunům pak program poskytoval nesmyslé údaje, které znepříjemňovaly celý vývoj. Jeden z nápadů byla konstrukce ze dřeva. Přesto byla nakonec vybrána česká stavebnice Merkur a celé rameno držící kameru a Raspberry Pi bylo postaveno s její pomocí. Konstrukce sestavená pomocí stavebnice Merkur se ukázala jako dostatečná pro vývoj a její podobu lze vidět na obrázcích 4.2 a 4.3. Je však nadmíru jasné, že pokud by se měl projekt dále rozšířit mimo oblast prototypování, bylo by zapotřebí využít zmíněného dřeva nebo vytisknout díly pomocí 3D tiskárny.

4.4 Strana klientské aplikace

Jako framework pro tvorbu grafického uživatelského rozhraní pro klientskou stranu byl zvolen Qt. Prvním důvodem volby zmíněného frameworku je existence velice rozsáhlé a kvalitně zpracované referenční dokumentace. Na webových stránkách s dokumentací lze velice snadno dohledat všechny třídy obsažené v knihovnách Qt a zjistit tak informace o metodách a atributech. Druhým důvodem pak byla teoretická přenositelnost. Pokud by se



Obrázek 4.2: Celková konstrukce zařízení vytvořená pomocí stavebnice Merkur.
Zdroj: Foto



Obrázek 4.3: Uchycení Raspberry Pi na horní části samotné konstrukce.
Zdroj: Foto

nevyužily systémové knihovny a používaly se pouze standardní knihovny a knihovny jazyka Qt, bylo by možné vytvořit aplikaci běžící nejen pod systémy Windows, ale i systémy Linux a MacOS. Posledním z důvodů byla již částečná zkušenost s vývojem pomocí frameworku Qt. Ze zmíněných třech důvodů byl tedy vybrán framework Qt společně s jazykem C++. Jako další byl zvažován jazyk C# společně s WPF a jazyk Java s JavaFX. Každé z těchto dvou kombinací však chyběl alespoň jeden z důvodů, kvůli kterým došlo ke zvolení Qt a jazyka C++. [8]

Samotná klientská aplikace by pak měla být velice jednoduchá a uživateli nabízet jen několik málo možností. Aplikace by měla sloužit pouze pro navázání spojení, přijímání zpráv od vstupního zařízení a simulaci stisku kláves. Uživatel tak bude s aplikací pracovat tím způsobem, že se pouze připojí na vstupní zařízení a o vše další se postará samotná aplikace. Pro navázání spojení je však nutné znát IP adresu zařízení. Pokud ji uživatel zná, bude mít možnost ji zadat do textového pole. Nebude-li ji znát, využije možnosti automatického vyhledání zařízení na lokální síti. Nalezené zařízení následně automaticky vyplní ono textové pole svojí IP adresou. Poté co uživatel jedním ze způsobů vyplní adresu, bude mít možnost se na zařízení připojit a po úvodní kalibraci jej ihned používat. Pro ukončení spojení pak bude sloužit další tlačítko dostupné pouze při navázaném spojení. Celé uživatelské rozhraní tak bude obsahovat tlačítko pro vyhledání vstupního zařízení, místo pro zadání adresy a tlačítko pro připojení a odpojení. Možností je i přidání prostoru pro výpis informačních a diagnostických dat.

4.5 Strana vstupního zařízení

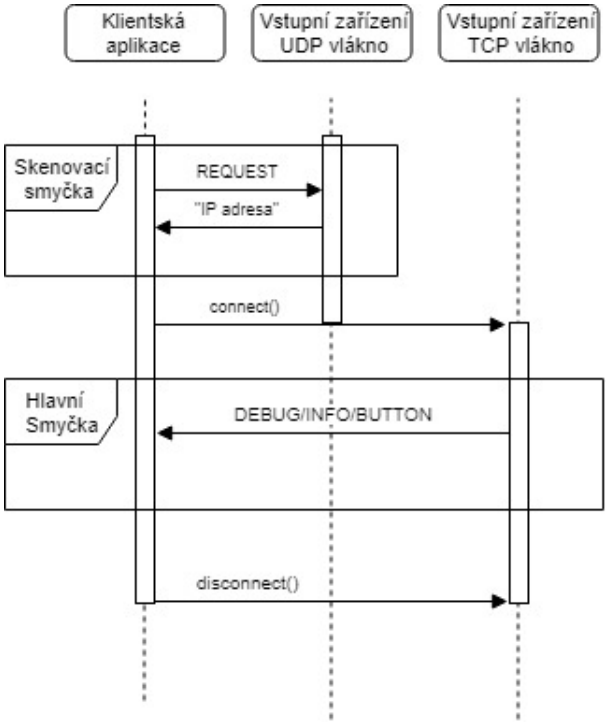
Pro vývoj aplikace protistrany běžící na Raspberry Pi byl zvolen opět jazyk C++, nyní však bez zde nepotřebného grafického rozhraní, jelikož veškeré ovládání vstupního zařízení bude probíhat skrze rozhraní poskytnuté klientskou aplikací. Jazyk C++ byl vybrán především kvůli existenci C++ API pro práci s kamerou a kvůli jeho rychlosti při zpracování obrazu oproti například Javě. Přestože byla aplikace psána v jazyce C++, nebylo počítáno s přenositelností, především kvůli specializaci celého systému.

Význam a funkčnost ze strany vstupního zařízení bude veskrze jednoduchá, popisujeme-li jen vnější chování a interakci s uživatelem. Uživatel pouze aplikaci na Raspberry Pi spustí a dále bude pracovat pomocí klientské aplikace. Možností je i zavedení programu do pospuštění, aby se uživatel nemusel starat o nic víc než jen o zapínání celého zařízení a čekání na start celého systému včetně poskytované aplikace. Po zapnutí bude program vyčkávat na spojení od klientské strany a zároveň poslouchat dotazy ohledně své IP adresy. Dostane-li takový požadavek, odešle svoji IP adresu fyzického rozhraní, a opět bude vyčkávat na další dotaz či spojení. Po přímém navázání spojení dojde k prvotní kalibraci, kde bude rozpoznána poloha čtverců. Následně bude spuštěna smyčka kontrolující pohyb a vyhodnocující, zda byl některý ze čtverců vybrán. V případě, že bude vyhodnocen jeden ze čtverců jako vybraný, bude odeslána zpráva přes síť oznamující zachycení společně s číselnou identifikací čtverce. Tato informace bude nadále zpracovávána v klientské aplikaci.

4.6 Návrh komunikace

Komunikace mezi oběma koncovými body bude velice jednoduchá, jelikož zde není zapotřebí přenosu velkých objemů dat, ani nebude potřeba data nikterak strukturovat. Posílané zprávy budou obsahovat vždy jedinou informaci. Zároveň nebude zapotřebí oboustranné výměny dat, jelikož v plánu je pouze informování klientské aplikace o zvoleném čtverci. Z toho důvodu jedna strana zprávy bude pouze odesílat a druhá zprávy přijímat. Jedinou výjimkou představuje vyhledávání zařízení na vnitřní síti. Komunikace se tedy rozdělí na dva proudy. Prvním by byla komunikace sloužící k vyhledávání zařízení a druhým jednoduché předávání informací o rozpoznávání za běhu.

Při vyhledávání se na síť vyšle požadavek znějící jednoduše „REQUEST“. Existuje-li vstupní zařízení připojené do stejné sítě se zapnutou aplikací, pak tento požadavek roz-



Obrázek 4.4: Komunikační diagram znázorňující návrh komunikace mezi klientskou aplikací a vlákny běžícími ve vstupním zařízení.

pozná a vyšle zprávu určenou pro klientskou aplikaci obsahující platnou IP adresu fyzického rozhraní. Jelikož v okamžiku vyhledávání vstupního zařízení nebude známa IP adresa protistrany ani jednomu zařízení, dojde k nespojované komunikaci přes broadcast pomocí protokolu UDP.

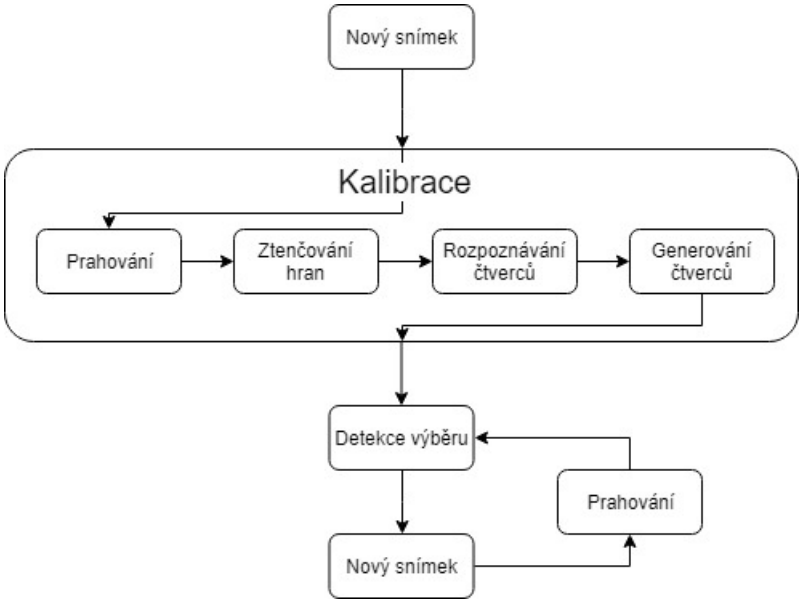
Po získání IP adresy může dojít k navázání spojení a zahájení spojované komunikace. Samotná komunikace a spojení bude zajištěna pomocí spolehlivého protokolu TCP garantujícího příchod packetů. Po navázání spojení pomocí poskytnuté IP adresy bude docházet již jen výhradně k jednostranné komunikaci, kde budou zprávy posílány pouze ze vstupního zařízení. Vstupní zařízení pak odesílá jeden ze tří druhů zpráv. Zprávy začínající řetězcem „INFO“ budou mít pouze informační charakter obsahující například informace o průběhu kalibrace, nebo zda je zařízení připraveno pracovat. Další typ zamýšlených zpráv začíná slovy „BUTTON“. Tento typ zprávy informuje klientskou aplikaci o detekci výběru jednoho ze čtverců. Identifikace čtverce bude zapsána ve zprávě ihned za označením „BUTTON“ pomocí číselného ID začínajícího od nuly. Posledním typem zprávy bude „DEBUG“ sloužící převážně pro diagnostické účely při vývoji a ve finální verzi nebudou tyto zprávy posílány. Výsledný diagram znázorňující komunikaci mezi jednotlivými vlákny je vidět na obrázku 4.4.

Kapitola 5

Implementace

V následujících podkapitolách je řešena především implementace vstupního zařízení, které je hlavním jádrem celé této práce. Implementace byla rozdělena na dvě hlavní části a to kalibraci a následné rozpoznávání čtverců. Postup celým systémem rozpoznávání lze vidět na diagramu 5.1. Kýženého výsledku bylo dosaženo implementací algoritmů vlastními silami bez použití externích knihoven, jako je například OpenCV. Jedinou použitou externí knihovnou se stala knihovna RaspiCam pro získávání obrazových dat z kamery. Pomocí této knihovny jsou získávány snímky v odstínech šedi s rozlišením 640x480.

Strana klientské aplikace byla záměrně vynechána, jelikož se stará především o komunikaci a simulaci stisku kláves a nenastává zde žádné zpracování obrazu.



Obrázek 5.1: Diagram systému.
Na diagramu je znázorněn prostup systémem a transformace získaných snímků.

5.1 Použité nástroje pro vývoj

Před samotným popisem implementace jsou níže zmíněny hlavní nástroje, kterých bylo využito k dosažení cíle.

Git

Git je velice užitečný nástroj umožňující přehlednou a jednoduchou správu kódu při týmovém vývoji. Umožňuje uchovávat historii kódu ve formě tzv. commitů reprezentujících provedené změny v kódu. Všechny zdrojové kódy jsou většinou uloženy na místě mimo počítače vývojářů a zajišťují určitý stupeň zálohy.

Nástroj git byl využit především pro zálohu kódu v případě selhání počítače či Raspberry Pi. K tomu účelu bylo využito webové služby BitBucket umožnující vytváření privátních repozitářů. Zmíněná služba byla vybrána z důvodu pozitivních zkušeností při používání v minulosti.

Qt Creator

Qt Creator je vývojové prostředí umožňující jednoduchý vývoj aplikací s pomocí frameworku Qt. Obsahuje jak editor kódu, tak designér umožňující rychlé a jednoduché vytváření uživatelského rozhraní pomocí operace „Drag and Drop“. Pomocí zmíněného editoru lze vytvářet logiku schovanou za komponenty vytvořenými pomocí designéru. Toto vývojové prostředí bylo využito k tvorbě klientské aplikace. K implemetaci bylo využito nástroje Qt Creator ve verzi 4.4.1 společně s Qt knihovnami ve verzi 5.9.2.

Vim

Vim je vylepšená verze textového editoru Vi (Vim - Vi Improved). Tento editor byl používán pro psaní zdrojových kódů na platformu Raspberry Pi. Práce probíhala nejenom v editoru Vim, ale i gVim což je jeho grafická verze. Pro usnadnění psaní bylo využito pluginu YouCompleteMe ulehčujícího práci pomocí napovídání a doplňování při psaní.

Vývojové nástroje

Zdrojový kód byl překládán pomocí nástrojů make a cmake. Nástroj make slouží pro pohodlný překlad programu bez nutnosti vypisovat veškeré parametry při překladu. K tomu využívá souboru Makefile obsahujícího informace, jak daný program přeložit. Nástrojem cmake lze tento soubor vygenerovat takovým způsobem, aby při překladu poskytoval programátorovi informace o probíhajícím překladu a zároveň optimalizoval celý proces překladu. Optimalizace probíhá především způsobem, kdy jsou pro linkování vytvářeny objektové soubory pouze změněného kódu. Kód byl nakonec kompilován pomocí g++, jenž je součástí kolekce překladačů GCC (GNU Compiler Collection). GCC poskytuje kompilátory a knihovny pro překlad několika jazyků, mezi které patří především jazyky C a C++.

VNC Viewer

VNC Viewer je program umožňující využívaní služeb vzdálené plochy. Za pomocí tohoto programu se lze jednoduše připojit na vzdálenou stanici a ovládat ji bez potřeby být přímo u ní. Ovládat počítač tímto způsobem lze z vnitřní sítě, případně i přes internet po provedení nastavení směrování v routeru. Pro správnou funkčnost musí protistrana obsahovat funkční a běžící VNC Server zajišťující výměnu dat mezi koncovými body.

Zmíněný program byl používán pro připojení k Raspberry Pi, který neměl připojen monitor, klávesnici ani myš. Program velice pomohl nejen při celkové správě zařízení, ale především při počáteční fázi vývoje, kdy docházelo ke kontrole stavu pořízených snímků z kamery. VNC Viewer byl využit ve verzi 6.17

Knihovna RaspiCam

Tato knihovna poskytuje API pro ovládání připojené kamery k Raspberry Pi pomocí CSI portu. Oficiální knihovna vznikla pouze pro jazyk Python, pro jazyk C++ pak vznikla tato knihovna od třetí strany. Knihovna poskytuje možnosti, jak získávat snímky z kamery a zároveň nastavovat parametry jako barevný prostor výstupního snímku, rozlišení a mnoho dalšího. Knihovna je dostupná ve dvou verzích, přičemž jedna verze využívá knihovny OpenCV pro počítačové vidění a druhá pracuje bez jejího použití. Pro projekt byla využita knihovna ve verzi 1.6 nevyužívající OpenCV. Knihovna je distribuována pod běžnou BSD licencí a díky tomu ji lze využít bez jakýchkoliv větších omezení. [24]

5.2 Komunikace

Komunikace mezi klientskou aplikací a vstupním zařízením probíhá tak, jak bylo popsáno v návrhu v předchozí kapitole. Na vstupním zařízení běží v samostatném vlákně smyčka, která čeká a poslouchá broadcastové zprávy. Pokud je určená jako dotaz pro zařízení, odešle zpět zprávu se svojí adresou. Mezitím v hlavním vlákně program čeká na navázání přímého TCP spojení, po němž ukončí běžící vlákno čekající na broadcastové zprávy. Většinu těchto operací zajišťuje třída `UDP` implementující jednoduchý server reagující na broadcast a třída `TCP` schopná navázat komunikaci a odesílat a přijímat zprávy, přičemž druhá možnost není při implementaci využita.

5.3 Kalibrace

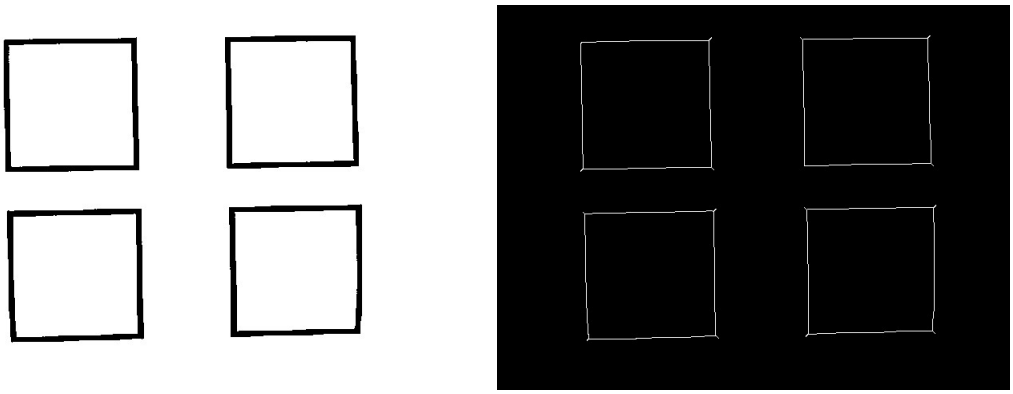
Před samotnou detekcí výběru čtverce je nutno provést kalibraci. Ta slouží k rozpoznání čtverců a poskytnutí jejich polohy dalším algoritmům. Postup kalibrace sestává z několika po sobě následujících kroků, kde výstup jednoho kroku je vstupem pro další. Prvním krokem je získání binárního snímku získaného pomocí prahování, následujícího ztenčováním hran. Dalším krokem je aplikování Houghova algoritmu pro detekci. Výstup algoritmu je následně zpracován, a nakonec jsou získány polohy čtverců v obraze.

5.3.1 Ztenčování hran

Vstupem pro většinu ztenčujících algoritmů je binární obraz po detekci hran. Jednou z možností detekce hran je například konvoluce vstupního obrazu se Sobelovým operátorem. Po této operaci pak dostaneme binární obraz, ve kterém bílá místa reprezentují hrany. V našem případě to ale není nutné. Náš obraz se skládá pouze ze čtverců, tj. hran, a proto nám postačí vytvořit binární obraz prahováním a výsledek invertovat tak, aby odpovídal výstupu po detekci hran.

Výše popsaný algoritmus v sekci 3.2.1 byl implementován v třídě `EdgeThinning`, kde jádrem třídy je metoda `zhangSuen`. Pro správné fungování třídy je nutno poskytnout konstruktoru ukazatel na pole pixelů a jeho rozměry. Po zavolání jmenované metody dojde v programu k zahájení algoritmu. Uvnitř jsou pak volány privátní metody `zhangSuen1` a `zhangSuen2` reprezentující dvě fáze algoritmu. Každá z metod vrací odpověď, zda byly splněny podmínky definované algoritmem. V hlavní metodě dojde k případnému uložení pixelu do vektoru pro následné smazání. Dojde-li k vymazání nějakého z pixelů, je tato skutečnost zaznamenána a hlavní metoda je po projití obrazu rekurzivně volána znova.

V případě této aplikace je technika ztenčování využita pro vytvoření jednoduché reprezentace čtverců vhodné pro další zpracování. Houghův algoritmus, který je následně využíván, by při využití neztenčeného obrazu díky svému principu detekoval mnohonásobně více čar, než by se ve skutečnosti nacházelo v obraze. Výsledek algoritmu lze vidět na obrázku 5.2. Konkrétní algoritmus byl vybrán především kvůli jednoduché implementaci, prezentované rychlosti, nízké paměťové náročnosti a v neposlední řadě jeho popularitě.

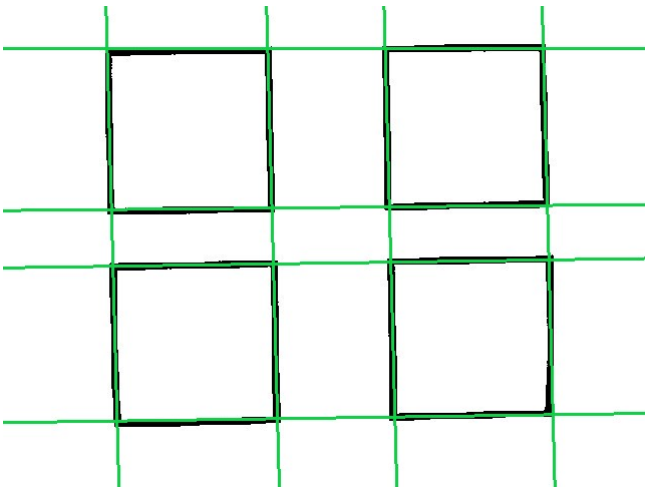


(a) Binární obraz před ztenčováním (b) Binární obraz po ztenčování

Obrázek 5.2: Výsledek ztenčování provedený nad samotnou klávesnicí.

5.3.2 Rozpoznávání čtverců

Hledání samotných čtverců v obraze probíhá ve dvou krocích. Prvním krokem je nalezení veškerých možných přímek v obraze. Tyto přímky představují hranice čtverců na papíře, z jejichž pomocí jsou dopočítány jejich pozice v obraze. Tento postup však vytváří jisté omezení na způsob, jak může vypadat papírová klávesnice. Čtverce na papíře tak nesmí být rozmístěny náhodně, nýbrž musí dodržovat stanovené pravidlo. Pro správné fungování musí platit, že sousední čtverce spolu musí vytvářet dvě pomyslné přímky spojující buď horní a spodní hrany, nebo levé a pravé hrany čtverců (viz obr. 5.3).



Obrázek 5.3: Linie které musí tvořit čtverce pro jejich správnou detekci.

Rozpoznávání přímek

Po získání ztenčeného obrazu přichází na řadu detekce přímek v obraze. Detekce přímek byla implementvána za pomoci Houghovy transformace popsané v sekci 3.2.2. Ta byla implementována ve třídě `HoughTransform`, pomocí metod `transfrom` a `findLines`. První ze zmíněných metod je privátní, stará se o převod snímku do Houghova prostoru a je volána z druhé metody `findLines`. Tato metoda má pouze jeden parametr a tím je velikost Moorova okolí při hledání, zda bod přesahující prahovou hodnotu je zároveň lokálním maximem. Prahovou hodnotu lze pak nastavovat pomocí veřejné metody `setThreshold`.

Na konci algoritmu je využito nutnosti vypočítat koncové body rozdílně pro vertikální a horizontální přímky. Přímky jsou taktko rozděleny do dvou vektorů, což usnadnujě další práci a odstraňuje nutnost vytvářet algoritmus rozdělující přímky na horizontální a vertikální. Tohoto rozdělení je zapotřebí v následujícím kroku při vytváření čtverců. Jak je v následující části popsáno, při tvorbě čtverců je využito dvou vertikálních a dvou horizontálních přímek ohraničujících čtverec.

Tvorba čtverců

Jak již bylo zmíněno, po dokončení vyhledávání přímek jsou k dispozici dva vektory, jeden nesoucí přímky vodorovné a druhý s přímkami vertikálními. Pro další postup je však zapotřebí vektory seřadit a „pročistit“. Seřazení je poměrně jednoduchá záležitost a k jejímu dosažení je využito standardní funkce `sort()`. Po seřazení následuje pročištění, které zajistí, že ve vektoru nebudou dvě nebo více přímek několik pixelů vedle sebe. K tomuto jevu při vývoji několikrát došlo a následně vytvořené čtverce neodpovídaly reálným čtvercům na papíře. Pročištění probíhá hledáním dvou sousedních přímek, jež jsou od sebe vzdálené méně jak deset pixelů. Je těžké určit správnou přímku reprezentující hranu čtverce, z toho důvodu je vytvořena nová přímka reprezentující průměr oněch dvou blízkých přímek.

Posledním procesem nejen při detekci čtverců, ale i během celé kalibrace, je tvorba objektů reprezentujících reálné čtverce. Tvorba objektů pak probíhá iterací přes vektory přímek, kdy v každé iteraci jsou konstruktoru poskytnuty jeho čtyři hraniční přímky pro tvorbu čtverce. Během konstrukce se iteruje původním snímkem všech čtverců v mezích vytvořených hraničními přímkami a ukazatelé na všechny takto navštívené pixely jsou uloženy do atributu objektu. Je nutno poznamenat, že při tvorbě objektu nejsou poskytnuty celé přímky, avšak pouze počáteční body. Tento postup vytváří určitou nepřesnost pozice každého čtverce v závislosti na vzdálenosti od počátku přímky. Tyto nepřesnosti jsou zanedbatelné při správném položení papíru, případně mírném natočení. Dojde-li však ke většímu pootočení papíru se čtverci, může nastat silná chybovost při vytváření objektů se čtverci.

5.4 Detekce vybrání čtverce

V tomto momentě jsou dostupné veškeré potřebné prostředky pro samotnou detekci vybraného čtverce. Detekce však nenastává nad každým snímkem, ale pouze za splnění určitých podmínek. Ne snad proto, že by nebylo dostatek výkonu na zařízení, nýbrž proto, aby bylo správně detekováno, kdy uživatel čtverec opravdu vybral a kdy přes něj pouze pohnul rukou, aby vybral sousední čtverec. Zmíněného požadavku pro správnou funkčnost detektoru je dosaženo velice jednoduše pomocí detekce pohybu. Detekce pohybu byla implementována opět nad binárním obrazem po prahování. Při detekci pohybu jde o prosté porovnávání současného a předchozího snímku, přičemž je zjišťováno, kolik pixelů změnilo svoji hodnotu.

Pokud je počet změněných pixelů nad určitou hranicí, je přechod mezi snímky vyhodnocen jako obsahující pohyb. Dochází-li tedy k pohybu, zařízení nevyhodnocuje žádný ze čtverců. Není-li v prostoru nad klávesnicí detekován žádný pohyb, ani tehdy nedochází k vyhodnocování. K vyhodnocení dojde pouze v případě přechodu ze stavu, kdy je detekován pohyb, na stav, kdy není detekován pohyb. Tímto přechodem je reprezentováno dokončení pohybu, a tudíž teoretické vybrání jednoho ze čtverců. Celý tento proces je navíce obohacen o klidový stav (z angl. cooldown). V klidovém stavu se zařízení nachází několik snímků po reálně detekovaném výběru čtverce. Zmíněný stav byl přidán, především aby se zamezilo nechtěné vícenásobné detekci stejného čtverce.

(a) (b) (c) (d)

Obrázek 5.4: Snímky jendotlivých čtverců při detekovaném výběru.

Při zjišťování, který ze čtverců byl vybrán, aplikace postupuje způsobem, kdy postupně projde všechny čtverce a spočítá, kolik čtverec obsahuje černých pixelů reprezentujících překážku mezi klávesnicí a kamerou. Reprezentace čtverců je znázorněna na obrázku 5.4 společně se čtvercem, který byl uživatelem vybrán (viz obr. 5.4c). Ze všech čtverců je následně vybrán čtverec obsahující největší počet černých pixelů. Pokud je zároveň počet nad stanoveným prahem, je čtverec vyhodnocen jako vybraný a jeho identifikace je odeslána klientské aplikaci. Během procesu výběru je ovšem nutné počítat s tím, že při výběru čtverců v horní řadě je více než pravděpodobné částečné či úplné zakrytí čtverce ve spodní řadě. Z toho důvodu mají horní čtverce prioritu ve výběru před spodními. Tato priorita je nastavována během tvorby objektů čtverců, kde každá další řada má vyšší prioritu.

Kapitola 6

Testování

Testování probíhalo především během vývoje. Po přidání každé nové funkčnosti bylo testováno nejen, zda se daná funkce chová korektně, ale i zda si celý program zachoval funkčnost, jakou měl před přidáním nové funkce. Testování klientské aplikace se zaměřilo především na správnost komunikace a její zpracování. První testování se zabývalo správným příjmem zpráv a následná simulace kláves byla testována pomocí poznámkového bloku a simulace stisku písmen. Testování samotných multimediálních funkcí probíhalo za pomocí programu KMPlayer, ve kterém se přehrávala hudba a pomocí vstupního zařízení docházelo k pokusům ovládat přehrávání hudby.

Testování vstupního zařízení probíhalo postupně. První částí byla detekce pohybu, poté následovalo rozpoznávání čtverců. Rozpoznávání bylo prováděno nejprve nad jedním čtvercem, poté nad dvěma a nakonec nad čtyřmi poskládanými ve dvou řadách. Zde probíhalo testování nejen správného rozpoznání, ale i přiřazení priority a identifikace. Při detekci výběru byla ověřena správná práce s prioritami a zároveň výběr správného momentu, kdy provést detekci. Během všech těchto testů byly zároveň prováděny experimenty s celým systémem s cílem nalézt optimální hodnoty všech mezních hodnot pro práci s obrazem.

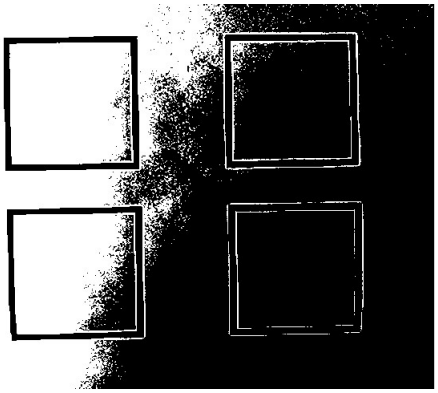
Problém se světlem

Největším odhaleným problémem byl nedostatek nebo naopak nadbytek světla. Při nedostatku světla se na binárním obraze vyskytoval šum v podobě „pobíhajících“ pixelů. Nekorektní výstup celé aplikace při nedostatku světla je vidět na obrázcích 6.1a až 6.1c. Naopak při nadbytku světla docházelo k odrazu od prstu, především nehtu, a na binárním obraze nebyl prst přítomen vůbec nebo jen částečně. Jedním z řešení bylo ovládání prahu pro vytváření binárního obrazu pomocí klientské aplikace. Po delší uváze však bylo dospěno k názoru, že by na uživateli měla aplikace záviset co nejméně. Z toho důvodu došlo k výměně klasického prahování za adaptivní prahování schopné poradit si i se stíny a nerovnoměrností světla dopadajícího na klávesnici.

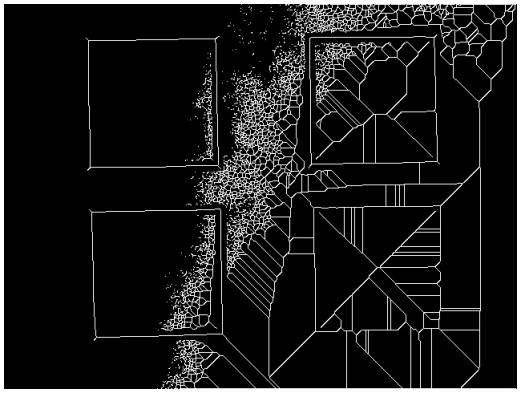
Funkčnost v reálném čase

Důležitým parametrem, který byl po celou dobu vývoje sledován, byla schopnost systému reagovat v reálném čase. Aby mohl být systém použitelný, bylo zapotřebí, aby reagoval na výběr čtverce okamžitě, případně s minimálním prodlením. Veškeré diagnostické výstupy, jako informace o prováděném úkonu nebo ukládání snímků na disk, samozřejmě celý proces zpomalují, proto byly omezeny na minimum. K testování byly využity kratší i delší časové

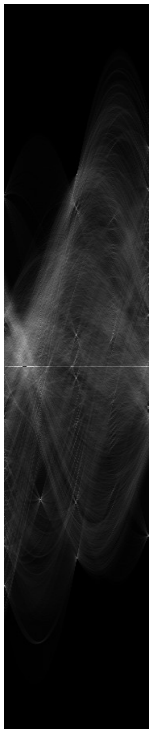
okamžiky, systém delší dobu běžel a stále bylo zkoušeno, zda reaguje včas. Během testování však nebyly odhaleny žádné problémy.



(a) Binární obraz obsahující šum po prahování.



(b) Výsledek ztenčování po použití obrazu se šumem.



(c) Výsledný akumulátor při hledání přímek po použití předchozího obrazu.

Obrázek 6.1: Jednotlivé výstupy mezikroků při kalibraci za nedostatku světla.

Kapitola 7

Závěr

Cílem této práce bylo vytvořit vstupní zařízení na bázi počítačového vidění, a tím se zároveň seznámit s principy zpracování obrazu. K dosažení tohoto cíle bylo nutné seznámit se s již existujícími řešeními a nastudovat nutné podklady pro vytvoření zmíněného zařízení. Analýza existujících řešení byla zpracována s ohledem na jejich principy, přednosti a slabiny. Samotné studium základních znalostí pro vyhotovení této práce se zaměřilo na znalosti nejen ohledně zpracování obrazu, ale i obrazu jako takového a jeho reprezentaci v počítači.

Při návrhu celého zařízení byl projekt rozdělen do dvou celků. Prvním z celků, kterým se tato práce věnovala, byla klientská aplikace. Zde bylo při návrhu především dbáno na jednoduchost a snadné použití. Druhým celkem byla strana vstupního zařízení. Zde se hlavním tématem stal výběr vhodné platformy a vývoj konstrukce vstupního zařízení. Během výběru platformy bylo prozkoumáno několik existujících platforem. Byly mezi sebou porovnány, a jako nejvhodnější byla vybrána platforma Raspberry Pi. Po samotné implementaci bylo vyhodnoceno, že tato platforma je více než dostatečná pro běh systému v reálném čase. Zajímavou možností by však bylo vyměnit současné Raspberry Pi 3 model B za méně výkonější, ale mnohem úspornější Raspberry Pi Zero W. Pokud by výkon modelu Zero W dostačoval, mohlo by dojít k výraznému ušetření spotřeby elektrické energie.

Pro implementaci obou koncových bodů bylo využito několik nástrojů a knihoven. Pro implementaci uživatelského rozhraní klientské aplikace posloužil framework Qt ve verzi 5, nicméně pro simulaci kláves bylo využito několika knihoven systému Windows, a proto byla ztracena přenositelnost. Strana vestavného zařízení využila knihovny třetí strany RaspiCam, která zajistila získávání obrazových dat a kontrolu kamery. Obě strany pak byly vytvořeny pomocí jazyka C++ ve standardu z roku 2011.

Výsledný systém je funkční, nicméně vytváří jistá omezení pro jeho používání. Jedním z hlavních omezení je vzhled samotné papírové klávesnice se čtverci. Tímto omezením je nutnost mít čtverce na papíře v předem daném seskupení. Další vývoj systému by se tedy mohl zabývat odstraněním těchto omezení, jakožto i přidáním dalších funkčností.

Během implementace, ale i návrhu celeho systému, byly vytvořeny dvě následující možné rozšiřující funkce. První možností by bylo přidání vlastního osvětlení k vstupnímu zařízení. Pomocí tohoto rozšíření by uživatel následně mohl využívat zařízení i za špatných světelných podmínek. Druhou možností je umožnit uživateli definovat funkce jednotlivých kláves pomocí konfiguračních souborů, případně mít možnost jednotlivé listy se čtverci svázat s těmito soubory, a měnit tak funkčnost kláves změnou listu. Díky této funkčnosti by pak aplikace mohla sloužit více účelům, nejenom jako multimediální ovladač.

Literatura

[1] Adafruit Industries, Unique & fun DIY electronics and kits. [Online; navštíveno 27.04.2018].
URL https://www.adafruit.com/

[2] Arduino. [Online; navštíveno 25.04.2018].
URL https://arduino.cc

[3] Electronics, Cars, Fashion, Collectibles, Coupons and More | eBay. [Online; navštíveno 25.04.2018].
URL https://arduino.cc

[4] Forum: Support Community |Intel Communities. [Online; navštíveno 27.04.2018].
URL https://communities.intel.com/community/tech

[5] Maker & Innovator Products | IoT | Intel® Software. [Online; navštíveno 27.04.2018].
URL https://software.intel.com/en-us/iot/hardware/discontinued

[6] Ochutnejte s námi Arduino! | HWKitchen.cz. [Online; navštíveno 25.04.2018].
URL https://www.hwkitchen.cz/

[7] Product Change Notification. [Online; navštíveno 27.04.2018].
URL http://qdms.intel.com/dm/i.aspx/C5E58142-4E04-4CBD-A7A6-BF330573055D/PCN115579-00.pdf

[8] Qt | Cross-platform software development for embedded & desktop. [Online; navštíveno 29.04.2018].
URL https://www.qt.io/

[9] Raspberry Pi - Teach, Learn, and Make with Raspberry Pi. [Online; navštíveno 25.04.2018].
URL https://www.raspberrypi.org/

[10] Dobeš, M.: *Zpracování obrazu a algoritmy v C#*. Praha : BEN - technická literatura, 2008, ISBN 978-80-7300-233-6.

[11] Fry, M.: Microsoft to consolidate the Kinect for Windows experience around a single sensor. [Online; navštíveno 12.04.2018].
URL https://blogs.msdn.microsoft.com/kinectforwindows/2015/04/02/microsoft-to-consolidate-the-kinect-for-windows-experience-around-a-single-sensor/

[12] Fuller, J.: How Virtual Laser Keyboards Work. [Online; navštíveno 14.04.2018].
URL https://electronics.howstuffworks.com/gadgets/travel/virtual-laser-keyboards.htm

[13] Henderson, H.: *Encyclopedia of Computer Science and Technology*. Facts on File science library, Facts On File, Incorporated, 2009, ISBN 9781438110035.
URL https://books.google.cz/books?id=3Tla6d153uwC

[14] Ing. Přemysl Kršek, P.: Základy počítačové grafiky IZG Studijní opora. [Online; navštíveno 16.04.2018].
URL https://wis.fit.vutbr.cz/FIT/st/course-files-st.php?file=%2Fcourse%2FIZG-IT%2Ftexts%2Fizg_opora.pdf&cid=10359

[15] Josef Schwarz, R. R., Josef Strnadel: Microprocessors and Embedded Systems IMP Study text Section 1. [Online; navštíveno 18.04.2018].
URL https://wis.fit.vutbr.cz/FIT/st/course-files-st.php.cs?file=%2Fcourse%2FIMP-IT%2Ftexts%2FAnglicka_opora%2FSection_1.pdf&cid=11452

[16] Kobie, N.: Intel Edison: an SD-card sized PC for wearable computing. [Online; navštíveno 27.04.2018].
URL http://www.alphr.com/news/386362/intel-edison-an-sd-card-sized-pc-for-wearable-computing

[17] KORTH, D.-P., HANS-E.: *EP0554492 (A1) - Method and device for optical input of commands or data*. 1992, [Online; navštíveno 15.04.2018].
URL https://worldwide.espacenet.com/publicationDetails/biblio?FT=D&date=19930811&DB=&locale=en_EP&CC=EP&NR=0554492A1&KC=A1&ND=1

[18] Moorhead, J.: Raspberry Pi device will reboot computing in schools. [Online; navštíveno 26.04.2018].
URL https://www.theguardian.com/education/2012/jan/09/raspberry-pi-computer-revolutionise-computing-schools

[19] Neznámý: 3D scanning with Windows 10. [Online; navštíveno 11.04.2018].
URL https://developer.microsoft.com/en-us/windows/hardware/3d-print/scanning-with-kinect

[20] Neznámý: Kinect for Windows SDK. [Online; navštíveno 12.04.2018].
URL https://msdn.microsoft.com/en-us/library/hh855347.aspx

[21] Neznámý: Kinect for Windows Sensor Components and Specifications. [Online; navštíveno 11.04.2018].
URL https://msdn.microsoft.com/en-us/library/jj131033.aspx

[22] Neznámý: Protocol Documentation - OpenKinect. [Online; navštíveno 12.04.2018].
URL https://openkinect.org/wiki/Protocol_Documentation#Control_Commands

[23] Parker, J.: *Algorithms for image processing and computer vision*. New York : Wiley Computer Publishing, 1997, ISBN 0-471-14056-2.

[24] Salinas, R. M.: RaspiCam: C++ API for using Raspberry camera with/without OpenCv. [Online; navštíveno 29.04.2018].
URL https://www.uco.es/investiga/grupos/ava/node/40

[25] Wilson, A.; Cutrell, E.: Flowmouse: A computer vision-based pointing and gesture input device. In *Francesca Costabile, M. & Paternò, F. (Eds.), Human-Computer Interaction INTERACT '05*, ACM, September 2005, ISBN 978-3-540-28943-2, s. 565–578.
URL https://www.microsoft.com/en-us/research/publication/flowmouse-a-computer-vision-based-pointing-and-gesture-input-device/

[26] Wilson, A. D.: Robust Computer Vision-based Detection of Pinching for One and Two-handed Gesture Input. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*, UIST '06, New York, NY, USA: ACM, 2006, ISBN 1-59593-313-1, s. 255–258, doi:10.1145/1166253.1166292.
URL http://doi.acm.org.ezproxy.lib.vutbr.cz/10.1145/1166253.1166292

[27] Wilson, M.: Exclusive: Microsoft Has Stopped Manufacturing The Kinect. [Online; navštíveno 12.04.2018].
URL https://www.fastcodesign.com/90147868/exclusive-microsoft-has-stopped-manufacturing-the-kinect

[28] Wu, G.: Kinect Sign Language Translator – part 1. [Online; navštíveno 11.04.2018].
URL https://www.microsoft.com/en-us/research/blog/kinect-sign-language-translator-part-1/

[29] Žára, J.; Beneš, B.; Sochor, J.; aj.: *Moderní počítačová grafika*. Computer Press, 2004, ISBN 9788025104545.
URL https://books.google.cz/books?id=USQnAAAACAAJ

[30] Zemčík, P.; Španěl, M.; Beran, V.; aj.: Zpracování obrazu ZPO Studijní opora. [Online; navštíveno 17.04.2018].
URL https://wis.fit.vutbr.cz/FIT/st/course-files-st.php.cs?file=%2Fcourse%2FZPO-IT%2Ftexts%2FOpora-ZPO-2011.pdf

[31] Zhang, T. Y.; Suen, C. Y.: A Fast Parallel Algorithm for Thinning Digital Patterns. *Commun. ACM*, ročník 27, č. 3, Březen 1984: s. 236–239, ISSN 0001-0782, doi:10.1145/357994.358023.
URL http://doi.acm.org/10.1145/357994.358023

Příloha A

Obsah přiloženého paměťového média

- `README.TXT` - Obsahuje informace o aplikacích, jak je spustit a používat
- `doc/` - Složka obsahující dokumentace
  - `thesis/` - Text a zdrojové soubory této práce
  - `doxygen/` - Vygenerovaná programová dokumentace
- `src/` - Zdrojové soubory
  - `camera/` - Zdrojové soubory strany vstupního zařízení
  - `client/` - Zdrojové soubory strany klientské aplikace
- `bin/` - Složka s binárními soubory