VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
FACULTY OF MECHANICAL ENGINEERING

ÚSTAV AUTOMATIZACE A INFORMATIKY
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

Detekce objektu pro průmyslového robota s využitím počítačového vidění

DETECTION OF OBJECTS FOR INDUSTRIAL ROBOTS USING COMPUTER VISION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Michal Huber
AUTHOR

VEDOUCÍ PRÁCE — Ing. Roman Parák
SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání bakalářské práce

Ústav:	Ústav automatizace a informatiky
Student:	**Michal Huber**
Studijní program:	Strojírenství
Studijní obor:	Základy strojního inženýrství
Vedoucí práce:	**Ing. Roman Parák**
Akademický rok:	2018/19

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Detekce objektu pro průmyslového robota s využitím počítačového vidění

Stručná charakteristika problematiky úkolu:

Práce bude zahrnovat rešerši knihovny OpenCV a její využií v oblasti detekci objektu, seznámení se s problematikou detekci objektu v průmyslových aplikacích, rovněž s vývojovou platformou Raspberry Pi.
Predmětem práce bude návrh řídíciho programu pro zpracování obrazových dat z kamery pomocí Raspberry Pi. Další částí práce bude návrh a implementace programu pro komunikaci mezi Raspberry pi a řídicí jednotkou průmyslového robotu (volba řídicí jednotky bude určena vedoucím práce). Zavěr práce bude venován implementaci návrhu řídiciho algoritmu pro detekci objektu a ověření funkčnosti vytvořeného řešení.

Práce předpokládá aktivní přístup studenta a nutnost práce v laboratoři.

Cíle bakalářské práce:

Seznamte se s problematikou detekci objektu v průmyslových aplikacích. Zpracujte prehled aktualniho stavu v dané oblasti.
Seznamte se s vývojovou platformou Raspberry Pi.
Proveďte rešerši knihovny OpenCV a její využití v oblasti detekci objektu.
Navrhněte řídicí program pro zpracování obrazových dat z kamery.
Implementujte návrh řídiciho algoritmu.
Navrhněte a implementujte program pro komunikaci mezi Raspberry pi a řídicí jednotkou průmyslového robotu.
Ověřte funkčnost vytvořeného řešení.

Seznam doporučené literatury:

SONKA, M., HLAVAC, V., BOYLE, R.: Image Processing, Analysis, and Machine Vision, Third Edition, Thornson, 2008.

UPTON, E., HALFACREE, R.: Raspberry Pi, COMPUTER PRESS, 2016. Martin Stříž, Bučovice, 2015. 200 s. ISBN: 978-80-251-4819-8.

OpenCV: Open Computer Vision library.https://opencv.org/

SHAPIRO, L., STOCKMAN, G.: Computer vision. Prentice Hall, 2001.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2018/19

V Brně, dne

L. S.

doc. Ing. Radomil Matoušek, Ph.D. ředitel ústavu	doc. Ing. Jaroslav Katolický, Ph.D. děkan fakulty

ABSTRAKT

Cílem této bakalářské práce je vytvoření řídícího programu pro zpracování obrazových dat z kamery. Úvodní část práce pojednává o současném stavu detekce objektů v průmyslových aplikacích, a také představuje vývojovou platformu Raspberry Pi a OpenCV knihovnu. Další kapitoly se pak zabývají přípravou Raspberry Pi, návrhem testovacích objektů, úpravou zachycených snímků a tvorbou samotného řídícího programu pro detekci správně vyrobené součásti. Závěr práce je věnován návrhu vizualizačního rozhraní, a především pak ověření funkčnosti vytvořeného řídícího programu.

ABSTRACT

The aim of this bachelor thesis is to create an image processing algorithm based on data captured by camera. Introduction of this thesis deals with the current situation of object detection in industrial applications and briefly presents Raspberry Pi and OpenCV library. Following chapters deal with Raspberry Pi setup, test objects design and captured pictures modifications. Last section of these chapters is devoted to a design of an image processing algorithm. The conclusion of this thesis deals with a design of vizualization interface and also describes a laboratory experiment to test the functionality of a designed algorithm.

KLÍČOVÁ SLOVA

Počítačové vidění, Python, OpenCV knihovna, Raspberry Pi, kontrola kvality, detekce objektu, PLC

KEYWORDS

Computer vision, Python, OpenCV library, Raspberry Pi, quality check, object detection, PLC

BIBLIOGRAFICKÁ CITACE

HUBER, Michal. *Detekce objektu pro průmyslového robota s využitím počítačového vidění.* Brno, 2019. Bakalářská práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav automatizace a informatiky. Vedoucí práce Ing. Roman Parák.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce, Ing. Romanu Parákovi, který mi poskytl odbornou pomoc, mnoho cenných rad, studijních materiálů a přátelský přístup při zpracování bakalářské práce.

ČESTNÉ PROHLÁŠENÍ

Prohlašuji, že tato práce je mým původním dílem, zpracoval jsem ji samostatně pod vedením Ing. Romana Paráka a s použitím literatury uvedené v seznamu literatury.

V Brně dne 21. 5. 2019 ……

Michal Huber

OBSAH

1 ÚVOD

V současnosti je průmyslová výroba reformována pronikáním velkého množství nových technologií. Jednou z nich je i oblast počítačového vidění. Právě touto oblastí se také bude zaobírat tato bakalářská práce. Jejím cílem je návrh řídícího kódu pro zpracování dat z kamery. Takto získaná data jsou následně využita pro detekci špatně vyrobených součástí. Praktické využití podobného návrhu spadá do oblasti průmyslové automatizace.

V úvodní kapitole je představen současný stav detekce objektu v průmyslové výrobě a také jsou nastíněny některé aktuální trendy v této oblasti. Po rešerši následuje stručné představení několika současných aplikací počítačového vidění ve výrobě nebo i u produktu.

Další kapitola se zabývá představením vývojové platformy Raspberry Pi, která bude sloužit jako řídící deska pro zamýšlenou aplikaci počítačového vidění. Dále je stručně představen hardware i software tohoto jednodeskového počítače. Zvláštní pozornost je pak věnována kamerovému modulu a porovnání použitelných kamer pro Raspberry Pi.

Teoretickou část práce zakončuje stručné představení OpenCV knihovny, která poskytuje algoritmy počítačového vidění a strojového učení s cílem urychlit jejich uplatnění. V další části této kapitoly je také představen stručný přehled současného využití algoritmů OpenCV knihovny.

Praktickou část práce poté zahajuje konkrétní nastavení Raspberry Pi a představení nutných knihoven pro detekci obrazu. Následně je představen proces návrhu a výroby testovacích objektů. Další část této kapitoly se poté zaobírá úpravami obrázku před detekcí využitím algoritmů OpenCV knihovny. Tato kapitola je zakončena samotným návrhem řídícího kódu pro detekci.

V rámci poslední kapitoly je navržený řídící program pro detekci otestován. V této kapitole je tak přiblížen návrh stojanu, komunikace a vizualizace pro ověření funkčnosti řídícího kódu. Na základě dat získaných při reálném testování jsou vyvozeny závěry o funkčnosti.

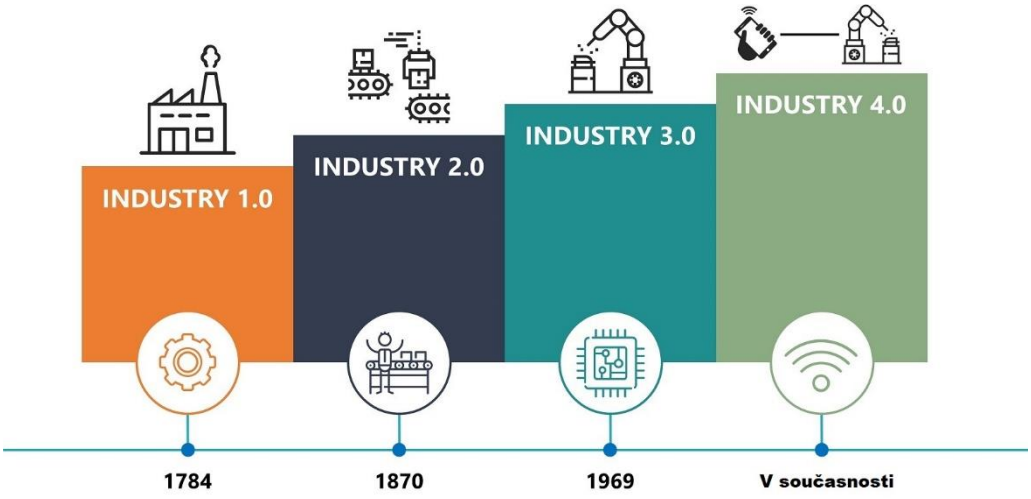
2 PŘEHLED SOUČASNÉHO STAVU DETEKCE OBJEKTU V PRŮMYSLOVÝCH APLIKACÍCH

Aktuální světový trend vývoje se zaměřuje především na inovace. Ve všech oblastech lidského snažení tak dochází k rozvoji nových myšlenek a postupů. Jednou z oblastí, která doznává největších změn, je technika. V současnosti tak pozorujeme velké technologické změny, které postupně mění tvář celé společnosti. Mění se průmyslová výroba, dochází k robotizaci a automatizaci a postupně zaniká velké množství nekvalifikovaných profesí. Aktuální trend v této oblasti, je souhrnně nazýván období Průmyslu 4.0.

2.1 Průmysl 4.0

Industry 4.0 (v ČR Průmysl 4.0) je celoevropská iniciativa, která si klade za cíl udržet Evropskou unii na špičce technologického vývoje. Podobné programy byly představeny i Japonskem, USA a Čínou. [1]

Označení Průmysl 4.0 představila německá vláda v rámci Hannoverského veletrhu v roce 2011 za účelem podpory technologických společností, které mají v Německu velký ekonomický význam. Myšlenka této iniciativy spočívala v podpoře rozvoje a aplikace nových technologií do domácností i průmyslu, s cílem udržet německé firmy v absolutní špičce. Tuto iniciativu později přejala celá Evropská unie a pojem Průmysl 4.0 se stal výrazem, který sjednocuje rozvoj a uplatnění nových technologií. Číslo čtyři v označení Průmysl 4.0 odkazuje na čtvrtou průmyslovou revoluci. [1, 3]



Obr. 1: Historie průmyslu – upraveno [2]

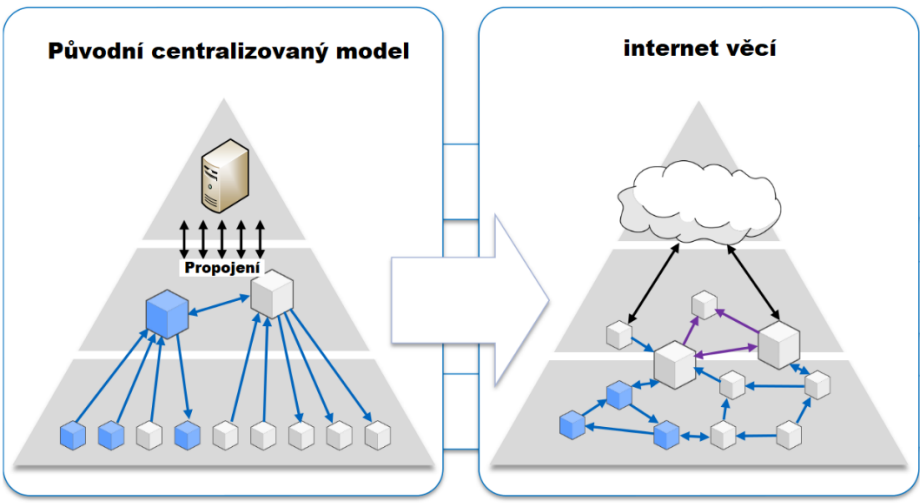
Termínem průmyslová revoluce je označováno období, kdy v důsledku technologického rozvoje, dochází k velkým změnám v uspořádání společnosti. O první průmyslové revoluci hovoříme s vynálezem parního stroje na konci 18. století. Následovala druhá průmyslová revoluce, která poprvé představila výrobní linky a elektrický proud. Třetí průmyslová revoluce pak přinesla do výrobního procesu informační technologie a automatizaci. [3]

V současnosti dochází k nástupu čtvrté průmyslové revoluce. V důsledku kompletní digitalizace, automatizace a robotizace se tak opět mění způsob průmyslové výroby. Vznikají první chytré továrny, které budou detailněji představeny na konci této kapitoly. Hlavní cíle čtvrté průmyslové revoluce spočívají ve větší rychlosti a efektivitě výroby, snížení nákladů a ekologičtějším využití materiálu. Mezi hlavní podmiňující faktory, které umožňují nástup čtvrté průmyslové revoluce, patří zlepšení v oblasti komunikačních technologií (internet a jiné bezdrátové systémy komunikace), schopnost zpracovávat velké množství dat a kolaborace člověka se strojem. Jednou z nejvýznamnějších inovací je pak internet věcí. [3]

2.1.1 Internet věcí

Pojem *Internet of Things* (zkráceně IoT, česky Internet věcí) byl poprvé představen britským podnikatelem Kevinem Ashtonem v roce 1999, během jeho práce na globální síti objektů, propojených pomocí rádiové frekvence (RFID). [3]

Internet věcí označuje síť fyzických zařízení, vybavených elektronikou, pohyblivými částmi, softwarem, senzory a síťovou konektivitou, která umožňuje zařízením vzájemné propojení a výměnu dat. Velmi časté je také uplatnění prvků umělé inteligence. Jednotlivá zařízení nashromáždí velké množství dat o vyráběném produktu a následně rozhodnou, která data předat dalším zařízením. Výsledkem je tak decentralizovaný systém, který umožňuje rychlejší a efektivnější výrobu a vzdálený přístup. [3]



Obr. 2: Internet věcí – upraveno [4]

S větším množství propojení však dochází také ke značnému nárůstu množství dat. Dále jsou proto rozvíjeny oblasti efektivního skladování, indexování a zpracování dat. Všechny data jsou však v současnosti vystavena velkým kybernetickým hrozbám. Z toho důvodu jsou vynakládány velké finanční prostředky na zabezpečení továren, využívajících internet věcí. [1, 3]

2.2 Počítačové vidění

V souladu s iniciativou Průmysl 4.0 dochází v současnosti k robotizaci a automatizaci většiny výrobních strojů v průmyslové výrobě, s cílem vytvořit stroj, schopný přijímat data v rámci IoT, a také schopný vyhodnotit situaci a případně reagovat na změny. Tento stroj tak ve výrobě nahradí lidskou obsluhu, neboť poskytne vyšší produktivitu. [1, 3]

Aby se však takový stroj mohl zařadit do většího technologického celku a zároveň plnit svou technologickou funkci, vyvstává potřeba rozvoje některých, pro stroj specifických, funkcí. Příkladem může být funkce počítačového vidění.

2.2.1 Počítačové vidění

Computer vision (zkráceně CV, česky počítačové vidění) je odvětví informačních technologií, které zahrnuje techniky umožňující počítačům „vidět“ a porozumět obsahu snímků na podobném principu, jako u lidského oka a následně vydat rozhodnutí, nebo novou interpretaci. [5]

Problematika počítačového vidění tak může na první pohled vypadat jednoduše, neboť problémy, jako je například identifikace auta na obr. 3, řeší člověk až triviálním způsobem. Vydat podobné rozhodnutí je však pro počítač obtížné. Počítač totiž každou oblast vidí pouze jako mřížku čísel, kde kterékoliv z těchto čísel může být znehodnoceno šumem nebo zkreslením. Tyto znehodnocení vznikají například z důvodu osvětlení, odrazů, pohybu nebo nedostatky čoček. [5]

Obr. 3: Způsob čtení obrázku počítačem [5]

K odstranění šumu je většinou přistupováno pomocí statistických metod. Příkladem může být snaha nalézt okraj objektu na obrázku. Pouhé porovnání hodnot dvou vedlejších pixelů nepřinese žádné výsledky. Pokud je však statisticky stanovena hodnota v určité oblasti a porovnána s jinou oblastí, stane se detekce okraje mnohem efektivnější. [5]

K některým jiným nedokonalostem obrázku můžeme přistupovat také na základě známých dat. Příkladem může být zkreslení způsobené čočkami. Zde stačí využít jednoduchého modelu polynomické regrese, sestaveného na základě takto deformovaných obrázků. Výsledkem je téměř dokonalé potlačení této vady. [5]

Průmyslové aplikace počítačového vidění se dělí na dvě hlavní oblasti. Prvním je využití při výrobě, například v rámci inteligentních výrobních linek. Druhým je pak využití v rámci samotného produktu.

2.2.2 Průmyslové využití počítačového vidění při výrobě

Chytrá továrna

Smart factory (česky chytrá továrna) je koncept továrny v souladu s iniciativou Průmyslu 4.0. Unikátnost této továrny spočívá ve schopnosti samostatného řízení ostatních strojů, za účelem dosažení požadovaného výstupu. Název chytrá továrna odkazuje na analogii s lidským tělem, kdy tato továrna musí být schopna vnímat, provádět rozhodnutí a na jejich základě konat. [1, 6, 7]

Vize konceptu chytré továrny je následující. Zákazník zadá požadavky počítačovému programu, který naplánuje výrobu a zjistí dostupnost materiálu. V případě nedostatku materiál objedná. Pro přepravu mezi skladem a výrobním prostorem jsou využívány autonomní vozítka a kontrola dostupnosti materiálu je zajištěna pomocí dronů. Údaje o výrobku jsou pomocí internetu věcí sdíleny v reálném čase se všemi komponenty výrobní linky. Jednotlivé stroje tak provedou svůj technologický úkon a odešlou informace ostatním výrobním strojům. Zde je důležité zmínit, že jednotlivé výrobní stroje nejsou před každou operací speciálně nastavovány. Velmi zásadní je tak schopnost stroje přizpůsobit se výrobě daného výrobku. Právě v těchto oblastech se nejčastěji uplatňují zmíněné techniky počítačového vidění, kdy výrobní stroj opticky kontroluje proces výroby a na základě „viditelných“ dat upravuje další technologický postup. [7]

V celém procesu výroby tak bude člověk plnit pouze pozorovací funkci. V samotné továrně postačí několik pracovníků, kteří budou kontrolovat proces výroby a zabývat se vylepšováním softwaru továrny a fyzickou kontrolou jednotlivých výrobních strojů. Tento proces automatizace továren zvyšuje řádově produktivitu, šetří materiál a umožňuje vzdálený přístup. Zároveň však také eliminuje velké množství profesí, neboť doposud dělali snadno opakovatelné úkony lidé. Jejich práce se tak stává nadbytečnou, což povede k velkým změnám v uspořádání společnosti, v souladu s myšlenkami čtvrté průmyslové revoluce.

Obr. 4: Automatická výrobní linka [8]

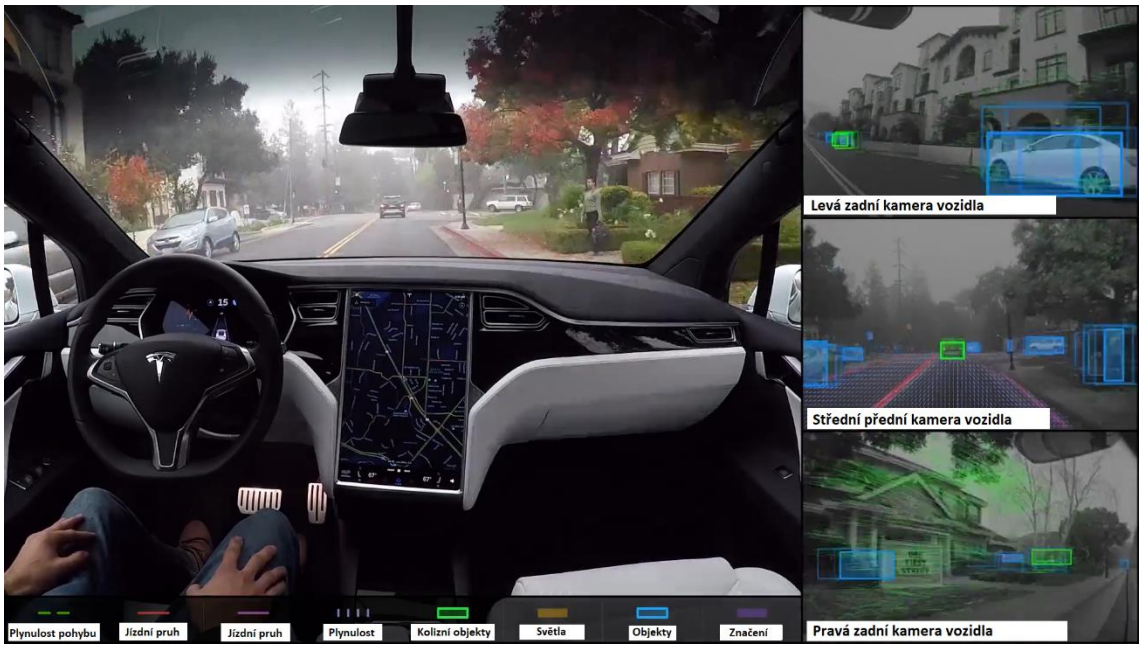
Koncept chytré továrny zaručuje velký pokrok v průmyslové výrobě. Momentálně však zůstává u konceptu, a většina firem se snaží o postupnou robotizaci a automatizaci a uplatňují prvky chytrých továren postupně. Došlo totiž k několika situacím, kdy firma byla nucena pozdržet výrobu, z důvodu přílišné automatizace.

2.2.3 Průmyslové využití počítačového vidění u produktu

Jednou z nejvíce viditelných využití technik počítačového vidění je automobilový průmysl. Moderní automobily jsou již standartně vybaveny parkovacími asistenty, měřiči vzdáleností mezi auty nebo systémy, jež zabraňují kolizi s chodci. Všechny tyto systémy jsou založeny právě na oblasti počítačového vidění. Současný trend však směřuje k samořídícím automobilům (tzv. *autonomous cars*). Tyto autonomní automobily jsou na základě dat pořízených z velkého množství kamer schopny nahradit lidského řidiče.

Technologie samořídících automobilů je velmi perspektivní, avšak její reálné využití je na počátku svého vývoje. V současné době umožňuje legislativa používat samořídící nastavení pouze pod aktivním dohledem řidiče. Nezřídka totiž dochází k incidentům, kdy dojde k mylné identifikaci chodce, nebo překážky na silnici a následné havárii. Tento trend, kdy dochází k postupné automatizaci řízení automobilů, podporuje většina výrobců. Dochází tak k dalšímu zdokonalování již používaných technologií a prvním pokusům o vytvoření komplexního samořídícího systému. Na čele těchto snah stojí především společnosti Tesla Motors, Uber, Google nebo Navya, která vyvíjí i samořídící autobusy. [9, 10]

Společnost Tesla Motors představila první verzi samořídícího systému v roce 2014. Tato prvotní verze umožňovala polo-autonomní řízení a plně automatické parkování. Současné verze jsou schopny plně autonomního řízení, které však legislativa z bezpečnostních důvodů zakazuje. Příklad samořídícího automobilu v provozu je zobrazen na obr. 5. [10]

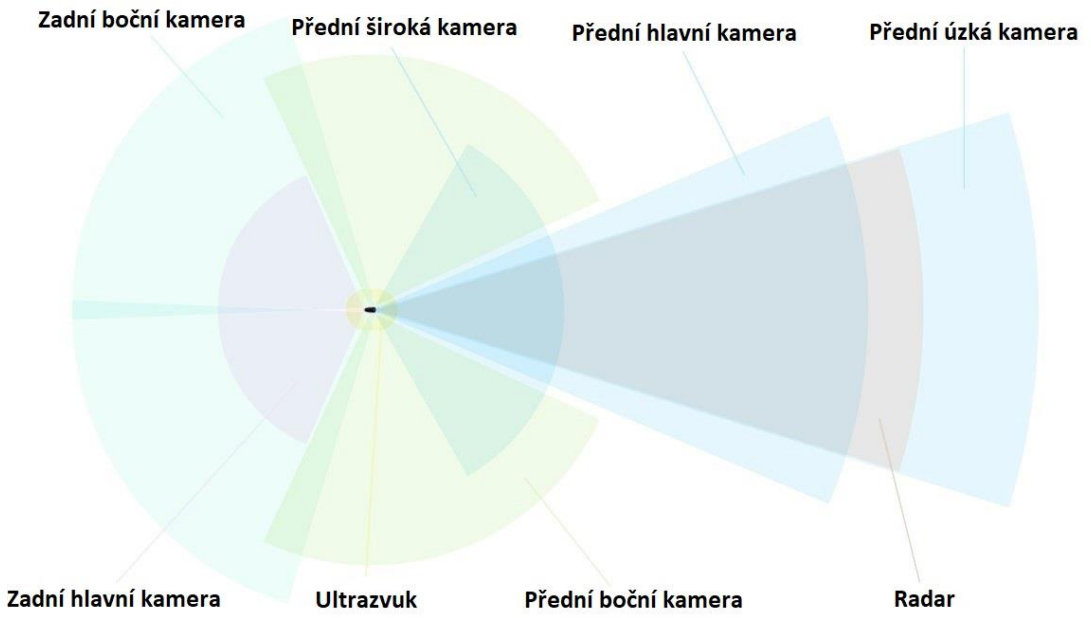


Obr. 5: Využití počítačového vidění u automobilu Tesla – upraveno [10]

Samořídící automobily využívají komplexní kamerový systém, který je postaven právě na metodách počítačového vidění a detekce objektu. Pomocí kamerového systému tak vozidlo například detekuje vzdálenost mezi vozidly, detekuje dopravní značky a jiné dopravní signály, nebo odhaduje zakřivení cesty. Při zpracování dat získaných z kamer je často využíváno strojové učení, především pak neuronové sítě. [9]

Neuronové sítě jsou výpočetní moduly využívané při detekci objektů v okolí vozidla. Tyto sítě jsou trénovány pomocí velké databáze pozitivních a negativních snímků, na jejichž základě se naučí, jak daný objekt vypadá a následně jsou schopny jej detekovat v blízkosti vozidla. Výsledkem je pak vykreslení tzv. *bounding boxu* (ohraničujícího obdélníku) při pozitivní detekci objektu. [9]

Tesla pro svůj samořídící systém využívá kamerový systém sestávající z 6 hlavních kamer, které jsou doplněny řadou senzorů a dohromady poskytují 360° viditelnost, a to až do vzdálenosti 250 metrů od vozidla. [9, 10]



Obr. 6: Kamerový systém Tesla – upraveno [10]

Samotné kamery však nejsou schopny poskytnout dostatek informací o rychlosti a vzdálenostech. Proto jsou doplněny o radar, ultrazvuk, laser a další senzory. Na základě údajů z kamerového systému a pomocných detekčních zařízení jsou informace v reálném čase zpracovány a vyhodnoceny. Celý tento proces je tak závislý na velkém množství proměnných, a proto může poměrně snadno dojít k chybě. Nejčastější chyby se vyskytují při samotné detekci počítačovým viděním, kdy dojde k některé, z již zmíněných chyb. I při drobné chybě však mohou nastat fatální důsledky. [9]

2.2.4 Společnosti zabývající se průmyslovou automatizací

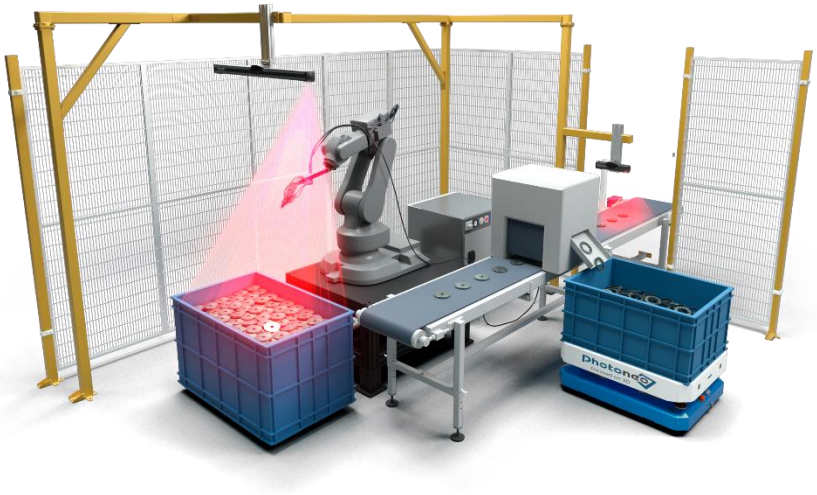
Automatizace procesu výroby je poměrně novou oblastí průmyslu. Většina společností, tak nemá příliš dlouhou tradici a ani trh prozatím neregistruje velký konkurenční boj. V této práci budou představeny dvě společnosti, které do jisté míry určují směr vývoje v této úzce specializované oblasti. Obě tyto společnosti také mají pobočky ve střední Evropě. Konkrétněji se jedná o společnosti Photoneo a SANEZOO.

Start-up Photoneo, který v roce 2015 obdržel jednu z historicky největších investic do start-upu ve střední Evropě, je slovenská firma specializující se na oblast počítačového vidění a průmyslové automatizace. [11]

Photoneo vyvinulo unikátní metodu 3D snímání objektu, která společnosti umožňuje produkci 3D kamer s vysokým rozlišením, a to až při 30 snímcích za sekundu při současné detekci. Tato kamera proto nalézá uplatnění v logistice, vzdušné dopravě nebo v rámci lékařských přístrojů. [11]

Pro průmyslovou výrobu má však největší význam její možné uplatnění v rámci automatizovaných linek. Kamera, v kombinaci s robotickým ramenem, slouží pro tzv. *bin-picking* (vytahování součástí z přepravky). V kombinaci s dalšími moderními

technologiemi, jako například autonomními roboty, nebo kamerou pro kontrolu kvality, vzniká komplexní výrobní linka, jejíž příklad je zobrazen na obr. 7. [11]



Obr. 7: Komplexní automatizované pracoviště společnosti Photoneo [12]

Další společností, která se zabývá oblastí průmyslové automatizaci a počítačového vidění, je SANEZOO.

Tato společnost, založená v roce 2016 v Kalifornii, se zaměřuje na tvorbu kamer, které spojují rychlost a přesnost stroje, s téměř lidskou schopností přizpůsobení se situaci. Z tohoto důvodu jsou kamery společnosti SANEZOO schopny řešit i komplikované úkoly počítačového vidění, jako například komplikované měření, detekci nebo inspekci. [13, 14]

Společnost SANEZOO využívá ve svých produktech kromě prvků počítačového vidění, také umělé inteligence a strojového učení. Výsledkem je tak systém, který pomocí průmyslových kamer a superpočítače vydává téměř okamžitá rozhodnutí. [13, 14]

Obr. 8: Detekce povrchu pneumatiky [13]

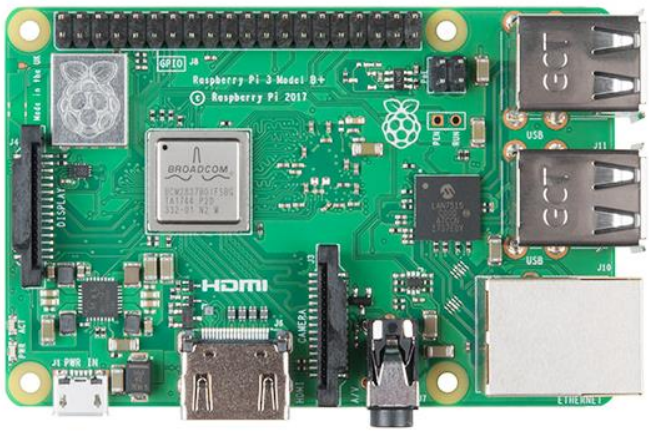
Uplatnění kamer společnosti SANEZOO do výrobního procesu tak násobně zrychluje inspekční proces, zvyšuje efektivitu, zkracuje čas výroby a tím zlevňuje finální produkt.

3 RASPBERRY PI

3.1 Volba řídící jednotky

Pro reálnou aplikaci detekce objektu je velmi výhodné použít jednodeskový počítač. Ten si lze představit jako miniaturní počítač, často velikosti kreditní karty, který obsahuje všechny komponenty počítače s výjimkou zobrazovacího média a ovládacích prvků. Jeho rozměry, nízká cena a snadná dostupnost jej tak činní použitelným pro široké spektrum využití. První jednodeskový počítač byl představen britskou nadací *Raspberry Pi Foundation*. (*RPi Foundation*) [15, 16]

Tato nadace, v čele s Ebenem Uptonem, vyvinula prvotní model z důvodu klesajících schopností studentů programovat, a naopak narůstající poptávky po pracovnících v oblasti IT. Současný trh je totiž ovládán trendem hotových výrobků s uzavřeným operačním systémem. Cílem *RPi Foundation* je tak probudit v lidech, především pak studentech, opětovně zájem o programování a poskytnout jim pro to dostupný prostředek. Především z důvodů nízké ceny a kvalitního zpracování, si tato platforma drží dominantní postavení na trhu. Na její úspěch se pokouší navázat podobné jednodeskové počítače, které se snaží dále vylepšovat přednosti Raspberry Pi, ale v současnosti tvoří minoritní podíl na trhu. Jejich příklady jsou Banana Pi, Orange Pi, Le Potato nebo Pocket Beagle. [15, 16]



Obr. 9: Raspberry Pi 3B+ v horním pohledu [17]

Na prvotní model navazuje několik dalších verzí (1A, 1B, 1A+, 1B+, 2B, Zero, W/WH, 3B, 3A+, 3B+). Tyto verze se vyznačují postupně narůstající výpočetní kapacitou, počtem GPIO konektorů a novější verze již mají zabudován hotspot pro Wi-Fi. Všechny také sdílí velikost kreditní karty, s výjimkou Raspberry Pi Zero a W/WH, jejichž rozměry jsou ještě menší. Nejdražší verze stojí $35. [17]

Obr. 10: Raspberry Pi Zero [17]

Po důkladném zvážení byla pro návrh programu na detekci objektu v dalších kapitolách zvolena verze Raspberry Pi 3 B+ (v následujícím textu bude používána zkratka RPi)

3.2 Hardware RPi 3B+

Základem počítače je čip Broadcom BCM2387B0. Jedná se o stejný typ čipu, používaného i u předchozí verze. Nově je však překryt rozvaděčem tepla a RPi tak funguje i na vyšších frekvencích. Jeho čtyři 64bitová jádra procesoru ARM Cortex-A53 umožňují při teplotě do 70 °C využít frekvenci 1,4 GHz. Při vyšších teplotách se procesor podtaktuje zpět na původních 1,2 GHz. Drobnou nevýhodou je kapacita RAM (*Random-access memory*). Ta je v současnosti limitována na 1 GB. Pro budoucí Model 4 je proto její navýšení jednou z priorit. [16, 18]

S narůstajícím číslem verze dochází postupně k velkému vývoji v oblasti připojení. Model 3B+ obsahuje gigabitový Ethernet, který je však stále připojen přes rozhraní USB 2.0, takže nelze využít plnou propustnost. Výrobce uvádí, že i přes tento nedostatek naměřil rychlost 315Mb/s, tedy více než trojnásobnou v porovnání s předchozí verzí. [18]

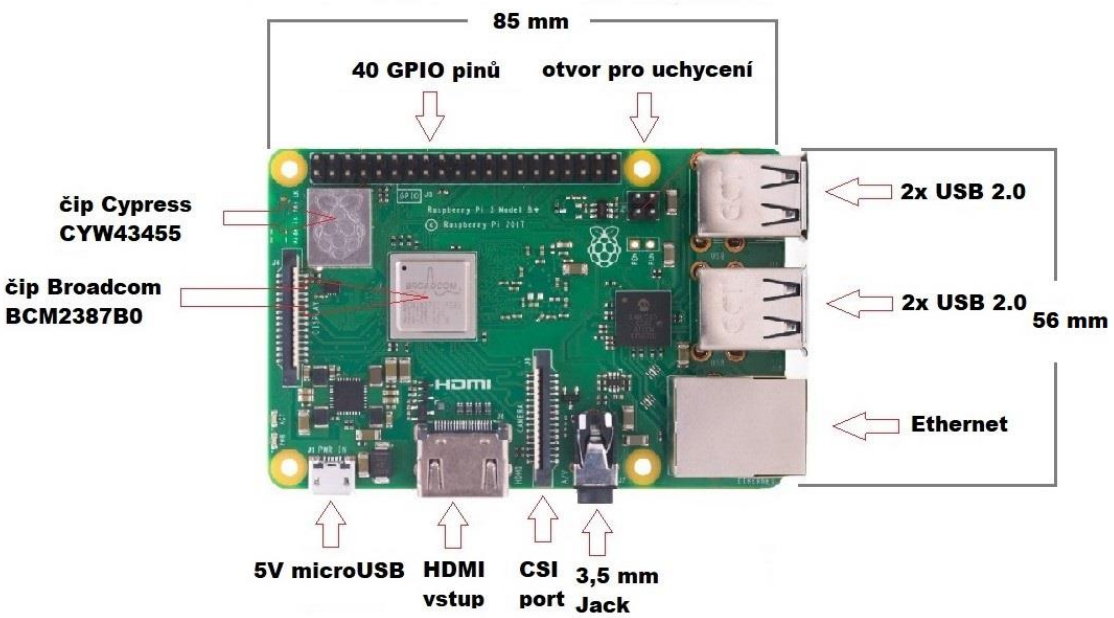
Tab. 1: Výrobcem naměřené hodnoty [18]

	Rychlost přenosu dat [Mb/s]
RPi 3B	94,1
RPi 3B+	315

Do této verze je také poprvé zahrnut čip Cypress CYW43455, který modelu 3B+ poskytuje, Bluetooth 4.2 a především Wi-Fi 802.11ac. Tento modul poskytuje RPi lepší

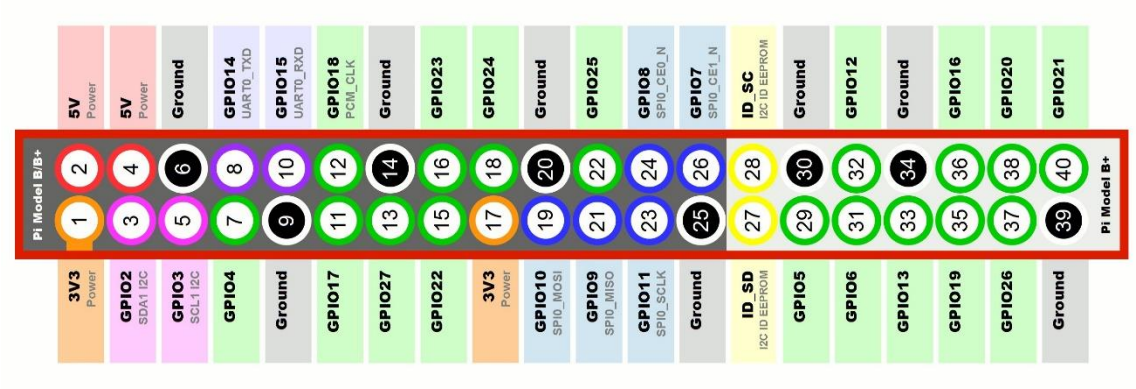
výkon v pásmu 2,4 GHz a nově také umožňuje využití pásma 5GHz. Výsledkem je tak násobně vyšší přenosová rychlost. [16, 18]

Na desce se nachází všechny konektory, které umožňují využití RPi jako malého stolního počítače, nebo jeho zapojení do projektů. Základem jsou 4 USB (*Universal Serial bus*) porty verze 2.0, již zmíněný gigabitový Ethernet, 3,5 mm Jack a také *High Definition Multimedia Interface* (zkráceně HDMI) vstup, který umožňuje desku připojit k obrazovce a při současném zapojení klávesnice a myši může být RPI využito jako malý stolní počítač. Velmi důležitý je také výskyt slotu pro microSD kartu, neboť RPi nemá integrované žádné vnitřní uložiště. Operační systém je tak načítán přímo z microSD karty, případne přes USB port. [16, 18]



Obr. 11: Raspberry Pi 3B+ konektory – upraveno [19]

RPi však obsahuje i konektory, které jsou pro jiná zařízení méně standartní. Příkladem je 40 GPIO pinů, které umožňují připojení různých senzorů, nebo přidavných modulů. Dále pak DSI (*Display Serial Interface*) port, který umožňuje připojit RPi k dotykové obrazovce, a především CSI (*Camera Serial Interface*) port, který umožňuje k RPi připojit kameru. [16]



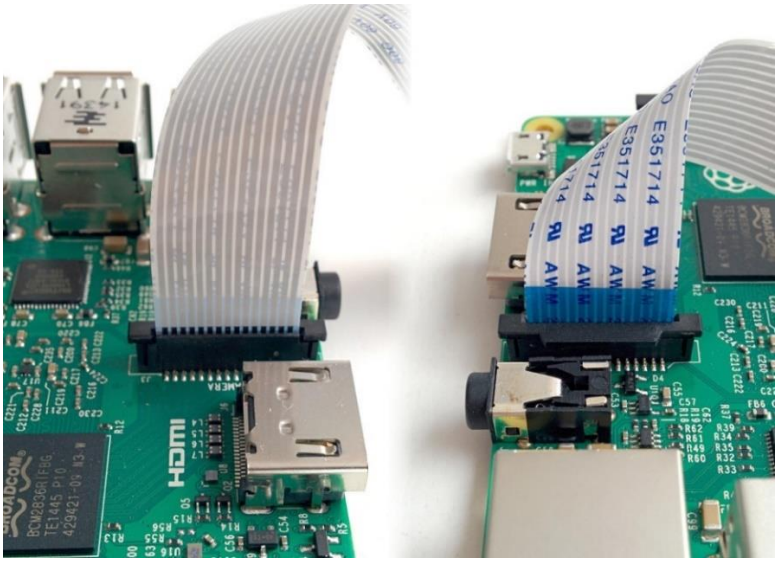
Obr. 12: Detail GPIO pinů [20]

Tento počítač je primárně napájen pomocí microUSB zdroje, který poskytuje 5V, ovšem může být nahrazen i zdrojem připojeným přes USB. Tato verze však v porovnání s předchozími využívá znatelně více energie. Proto vydal výrobce doporučení pro využívání kvalitního 2,5 A zdroje, např. oficiálního Raspberry Pi zdroje. [16]

3.3 Rozšíření Raspberry Pi o kamerový modul

V současné době jsou dostupné dvě možnosti rozšíření Raspberry Pi o modul kamery. Prvním řešením je využití zmíněného CSI portu, který umožňuje připojení oficiálních kamer pro Raspberry Pi (tzv. PiCamera). Druhým, univerzálnějším, je připojení webkamery pomocí USB portu.

Rozhraní CSI zajišťuje přímé propojení kamerového modulu s procesorem. Podobné rozhraní se využívá i u mobilních telefonů, pro které bylo původně navrženo. Komunikace je pouze jednosměrná (z kamery do procesoru) a dosahuje až 1Gb/s. Samotný konektor se na desce nachází mezi 3,5 mm Jackem a HDMI vstupem. [21]



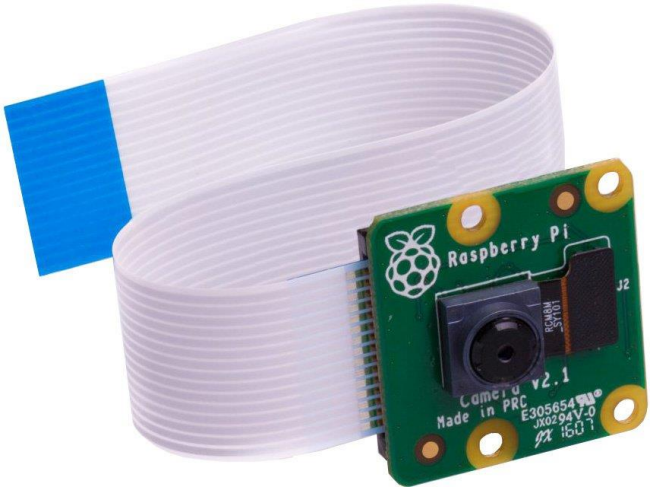
Obr. 13: Pozice CSI portu [17]

USB webkamery představují jednodušší řešení kamerového rozšíření. Problém však nastává s dostupností ovladačů pro jednotlivé kamery a v porovnání s CSI kamerami nedosahují takových rychlostí. Dalším aspektem při volbě je cena. Kvalitní webkamery, které by vyhovovaly svými parametry, jsou velmi nákladné. Pro následující praktickou aplikaci tak bylo zvoleno rozšíření pomocí CSI rozhraní připojením PiCamery. [21]

3.3.1 Raspberry Pi kamera

Raspberry Pi Camera (PiCamera) je oficiální kamera vydaná nadací RPi. K samotné desce je připojena pomocí 15-žilého flex kabelu.

PiCamera funguje obdobně, jako kamera mobilního telefonu. Ke snímání obrazu dochází na základě tzv. *rolling shutter* principu, kdy při zachycení snímku kamera čte hodnoty pixelů ze senzoru po řádcích, namísto zachycení všech hodnot najednou. Může tak dojít ke vzniku tzv. *rolling shutter* efektu, kdy dochází ke zkreslení u rychle se pohybujících součástí (např. vrtule leteckého motoru). Opakem jsou digitální zrcadlovky, využívající tzv. *global shutter* principu. I v tomto případě kamera většinou čte hodnoty pixelů po řádcích. Rozdíl je však v přítomnosti optické závěrky, kdy při zachycení obrázku dojde k otevření optické závěrky, samotnému snímání scény senzorem a zavření optické závěrky. Sejmutý obrázek je tak zachycen v jednom okamžiku a až následně jsou odečítány hodnoty. [22]



Obr. 14: Raspberry Pi Camera [23]

Pro práci na praktické aplikaci bude využita kamera Raspberry Pi V2, zobrazená na obr. 14. Tento kamerový modul byl poprvé představen v dubnu 2016 a nahradil svého předchůdce, s označením V1. Porovnání těchto modulů se nachází v tabulce 2. Oba tyto moduly obsahují také infračervený filtr.

Tab. 2: Porovnání kamerových modulů [24]

	Kamerový modul V1	**Kamerový modul V2**
Cena	$25	$25
Váha	3 g	3 g
Rozlišení	5 Mpx	8 Mpx
Senzor	OmniVision OV5647	Sony IMX219
Rozlišení senzoru	2592 x 1944 px	3280 x 2464 px
Velikost pixelu	1,4 μm x 1,4 μm	1,12 μm x 1,12 μm

Spektrum využití PiCamery je stejně široké, jako využití RPi samotného. Kamerový modul se využívá například v rámci domácích bezpečnostních systémů, fotopastí v divočině nebo při rozpoznávání obličeje. Pro PiCameru jsou také dostupné různé knihovny třetích stran, například knihovna Python pro PiCameru, kterou je možno využít při zpracování obrazu.

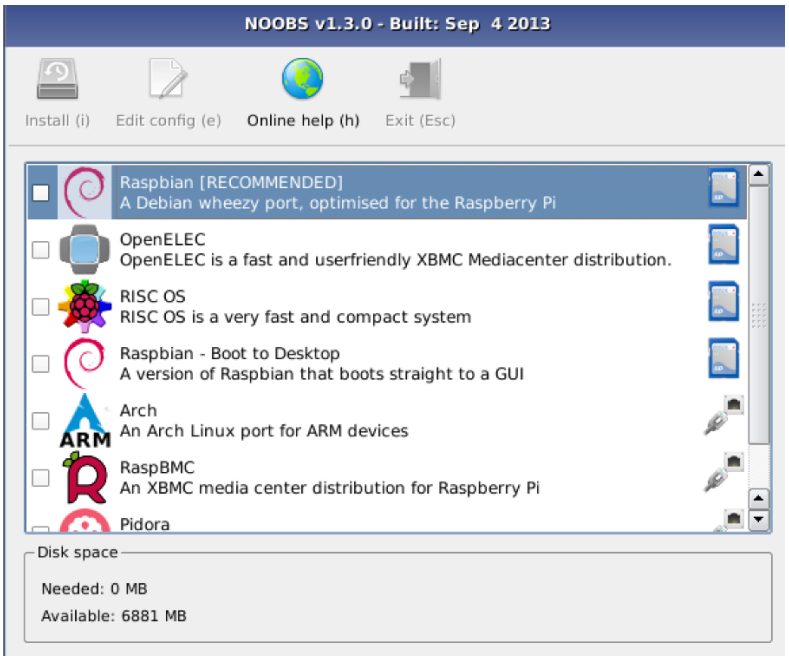
3.4 Software Raspberry Pi 3 B+

V souladu s cíli nadace Raspberry Pi, je celý návrh desky, stejně tak i doporučovaný software uvolněn pod svobodnou licencí. Tímto způsobem se nadace snaží zajistit rovné podmínky pro všechny studenty, nejen programování.

Jak již bylo zmíněno, řídící deska nemá integrované žádné uložiště. K zavedení operačního systému je tak nutno využít slotu na microSD kartu. Tuto kartu je doporučováno naformátovat před prvním použitím a dále při každé přeinstalaci operačního systému.

Při prvotním zapojení RPi je vhodné využít NOOBS (*new out of box system*). Výrobcem je prezentován jako nejjednodušší možnost instalace operačního systému, neboť uživatelsky přívětivě navádí koncového uživatele a nedá nezkušenému uživateli možnost udělat chybu. [25]

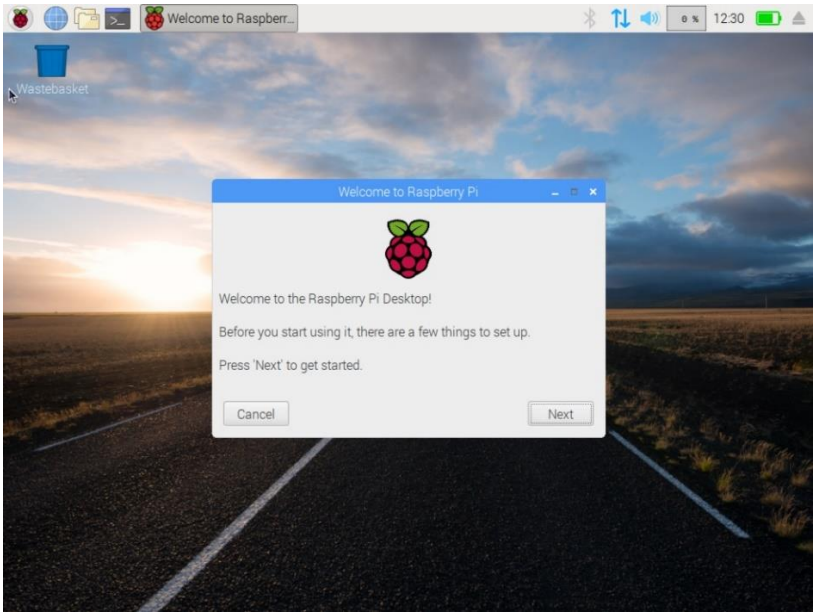
Po připojení k monitoru nabídne koncovému uživateli možné operační systémy, případně umožní připojení k Wi-Fi pro další možnosti.



Obr. 15: Instalace pomocí NOOBS [25]

Nadace Raspberry Pi doporučuje uživatelům využít operační systém Raspbian, klon linuxové distribuce Debian.

Jedním z cílů nadace byla i snadná orientace začátečníků, proto se prostředí Raspbianu podobá prostředí, jaké poskytuje většina stolních počítačů.



Obr. 16: Úvodní obrazovka Raspbianu [25]

Základem softwaru je několik programovacích editorů. Jedním z nich je Geany, editor podobný Notepadu++. Tento editor podporuje využití širokého spektra programovacích jazyků, například C, Java, PHP, HTML, Python, nebo Pascal. Dále pak

poskytuje nástroje pro výuku Pythonu, integrované prostředí Python3. Pro svoji jednoduchost, širokou použitelnost a intuitivní orientaci si získal velkou oblibu mezi studenty, využívajícími RPi k výuce programování. Dalšími oblíbenými programovacími editory jsou například BlueJ, Greenfoot nebo Scratch. [15, 25]

Raspbian obsahuje také základní kancelářský software. LibreOffice je balíček programů, zahrnující textový a tabulkový editor, dále pak software pro tvorbu prezentací, malování nebo komplexní matematickou aplikaci. Pro přístup k internetu slouží Chromium, otevřená verze prohlížeče Google Chrome. Raspbian také podporuje hraní her. Všechny tyto vestavěné aplikace, připomínající stolní počítače, tak dělají Raspberry Pi snadno osvojitelné i pro uživatele stolních počítačů. [15, 25]

4 OPENCV KNIHOVNA

Open source computer vision library (zkráceně OpenCV knihovna) je otevřená knihovna počítačového vidění. Knihovna je napsána v programovacích jazycích C a C++ a je podporována operačními systémy Windows, Linux, Mac OS, a Android. V současné době však umožňuje také práci pomocí jazyků Python, Java, Ruby, MATLAB a několika dalších jazyků. [5]

Obr. 17: Logo OpenCV knihovny [5]

Jedním z primárních cílů OpenCV knihovny je poskytnout jejím uživatelům snadno použitelnou infrastrukturu pro uplatnění počítačového vidění a urychlit tak využití strojového vidění v reálných aplikacích. V současnosti obsahuje tato knihovna více než 2500 algoritmů, které zahrnují vše, od základních, po nejkomplexnější algoritmy počítačového vidění a strojového učení. Právě počítačové vidění a strojové učení jsou často využívány společně. Z tohoto důvodu obsahuje OpenCV také podknihovnu pro strojové učení (*Machine Learning library*). [5]

Pro účely této práce byla využita verze OpenCV 4.0.

4.1 Historie OpenCV knihovny

Projekt OpenCV knihovny byl poprvé představen v roce 1999, jako jeden z výsledků iniciativy společnosti Intel, která si kladla za cíl vylepšení funkcí centrálních procesorových jednotek. Samotný nápad tvorby otevřené knihovny obsahující algoritmy přišel na základě návštěvy jednoho pracovníka Intelu na MIT, kde si v té době studenti mezi sebou předávali navzájem velké množství algoritmů, aby si navzájem ulehčili začátky studia a mohli se učit a navazovat na algoritmy svých předchůdců. [5]

Po této návštěvě se Intel rozhodl vyvinout otevřenou knihovnu počítačového vidění, která by umožnila získat podobný náskok ve studiu i všem ostatním studentům počítačového vidění na světě a podnítit tak jeho rozvoj a uplatnění. Klíčovou roli při

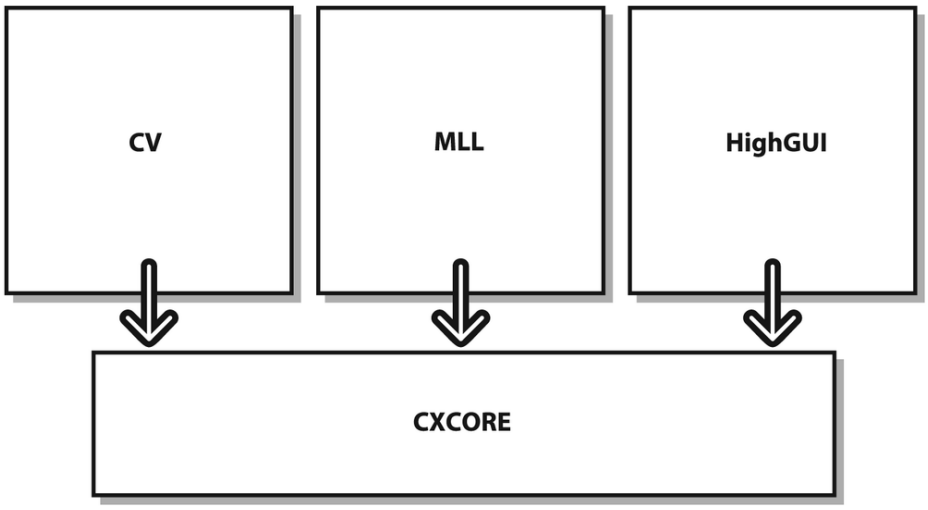
vývoji knihovny sehrálo, kromě výzkumných laboratoří Intelu, také několik ruských expertů na optimalizaci. Největší zásluhy v této oblasti se přisuzují Vadimu Pisarevskemu, který napsal a optimalizoval velkou část OpenCV knihovny a do dnešní doby je v této oblasti velmi aktivní. V současnosti je však OpenCV knihovna rozvíjena především jejími samotnými uživateli, kterými jsou studenti, vývojáři a firmy po celém světě. [5]

Na počátku vývoje měla OpenCV knihovna následující cíle: [5]

- Umožnit rozvoj počítačového vidění poskytnutím otevřeného a optimalizovaného kódu.
- Šířit znalosti počítačového vidění vytvořením společné infrastruktury, na které by mohli vývojáři postavit své projekty a jejich kód tak byl čitelnější a lépe přenositelný.
- Poskytnutí přenosného optimalizovaného kódu s otevřenou licencí pro komerční využití a s tím související komerční aplikace v oblasti vidění.

4.2 Struktura OpenCV knihovny

OpenCV knihovna je strukturována do pěti hlavních oblastí, z nichž čtyři jsou znázorněny na obr. 18.



Obr. 18: Struktura OpenCV knihovny [5]

Oblast CV (*Computer Vision*) obsahuje algoritmy zpracování obrazu a další techniky počítačového vidění. Již zmíněná oblast MLL (*Machine Learning library*) obsahuje velké množství klasifikátorů a nástrojů shlukové analýzy. Oblast HighGUI obsahuje funkce pro ukládání a načítání obrázků a videa a CXCore pak obsahuje základní struktury a algoritmy. Na obr. 18 chybí oblast CvAux. Tato podknihovna obsahuje

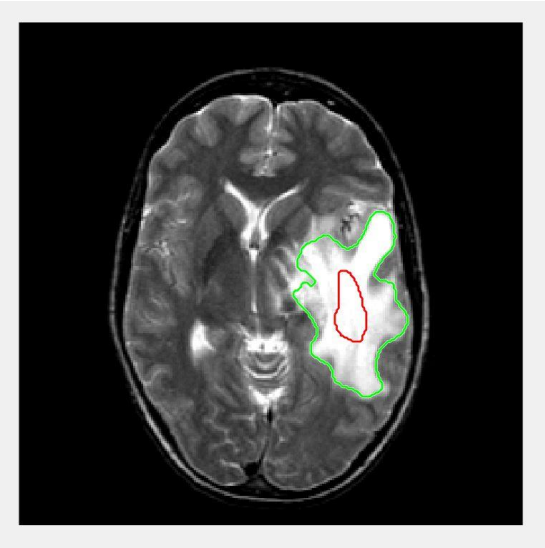
zaniklé oblasti (například při rozpoznání obličeje) a experimentální algoritmy pro segmentaci popředí, respektive pozadí. [5]

4.3 Přehled uplatnění OpenCV knihovny

Již od prvotní verze z roku 1999 si OpenCV knihovna nachází uplatnění téměř ve všech oborech lidského snažení. Její význam je nepopíratelný pro vědecké pracovníky, ale aplikace založené na metodách počítačového vidění rychle pronikají i mimo vědecké kruhy.

Jedním z využití OpenCV a počítačového vidění obecně je i spojování více snímků za účelem vytvoření komplexních satelitních a internetových map. OpenCV knihovna je tak využívána pro tvorbu map například společností Google. Dalšími světovými firmami využívajícími tuto technologii jsou například Yahoo, Microsoft, Intel, IBM, Sony, Honda nebo Toyota. [5, 26]

Velký potenciál má uplatnění metod počítačového vidění v oblasti medicíny, kde v kombinaci se strojovým učením a umělou inteligencí vyhodnocuje údaje o lidském těle. Podobné algoritmy se v současnosti využívají především při analýze rentgenových a ultrazvukových snímků, kdy pomáhají lékařům zpracovat velké množství dat a odhalit případné anomálie. [5, 26]



Obr. 19: Využití OpenCV v medicíně [27]

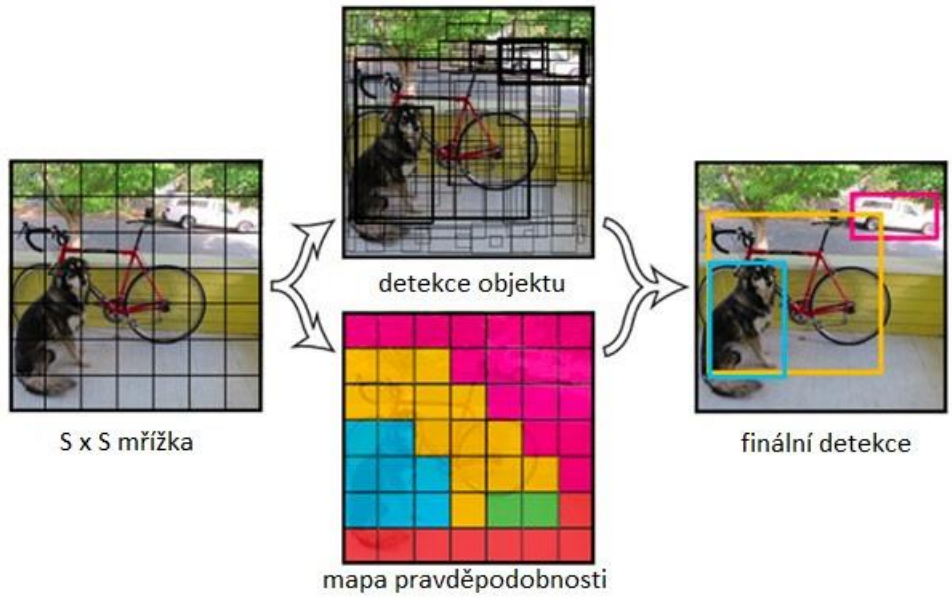
Dalšími příklady využití OpenCV jsou například monitorování výbušnin v dolech, detekce topících se plavců, kontrola vzletových a přistávacích drah letišť nebo bezpečnostní systémy na rozpoznání obličeje. [5]

Pro technickou praxi má největší význam její využití v oblasti automatizace a robotiky, v souladu s iniciativou Průmysl 4.0.

4.4 Aktuální využití OpenCV pro detekci objektu

V současnosti zaznamenávají metody detekce objektu s využitím OpenCV knihovny velký pokrok. Ve většině případů jsou tyto metody kombinovány se strojovým učením a umělou inteligencí. Vznikají tak často velmi komplexní algoritmy, které jsou schopny lépe reagovat na měnící se prostředí a v některých případech poskytovat lepší a rychlejší výsledky. Ve zbylých, neojedinělých případech však dochází ke špatné identifikaci objektu, což zabraňuje ještě většímu využití metod strojového učení a umělé inteligence.

Jedním z příkladů pokročilejších algoritmů může být tzv. *You Only Look Once* (zkráceně YOLO) detektor objektů. Jeho využití představuje kombinaci OpenCV knihovny a strojového učení. [28]



Obr. 20: Schéma YOLO detektoru - upraveno [28]

YOLO detektor provádí detekci objektu na základě konvoluční sítě využívající matematickou regresní analýzu. Na jejím základě pak vytvoří tzv. *bounding box* (ohraničující obdélníky), pro které vypočítá pravděpodobnost, že se skutečně jedná o daný objekt. Pokud tato vypočtená pravděpodobnost překročí nastavenou hranici, dojde k vykreslení konkrétního ohraničujícího obdelníku a k identifikaci objektu. [28]

Výhoda tohoto detektoru spočívá především v jeho rychlosti, kdy je schopen dosáhnout až 150 snímků za sekundu při přijatelné chybě. V praxi bývá velmi často využíván v rámci bezpečnostních kamer, popřípadě kamer monitorujících vozidla na dálnicích. Stejně tak může však být využit i v technické praxi, například při kontrole kvality výrobku.

Tento příklad demonstruje, jakých výstupů je OpenCV knihovna ve spojení s prvky strojového učení, popřípadě umělé inteligence, v současnosti schopna dosáhnout. Pro samotný návrh algoritmu na zpracování obrazu však prozatím nebude využito prvků

strojového učení a daný algoritmus bude vyvíjen pouze na základě možností OpenCV knihovny.

5 NÁVRH ŘÍDÍCÍHO PROGRAMU PRO DETEKCI OBJEKTU

5.1 Příprava Raspberry Pi 3B+

Aby mohlo RPi sloužit jako řídící deska pro program zpracovávající dát z kamery, musí následovat několik konkrétních kroků a modifikací jeho softwaru. Tyto modifikace spočívají v rozšíření RPi o konkrétní programovací jazyk a další knihovny a balíčky, jež dohromady umožní z RPi vytvořit funkční platformu pro zpracování obrazu na základě dat pocházejících z PiCamery.

5.1.1 Volba programovacího jazyka

Po důkladném zvážení byl pro následující aplikaci využit programovací jazyk Python. Mezi nejvýznamnější faktory, které ovlivnily tuto volbu, patří dobrá čitelnost kódu, a především pak množství dostupných materiálů, neboť jazyk Python je v oblasti počítačového vidění nejčastěji používanou variantou. Z tohoto důvodu je tak dostupná větší komunita a také větší množství internetových i knižních zdrojů. Další výhodou, v porovnání například se softwarem MATLAB, je pak ekonomická stránka, kdy Python je šířen pod svobodnou licencí a v kombinaci se správnými rozšířeními dokáže nahradit software MATLAB, při prakticky nulových nákladech. [30]

Python je víceúrovňový programovací jazyk, který klade důraz na čitelnost kódu a podporuje rozdílné způsoby programování. Od ostatních programovacích jazyků se odlišuje rozsáhlou knihovnou, na kterou navazují další rozšíření a balíčky. Cílem jazyku Python je poskytnout výkonný programovací jazyk, který bude zároveň snadno čitelný a přehledný. [29, 32]

Obr. 21: Logo Python [30]

Programovací jazyk Python byl poprvé představen v roce 1991 Guido van Rossem. Navazoval na programovací jazyk ABC, který vynikal především v kombinaci s operačním systémem Amoeba, jehož struktura však byla příliš nepřehledná a těžko čitelná. [29]

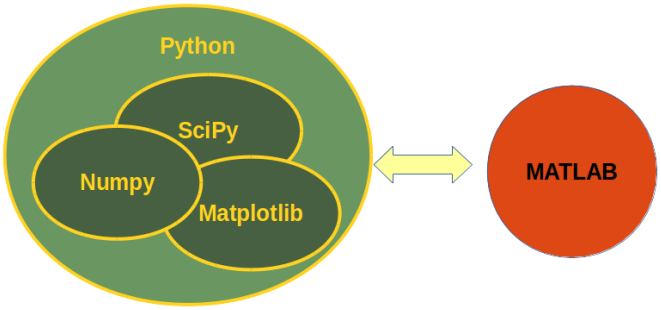
Na původní verzi Pythonu navázal v roce 2000 Python 2.0, který umožnil větší podporu od komunity vývojářů. V prosinci 2008 pak byl představen Python 3.0, který se

zaměřoval na odstranění nadbytečných konceptů a modulů s cílem ponechat pouze jeden způsob řešení problému. I v současnosti je využíváno této verze, nejaktuálnější modifikace se pak nazývá Python 3.7. [29]

Při snaze o rozšíření základní verze Pythonu je uživatelsky velmi přívětivé využít tzv. *pip*. Jedná se o rozhraní pro instalaci balíčků a knihoven, kdy v několika řádcích kódu umožňuje rozšíření jazyku Python do požadované podoby.

5.1.2 Využívané knihovny

Pro využití RPi, jako řídící jednotky pro zpracování dat z kamery je v kombinaci s programovacím jazykem Python, využito především knihoven NumPy, SciPy a Matplotlib, které je možno nainstalovat právě pomocí příkazu pip. Všechny tyto rozšíření tak vytváří výsledný produkt, který se velmi podobá softwaru MATLAB, avšak je volně dostupný. [30]



Obr. 22: Podobnost se softwarem MATLAB [30]

Numerical Python (zkráceně NumPy) je knihovna programovacího jazyku Python, která umožňuje práci s vektory a vícedimenzionálními maticemi. Tato knihovna tak poskytuje velké množství matematických funkcí, například základní metody lineární algebry, nebo diskrétní Fourierovu transformaci. Pro zpracování obrazu je však velmi významná především možnost interpretovat obrázky pomocí vícedimenzionálních matic. Tato schopnost NumPy knihovny totiž umožňuje numerickou analýzu obrazu. [31, 32]

NumPy je volně dostupný software, který je napsán v programovacích jazycích Python a C. Navazuje na svého předchůdce Numeric, který byl ve vývoji od roku 1995. Název NumPy je využíván od roku 2006, kdy došlo ke sloučení několika knihoven Pythonu podobného charakteru do jedné. [31]

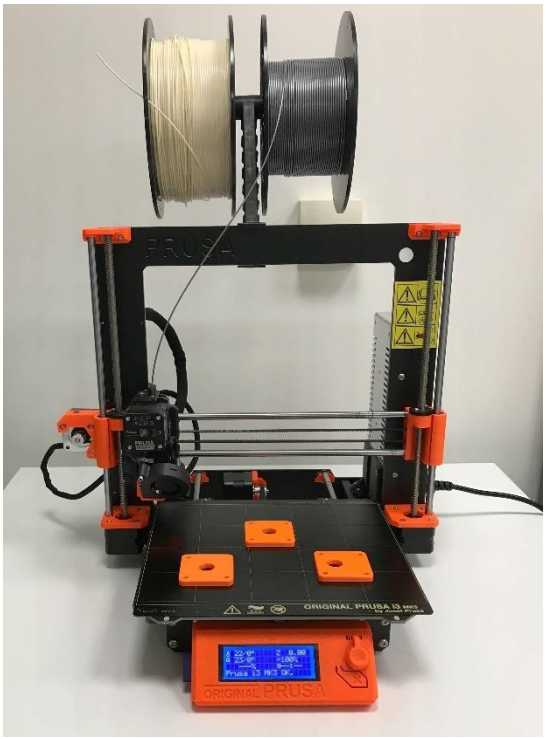
Další knihovna, vhodná pro zpracování obrazu, je tzv. *Scientific Python* (zkráceně SciPy). Knihovna SciPy, která stejně jako NumPy je šířena ve formě volně dostupného softwaru, poskytuje prostředky pro vědecké využití jazyka Python. Umožňuje tak například numerické vyčíslování integrálů, řešení diferenciálních rovnic, optimalizace, nebo využití tzv. řídkých matic, jež obsahují většinu nulových prvků a představují efektivnější způsob zapisování matic do paměti počítače. [31, 32]

Knihovny NumPy i Scipy však postrádají vykreslovací schopnost. Pro jazyk Python existuje několik balíčků, které tuto funkci poskytují. Nejčastěji využívaným je

pak tzv. Matplotlib. Tato 2D vykreslovací knihovna pro jazyk Python byla poprvé publikována v roce 2003 Johnem D. Hunterem a v současnosti se vyznačuje velmi aktivní vývojářskou komunitou. [32]

5.2 Návrh testovacího objektu

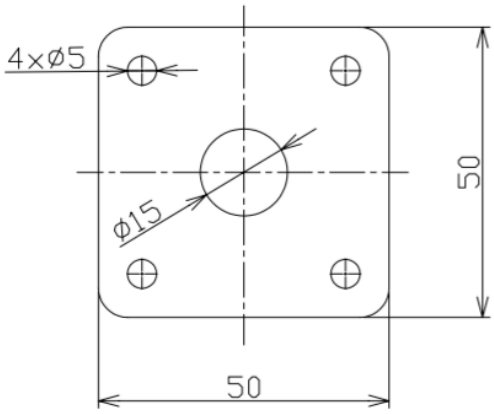
Pro reálné ověření funkčnosti navrhovaného řídícího kódu pro detekci špatně vyrobených součástí, bylo navrženo několik verzí testovacích objektů. Při jejich návrhu bylo dbáno na jejich jednoduchost a snadnou vyrobitelnost, aby tyto objekty mohly být vytištěny na 3D tiskárně. Pro 3D tisk byla využita tiskárna Prusa i3 mk3 (na obr. 23). Pro tvorbu součástí byl využit materiál PLA.



Obr. 23: Tiskárna Prusa i3 mk3

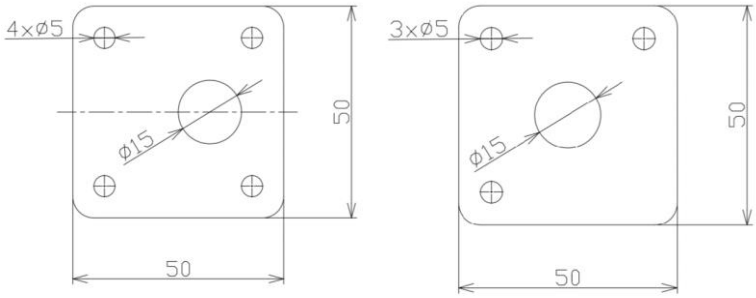
Návrh součástí byl proveden v kolaboraci s vedoucím práce a také s přihlédnutím k práci dalšího studenta [42]

Samotný návrh tvaru a rozměrů testovacího objektu tak byl proveden s přihlédnutím k rozlišovací schopnosti kamery, předpokládané výšce kamery a případnému využití v průmyslové výrobě. Pro správně vyrobený testovací objekt byla stanovena následující geometrie.



Obr. 24: Návrh správně vyrobené součásti

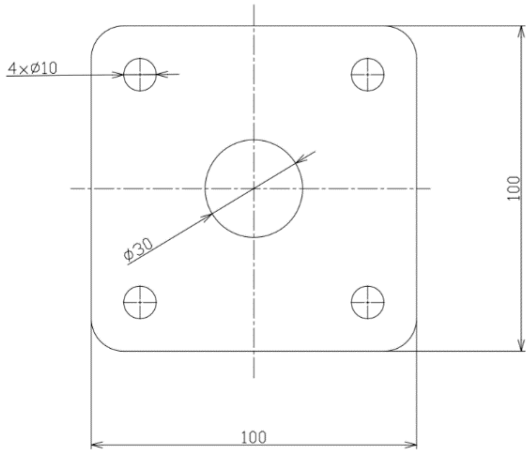
Následně byly navrženy dvě varianty špatně vyrobených součástí, z nichž jedna je opravitelná. Součást na obr. 25 vlevo má špatně vyvrtanou středovou díru, což ji dělá neopravitelnou. Součást vpravo pak nemá vyvrtané všechny díry. Tato součást je opravitelná.



Obr. 25: Návrhy špatně vyrobených součástí

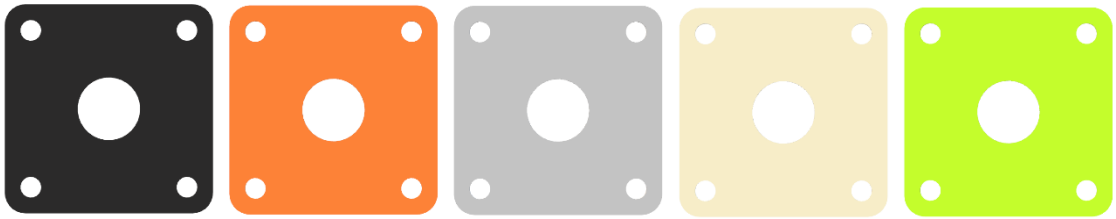
Cílem řídícího kódu pro zpracování dat z kamery a detekci objektu tak bude identifikaci chybně vyrobených součástí.

Vedlejším cílem pak bude experimentální ověření závislosti velikosti testovacího objektu na kvalitě detekce řídícím kódem. Pro tyto účely tak byla vytvořena jedna série testovacího objektu, o jiných rozměrech. Tyto rozměry jsou znázorněny na obr. 26.



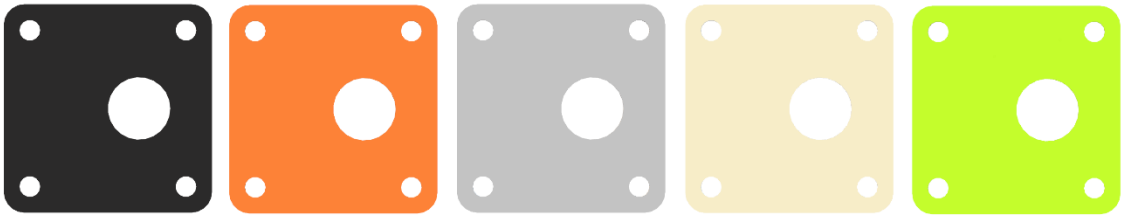
Obr. 26: Návrh zvětšené součásti

Dalším cílem při návrhu testovacího objektu, bylo porovnání účinnosti detekce pro několik základních barev. Tyto barvy byly úmyslně vybrány tak, aby pokrývaly většinu barevného spektra a umožnily tak vzájemné porovnání a stanovení nejvhodnějších barev. Po dohodě s vedoucím práce bylo stanoveno pět základních barev – černá, oranžová, šedá, bílá a zelená. Tyto testovací objekty jsou zobrazeny na obr. 27. V následujícím textu budou správně vyrobené testovací součásti souhrnně označovány jako *OK*.



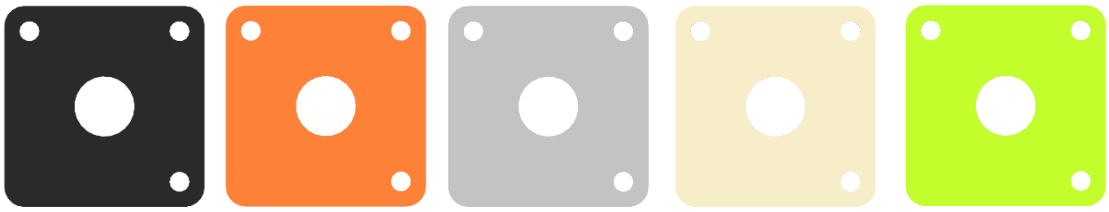
Obr. 27: Správně vyrobená součást

Ve stejném barevném provedení pak byly vytisknuty také součásti, které mají špatně umístěnou středovou díru. Znázorněny jsou na obr. 28. V dalším textu bude součást s touto vadou souhrnně označována jako *BAD*. Tato vada je neopravitelná.



Obr. 28: Součást typu *BAD*

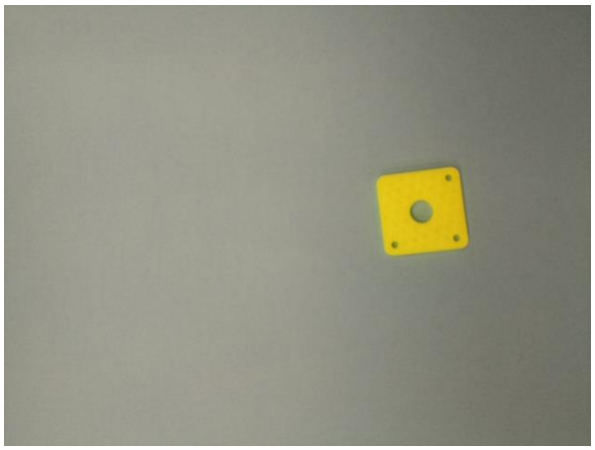
Poslední variantou jsou pak součásti, jež mají správně umístěnou středovou díru, ale nemají vyvrtanou jednu malou díru. I tyto součásti byly vytištěny v pěti barevných kombinacích. Znázorněny jsou na obr. 29. V dalším textu budou tyto součásti souhrnně označovány jako *REPAIR*. Tato vada je opravitelná, chybějící díru je možno vyvrtat.



Obr. 29: Součást typu REPAIR

Velmi významnou součástí práce byla také volba pozadí. Navrhnuté součásti zahrnují velmi široké spektrum barev, a proto bylo nutné najít dostatečně kontrastní barvu pozadí. Po zvážení několika variant byla nakonec vybrána světle modrá barva.

Na základě právě zmíněných údajů tak byla ve spolupráci s vedoucím práce vytvořena trénovací množina obrázků, určená k ověření funkčnosti řídícího kódu, jež bude následně také otestován v laboratoři. Tato množina je k práci přiložena v příloze A.



Obr. 30: Ukázka trénovací množiny obrázků

5.3 Příprava obrázku pro detekci

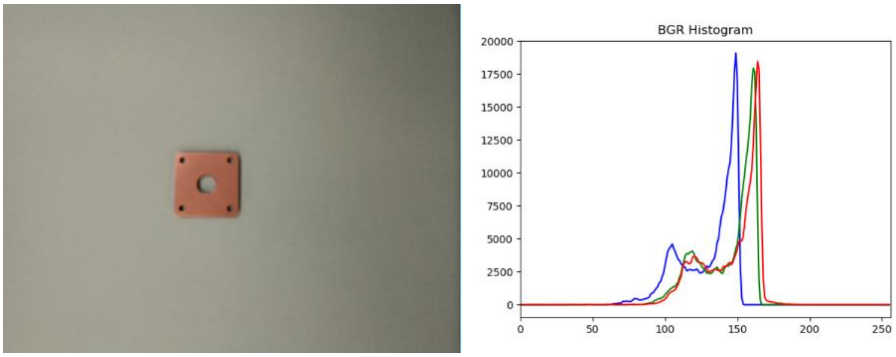
Aby mohl řídící program provést detekci součásti, musí být zachycený snímek upraven do stavu, kdy jsou maximalizovány rozlišovací schopnosti řídícího kódu. Pro tento účel je tak snímek podroben uplatnění několika filtrů a úprav, jež umožní detekovat správnost vyrobené součásti. V této podkapitole tak budou postupně představeny čtyři hlavní způsoby úpravy zachyceného snímku.

5.3.1 Konverze barvy

Elementárním prvkem každého obrázku jsou pixely. Tyto nejmenší bezrozměrné jednotky digitální grafiky lze v obrázku identifikovat na základě jejich polohy. Pixelů je využíváno také při stanovení velikosti obrázku. Každý pixel má také přiřazenu právě jednu barvu a barevné provedení obrázku tak vzniká na základě sloučení velkého množství barevných pixelů do jednoho celku.

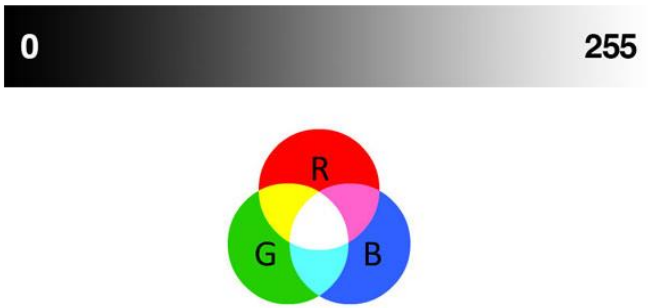
Základní vyjádření barev pro OpenCV knihovnu je pomocí tzv. RGB (*Red, Green, Blue*). Jelikož však byla OpenCV knihovna vyvíjena v období, kdy standartní vyjádření barev bylo ve formátu BGR, namísto RGB, je tato konvence stále dodržována a základní vyjádření barev pro OpenCV knihovnu je tak BGR. [32, 33]

Při této reprezentaci je tak barva každého pixelu reprezentována maticí (B, G, R), kde je definována hodnota jednotlivých složek, ze kterých je následně složena výsledná barva pixelu. Každá složka matice (B, G, R) pak může nabývat hodnot od 0 do 255. Toto rozdělení je znázorněno na histogramu na obr. 31. [33]



Obr. 31: Histogram barev BGR

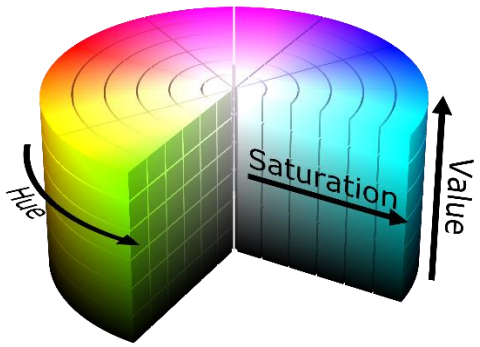
Pro některé filtry obsažené v OpenCV knihovně je však výhodnější pracovat s obrázkem reprezentovaným pouze dvěma barvami. V těchto případech se uplatňuje reprezentace barev pomocí tzv. *grayscale* (česky odstínů šedi). V tomto případě je tak hodnota každého pixelu vyjádřena pomocí jednoho čísla v rozmezí 0 a 255, kde nula reprezentuje černou a 255 bílou, obr. 32.



Obr. 32: Odstíny šedi (nahoře) a RGB barvy [33]

Pro potřeby počítačového vidění a zpracování obrazu je také často výhodné užití vyjádření barev pomocí tzv. HSV (*Hue Saturation Value*). V tomto případě je výsledná barva vyjádřena maticí (H, S, V).

Hue (česky odstín) je jediná složka, která popisuje barvu. Její hodnota odpovídá úhlu na barevném kole, zobrazeném na obr. 33. *Saturation* (česky nasycení) je vyjadřováno v procentech a definuje nasycení jednotlivou barvou. *Value* (česky hodnota) pak specifikuje množství jasu v barvě. [34]



Obr. 33: HSV barvy [34]

Tento model vyjádření barev je bližší lidskému oku ve způsobu vnímaní a identifikace barev a také samostatně odděluje odstín. Především z těchto důvodů a více intuitivní reprezentace barvy, je často využíván při zpracování obrazu.

Obr. 34: Zleva BGR, odstíny šedi, HSV

5.3.2 Rozmazání

Blurring (česky rozmazání) je efekt, při kterém je hodnota intenzity pixelu upravena v závislosti na hodnotách okolních pixelů. Tento efekt má ve většině oblastí negativní charakter. V oblasti zpracování obrazu je však často záměrně využíván, neboť napomáhá docílit přesnějších výsledků, odstraněním šumu.

Jednou z využívaných metod je tzv. *averaging* (česky metoda průměrování). Tato metoda spočívá v průměrování hodnoty prostředního pixelu na základě 8 okolních pixelů. V rovnici 1 je hodnota tohoto pixelu pro přehlednost zvýrazněna modře. [35]

$$K = \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \tag{1}$$

Na obdobném principu jako metoda průměrování, funguje i jedna z nejpoužívanějších metod, zvaná *Gaussian blurring* (česky Gaussovo rozmazání). Při této metodě však namísto průměrování hodnot dochází k využití Gaussovi distribuce. V praxi bývá právě tato metoda využívána pro úpravu snímků před dalším zpracováním. Příklad využití Gaussova rozmazání je zobrazen na obr. 35. [35]

Obr. 35: Využití Gaussova rozmazání

5.3.3 Thresholding

Další možností úpravy snímku je tzv. *thresholding* (česky prahování), který zjednodušuje data pro analýzu a napomáhá odstranění šumu. Vstupní snímek je často vyveden v odstínech šedi, což umožňuje nejjednodušší variantu thresholdingu zvanou binární thresholding. Binární thresholding spočívá v porovnání hodnoty každého pixelu snímku s hraniční hodnotou, nazývanou hodnota thresholdu. Toto porovnání je souhrnně vyjádřeno rovnicí 2. Pokud je hodnota pixelu menší, než hodnota thresholdu, je vykreslena černá barva (její hodnota v BGR je 0). Pokud je naopak hodnota rovna, nebo větší, je vykreslena bílá barva (její hodnota v BGR je 255). [32, 36]

$$f(x) = \begin{cases} 0 & pokud\ x < hodnota\ thresholdu \\ 255 & pokud\ x \geq hodnota\ thresholdu \end{cases} \tag{2}$$

Thresholding je tedy funkcí, jež umožňuje filtrování snímku na základě informací uložených v jednotlivých pixelech. Dále se také jedná o efektivní variantu odstranění šumu a identifikace oblasti zájmu a umožňuje tak zvýraznění objektů v popředí. Praktická aplikace tohoto způsobu filtrování je zobrazena na obr. 36.

Obr. 36: Využití thresholdingu

5.3.4 Nalezení hran a stanovení gradientu

Jednou z nejdůležitějších úprav snímku pro detekci součásti je nalezení hran (tzv. *edge detection*). Nalezení hran na obrázku spočívá ve využití matematických metod k nalezení bodů snímku, kde se skokově mění hodnoty jasu. Tento postup tak umožnuje identifikaci důležitých oblastí snímku a výrazně snižuje množství dat, která musí být zpracována.
Pro účely této práce je využito tzv. *Canny edge detection* (česky Cannyho detektoru hran).

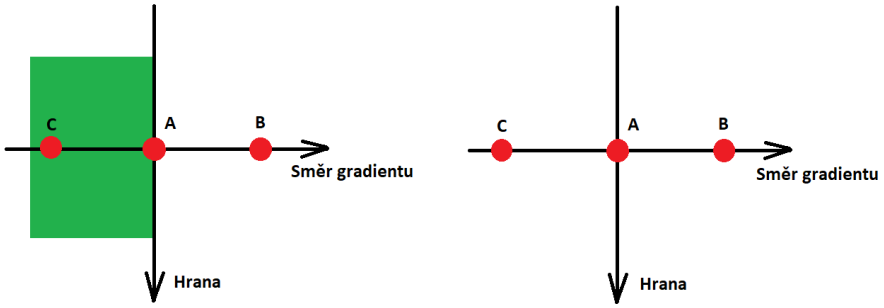
Cannyho detektor hran je vícekrokový algoritmus, který je nejčastěji využíván při hledání hran obrázku.

Jelikož je detekce hran velmi náchylná k šumu, prvním krokem Cannyho algoritmu je právě redukce šumu. V praxi bývá často využito Gaussova filtru (tzv. *Gaussian filter*) o rozměrech 5x5. Takto upravený snímek je následně filtrován v horizontálním i vertikálním směru pomocí Sobelova operátoru. Výsledkem této úpravy je první derivace v horizontálním směru (G_x) a vertikálním směru (G_y). Na základě těchto derivací je podle rovnic 3,4 stanoven gradient, který je vždy kolmý k hranám. Úhel je pak zaokrouhlen do horizontálního, vertikálního a dvou diagonálních směrů. [32, 37]

$$gradient(G) = \sqrt{G_x^2 + G_y^2} \tag{3}$$

$$úhel(\theta) = tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{4}$$

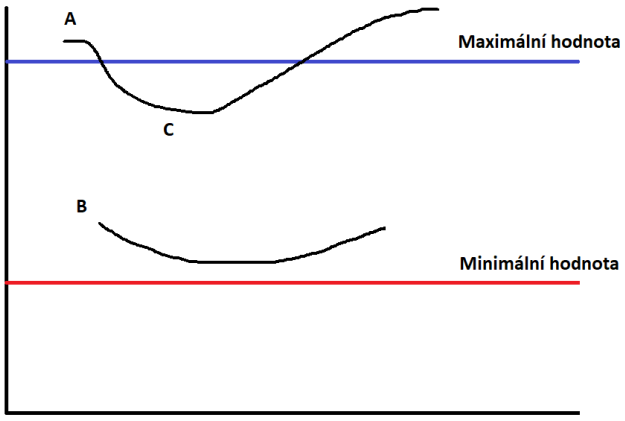
Po stanovení velikosti a směru gradientů dochází k odstranění pixelů, které netvoří hrany. K tvorbě hran dochází na základě stanovení hodnot lokálního maxima v okolí pixelu ve směru gradientu. Tento proces je znázorněn na obr. 37.



Obr. 37: Tvorba hran – upraveno [37]

Dochází tak k ověření, zda bod A je v porovnání s body B a C lokálním maximem. Pokud je tato podmínka splněna, je bod zachován. V opačném případě je potlačen a v dalších fázích se již nevyskytuje. Tímto procesem jsou vytvořeny hrany. [37]

V posledním kroku je následně rozhodnuto, zda budou hrany vykresleny. Určujícím faktorem jsou maximální a minimální hodnota. Pokud je hodnota intenzity gradientu menší než minimální hodnota, algoritmus ji neuvažuje jako hranu a nevykreslí ji. Hrany s hodnotou gradientu větší, než maximální hodnota jsou identifikovány jako „jisté“ hrany a vykresleny. Hrany, jež se nachází mezi těmito hodnotami, jsou vykresleny jen v případě, že jsou spojeny s hodnotami „jistých“ hran. Tento postup je ilustrován na obr. 38 [37]



Obr. 38: Rozhodující faktory pro vykreslení hran – upraveno [37]

V této fázi dochází také k odstranění malých hodnot šumu, neboť nesplňují definici hrany, jakožto dlouhé čáry.

Výsledkem využití Cannyho detektoru je tak vykreslení hran součásti, které slouží jako určující prvky pro vznik kontur, které budou rozebrány v další podkapitole. Na obr.39 je pak zobrazeno využití Cannyho detektoru na snímku z trénovací množiny obrázků.

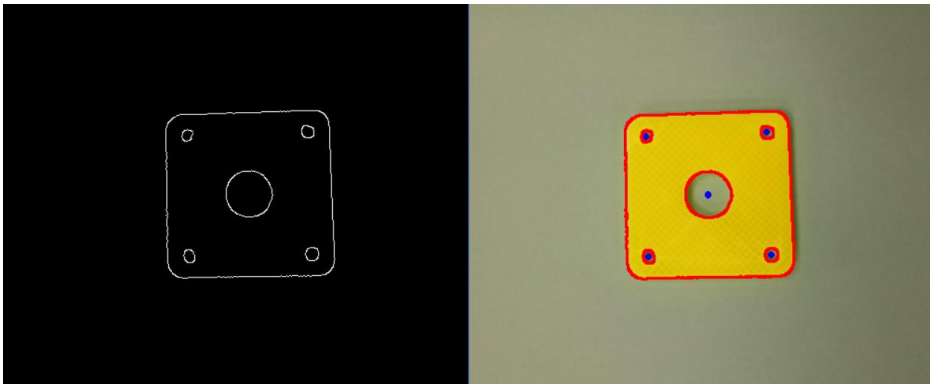
Obr. 39: Příklad využití Cannyho detektoru

5.4 Nalezení kontur

Kontura je křivka, která propojuje všechny body stejné barvy nebo intenzity. V oblasti zpracování obrazu se jedná o užitečný nástroj při analýze tvarů, detekci objektů a jejich následné rozpoznávání. Pro úspěšné vykreslení kontur je nutné mít vstupní obrázek upraven pomocí *thresholdingu*, nebo Cannyho detektoru, aby byl dostatečně kontrastní přechod barev. [38]

Přítomnost kontur na snímku pak umožňuje stanovení počtu geometrických útvarů, jejich tvaru, nebo měření jejich vzájemné vzdálenosti. Právě tyto vlastnosti bývají nejčastěji využity při samotné detekci objektu. [32, 38]

Samotná kontura je vykreslena pomocí knihovny NumPy jako matice souřadnic bodů. Tento postup je vizuálně zobrazen na obr. 40, kde jsou kontury vykresleny na základě Cannyho detektoru. Samotné kontury jsou vykresleny červenou barvou, jejich středy jsou znázorněny modrými body.



Obr. 40: Vykreslení kontur

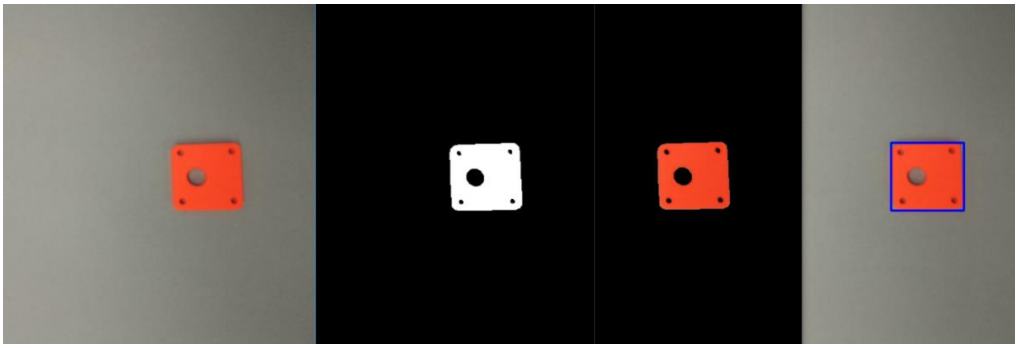
Vykreslování kontur a detekce hran jsou však také oblasti, které jsou nejvíce náchylné na přítomnost šumu. K nejčastějším případům chyby řídícího kódu tak dochází

především jako následek špatného osvětlení, jež vede ke špatnému odrazu světla a tvorbě stínu.

5.5 Detekce na základě barvy

Princip detekce na základě barvy spočívá v prokládání obrázku pomocí předem vytvořených masek jednotlivých barev. Následně je tak detekována oblast, kde je tato barva přítomna. Pro detekci barvy je výhodnější užití vyjádření barev HSV.

Před samotnou detekcí je nutné stanovení masky. Tato maska je stanovena jako barevné rozmezí, pro které má být barva detekována, jako například oranžová. Pro každou z pěti barev testovacího objektu je nutné stanovit vlastní rozmezí. Pokud tak maska na vstupním obrázku zachytí pixely v rozmezí nadefinovaném pro barvu, dojde k jejich vykreslení. Výsledný obrázek je v binárním formátu, tedy jen bílá nebo černá. Příklad vykreslení této masky je na obr. 41, jedná se o druhý obrázek zleva. Algoritmus tak získá definovanou oblast zájmu. Následně dojde ke složení vstupního snímku a masky, kde pouze oblast zájmu je vykreslena na základě vstupního snímku, tato situace je zobrazena na stejném obrázku, v pořadí třetí zleva. Tato oblast je tak algoritmem definována, jako oranžová barva a tento výstup je vykreslen do původního obrázku.



Obr. 41: Detekce na základě barvy

Tato část řídícího algoritmu je však velmi citlivá na správné osvětlení. Při změně osvětlení se také mění barevné hodnoty pixelů, což může způsobit špatnou identifikaci barvy.

5.6 Detekce špatně vyrobené součásti

Samotný princip detekce špatně vyrobené součásti se bude lišit v závislosti na typu součásti (zda se jedná o typ *BAD*, nebo typ *REPAIR*). Pro oba typy je však shodná úprava snímku, před samotnou detekcí. V praxi je tak využito konverze barvy, Gaussova rozmazání, *thresholdingu* a Cannyho detektoru.

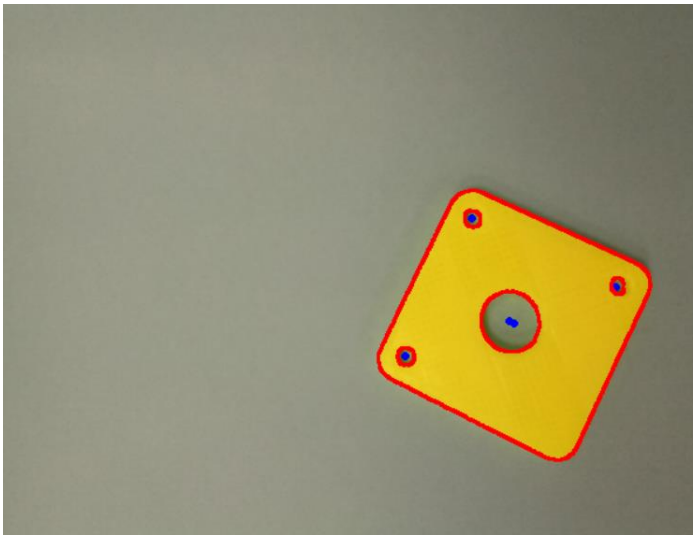
5.6.1 Typ *REPAIR*

Při posuzování součásti *REPAIR* je využito znalosti počtu kontur.

Zachycený snímek je tak upraven a následně jsou vykresleny kontury, na obr. 42 červeně. Tyto kontury jsou následně aproximovány, takže například kontura mající tvar elipsy, je zobrazena s menší excentricitou a je tak více podobná kruhu.

Pro tyto kontury jsou následně vypočteny středy kontur, na obr. 42 znázorněny modrými body.

Tyto body jsou následně uloženy do příslušné datové struktury a na základě počtu prvků v této datové struktuře je rozhodnuto, zda obsahuje součást všechny díry.



Obr. 42: Detekce součásti typu REPAIR

Pro zabránění špatné interpretace výsledků jsou využity jen ty kontury, jejichž velikost a lokace odpovídá přibližným rozměrům součásti.

5.6.2 Typ *BAD*

Pro detekci špatně vyrobených součástí typu *BAD* je využito schopnosti OpenCV knihovny měřit vzdálenost mezi dvěma body.

Pro stanovení vzdálenosti mezi dvěma body je nutné definovat referenční rozměr, ke kterému se následně vztahují požadované vzdálenosti. Rozměry referenční vzdálenosti musí být známy, a také musí být jednoznačně definovány na obrázku. [39]

Pro účely této práce je jako referenční vzdálenost zvolena šířka součásti. Na obr. 43 je zvýrazněna bíle. Jelikož má testovací objekt tvar čtverce, není nutné rozlišovat mezi šířkou a výškou součásti a jako referenční tak mohou být využity obě.

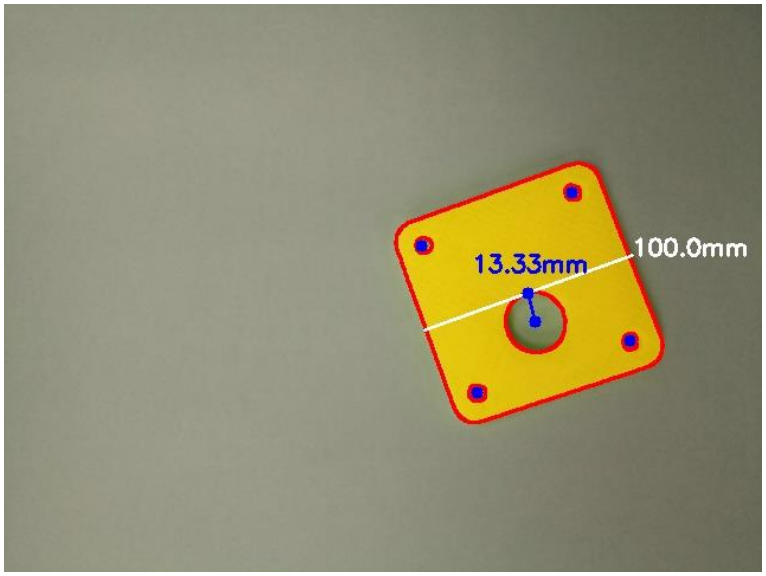
Této referenční vzdálenosti (v milimetrech) odpovídá v obrázku tzv. *Euclidean distance* (česky eukleidovská metrika). Jedná se o stejnou vzdálenost, jako referenční, ale vypočtenou na základě vzdáleností pixelů.

Pro stanovení vzdálenosti mezi libovolnými dvěma body se pak využije poměru, tzv. *pixels per metric* (česky pixel na vzdálenost). Tento poměr se stanoví na základě

poměru *euclidean distance* a známé vzdálenosti. Zjednodušeně se tak dá říct, že se jedná o vzdálenost pixelů, podělený skutečnou vzdáleností. [39]

$$poměr = \frac{euclidean\ distance}{známá\ vzdálenost} \tag{5}$$

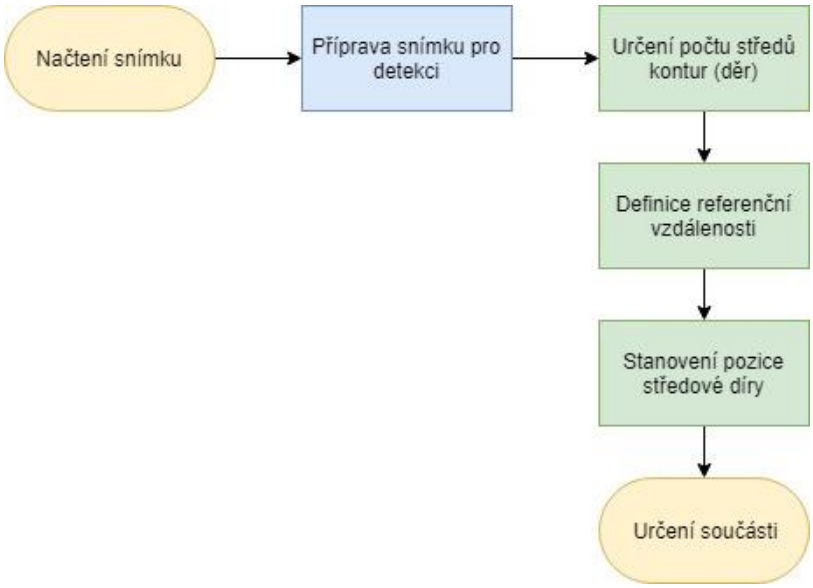
Tohoto poměru se následně využije pro výpočet vzdálenosti mezi dvěma body. V tomto případě se jedná o bod středu součásti a středu vyvrtané velké díry. Na obr. 43 znázorněno modrou barvou.



Obr. 43: Detekce součásti typu *BAD*

Tento postup tak byl využit pro stanovení součásti typu *BAD*, kdy platí, že pokud je tato naměřená vzdálenost mezi body větší než přípustná odchylka způsobená zobrazovacími vadami, je součást zhodnocena jako vadná.

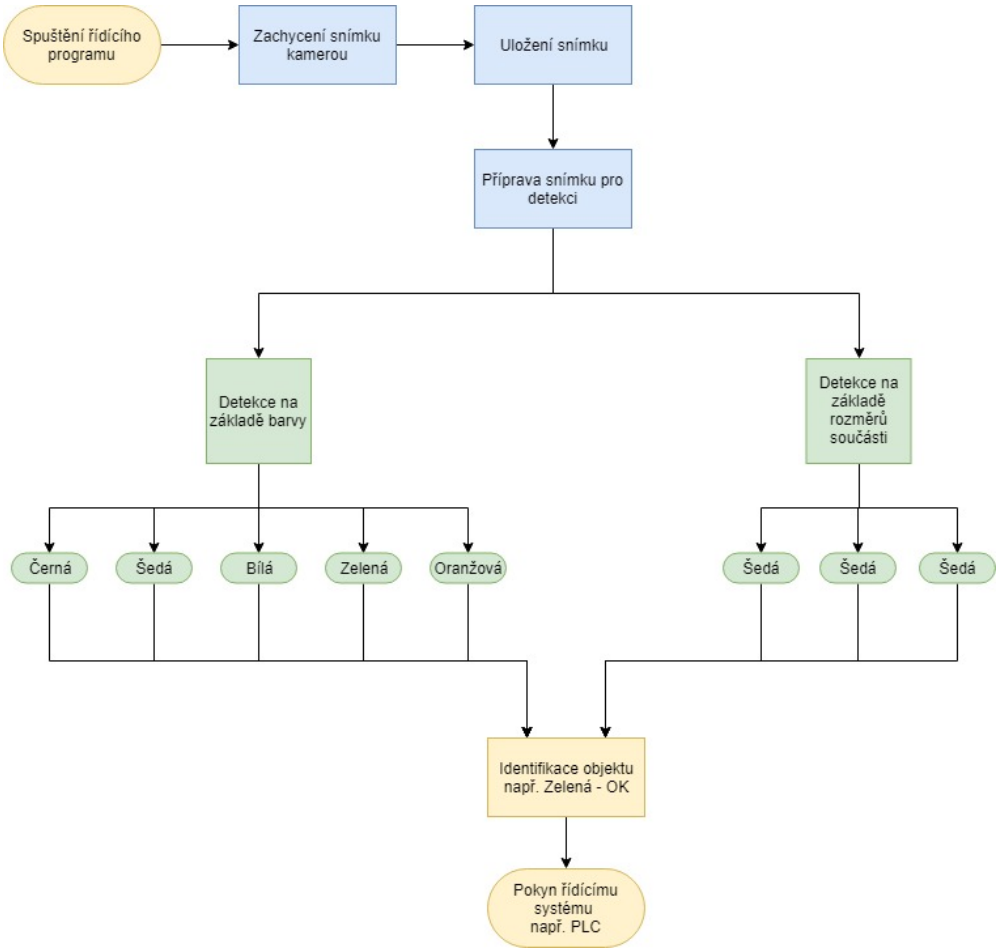
Souhrnně je pak celý princip posuzování správnosti vyrobených součástí uveden na vývojovém diagramu na obr. 44. Řídící program zabývající se detekcí je pak přiložen v příloze A.



Obr. 44: Vývojový diagram detekce špatně vyrobené součásti

5.7 Navržený řídící kód

Souhrnný vývojový diagram řídícího kódu, spojující detekci na základě barvy a detekci správně vyrobené součásti, je zobrazen na obr. 45. Samotný kód je pak přiložen v příloze A.



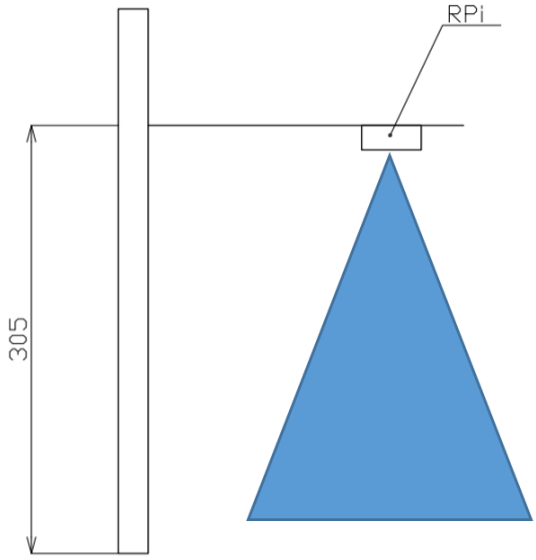
Obr. 45: Vývojový diagram navrženého řídícího kódu

6 OVĚŘENÍ FUNKČNOSTI NAVRŽENÉHO ŘÍDÍCÍHO PROGRAMU

Pro ověření návrhu funkčnosti řídícího kódu byl vytvořen stojan a následně byla otestována funkčnost navrženého řídícího algoritmu. Další částí této kapitoly je návrh uživatelského rozhraní, jež napomáhá přehlednější identifikaci výsledků detekce a v průmyslové praxi by umožňovala efektivnější vzdálený přístup pracovníka. Stavba stojanu i tvorba uživatelského rozhraní proběhla ve spolupráci s vedoucím této práce.

6.1 Návrh stojanu

Aby mohla být otestována funkčnost řídícího algoritmu, bylo nutné navrhnout stojan pro uchycení RPi a stabilizaci PiCamery. Návrh tohoto stojanu je zobrazen na obr.46. Praktická realizace je pak zobrazena na obr. 47.



Obr. 46: Návrhový výkres stojanu

Obr. 47: Vlevo realizace stojanu, vpravo detail uchycení RPi

Při samotném návrhu hrála důležitou roli především výška uchycení RPi s PiCamerou, neboť tato výška určuje velikost zobrazení součásti. Po několika experimentech byla tato výška stanovena na 30,5 cm.

Při samotné detekci je pak využíváno pouze kancelářské světlo v místnosti. Žádná další osvětlení přítomna nejsou.

6.2 Komunikace

Při ověřování funkčnosti byly využity výrobky společnosti B&R. Tato rakouská společnost s pobočkou v Brně se specializuje na automatizaci průmyslových procesů. Do jejich portfolia tak patří například PLC, průmyslové počítače, servomotory nebo části inteligentních výrobních linek. V roce 2017 byla společnost B&R převzata společností ABB. [40]

Obr. 48: B&R logo [40]

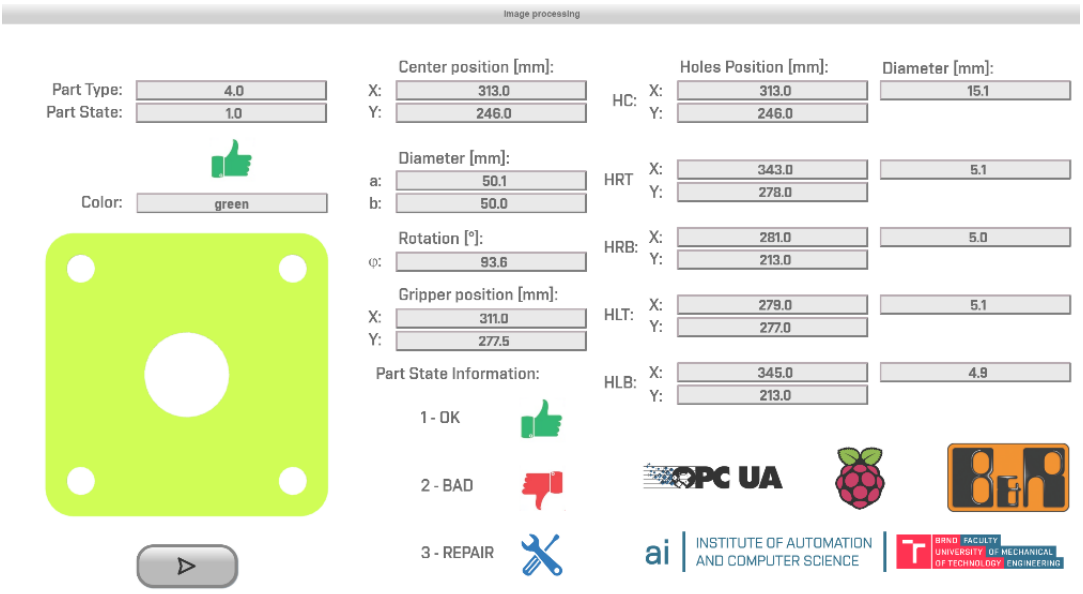
PLC (*Programmable Logic Computer*, česky programovatelný logický automat), je zařízení využívané pro automatizaci procesů. PLC získá informaci z připojených zařízení a senzorů, tuto informaci zpracuje a na základě předem naprogramovaného algoritmu vykoná výstup. Při reálném ověřování bylo využito PLC X20CP1584 od společnosti B&R. [40]

Obr. 49: Využité PLC [40]

Při komunikaci mezi jednotlivým zařízeními je využíváno komunikačního protokolu OPC UA (*OPC Unified Architecture*). Tento protokol byl vyvinut na základě

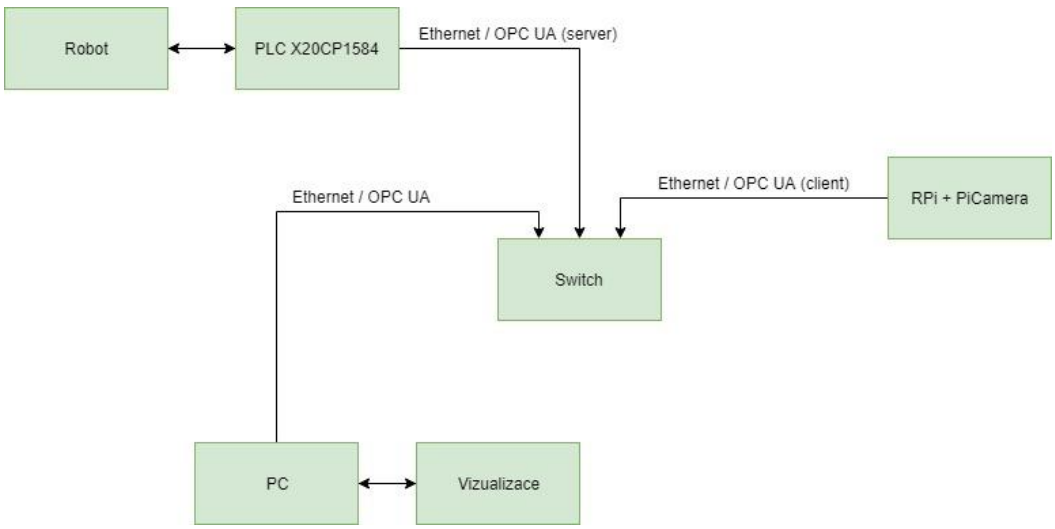
OPC Classic a umožňuje bezpečnou a spolehlivou výměnu dat v oblasti průmyslové automatizace. [41]

Při tvorbě samotné vizualizace pak byla využita technologie společnosti B&R zvaná *mappView*. Pomocí této technologie společnost B&R umožňuje třetím stranám přístup k webovým technologiím, využitelným pro automatizační aplikace. Její nevýhodou jsou však velké požadavky na procesor počítače. [41]



Obr. 50: Příklad vizualizace

Souhrnně je pak celý proces komunikace vyjádřen následujícím vývojovým diagramem.



Obr. 51: Vývojový diagram

6.3 Zhodnocení závislosti barvy na detekci

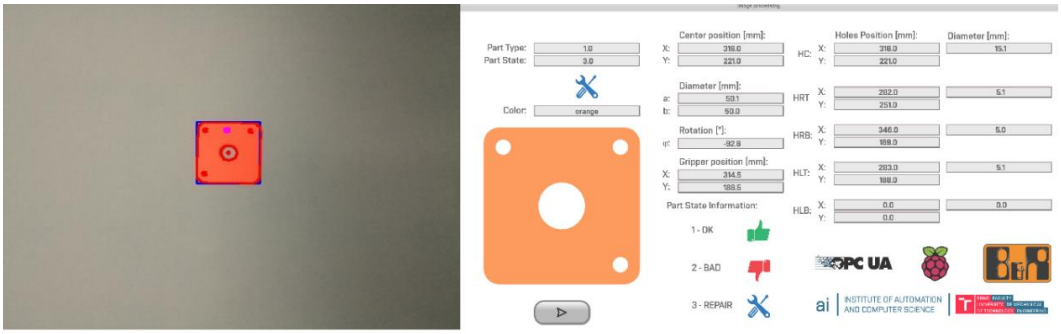
Při reálném ověření detekce barvy bylo dosaženo následujících výstupů.

Šedá barva součásti se při současných osvětlovacích podmínkách jeví velmi podobně jako barva pozadí. Z tohoto důvodu má řídící algoritmus při její detekci poměrně velké množství nepřesných výstupů. Šedá barva však také není ideální při detekci špatně vyrobených součástí, neboť malé díry na okrajích součásti neodrazí dostatek světla, které je pohlceno zbylou plochou šedé součásti.

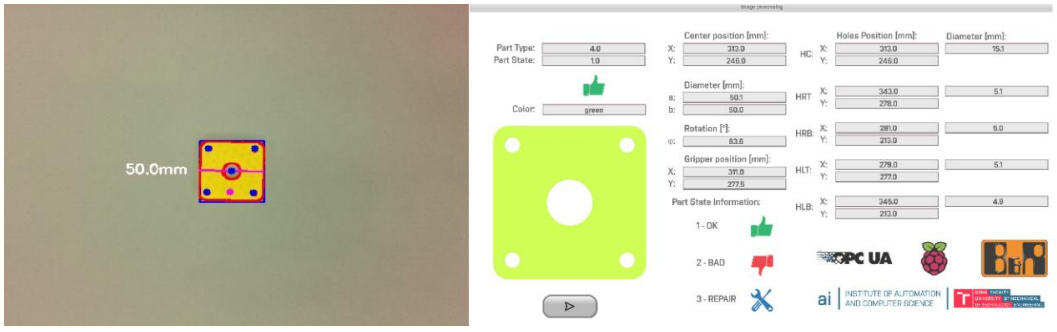
Při současné konfiguraci není ideální ani volba bílé součásti. Detekce těchto součástí je totiž velmi závislá na intenzitě a úhlu osvětlení. Bílá barva velmi dobře odráží světlo, což však není příliš výhodné v oblasti počítačového vidění. Z důvodu nadměrného množství odrazů tak vzniká velké množství šumu a stínů, které detekci komplikují.

Nejhorších výsledků detekce však bylo dosaženo pro černou součást. Tato barva již ze své podstaty pohlcuje světlo, a proto nedocházelo k dostatečnému odrazu světla v malých dírách a řídící algoritmus tak nenacházel dostatečně výrazné přechody pro vykreslení těchto kontur.

Ze souboru testovaných objektů tak bylo dosaženo nejlepších výsledků pro zelenou a oranžovou součást. Obě tyto barvy se velmi odlišují od barvy pozadí, a proto jsou tyto barvy řídícím kódem poměrně přesně detekovány. Při detekci správnosti vyrobené součásti poskytují obě barvy dostatečně výrazné přechody, čímž umožňují kvalitní vykreslení kontur a s tím související detekci. Konkrétní příklady detekce pro součásti těchto barev včetně vizualizací výstupu jsou znázorněny na obr. 51 a obr. 52.



Obr. 52: Detekce oranžové součásti a vizualizace výsledků



Obr. 53: Detekce zelené součásti a vizualizace výsledků

Pro reálné využití v průmyslové praxi by však bylo výhodné dále upravit řídící algoritmus i hardwarovou konfiguraci. Samotná PiCamera poskytuje s přihlédnutím k ekonomické stránce dobré snímky, avšak pro reálné využití by bylo potřeba zahrnout kvalitnější, nejspíš USB kameru.

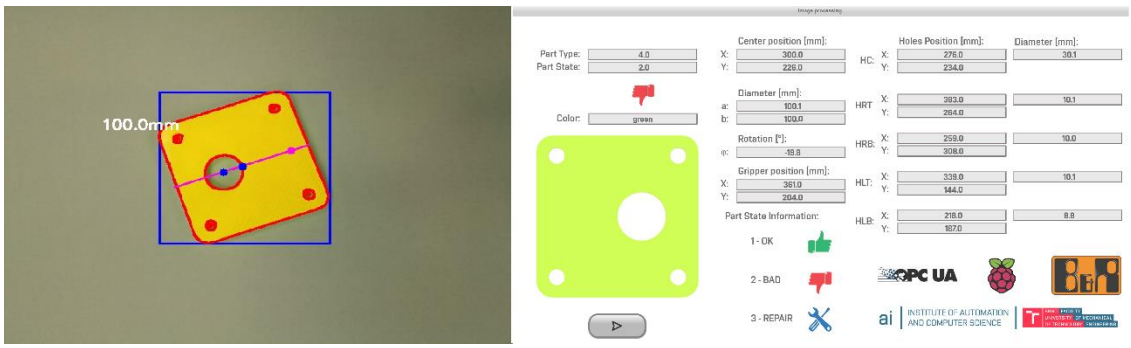
Další oblast, která má velký vliv na samotný proces detekce, je kvalita osvětlení. Současný návrh se touto oblastí nezaobírá a pracuje pouze při standartním kancelářském osvětlení. Ke zlepšení výsledků především u tmavých součástí by však mohlo dojít v důsledku zahrnutí bodového osvětlení, dostatečně silného na eliminaci okolního světla.

6.4 Zhodnocení závislosti velikosti součásti na kvalitě detekce

Vedlejším cílem ověření detekce bylo stanovení závislosti velikosti součásti na kvalitě detekce. Z tohoto důvodu byla vytištěna také jedna zelená série součástí o dvojnásobných rozměrech.

Na základě reálného testování bylo stanoveno, že větší rozměry součásti, při současné hardwarové konfiguraci, přispívají k lepší detekci, neboť tyto součásti mají rozměrově větší přechody mezi součástí a pozadím. Pro řídící algoritmus je tak snazší odlišit skutečnou kontury součásti od šumu pozadí. Příklad detekce je zobrazen na obr.53.

Velikost součásti nehraje významnou roli v oblasti detekce barvy.



Obr. 54: Detekce velké zelené součásti a vizualizace výsledků

Nevýhodou při tvorbě zvětšených součástí je však spotřeba materiálu, a především pak délka tisku. Z tohoto důvodu tak byla vytvořena jen zelená série součástí.

7 ZÁVĚR

V rámci bakalářské práce byl navržen řídící algoritmus pro zpracování dat z kamery a následnou detekci špatně vyrobených součástí. Tato povinná část práce byla následně rozšířena o dvě nepovinné části. Konkrétněji se jedná o testování závislosti velikosti součásti na kvalitě detekce a také návrh vizualizačního prostředí (obě kap. 6).

Prvním krokem při návrhu algoritmu bylo přizpůsobení Raspberry Pi 3B+ na počítač vhodný pro zpracování dat a detekci obrazu. Tato modifikace softwaru spočívala v instalaci OpenCV knihovny, programovacího jazyku Python a také jeho rozšíření. Následně byla zvolena možnost rozšíření Raspberry Pi o kamerový modul. Především z ekonomického hlediska byla zvolena PiCamera (kap. 3). Dalším krokem pak byl návrh testovacích objektů. Tyto součásti byly ve spolupráci s vedoucím práce vytisknuty na 3D tiskárně.

Druhý krok již zahrnoval samotný návrh řídícího kódu pro detekci špatně vyrobených součástí (kap. 5). Samotnou detekci součásti je možno rozdělit na dvě části. V prvé části byla součást detekována na základě barvy, jelikož testovací objekty byly vyhotoveny v pěti barevných kombinacích. V druhé části pak byla součást detekována na základě rozměrů a vzdáleností děr. Právě tato forma detekce ověřuje, zda je součást vyrobena správně. Výstupem tohoto řídícího kódu je pak barva součásti a rozhodnutí, zda je součást vyrobena správně nebo chybně (a zde se dá případně opravit).

Aby mohl být vzniklý řídící program otestován v laboratorním prostředí, byl ve spolupráci s vedoucím práce navržen stojan a také bylo navrženo vizualizační prostředí. Vzniklý řídící algoritmus tak byl následně otestován (kap. 6). Na základě tohoto testování bylo stanoveno, že algoritmus nejlépe detekuje součásti, jež mají výraznou barvu (např. zelená nebo oranžová). U ostatních navržených barev by mohlo dojít ke zlepšení výsledků aplikací přídavného osvětlení, neboť součásti tmavých barev (např. černá a šedá) pohlcují velké množství světla a neumožňují dostatečný odraz světla v dírách součástí.

Detekci špatně vyrobených součástí je závislá na podobných proměnných, jako detekce na základě barvy. U kontrastních součástí (zelená, oranžová) vykazuje navržený algoritmus jen malé množství nepřesných výsledků. Pro součásti ostatních barev však podíl chybných výsledků narůstá. I v tomto případě by mohla problém vyřešit úprava osvětlení, případně kvalitnější kamera.

V poslední části práce pak bylo stanoveno, že větší velikost součásti má pozitivní dopad na kvalitu detekce, neboť tato součást má výraznější přechody mezi součástí a pozadím. Řídící algoritmus tak snáze odliší konturu součásti od šumu.

8 SEZNAM POUŽITÉ LITERATURY

[1] ROJKO, Andreja. Industry 4.0 Concept: Background and Overview. *International Journal of Interactive Mobile Technologies*. 2017, **11**(5), 77-82.

[2] THE RISE OF BIG DATA AND INDUSTRY 4.0. In: *Trillium Network* [online]. 2017 [cit. 2019-05-19]. Dostupné z: http://trilliummfg.ca/the-rise-of-big-data-and-industry-4-0/

[3] CHEN, Yubao. Integrated and Intelligent Manufacturing: Perspectives and Enablers. *Engineering* [online]. 2017, **3**(5), 588-591 [cit. 2019-05-19]. Dostupné z: https://www.sciencedirect.com/science/article/pii/S2095809917307105

[4] *Blockseminar: The "Internet of Things" for industrial applications* [online]. In: . [cit. 2019-05-19]. Dostupné z: http://www.db.in.tum.de/teaching/ws1314/industrialIoT/

[5] BRADSKI, Gary a Adrian KAEHLER. *Learning OpenCV*. O´Reilly Media, 2008. ISBN 978-0-596-51613-0.

[6] *Kyberfyzikální systémy* [online]. 2016 [cit. 2019-05-19]. Dostupné z: https://www.iot-portal.cz/2016/08/22/kyberfyzikalni-systemy/

[7] *Chytrá továrna v Průmyslu 4.0* [online]. [cit. 2019-05-19]. Dostupné z: http://www.prumysloveinzenyrstvi.cz/chytra-tovarna-prumyslu-4-0/

[8] In: *Smart manufecturing* [online]. [cit. 2019-05-19]. Dostupné z: https://www.smartmanufacturinginitiative.com/2015/12/11/automation-is-king-in-the-next-wave-of-globalization/

[9] SILVER, David. [online]. [cit. 2019-05-19]. Dostupné z: https://www.linkedin.com/pulse/how-computer-vision-works-self-driving-cars-david-silver

[10] *Tesla Motors* [online]. [cit. 2019-05-19]. Dostupné z: https://www.tesla.com/autopilot

[11] *Photoneo raises one of the largest seed rounds in Central Europe* [online]. Bratislava, 2015 [cit. 2019-05-19]. Dostupné z: https://www.photoneo.com/photoneo-raises-one-of-the-largest-seed-rounds-in-central-europe/

[12] In: *Photoneo* [online]. [cit. 2019-05-19]. Dostupné z: https://www.photoneo.com/

[13] *Sanezoo Europe* [online]. [cit. 2019-05-19]. Dostupné z: https://www.sanezoo.com/

[14] *Sanezoo* [online]. [cit. 2019-05-19]. Dostupné z: https://www.linkedin.com/company/sanezoo/

[15] BERÁNEK, O. Implementace prezentačního zařízení s Raspberry Pi. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2018. 53 s. Vedoucí bakalářské práce Ing. Ondřej Andrš, Ph.D.

[16] UPTON, Eben. *Raspberry Pi : uživatelská příručka*. Brno : Computer Press, 2016, 280 s. ISBN 978-80-251-4819-8.

[17] *Raspberry Pi* [online]. [cit. 2019-05-20]. Dostupné z: https://www.raspberrypi.org/

[18] UPTON, Eben. *Raspberry Pi 3 Model B+* [online]. [cit. 2019-05-20]. Dostupné z: https://www.raspberrypi.org/blog/raspberry-pi-3-model-bplus-sale-now-35/
[19] *RPi 3B+* [online]. [cit. 2019-05-20]. Dostupné z: https://www.theengineeringprojects.com/2018/07/introduction-to-raspberry-pi-3-b-plus.html
[20] *Simple Guide to the Raspberry Pi GPIO Header and Pins* [online]. [cit. 2019-05-20]. Dostupné z: https://www.raspberrypi-spy.co.uk/2012/06/simple-guide-to-the-rpi-gpio-header-and-pins/
[21] JENČ, Jiří. *Možnosti rozšíření Raspberry Pi o modul kamery: Projekt 3*. Fakulta strojní, 2014. ČVUT v Praze.
[22] Camera Hardware [online]. [cit. 2019-05-20]. Dostupné z: https://picamera.readthedocs.io/en/release-1.13/fov.html
[23] Raspberry Pi Camera Module V2 [online]. [cit. 2019-05-20]. Dostupné z: https://www.raspberrypi.org/products/camera-module-v2/
[24] UPTON, Eben [online]. [cit. 2019-05-20]. Dostupné z: https://www.raspberrypi.org/documentation/hardware/camera/
[25] UPTON, Eben. *Noobs* [online]. [cit. 2019-05-20]. Dostupné z: https://www.raspberrypi.org/blog/be-a-noobs-v1-3-beta-tester/
[26] *Opencv* [online]. [cit. 2019-05-20]. Dostupné z: https://opencv.org/about/
[27] *Medicine applications* [online]. [cit. 2019-05-20]. Dostupné z: https://i.stack.imgur.com/gE2V6.jpg
[28] REDMON, Joseph, Santosh DIVVALA, Ross GIRSHICK a Ali FARHADI. *You Only Look Once: Unified,Real-Time Object Detection* [online]. May 2016, , 1-8 [cit. 2019-05-20]. Dostupné z: https://arxiv.org/pdf/1506.02640.pdf
[29] KEYSER, Celine. *Python Programming Language & Applications* [online]. India: World Technologies, 2012 [cit. 2019-05-20]. ISBN 8132317580.
[30] *Numerical Python* [online]. [cit. 2019-05-20]. Dostupné z: https://www.python-course.eu/numerical_programming.php
[31] BRESSERT, Eli. *Scipy and Numpy* [online]. O´Reilly [cit. 2019-05-20]. Dostupné z: https://www.oreilly.com/library/view/scipy-and-numpy/9781449361600/ch01.html
[32] ROSEBROCK, Adrian. *Practical Python and OpenCV* [online]. 4th edition. [cit. 2019-05-20].
[33] ROSEBROCK, Adrian. *OpenCV* [online]. [cit. 2019-05-20]. Dostupné z: https://www.pyimagesearch.com/2018/07/19/opencv-tutorial-a-guide-to-learn-opencv/
[34] MALLICK, Satya. *Color spaces* [online]. [cit. 2019-05-20]. Dostupné z: https://www.learnopencv.com/color-spaces-in-opencv-cpp-python/
[35] *Blurring* [online]. [cit. 2019-05-20]. Dostupné z: https://docs.opencv.org/master/d4/d13/tutorial_py_filtering.html
[36] HARRISON. [online]. [cit. 2019-05-20]. Dostupné z: https://pythonprogramming.net/thresholding-image-analysis-python-opencv-tutorial/
[37] *Canny edge detection* [online]. [cit. 2019-05-20]. Dostupné z: https://docs.opencv.org/master/da/d22/tutorial_py_canny.html

[38] *Contours* [online]. [cit. 2019-05-20]. Dostupné z: https://docs.opencv.org/master/d4/d73/tutorial_py_contours_begin.html

[39] ROSEBROCK, Adrian. *Contours* [online]. [cit. 2019-05-20]. Dostupné z: https://www.pyimagesearch.com/2016/04/04/measuring-distance-between-objects-in-an-image-with-opencv/

[40] *B&R* [online]. [cit. 2019-05-20]. Dostupné z: https://www.br-automation.com/cs/

[41] PARÁK, Roman. Robotický stolní fotbal – herní strategie, Brno, 2017, 116 s. Diplomová práca. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav automatizace a informatiky. Vedúci diplomovej práce Ing. et Ing. Stanislav Lang.

[42] GAŠKO, V. Využití strojového učení pro kontrolu kvality v průmyslových aplikacích, Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2019. Vedoucí bakalářské práce Ing. Roman Parák

9 SEZNAM OBRÁZKŮ

10 SEZNAM PŘÍLOH

PŘÍLOHA A. CD – ROM

A. CD-ROM

Tab. 1: Obsah CD

Adresář	Soubor
Bakalářská práce	2019_BP_Huber_Michal_191419.pdf
Řídící kódy	detection_color.py detection_faultness.py detection_color_faultness_pc.py detection_color_faultness_rpi.py detection_green_big.py
Video ověření funkčnosti	video_orange.mp4 video_green.mp4 video_green_big.mp4
Trénovací množina obrázků	image-12-06-33.jpg … … image-12-32-40.jpg