VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2020 Bc. Václav Boček

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

DETEKCE VAD POTISKU
DETECTION OF PRINTING DEFECTS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Václav Boček**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Peter Honec, Ph.D.**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Kybernetika, automatizace a měření**

Ústav automatizace a měřicí techniky

Student: Bc. Václav Boček ***ID:*** 182589

Ročník: 2 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Detekce vad potisku

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je navrhnout a realizovat detekci vad v potisku propisek s využitím řádkové kamery a automatických algoritmů pro rozpoznáváni vad.

1. Seznamte se s problematikou detekce vad na soutisku na offsetových a sítotiskových strojích.
2. Navrhněte vhodnou metodu snímání potisku na tělech propisovacích tužek s využitím řádkové (line-scan) kamery.
3. Navrhněte a vytvořte pracoviště pro takové snímání potisku.
4. Vytvořte aplikaci pro řízení rotace, osvětlení a řízení řádkové kamery.
5. Navrhněte a implementujte algoritmy pro detekci vad v potisku.
6. Otestujte a zhodnoťte.

DOPORUČENÁ LITERATURA:

HLAVAC V., SONKA M., BOYLE R.: Image Processing, Analysis, and Machine Vision, ISBN 978-0495082521

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 1.6.2020

Vedoucí práce: Ing. Peter Honec, Ph.D.

doc. Ing. Václav Jirsík, CSc.
předseda oborové rady

ABSTRAKT

Tato práce se zabývá návrhem a realizací zařízení pro vizuální kontrolu potisku loga na propiskách. Ke snímání objektu je využito řádkové kamery. Řízení celé jednotky a zpracování pořízených dat zajišťuje mikropočítač Raspberry Pi 4, ke kterému je vytvořena rozšiřující deska pro ovládání periferií. Řízení jednotlivých prvků zařízení je implementováno v jazyku C++, algoritmy detekce v jazyku Python s využitím knihoven OpenCV a TensorFlow. Zařízení disponuje grafickým uživatelským rozhraním pro ovládání celého procesu kontroly. Na konci práce jsou uvedeny výsledky testu spolehlivosti celé kontrolní jednotky.

KLÍČOVÁ SLOVA

Detekce vad, potisk propisek, řádková kamera, Raspberry Pi, zpracování obrazu, OpenCV, Keras, neuronové sítě, GUI

ABSTRACT

This thesis deals with the design and subsequent implementation of a unit inspecting a printed logos on the pen surface. A line-scan camera is used to capture the object. Whole the unit including acquited data processing is controlled by Raspberry Pi 4 platform extended by perifery board. The control of the hardware parts is implemented in C++, the detection algorithms in Python using OpenCV and TensorFlow libraries. The unit has a graphical user interface for control of the inspection process. In the end of the thesis test of the unit reliability is shown.

KEYWORDS

Defects detection, pen printing, line-scan camera, Raspberry Pi, image processing, OpenCV, Keras, neural networks, GUI

BOČEK, Václav. *Detekce vad potisku*. Brno, 2020, 88 s. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce: Ing. Peter Honec, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Detekce vad potisku“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

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

Brno 1.6.2020 .....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Petrovi Honcovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno 1.6.2020 .....

podpis autora

Obsah

Seznam obrázků

Seznam tabulek

Úvod

V posledním desetiletí zažíváme obrovský rozvoj a praktické využití strojového vidění v aplikacích z mnoha různorodých oblastí. Uplatnění nachází například v oblasti medicíny, zabezpečení objektů, dopravy, běžného života a také v průmyslu. Právě tam jsou orientovány typické aplikace jako počítání objektů, čtení výrobních kódů, navádění robotů nebo také vizuální kontrola kvality výrobků.

S postupující globalizací a navyšováním kapacity výroby se stává stále aktuálnější a více žádaná vizuální inspekce. Dříve aplikovaný klasický model, kdy operátor kvality kontroluje kus po kusu či provede namátkovou kontrolu v rámci vyrobené série, se stává s ohledem na rostoucí cenu pracovní síly a zvyšujícími se požadavky na kvalitu nedostatečným. Proto bývá nahrazován automatizovanými systémy. Ty navíc umožňují odesílat kvalitativní data z výroby do nadřazeného systému, jako jsou databáze a cloudy. Na jejich základě je poté možné počítat statistické výstupy pro zvýšení efektivity výroby. Kromě toho v aplikačním procesu platí pravidlo: Čím dřív se chyba objeví, tím levnější bude její odstranění. To do platí do jisté míry i v odvětví výroby pro vady výrobků, zvláště pak těch tvořících komplexnější systémy. Dodání kvalitních, prověřených výrobků tedy zákonitě vede i k větší spokojenosti zákazníků a snížení výrobních nákladů.

Zadání této diplomová práce vychází z aktuálních potřeb průmyslu. Konkrétně se jedná o firmu Centropen Dačice, jednoho z největších výrobců kancelářských potřeb na tuzemském trhu, odkud pramení požadavek na vývoj systému pro kontrolu potisku loga značkovací propisky.

Jde tedy o kontrolu válcových vzorů, které lze rotací rozbalit na plošné. K tomu se namísto klasické prostorové kamery nabízí využití kamery řádkové. Její použití s sebou nese kromě vyššího rozlišení i výhodu snímání vždy pod stejným úhlem či přítomnost pouze jednorozměrné vinětace a geometrického zkreslení. Pro řešení této úlohy byla vedoucím práce poskytnuta monochromatická řádková kamera od firmy Basler. Celé pracoviště je pak řízeno výpočetní jednotkou Raspberry Pi 4B.

V první kapitole je čtenář pro úplnost a pro lepší vhled do problematiky vzniku a charakteru vad seznámen se základními principy offsetového a sítotiskového tisku. V druhé kapitole jsou uvedeny přístupy k detekci vad na podobném typu objektu, které byly úspěšně vyřešeny cizími autory. Protože k vyhodnocení, zda se jedná či nejedná o vadu, je využito neuronových sítí, je v třetí kapitole shrnuta jejich teorie. Čtvrtá kapitola se věnuje popisu měřicího pracoviště a výběru jednotlivých hardwarových komponent. Jsou zde také popsány jejich parametry a diskutovány výhody, případně nevýhody. V páté kapitole následuje rozbor softwarové vybavení, které bylo použito k vývoji řídící aplikace, detekčních algoritmů a grafického uživatelského rozhraní. Je zde také uveden podrobný návod instalace některých softwarů, aby čtenář mohl

stejný úkol případně s co nejmenší námahou zopakovat. V následujících třech kapitolách je podrobně rozebrána akvizice a předzpracování snímků, následuje detekce vad. V deváté kapitole jsou popsány výsledky testování systému. Desátá kapitola představuje vytvořené grafické uživatelské rozhraní a ovládání celého procesu.

1 Druhy tisku

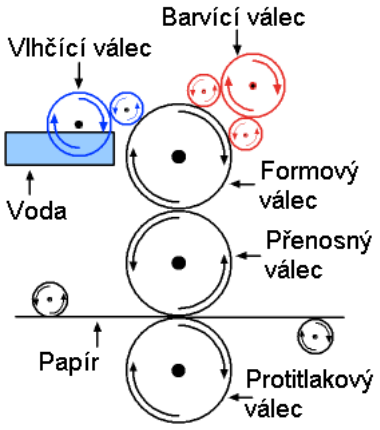
V této kapitole jsou uvedeny základní principy sítotiskového a offsetového tisku pro lepší vhled do problematiky vad těchto potisků, ke kterým s ohledem na různé výrobní faktory s nižší či vyšší frekvencí vždy dochází. Problematice detekce vad budou věnovány následující kapitoly.

1.1 Offsetový tisk

Offsetový tisk je v dnešní době nejrozšířenější tiskovou technikou. Z ekonomického hlediska je tento přístup vhodný pro velkoobjemový tisk ve vysoké kvalitě, proto se využívá zvláště pro tisk knih, novin, časopisů a letáků.

Princip offsetu spočívá v rozdílných chemicko-fyzikálních vlastnostech tiskových činitelů, konkrétně jde o vzájemné odpuzování mastných tiskových barev a vody na tiskové formě, která obepíná formový válec. Tisková forma je hliníková deska o tloušťce 0,1 – 0,5 mm, na jejíž jedné straně jsou naneseny dvě vlastnostmi rozdílné vrstvy [1]. Tyto různé vlastnosti jsou dosaženy úpravou povrchu, např. eloxací nebo zdrsněním povrchu, kdy na jeden typ povrchu je barva přitahována a druhým je odpuzována. Následně se tedy z barvicích válců přenese barva pouze na požadovaná místa formového válce. Z formového válce se barva přenáší na přenosný (nebo také offsetový) válec, který má pryžový povrch a z něj se tiskne na papír nebo jiný tiskový materiál. Papír je zespodu požadovaným tlakem přitlačován pomocí protitlakového válce.

Při tisku na válcovitý povrch tiskové těleso nahradí protitlakový válec. Systém válců je zobrazen na níže.



Obr. 1.1: Schéma vnitřního uspořádání válců v offsetové jednotce [1].

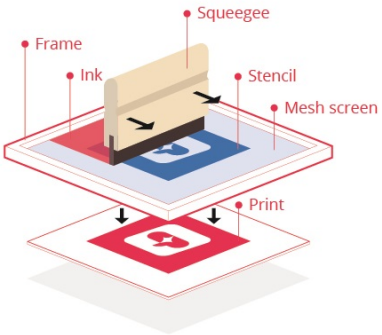
Takto se natiskne vždy jedna barva. V offsetovém tisku se zpravidla využívá CMYK barevného modelu. Proto při barevném tisku prochází list papíru postupně celou sérií všech čtyř barevných jednotek.

Obr. 1.2: Ilustrační obrázek offsetové tiskárny[2].

Mezi výhody této tiskové techniky patří nízká cena při větším tiskovém objemu, dále pak rychlost tisku a dobrá kvalita pro běžné použití.

1.2 Sítotisk

Princip sítotisku spočívá v protlačování – protírání - vazké barvy pomocí tříče (jinak také těrka nebo stěrka) propustnými místy sítotiskové šablony na průtiskový materiál [3].



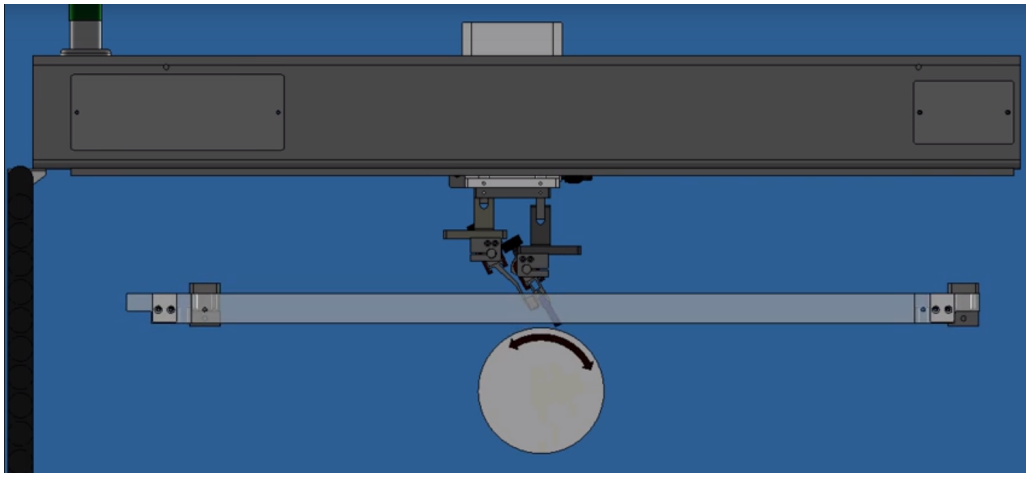
Obr. 1.3: Zobrazení principu sítotisku [4].

Šablona je vyrobena z přírodního nebo syntetického hedvábí nebo z kovových vláken a skládá se z propustných kresebných míst a nepropustných, zakrytých částí

netisknoucí plochy, zakotvených přímo mezi oky tkaniny. Nepropustné vrstvy jsou v šabloně zhotoveny různými přímými a nepřímými technikami, v průmyslu se nejčastěji využívá přímá fotomechanická metoda, kdy se na požadovaná místa nanese světlocitlivá emulze. Ta se absorbuje a následně dojde vlivem světelné expozice k ustálení a zakonzervování šablony.

Pak se do rámu, ve kterém je umístěna šablona, nanese barva a tříčem se protlačuje na materiál na potiskovaný objekt.

V případě válcového tvaru potiskovaného tělesa je tříč ve stabilní neměnné pozici, rovinná šablona vykonává rovnoměrný translační posun, kterým dochází k protlačování barvy na rotující potiskované těleso. Tento proces je zobrazen na přiloženém obrázku.



Obr. 1.4: Zobrazení principu sítotisku na válcovité těleso [5]

Výhodou sítotisku je relativně jednoduchá technologie nanesení barvy. Z ekonomického hlediska je výhodný až při větších sériích. Nevýhodou může být nedokonalé překrytí barvou.

2 Průzkum aplikovaných přístupů k detekci vad

Tato kapitola je věnována průzkumu studií, přístupů a metod, které se zabývají detekcí vad, zvláště pak detekcí vad potisků a typově podobných problémů. Názvy podkapitol jsou odvozeny od příslušných názvů článků, ze kterých bylo čerpáno.

2.1 Machine Vision based Defect Detection Approach using Image Processing

Následující přístup se zabývá řešením problému vizuální kontroly a detekce vad prokovů a děr na deskách plošných spojů. Autoři tohoto řešení jsou turečtí výzkumní pracovníci z Ardahanské a Firatské univerzity Mehmet Baygin, Mehmet Karakose, Alisan Sarimaden a Erhan Akin, kteří své řešení publikovali v roce 2017 v [6] článku nakladatelství IEEE.

Cílem této práce bylo na deskách plošných spojů pohybujících se na dopravníku určit, zda jsou mají správně vyvrtané díry a prokovy a u chybných kusů určit lokaci vady.

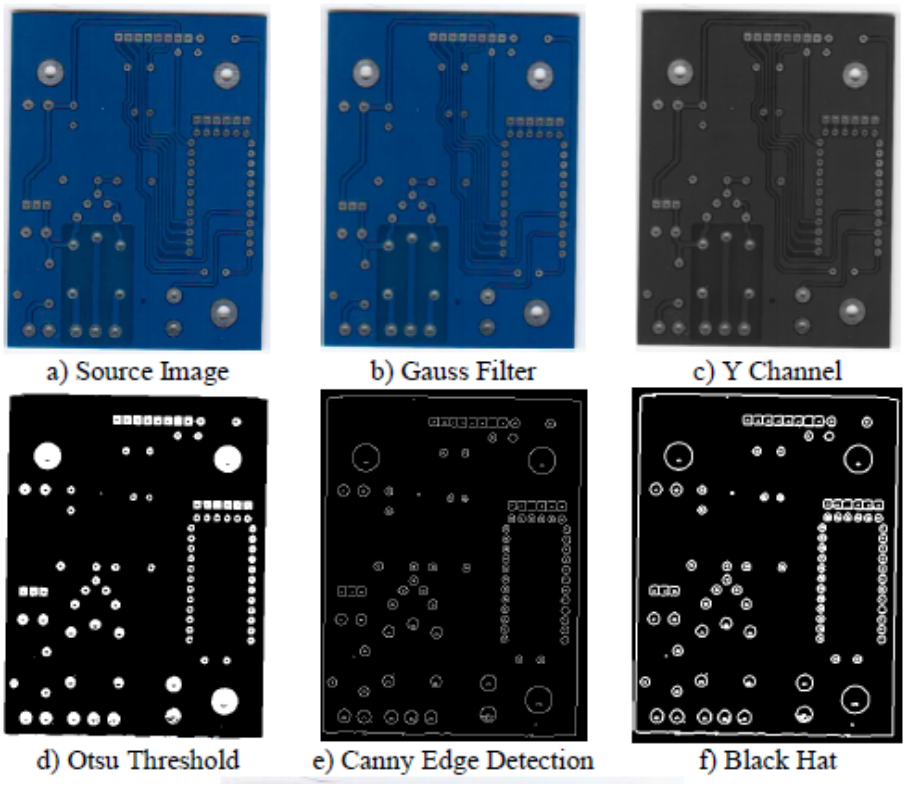
K vyhodnocení byla použita série operací zpracovávajících pořízený snímek. Po pořízení je RGB snímek je pro lepší manipulaci převeden do YUV barevného prostoru. Následně je šum v obrazu potlačen pomocí Gaussoba filtru

$$G(r) = \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} e^{\frac{-r^2}{2\sigma^2}} \tag{2.1}$$

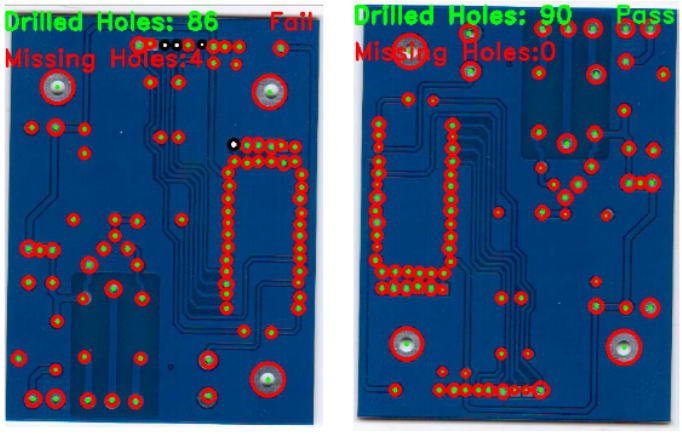
Poté je obraz zbinarizován pomocí Otsu metody se zvoleným prahem. V dalším kroku jsou pomocí Canny hranového detektoru detekovány v obraze hrany a pomocí morfologických operací, konkrétně procesu Black Hat jsou nalezené hrany zesíleny. Na tento obraz hran je v posledním kroku použita Houghova transformace, kterou jsou identifikovány jednotlivé kruhové otvory v desce, určena jejich pozice a průměr. Posloupnost těchto kroků je zobrazena na obrázcích desky plošných spojů v jednotlivých fázích zpracování na obr.2.1

Tímto procesem je zpracován snímek referenčního správného kusu i snímky testovaných kusů. Následným výsledkem porovnání referenčního a testovaného obrazu je buď shoda, pak jde o dobrý kus, nebo ke shodě nedojde a pak je kus klasifikován jako vadný.

Tento systém pracuje v reálném čase, je nezávislý na barvě testované desky, pozici a natočení předmětu a dokáže zachytit výrobní chybu menší než 2μm.



Obr. 2.1: Jednotlivé fáze zpracování snímku desky plošných spojů[6].



Obr. 2.2: Příklad špatně (v levo) a správně (v pravo) klasifikovaných kusů [6]

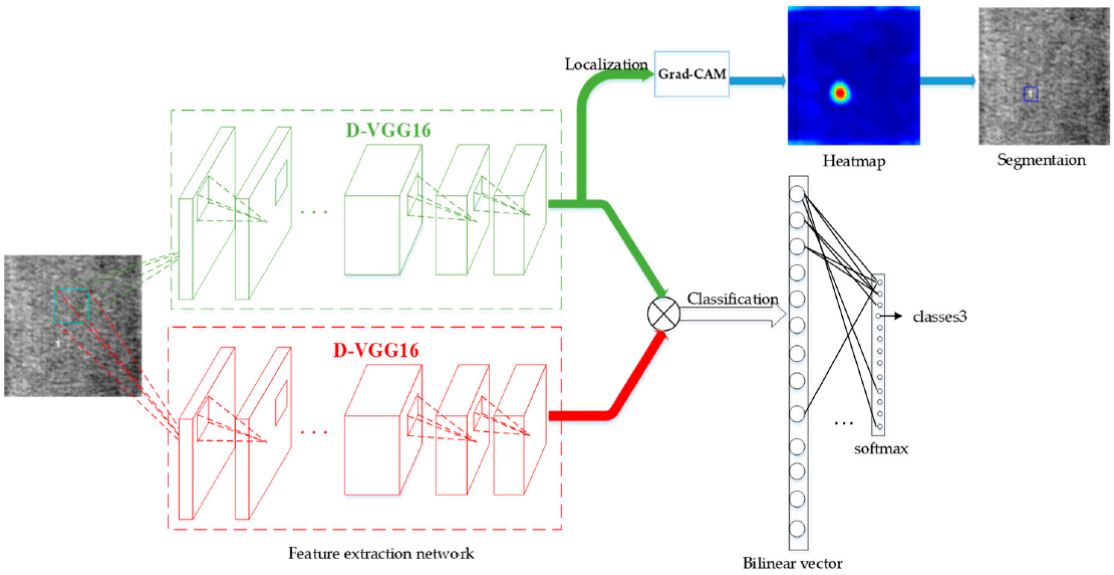
2.2 A Generic Automated Surface Defect Detection Based on a Bilinear Model

Metoda popsaná v tomto článku se zabývá automatickou průmyslovou detekcí vad na povrchu textury textilie. Navrhli ji čínští výzkumníci Fei Zhou, Guihua Liu, Feng Xu a Hao Deng, jejichž článek [7], ve kterém tento přístup popisují, byl vydán na platformě vědecké komunity ResearchGate.net.

Navržený systém se skládá ze dvou paralelních symetrických konvolučních neuronových podsítí, které slouží jako extraktor příznaků. Tyto vrstvy byly převzaty z konvoluční neuronové sítě VGG16, která byla naučena na datasetu ImageNet s 1000 výstupními třídami. Využívá se zde metody „transfer learningu“, kdy se převezmou první část přednaučené konvoluční sítě, následně se přidají dodatečné nenaučené vrstvy o požadovaných rozměrech dle úlohy a síť se poté doučí na požadované vzory. Výhodou této metody je daleko menší výpočetní náročnost, a potřeba nižšího množství trénovacích vzorů.

Dle mínění autorů tyto dva paralelní symetrické extraktory zvyšují citlivost systému na drobné vady. Tyto sítě jsou totiž v základu stejné, ale uzpůsobením vrstev s náhodnou inicializací a případným přidáním prvků náhody jako je například dropout, se tyto sítě na stejných příkladech naučí mírně odlišně.

Následně jsou tyto vektory z konvolučních vrstev obou sítí spojeny do jednoho vektoru. Tento výstup je dán na vstup softmax modulu, který klasifikuje daný snímek do příslušné kategorie.



Obr. 2.3: Architektura navrženého modelu[7].

Kromě toho výstup jednoho z extraktorů je dán na vstup tzv. Grad-CAM modulu, který z extrahovaných odezev poslední vrstvy extraktoru vytvoří tzv. Heatmapu. To je obraz, který intenzitou jasových hodnot udává pravděpodobnost, že se v příslušné oblasti vyskytuje vada. Následně se provede segmentace vady od pozadí a určí se lokace dané vady pomocí prahování získané „heat mapy“.

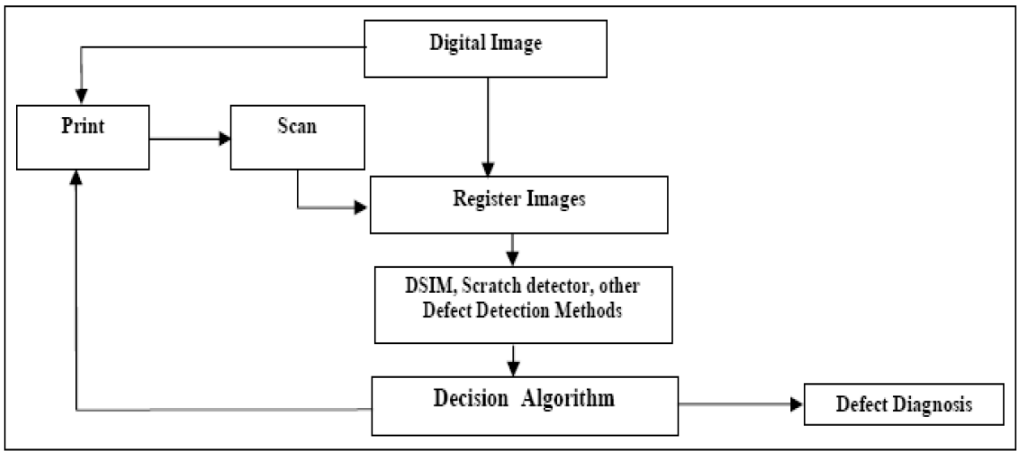
Těmito dvěma moduly lze tedy získat informaci o typu vady i o její lokaci.

Přehledový diagram modelu je uveden na obr.2.3

2.3 Automatic visual inspection and defect detection on Variable Data Prints

Následující přístup se zabývá detekcí vad výtisků s proměnným obsahem. Metodu uvedli výzkumníci HP Laboratories Marie Vans, Sagi Schein, Carl Staelin, Pavel Kisilev, Steven Simske, Ram Dagan, Shlomo Harush v roce 2011.

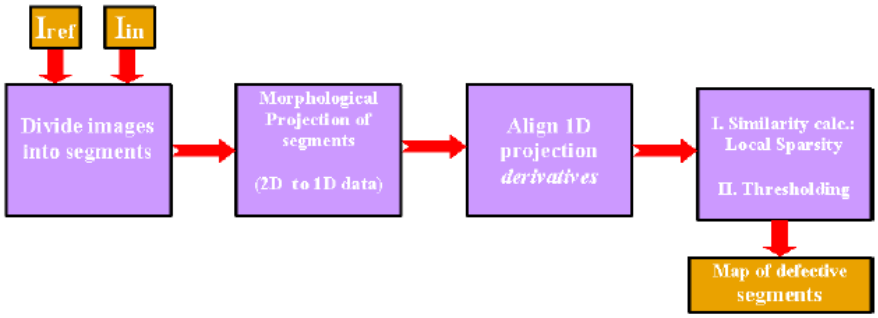
Navržený model využívá znalosti optimálního požadovaného výtisku, který je v digitální formě a který je poslán k vytisknutí. Po vytisknutí je výtisk oskenován. K dispozici je tedy ideální referenční a reálný vzorek výtisku, které jsou mezi sebou porovnávány.



Obr. 2.4: Diagram navrženého systému[8].

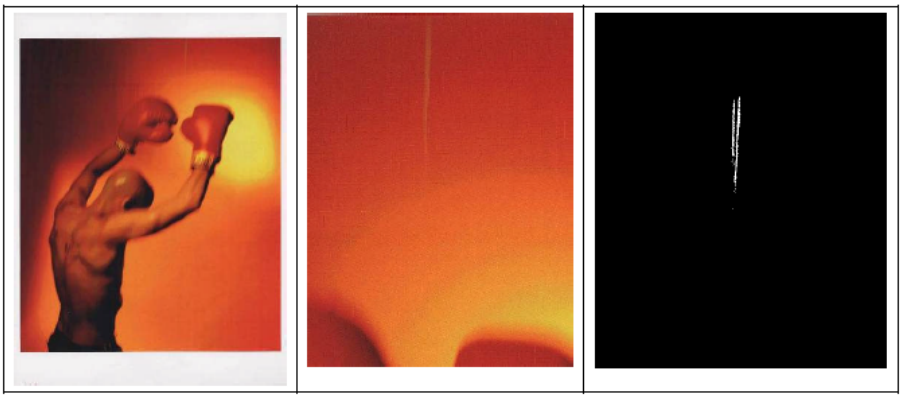
Model obsahuje dva detektory vad. První je Structural Dis-similarity, který využívá myšlenky, že každá oblast v tištěném obrázku má podobnou oblast v referenčním obrázku, pokud tedy neobsahuje vadu [8]. Poté, co jsou v obou obrázcích nalezeny odpovídající si oblasti, jsou měřeny některé jejich parametry jako střední jasová hodnota a kontrast daných oblastí a jejich vzájemná korelace a na základě těchto indexů je vada potvrzena či vyvrácena.

Tento první detektor ovšem má velice nízkou úspěšnost v detekci vad, které mají charakter tzv. škrábanců (angl. scratch), které se při tisknutí velice často vyskytují. Tyto vady mají několik společných vlastností, a to že bývají tenké, světlé, orientovány v jednom směru a mají velmi nízký nízký poměr signál-šum. Díky znalosti typických vlastností těchto vad byl navržen detektor Sparse Projection-Based Scratched Detector, který vady tohoto charakteru postihuje. Jeho schéma je na obr.2.5.



Obr. 2.5: Diagram sekundárního detektoru[8].

Nejdříve jsou vybrány odpovídající si části referenčního a testovaného obrazu, dále jsou na segmentech obou obrazů provedeny morfologické operace z důvodu zvýraznění vad s nízkým kontrastem, dále následuje transformace obrazu do vektoru pomocí sumací v jedné dimenzi. Následně je spočítána mapa podobností, kde jako vady jsou vyhodnoceny místa s hodnotou větší než zvolený práh. Tímto druhým detektorem jsou tedy kromě obecných vad nalezeny i vady specifické, které první detektor víceméně nepostihuje. Příklad detekce vady tohoto typu je zobrazen na obrázku 2.6.



Obr. 2.6: Ukázka nalezené vady typu „scratch“[8].

3 Neuronové sítě

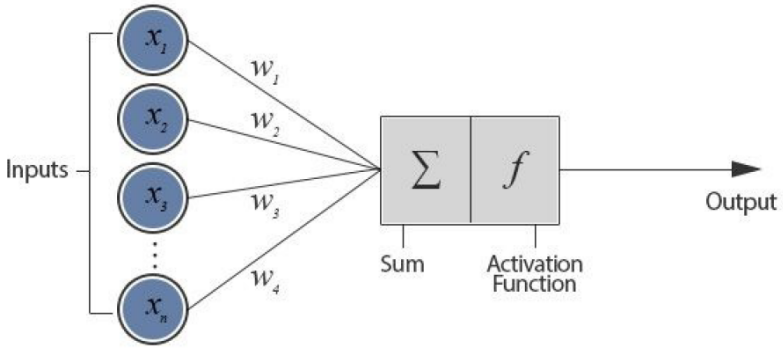
Na některé principy a technologie člověk v průběhu času přijde sám, jindy se nechal inspirovat některou z excelentních schopností přírodních organismů. Mezi takové případy patří například suchý zip, který byl inspirován lopuchem, nebo letadla, jejichž prvotní koncept nejspíš vznikl z pozorování letu ptáků. Je tedy naprosto logické, že právě lidský mozek byl pro mnohé inspirací ve vytvoření jakéhosi inteligentního systému.

Přestože ale letadla mají křídla jako ptáci, jako ptáci s nimi netřepetají, stejně tak i umělé neuronové sítě byly přírodou pouze inspirovány a jejich vnitřní principy jsou v mnohém odlišné.

3.1 Model neuronu

Neuron je základní jednotkou neuronové sítě. Byl vynalezen Frankem Rosenblattem v roce 1957.

Model neuronu má lineární část, kterou tvoří suma součinu vektoru vah neuronů s vektorem vstupů a část nelineární, kterou reprezentuje aktivační funkce.



Obr. 3.1: Model umělého neuronu [9]

Model neuronu popisuje následující rovnice.

$$y = f(\sum_{i=1}^{N} x_i w_i + b) \tag{3.1}$$

kde f je aktivační funkce, x jsou jednotlivé vstupy, w jsou váhy a b je bias.

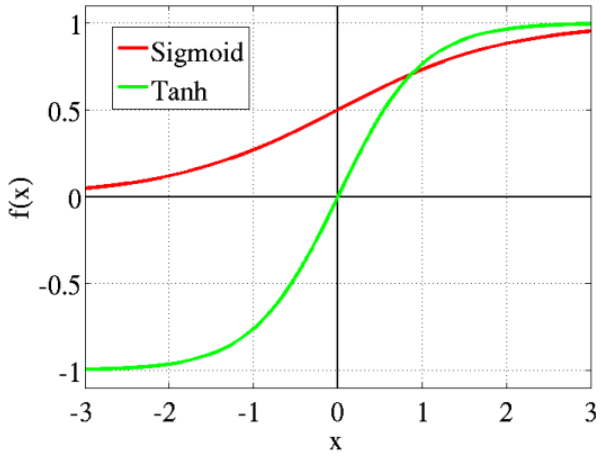
Aktivační funkce provede s vnitřním potenciálem neuronu, který je daný součinem příslušných vstupů a vah nelineární transformaci. Aktivačních funkcí existuje několik, u klasických dopředných neuronových sítí se nejčastěji využívá sigmoida, která má vzorec:

$$f(z) = \frac{1}{1 + e^{-\sigma.z}} \tag{3.2}$$

nebo hyperbolická tangenta:

$$f(z) = \frac{1 - e^{-\sigma.z}}{1 + e^{-\sigma.z}} \tag{3.3}$$

kde z je vnitřní potenciál neuronu a σ je strmost funkce. Na obrázku níže je zobrazen průběh těchto aktivačních funkcí.



Obr. 3.2: Aktivační funkce - Sigmoida a Hyperbolická tangenta [11]

Učením neuronu, resp. neuronové sítě, která je z těchto neuronů složena, se pak nastavují jednotlivé váhy neuronu a případně i strmost jejich aktivačních funkcí.

3.2 Architektura sítě

Umělé neuronové sítě jsou tvořeny uspořádáním jednotlivých neuronů do jednotlivých vrstev, které společně vytváří umělou neuronovou síť. Neuronová síť vždy má vstupní vrstvu, kterou je n-dimenzionální signál distribuován do vyšších vrstev, pak má zpravidla jednu nebo více skrytých vrstev a poslední vrstva je vrstvou výstupní, která tvoří rozhraní na které je dán predikovaný výstup sítě. Jsou ale i výjimky, jako je Hopfieldova síť, která tvořena pouze jednou vrstvou.

Velikost neuronové sítě je charakterizována počtem neuronů ve vstupní a výstupní vrstvě a dále pak počtem vrstev a počtem neuronů v těchto vrstvách. Čím je velikost neuronové sítě větší, tím je více nelineární a je schopna se naučit na složitější a komplexnější problémy.

Neuronové sítě mají různé typy struktury propojení. Můžou být buď přímovazební nebo zpětnovazební. Spoje, resp. vazby jednotlivých neuronů u přímovazební struktury vedou postupně z neuronů v nižší vrstvě na neurony ve vrstvě vyšší, nikdy ne na sousední neurony a na neurony v nižších vrstvách. Příkladem přímovazební topologie je například dopředná neuronová síť, RCE síť, Kohonenova SOM. U zpětnovazební topologie sítě rovněž nejsou spojení se sousedními neurony v jedné vrstvě, ale na rozdíl od přímovazební jsou neurony vyšších vrstev zpětně navázány na neurony vrstev nižších. Příkladem zpětnovazební topologie neuronové sítě je Hopfieldova síť.

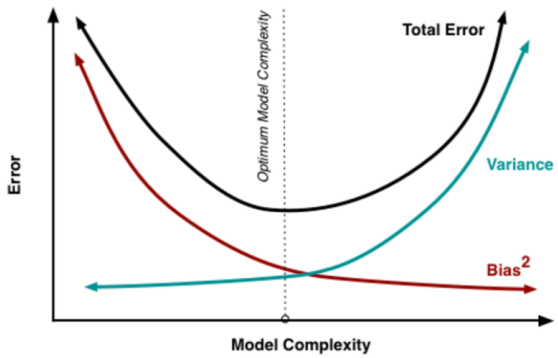
3.2.1 Složitost modelu

Chyba modelu neuronové je tvořena součtem tří dílčích složek, tvoří ji Bias, Variance a Neredukovatelná chyba.

$$Chyba = Bias + Variance + Neredukovatelnáchyba \tag{3.4}$$

Bias je chyba, kdy je model nemá dostatečnou velikost, aby dokázal daný problém dostatečně přesně generalizovat. Chyba biasu se projeví jak velkou chybou na trénovacích datech, tak i na datech validačních. Chyba biasu způsobuje nedoučení modelu.

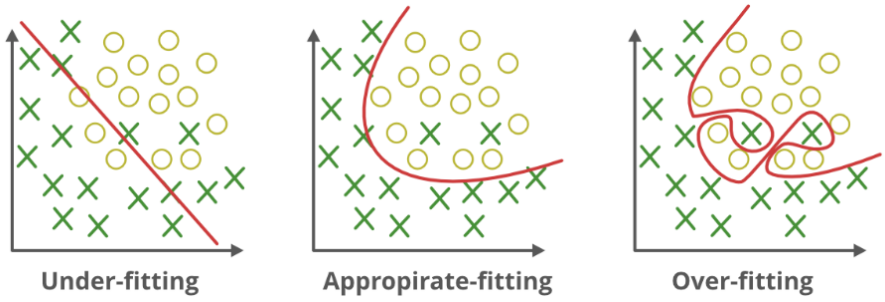
Variance je chyba dána přílišnou složitostí modelu. Model s vysokou variancí má tendenci místo generalizace souvislostí a vzorců v datech si v procesu učení jednotlivá data zapamatovávat. Tím se vlastně namísto užitečnému signálu učí šumu. Model s vysokou variancí se pozná tak, že má nízkou chybu na trénovací množině dat, ale vysokou chybu na validační množině. Variance modelu způsobuje jeho přeučení.



Obr. 3.3: Graf chyby modelu v závislosti na jeho složitosti [12]

Neredukovatelná chyba je chyba, která je v modelu vždy v určité míře přítomna, ať je model navržen jakkoliv, a není možné ji redukovat. Tato chyba je způsobena především šumem v datech.

Složitost modelu je pro daný problém optimální, jestliže je v celkové chybě modelu zvolen kompromis mezi oběma dílčími chybami, biasem i variancí. V takovém případě je celková chyba nejmenší. Složitost modelu je nutné odhadnout na základě komplexnosti a složitosti řešeného problému a na velikosti a kvalitě trénovací množiny.



Obr. 3.4: Míra naučenosti modelu [13]

3.3 Učení

Cílem učení neuronové sítě je nalézt takovou konfigurace sítě ve váhovém prostoru, která v aktivním režimu (vybavování) realizuje požadovanou funkci[14].

Na způsob učení daného modelu se lze dívat z různých pohledů. Jedním z nich je, zda daný model učí jednorázově, příkladem tohoto způsobu učení je Hopfieldova síť, nebo v iterativním procesu. Mezi neuronové sítě s iterativním učením se řadí například dopředná neuronová síť nebo RCE síť.

Dále se princip učení sítě může posuzovat podle toho zda se jedná o učení s učitelem či učení bez učitele.

Učení s učitelem

U učení s učitelem se modely učí na datech, u kterých je předem určena požadovaná hodnota nebo požadovaná třída. Proces učení probíhá tak, že je modelu předložen na vstup tréninkový vzor a model predikuje hodnotu, kterou vystaví na výstup. Tato hodnota je dále porovnána s požadovanou hodnotou a je odhadnuta chyba modelu. Dle optimalizačního algoritmu se pak upraví nastavení parametrů modelu tak, aby se minimalizovala nákladová funkce. Tento proces probíhá do splnění ukončovací podmínky.

Mezi algoritmy učení s učitelem patří kromě plně propojených, Hopfieldovy, RCE neuronových sítí také další modely strojového učení jako například rozhodovací stromy, SVM, k-NN.

Mezi hlavní nevýhody tohoto rozšířeného přístupu je nutnost přiřadit ke každé instanci trénovacích dat požadovanou hodnotu. Tento proces je většinou nutné provádět ručně, což bývá náročné jak po stránce časové, tak někdy i té psychické.

Učení bez učitele

Na rozdíl od učení s učitelem se modely při učení bez učitele učí na datech, kterým předem nebyla přiřazena určitá kategorie. Algoritmy se učí z předkládaných dat vzájemné vztahy mezi nimi a také strukturu a charakter vstupních dat.

Algoritmy učení bez učitele jsou nejčastěji používány ke shlukové analýze a k redukci dimenze dat. Proto jsou základem datové analýzy a použití těchto algoritmů může být někdy nezbytné vůbec pro pochopení dat předtím, než lze použít algoritmy učení s učitelem.

Jejich výhodou je také to, že zde není nutnost pořízená data předem manuálně opatřovat příslušností ke kategorii, z čehož pramení možnost získat data kvantitativně v daleko větší míře než v případě učení s učitelem.

Mezi algoritmy učení s učitelem patří například Kohonenova samoorganizační mapa (SOM), z ostatních algoritmů strojového učení pak například PCA nebo K-means.

3.3.1 Ztrátová funkce

Trénování neuronové sítě je založené na nastavování vah neuronů v síti. K tomu, aby bylo možné upravit potřebné váhy, je nezbytné pro každý tréninkový vzor vyjádřit vzdálenost aktuálního výstupu a výstupu požadovaného. Tu nám určuje ztrátová (chybová) funkce.

Podstatou ztrátové funkce je tedy vyjádřit odlišnost mezi modelem a realitou. Stejně, jako se od sebe liší různé typy úloh a tedy i výstupů modelů, tak se liší i používané ztrátové funkce. Nejzákladnější dělení spočívá v typu úlohy, která může být regresní nebo klasifikační.

Pro regresní typ úlohy je vhodné použití *metody nejmenších čtverců*:

$$L = \frac{1}{2}\sum_{i=1}^{N}[y_i - d_i]^2 \tag{3.5}$$

resp. metodu *binary cross entropy* pro případ logistické regrese:

$$L = -(d\log(y) + (1-d)\log(1-y)) \tag{3.6}$$

U úloh klasifikačního typu s více výstupními neurony se pak nejčastěji využívá metoda *categorical cross entropy*, která namísto jednoho výstupu sumuje chybu všech výstupů:

$$L = -\sum_{j=1}^{m}(d \log(y) + (1-d) \log(1-y)) \tag{3.7}$$

Průměrnou chybu, spočítanou přes celý tréninkový set, vyjadřuje tzv. nákladová funkce:

$$J = \frac{1}{p}\sum_{s=1}^{p} {}^{s}\mathrm{L} \tag{3.8}$$

Čím menší hodnotu nákladová funkce má, tím lépe je model na danou množinu dat naučen.

3.3.2 Optimalizační algoritmy

Aby se neuronová síť z vypočtených chyb naučila je potřeba chybu vhodně zpětně zpropagovat i do nižších vrstev neuronů a váhy spojení jednotlivých neuronů upravit. K tomuto procesu se používají optimalizační algoritmy, které v důsledku vedou k celkovému snížení chyby sítě a tedy k naučení na zvolený problém.

Gradient Descent

Gradient Descent je základní optimalizační algoritmus, ze kterého byly následně odvozeny další vylepšené algoritmy. Metodou Back Propagation jsou zpropagovány chyby k jednotlivým neuronům ve vrstvách. Metoda Gradient Descent pak využívá znalosti těchto chyb a dle parciálních derivací zpropagované chyby upraví příslušné váhy neuronů. Tento proces popisuje rovnice níže.

$$\delta_t = -\nabla_{\Theta}.F(\Theta_t) \tag{3.9}$$

$$\Theta_t = \Theta_{t-1} + \eta.\delta_t \tag{3.10}$$

kde Θ_t je nová váha, Θ_{t-1} je váha minulého kroku, η je koeficient učení, F značí funkci reprezentující neuronovou síť a ∇ reprezentuje gradient.

Gradient Descent počítá gradient a upravuje váhy neuronů na základě celé trénovací množiny. Proto může být v kombinaci s větší trénovací množinou nepřijatelně pomalý.

Algoritmus, který vychází z Gradient Descent a upravuje váhy po každém tréninkovém vzoru tréninkové množiny je Stochastic Gradient Descent. Tím, že každý tréninkový vzor má okamžitý dopad na nastavení vah, je do učení zanesen prvek stochasticity, proto název Stochastic Gradient Descent. Ve výsledku pak konverguje

v porovnání s Gradient Descent rychleji. Nevýhodou je jeho nestabilita, obzvláště pak při větší diverzitě trénovacích dat a relativně větším koeficientu učení.

Kompromisem mezi těmito dvěma přístupy je Mini-Batch Gradient Descent, kdy dochází k úpravě vah po vyhodnocení trénovací podmnožiny o zvolené velikosti.

Momentum

Přičtením momentu k hodnotě adaptace váhy se zohlední i změna váhy v předchozím kroku. To je vhodné k dosažení vyšší rychlosti konvergence chybové funkce k minimu a také může dopomoci k překonání lokálního minima, ve kterém by jinak učení neuronové sítě uvázlo. Výpočet adaptace vah s přidaným momentem je uveden v rovnici níže.

$$\Theta_t = \Theta_{t-1} + a_t \tag{3.11}$$

$$a_t = -\eta.\nabla_\Theta F(\Theta_t) - \gamma a_{t-1} \tag{3.12}$$

kde γ reprezentuje moment.

Algoritmy s adaptací parametrů

Algoritmus **AdaGrad** vychází z algoritmu Gradient Descent. Umožňuje ale automatické snižování parametru učení v průběhu učení. To je výhodné, protože z počátku učení se model rychle naučí a s přibývajícími iteracemi dochází k jeho stále jemnějšímu dolaďování.

$$\Theta_t^i = \Theta_{t-1}^i + \frac{\eta}{\sqrt{\sum_{t=1}^{t} {\delta_\tau^i}^2 + \epsilon}} \delta_t^i \tag{3.13}$$

Protože dělitel ve jmenovateli, který snižuje koeficient učení je akumulován přes všechny iterace, postupně může dojít téměř k vymizení koeficientu učení. To je hlavní nevýhodou tohoto algoritmu. Tento nedostatek řeší algoritmus **AdaDelta**, který je stejný s tím rozdílem, že dochází ke snižování koeficientu učení jen na základě x posledních gradientů.

Existují i další optimalizační algoritmy, které dále rozvíjí a vylepšují jejich trénovací funkci, z těch známých jsou to například RMSPROP nebo ADAM, který je velice často používán pro trénování konvolučních neuronových sítí.

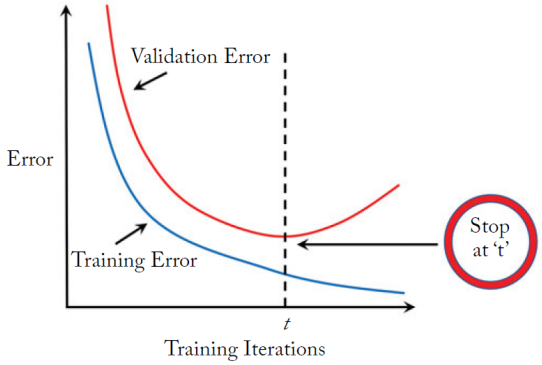
3.3.3 Regularizace

Regularizace je souborem metod používaných k podpoře generalizace v učeném modelu. Tyto techniky často ztěžují a zpomalují proces učení, ale zároveň podporují to, aby se model učil reálné obecné souvislosti na rozdíl od prostého memorování, které je typické při přeučení. Čili vede k tomu, že se neuronová síť učí užitečný signál a ignoruje šum.

Early Stopping

Nejjednoduší regularizací je včasné ukončení trénování.

Protože ve většině případů je velikost trénovací množiny omezená a není tedy možné trénovanému modelu předkládat stále nová data bez toho, aby se opakovala, existuje možnost, že se model přeučí. Aby k přeučení nedošlo, je třeba trénovací proces v pravý okamžik ukončit. Ukončení trénování je ideálně provedeno v okamžiku, kdy trend chyby na validační množině dat začne mít rostoucí tendenci. To je okamžik, kdy má model nejvyšší generalizační schopnost daného problému a je tedy nejlépe naučen. Na obrázku 3.5 je tento bod označen symbolem t.



Obr. 3.5: Typický graf průběhu chyby modelu na tréninkovém a validačním datasetu v závislosti na počtu iterací [15]

Penalizace velikostí váhy

Základní myšlenkou této regularizace je naučení modelu současně za co možná nejnižších hodnot vah jednotlivých spojení neuronů. Tím se rozhodovací činnost sítě distribuuje přibližně rovnoměrně na všechny neurony a předejde se situaci, kdy výsledek modelu určuje pouze malé množství velmi vlivných neuronů.

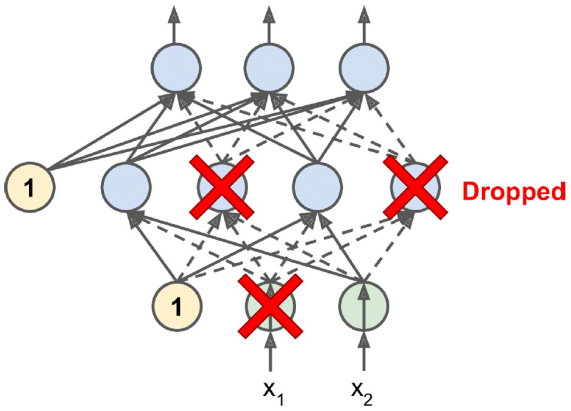
Požadovaného efektu je dosaženo přidáním regularizačního členu, který se přičítá k chybové funkci. Ten je buď prvního typu, kdy je penalizace počítána z absolutní hodnoty vah, nebo druhého typu, kde se penalizuje hodnota čtverce váhy neuronu.

$$L1 = \alpha . \sum ||w|| \qquad (3.14) \qquad\qquad L2 = \alpha . \sum ||w||^2 \qquad (3.15)$$

kde α je regularizační koeficient, který určuje míru penalizace a w jsou váhy neuronů.

Dropout

Dropout je jednou z nejvíce používaných regularizačních technik, která se používá při trénování neuronových sítí[16]. Technika spočívá v tom, že během procesu učení nejsou vždy aktivní všechny neurony, ale každý neuron je aktivován pouze s určitou pravděpodobností (volí se zpravidla do hodnoty 0,5). To opět vede k rovnoměrnějšímu rozložení vah v síti a tedy i vlivu na rozhodování.



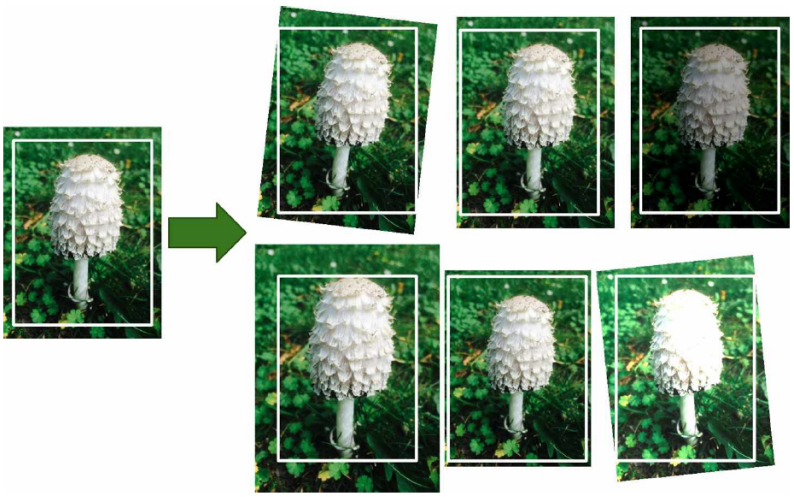
Obr. 3.6: Grafické znázornění deaktivace neuronů při technice dropout [17]

Data Augmentation

Tato regularizační technika je využívána hlavně v případech malého množství trénovacích dat. Jde o techniku, kdy jsou na základě zvolených operací či jejich kombinacemi prováděny transformace již existujících reálných dat, čímž je galerie dat rozšiřována. Výběr vhodných augmentačních operací závisí na charakteru řešeného problému. Obecně lze volit některé z těchto operací:

- Translace

- Rotace
- Střih
- Překlopení obrazu
- Výřez části obrazu
- Přidání šumu
- Změna jasu
- Změna kontrastu
- ...



Obr. 3.7: Ukázka rozšíření galerie dat vhodnou transformací [17].

3.4 Vybavování

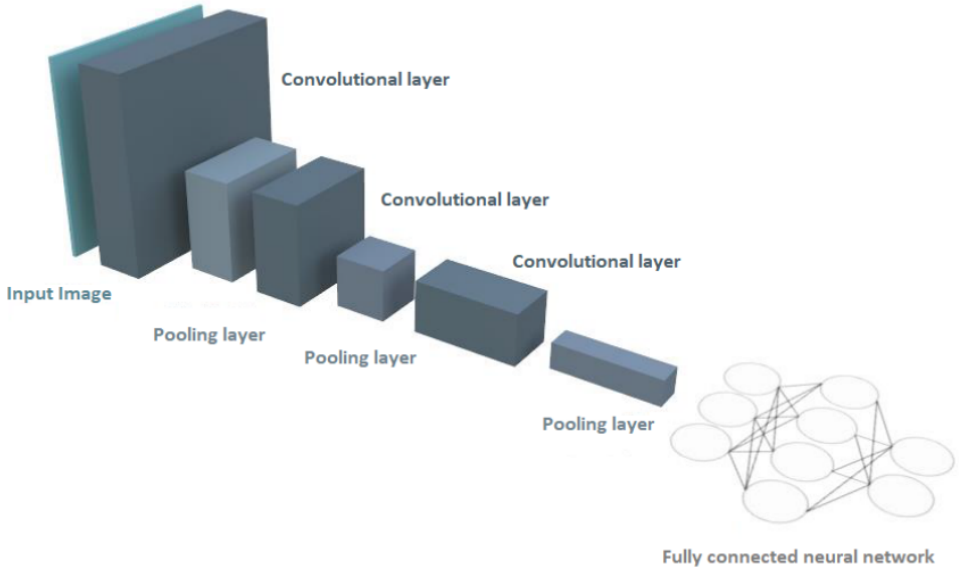
V závislosti na typu neuronové sítě vybavování může být buď jednorázový proces, který využívají například dopředné neuronové sítě, RCE sítě, Kohonenova samoorganizační mapa, nebo proces iterační, který je typický například pro Hopfieldovu neuronovou síť.

3.5 Konvoluční neuronové sítě

Konvoluční neuronové sítě (CNN) jsou hluboké dopředné neuronové sítě, ve kterých se využívá v některých vrstvách matematické operace konvoluce. Velmi časté je jejich využití v klasifikaci obrazových dat, resp. dat s daným vnitřním uspořádáním.

Jejich architektura je tvořena dvěma základními částmi. První částí je extraktor příznaků, který je složen z vrstev konvolučních a poolingových, které snižují šířku

a výšku map vzniklých ze vstupního obrazu a naopak zvyšují jeho hloubku. V ní dochází k získávání příznaků. V nižších vrstvách nejprve dochází k detekci jednoduchých primitiv jako jsou hrany a rohy a s postupem informace do vyšších vrstev jsou tyto informace spojovány a jsou detekovány složitější a komplexnější tvary a vzorce. Tyto příznaky jsou pak v druhé části sítě klasifikovány. Druhá část je tvořena plně dopřednými propojenými neuronovými vrstvami, které dle extrahovaných příznaků klasifikují vstupní obrázek do předdefinovaných kategorií.



Obr. 3.8: Uspořádání vrstev v konvoluční neuronové síti [18]

3.5.1 Konvoluční vrstva

Konvoluční vrstva je vrstva neuronové sítě, která vykonává operaci konvoluce vstupního signálu s definovaným počtem filtrů. Výstupem vrstvy je tzv. příznaková mapa (*feature map)*. Operace diskrétní dvourozměrné konvoluce se provádí dle následujícího vztahu:

$$g(x,y) = f(x,y) * h(x,y) = \sum_{i=-S/2}^{S/2} \sum_{j=-R/2}^{R/2} f(x-i, y-j).h(i,j) \tag{3.16}$$

kde g je příznaková mapa, f reprezentuje vstup a h reprezentuje filtr.

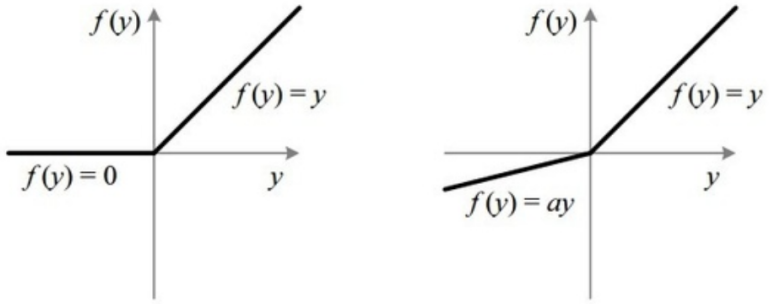
Váhy klasických neuronových sítí jsou zde použity jako hodnoty konvolučních filtrů konvoluční vrstvy. Proto i při učení jsou nastavovány právě hodnoty filtrů. Konvoluční filtr bývá typicky o rozměrech $3 \times 3 \times h$ nebo $5 \times 5 \times h$, přičemž jeho

hloubka h se odvíjí od hloubky vstupních dat dané konvoluční vrstvy. Hloubka výstupní příznakové mapy je pak dána počtem filtrů, které v dané vrstvě jsou použity, neboť výstup konvoluce s každým filtrem je přidán do výstupní příznakové mapy jako jedna vrstva.

U konvoluční vrstvy se nejčastěji využívá aktivační funkce *Rectified Linear Unit*(ReLU). Rovnici této aktivační funkce lze popsat jako:

$$f(x) = max(0, x) \tag{3.17}$$

Její výhodou je nízká výpočetní náročnost. Záporné hodnoty jsou u ní saturovány na nulu. To může být problematické vzhledem k učení, neboť v této oblasti je nulový gradient. Proto se někdy používá Leaky ReLU, která má mírný sklon i v záporné oblasti. Na obrázku 3.9 vpravo je zobrazen její průběh.



Obr. 3.9: Graf aktivační funkce ReLU (vlevo) a Leaky ReLU (vpravo) [19]

Volitelným u této vrstvy je parametr *stride*, který udává o kolik prvků se bude filtr během výpočtu konvoluce posunovat. Implicitní nastavení tohoto parametru bývá 1. Zvýšení této hodnoty může být výhodné v případě použití filtrů o větších rozměrech, kdy posunutím pouze o jeden krok dochází k získání velkého množství redundantních dat.

S každou konvoluční vrstvou se snižuje rozměr výstupních dat vzhledem k vstupním datům o *rozměr masky filtru méně jedna*. To může být nežádoucí u velmi hlubokých CNN. Toto podvzorkování je možné kompenzovat parametrem zero-padding, který vstupní signál - dle rozměru filtru použitého v dané vrstvě - doplní po okrajích nulami. Potom je šířka i výška výstupní matice rovna šířce a výšce matice vstupní.

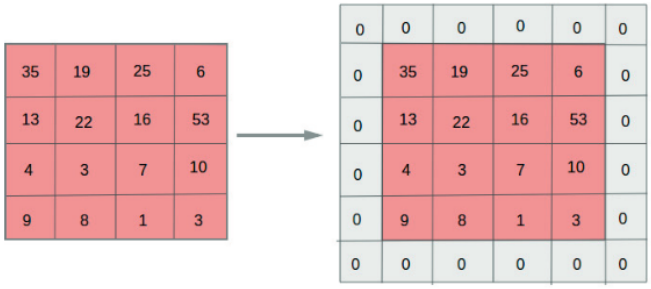
Rozměry výstupní matice, jsou dány vztahem:

$$W_2 = (W_1 - F + 2P)/S + 1 \tag{3.18}$$

$$H_2 = (H_1 - F + 2P)/S + 1 \tag{3.19}$$

$$D_2 = K \tag{3.20}$$

kde vstupní matice je o rozměrech $W_1 \times H_1 \times D_1$ a výstupní matice má rozměry $W_2 \times H_2 \times D_2$, K je počet filtrů, F je rozměr filtru v dané dimenzi, S je hodnota stride a P reprezentuje hodnotu zero-paddingu.

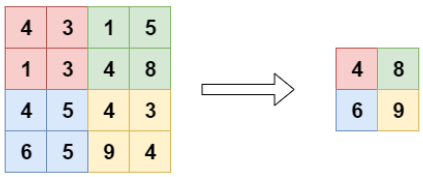


Obr. 3.10: Vizualizace matice se zero-paddingem [20]

3.5.2 Poolingová vrstva

Poolingová vrstva je v CNN umístěna za vrstvou konvoluční. Umožňuje redukci dimenze aktivační mapy, zatímco informace v ní obsažená zůstává zachována. To vede k požadovanému snížení výpočetní náročnosti.

Princip spočívá v procházení příznakové mapy výběrovým filtrem o zvolené velikosti a zvoleném kroku. V každé pozici se z příslušných hodnot spočítá výstup dle zvolené operace. Touto operací může být minimální nebo průměrná hodnota, ale nejčastěji se využívá volba maximální hodnoty, tzv. *Max pooling*.

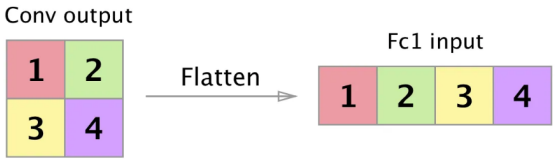


Obr. 3.11: Ukázka fungování Max poolingové operace. Vlevo je vstupní matice na níž je aplikován max pooling s filtrem 2×2 a krokem 2, vpravo je výsledek operace [21].

3.5.3 Flatten vrstva

Tato vrstva se v CNN vyskytuje pouze jednou. Její funkcí je transformace výstupu extraktoru příznaků CNN, který je tvořen konvolučními a poolingovými vrstvami,

na vstupní vektor do klasifikační části tvořené plně propojenými vrstvami. Příklad takové operace je zobrazen na obrázku 3.12



Obr. 3.12: Ukázka principu fungování flatten vrstvy [22].

3.5.4 Plně propojená vrstva

Plně propojené vrstvy jsou ve struktuře CNN za konvolučními a poolingovými vrstvami a tvoří klasifikační část CNN. V těchto vrstvách jsou neurony propojeny s každým neuronem z minulé vrstvy.

V poslední vrstvě neuronové sítě je možné spolu s One Hot Encoding výstupem použití aktivační funkce *Softmax*, jejíž vztah je uveden v rovnici 3.21

$$f(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_n}}, i = 1, ..., N \tag{3.21}$$

kde N je počet neuronů ve výstupní vrstvě. Tato funkce normalizuje výstupní vektor neuronové sítě a zároveň zvýrazní maximální hodnoty výstupního vektoru, přičemž součet výstupního vektoru je roven 1.

4 Snímací pracoviště a výběr hardwarových komponent

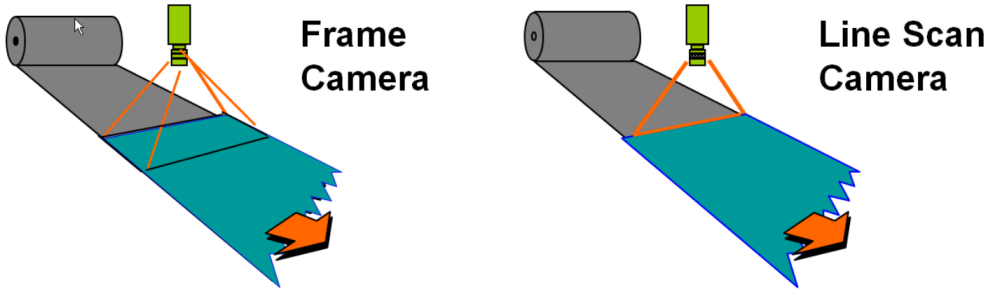
4.1 Kamera

Důležitým aspektem ve všech úlohách počítačového vidění je výběr vhodné kamery dle charakteru řešeného problému. V této úloze je využito řešení s řádkovou kamerou.

4.1.1 Řádková kamera

Řádková kamera se od klasické maticové kamery liší nejvíce v topologii senzoru. Zatímco klasická maticová kamera má snímač o rozlišení $m \times n$ pixelů, u řádkové kamery má rozměr $1 \times n$, čili je tvořen jedinou linií obrazových bodů. Dvojrozměrný snímek je pak vytvořen skládáním jednotlivých řádků do jedné obrazové matice.

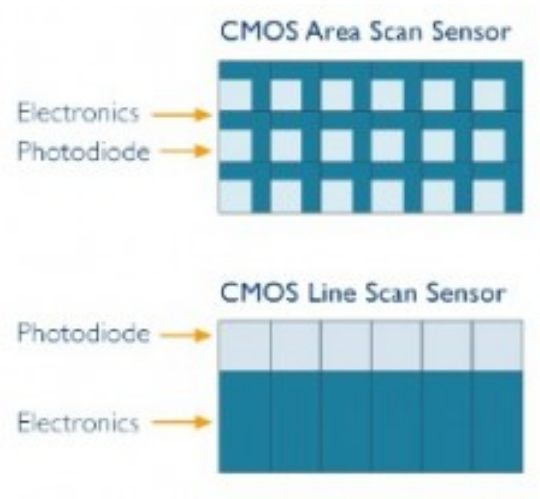
Oproti maticové kameře, ale vyžaduje, aby se buď samotná kamera nebo objekt pohybovaly.



Obr. 4.1: Zobrazení zorného pole maticové a řádkové kamery [23]

Řádkové kamery mají ve srovnání s maticovými kamerami výhodu v obrovském rozlišení ve snímané dimenzi. Abychom dosáhli srovnatelného rozlišení u maticové kamery, byl by potřeba velkoformátový snímač, u kterých ovšem bývají některé pixely vadné, což v důsledku snižuje kvalitu snímku a případně může maskovat chybná místa na potisku při procesu detekce vad. Navíc klasická kamera by vyžadovala velmi krátký expoziční čas, a tedy velmi silné zábleskové světlo, aby nedocházelo vlivem pohybu tělesa k rozmazání snímků. V případě řádkové kamery můžeme s výhodou použít méně výkonné liniové světlo, které celý svůj výkon soustředí do jedné úzké linky namísto světla prostorového.

Dalším přínosem řádkové kamery je lepší světelná citlivost. U každého maticového snímače je nutné kromě světlocitlivých buněk na plochu pixelu umístit i zpracovávající elektroniku. Tím se tedy redukuje plocha světlocitlivé části pixelu. U řádkových snímačů je elektronika umístěna bokem od jednotlivých pixelů, plocha pixelu je tedy maximálně zastavěna světlocitlivou částí, a proto dosahují čipy řádkových kamer v porovnání s maticovými až dvakrát vyšší citlivosti.



Obr. 4.2: Uspořádání fotocitlivých buněk a zpracovávající elektroniky u maticového a řádkového obrazového snímače [24].

Řádková kamera se tedy hodí na aplikace, kde je vyžadováno vysoké rozlišení nebo vysoká snímací rychlost, jako jsou např. kontrola textilií, kovů, kontrola na otáčených objektech nebo skenování pohybujících se objektů na železnicích nebo silnicích.

4.1.2 Použitá kamera Basler racer raL6144-16gm

Pro tuto úlohu detekce vad mi byla vedoucím práce zapůjčena kamera Basler racer – raL6144-16gm.

Jedná se o monochromatickou řádkovou kameru se snímací frekvencí až 17 kHz. Dosahuje maximálního rozlišení až 6144 px s hloubkou pixelu 8/12 bitů. Kamera má tři možnosti synchronizace, a to buď softwarový nebo hardwarový trigger anebo volně běžící režim. Má jedno pevné rozhraní, a to GigE. Parametry snímače uvádím v tabulce níže.

Obr. 4.3: Použitá kamera Basler racer raL6144-16gm [25]

Tab. 4.1: Parametry snímače použité kamery [25]

Resolution (HxV)	6144 px x 1 px
Sensor Vendor	Awaiba
Sensor	DR-6k-7
Shutter	Global Shutter
Sensor Type	CMOS
Sensor Size	43 mm
Pixel Size (H x V)	7 µm x 7 µm

4.2 Objektiv

Protože je využíván objektiv spolu s řádkovou kamerou jsou od něj vyžadovány specifické vlastnosti. U řádkových kamer není potřebné vysoké rozlišení objektivu. Co je daleko důležitější je schopnost vykreslit obraz i na snímač o velikosti až několika centimetrů. A to při co možná nejmenším vlivu vinětace a ztmavování okrajů řádku. Také je kladen velký důraz na rovnoměrné rozložení jasu a kontrastu v celém zorném poli.

Neboť budu snímat rotující objekt v konstantní vzdálenosti, není kladen důraz na možnost zoom. Naopak objektivy s pevným ohniskem ve srovnání se zoomy lépe kreslí. Zvolil jsem tedy objektiv Canon 50 mm f/1,8 II.

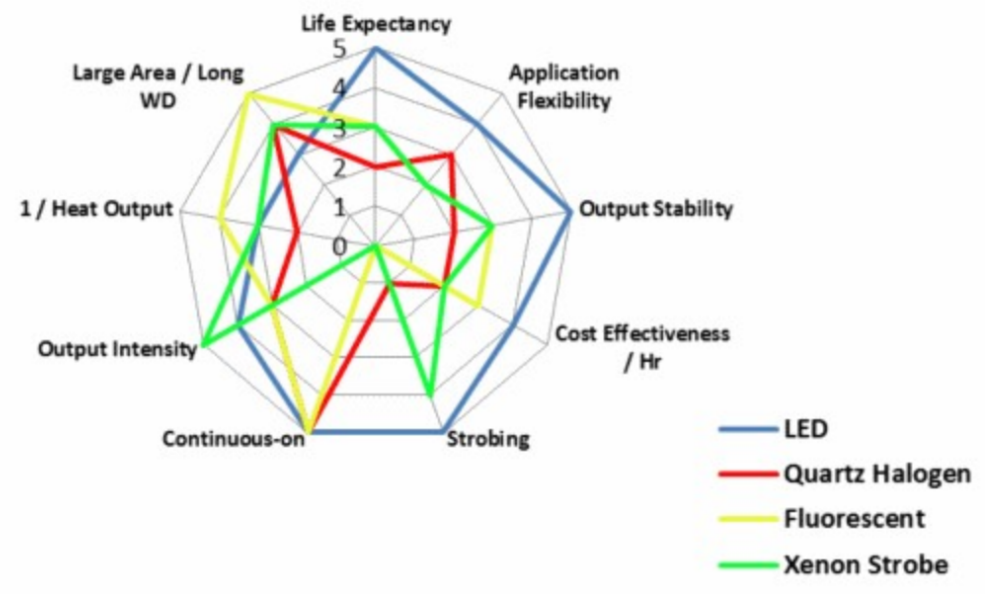
Obr. 4.4: Použitý objektiv Canon 50 mm f/1,8 II [26]

4.3 Osvětlení

Osvětlovače pro použití s řádkovou kamerou musí splňovat několik základních požadavků. V první řadě je to vysoká intenzita osvětlení kvůli velmi krátkým expozičním časům v řádech jednotek až desítek mikrosekund. Té se dosahuje zpravidla usměrněním světelného toku pomocí čočky nebo zrcadla. Zadruhé je nutné rovnoměrné osvětlení zorného pole a stálost parametrů osvětlovače v čase [27].

4.3.1 Zdroje světla

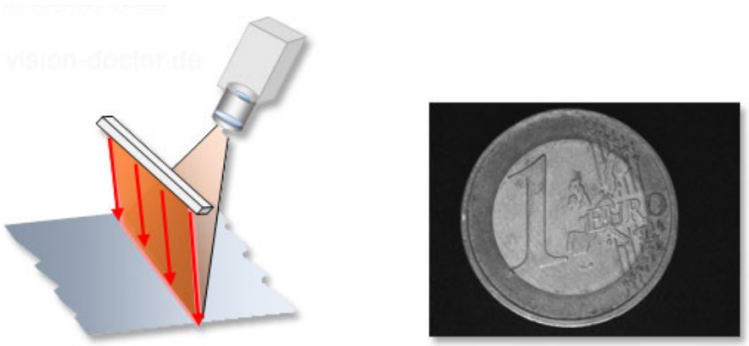
Fluorescenční, halogenové a LED jsou nejčastějšími druhy o osvětlení v oblasti strojového vidění zvláště v oblastech malých a středních kontrolních aplikací. Metalhalidové, xenonové a vysokotlaké sodíkové výbojky se pak zpravidla používají v aplikacích s požadavkem na velkou jasovou intenzitu světla. V grafu níže jsou zobrazeny jednotlivé zdroje světla s ohledem na dané požadavky.



Obr. 4.5: Graf jednotlivých světelných zdrojů v závislosti na různých požadavcích [28]

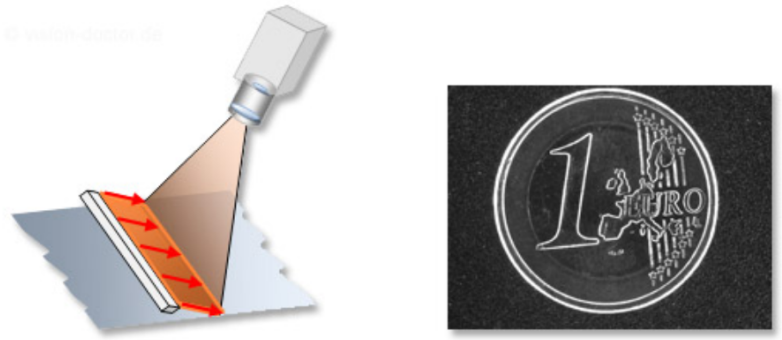
4.3.2 Geometrie osvětlení

Přímé incidentní světlo



Obr. 4.6: Geometrie přímého incidentního světla [29]

Výhody:	Dobré na odrazivých materiálech
Nevýhody:	Při drsném povrchu vznikají stíny

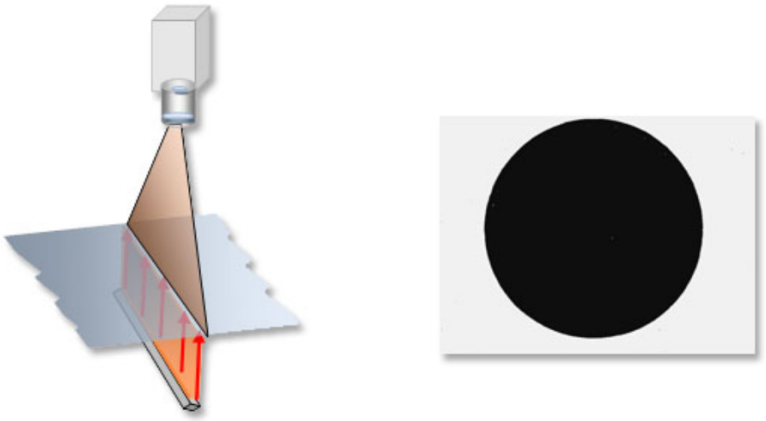


Obr. 4.7: Geometrie osvětlení v temném poli [29]

Osvětlení v temném poli

Výhody:	Vhodné pro kontrolu vad rovného povrchu Vysoký kontrast
Nevýhody:	Pouze pro úzký výběr aplikací Nevhodné na hladkém povrchu Vyžadován velmi výkonný zdroj světla

Zadní světlo (transmisní)

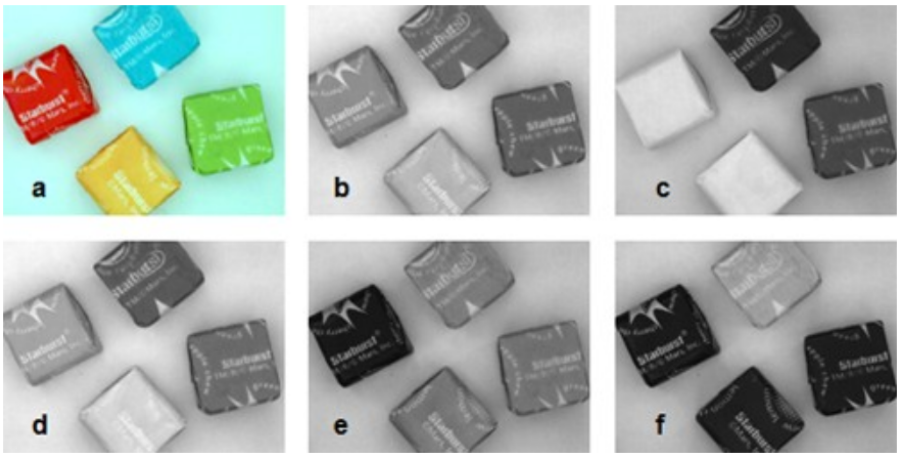


Obr. 4.8: Geometrie zadního světla [29]

Výhody:	K získání obrysu objektu – ostré hrany K měření rozměrů objektu U transparentních objektů
Nevýhody:	Je vyžadován prostor pod objektem

4.3.3 Volba vlnové délky osvětlení

Pro maximalizaci kontrastu na snímaném objektu je vhodné zvolit vhodnou barvu osvětlení, neboť různé barvy absorbují, resp. odrážejí odlišně světlo o různých vlnových délkách. Na obrázku níže je zobrazena stejná scéna různobarevných objektů v kombinaci s odlišnou barvou osvětlení v jednotlivých snímcích.



Obr. 4.9: Scéna s odlišným barevným osvětlením: (a) bílé světlo, barevná ccd kamera, (b)bílé světlo, monochromatická kamera – rovněž i v dalších případech, (c) červené světlo, (d) červené a zelené světlo, (e) zelené světlo, (f) modré světlo [28]

4.3.4 Zvolené osvětlení

U monochromatických kamer se zpravidla používá monochromatického světla (červená, zelená, ...), spíše než bílého, aby se eliminoval vliv barevné vady optiky.

Pro můj úkol se nejlépe hodí přímé LED liniové osvětlení. Zvolil jsem tedy LED pásek RGB 5050 EPISTAR 96LED 23W 24V, s ohledem na co možná nejvyšší světelný výkon a také co nejvyšší hustotu led, aby bylo zajištěno maximálně rovnoměrné osvětlení ve všech bodech snímaného objektu. LED čipy na pásku jsou RGB, takže je zde možnost si podle barvy objektu vybrat nejvhodnější světelné spektrum.

4.4 Pohon

4.4.1 Výběr typu motoru

Bylo zvažováno několik koncepcí otáčení těla fixy, z nichž každá s sebou nese určité výhody a nevýhody. Předběžný požadavek na aplikaci byl, aby výsledný snímek měl

vertikální rozměr alespoň cca 1000 px.

Uvažoval jsem tyto možnosti:

- **Krokový motor s využitím mikro krokování**
  Byla by možnost využít hojně rozšířený krokový motor NEMA 17, který se využívá např. v 3D tiskárnách. Motor má 200 kroků na otáčku, tzn. má krok o úhlovém natočení 1,8°. Přidáním vhodného kontroléru lze využít techniku mikro krokování, kdy se spínáním cívek motoru o proměnných napětích, zjemní krok až na dalších cca 64 mikro kroků v rámcic jediného kroku. Tzn. možné horizontální rozlišení by bylo až do 12800 poloh na otáčku. Toto rozlišení by zcela určitě bylo dostatečné. Problém ovšem je v nelinearitě rozdělení mikro kroků, kdy by jednotlivé kroky nebyly v konstantních vzdálenostech. Tuto možnost jsem z tohoto důvodu tedy z uvažování vyřadil.

- **Krokový motor bez převodovky s dostatečným množstvím kroků**
  Tato možnost byla rovněž zvažována, avšak krokové motory s cca 1000 kroky na otáčku jsou velmi drahé, pakliže je už některý z mála výrobců vyrábí.

- **Krokový motor s převodovkou**
  Jako vhodnější přístup se zdálo využití dostupného krokového motoru s klasickými 200 kroky na otáčku a následným zpřevodováním na cca 5x vyšší rozlišení. Obecně u krokových motorů hrozí riziko, že vlivem půlsního ovládání motoru, tedy nekonstantních otáček motoru s proměnným zrychlením, že daný řádek pixelů bude opět snímán s rozdílnými polohovými diferencemi mezi jednotlivými polohami. Je také možné, že z důvodu proměnné akcelerace při nevhodně zvolené expoziční době budou některé řádky rozmazané a některé nikoliv a tento problém půjde špatně identifikovat. Poslední riziko je, pokud by byl vzhledem k danému motoru příliš velký zátěžný moment, tak by krokový motor mohl vynechat jeden či několik kroků.

- **DC motor s převodovkou**
  Možnost s využitím zpřevodovaného DC motoru je spojena se skutečností, že řádková kamera nemusí být nutně trigerovaná externím signálem, jako u krokových motorů, kde je jeden řádek snímku pevně spojen s danou polohou natočení krokového motoru. Kromě možnosti externího trigerování lze využít i volnoběžný mód, který nabízí větší možnost variability rozměru snímku možností nastavení snímkovací frekvence podle budoucích požadavků na zpracování obrazu a vyhodnocení vad. To je ovšem podmíněno nutností hladkého

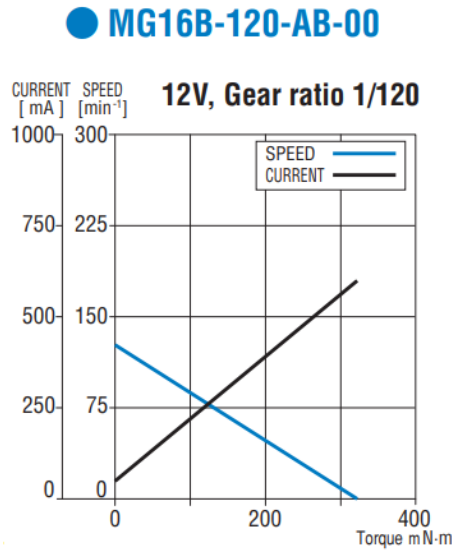
chodu motoru s konstantní rychlostí otáčení. Tento faktor plynulého chodu je podpořen díky přesnému zdroji napájecího napětí, ke kterému je motor připojen a dále pak převodovkou DC motoru, která poměrově zmenší drobné odchylky v otáčkách.

4.4.2 Zvolený motor

Nakonec po zvážení všech těchto možností jsem se rozhodl použít DC motor s převodovkou. Jako vhodný motor se zdál japonský Nidec MG16B-120-AB-00.

Obr. 4.10: Použitý motor Nidec MG16B-120-AB-00 [30]

Převodový poměr má 1/120, což by mělo být dostatečné pro případnou redukci výkyvů rychlosti způsobené kolísáním napájecího napětí.



Obr. 4.11: Graf otáček motoru v závislosti na zatěžovacím momentu je lineární [30]

Motor má jmenovitý moment 60 mNm, maximální moment je 90 mNm. Motor budeme v naší aplikaci v porovnání s uvedenými momenty provozovat téměř nezatížený. Otáčky nezatíženého motoru jsou 127 rpm, zatíženého na jmenovitý moment pak 100 rpm. Tyto hodnoty otáček se zdají být odpovídající typu řešené aplikace.

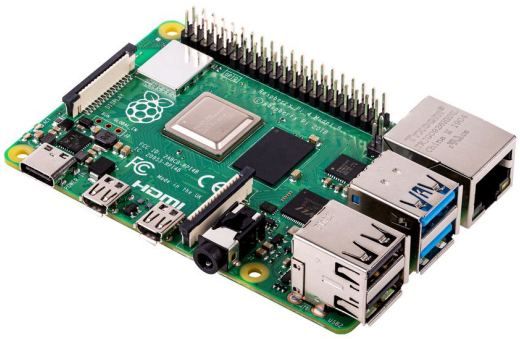
4.5 Raspberry Pi

Díky technologickému pokroku v oblasti informačních technologií a masivnímu rozvoji embedded elektroniky jsme se dostali do fáze, kdy i malá krabička jen o něco málo větší než je např. kreditní karta může velice dobře suplovat a v mnohém i rozšířit funkci stolního počítače. Takovým příkladem je právě i Raspberry Pi (dále jen RPi). Tento první embedded prototyp byl vyvinut nadací Raspberry Pi Foundation v roce 2012. Původně byl určen k podpoře výuky programování na středních školách, ale pro svou nízkou cenu, všestranost a možnost připojení různých periferií a rozšiřujících přídavných modulů se brzy uchytil i na poli vývoje a domácích projektů.

Tento pilotní projekt později inspiroval k vývoji dalších zařízení jako např. Banana Pi, Ondroid, Khadas, Nvidia Jetson, které se blíže zaměřují na práci ve specifických oblastech techniky.

4.5.1 Rasberry Pi 4 model B

Celé pracoviště je třeba řídit, aby vykonávalo požadovanou funkci, tzn. spínat jednotlivé výkonové členy, obsluhovat kameru, zajistit synchronizaci procesů a také s dostatečným výkonem zajistit výpočet algoritmů.



Obr. 4.12: Raspberry Pi 4B [31]

K tomu jsem zvolil právě Raspberry Pi 4 model B, která po zvážení výhod a nevýhod vyšla z uvažovaných možností nejlépe. Mezi požadavky na výběr byly aspekty:

- Operační systém musí podporovat Pylon 5, tedy framework k nastavování a ovládání řádkových kamer Basler s možností vizualizace v GUI Pylon Viewer
- Zařízení musí obsahovat nebo poskytovat připojení digitálního rozhraní ke spínání výkonnových členů
- Zařízení musí mít gigabitový ethernet ke grabbování obrázků z kamery
- Zařízení poskytuje dostatečný výpočetní výkon pro zpracování obrazu v přijatelném čase
- Existence dokumentace a uživatelské komunity

Tyto požadavky RPi 4B splňuje.
V tabulce níže uvádím technické parametry použitého RPi zařízení.

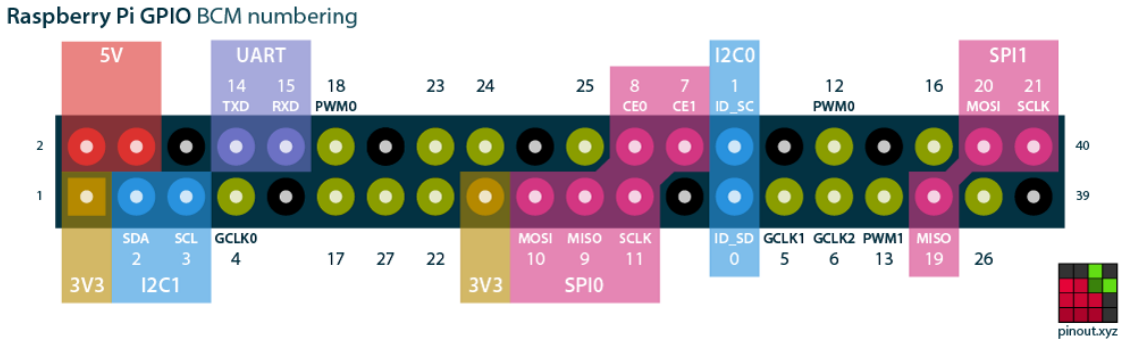
Tab. 4.2: Raspberry Pi 4 model B - technická specifikace

Parametry	Raspberry Pi model B
Model CPU	Cortex-A72 (ARM v8) 64-bit
Jader CPU	4
Frekvence CPU	1.5 GHz
SoC	Broadcom BCM2711
RAM	4 GB LPDDR4-3200 SDRAM
Conektivita	2.4 GHz a 5.0 GHz bezdrátově
	LAN, Bluetooth 5.0, BLE
	Gigabitový Ethernet
	2 × USB 3.0 port
	2 × USB 2.0 port
Video	2 × micro HDMI port (do 4Kp60)
	2-lane MIPI DSI display port
	2-lane MIPI CSI camera port
Zvuk	HDMI, 3,5 mm jack
GPIO	40-pin GPIO
Datové úložiště	Micro SD
Napájení	5V DC přes USB-C konektor (min. 3A)
	5V DC přes GPIO (min. 3A)

4.5.2 GPIO

Oproti klasickému stolnímu počítači má RPi výhodu v zabudovaných GPIO (resp. General Purpose Input / Output) pinech. Není tedy třeba tak jako u stolního počítače připojovat vstupně výstupní karty a moduly. Tyto piny slouží pro komunikaci s okolními zařízeními na nižší úrovni.

Konektor obsahuje celkem 40 pinů. Obsahuje piny napájecí 5V a 3,3V DC, zem a piny vyhrazené pro EEPROM. Ostatní piny jsou uživatelsky nakonfigurovatelné, zda budou vstupní nebo výstupní, s pull-up nebo pull-down rezistorem. Některé piny navíc mají možnost konfigurace jako UART, I2C nebo SPI.

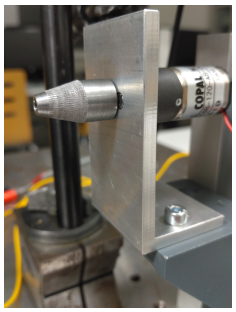


Obr. 4.13: GPIO piny na Raspberry Pi 4B [32]

4.6 Mechanické části - otočný a upínací mechanismus

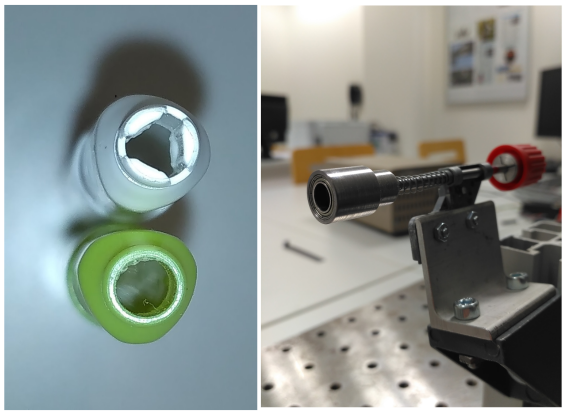
Struktura otočného mechanismu je principiálně podobná soustruhu. Obsahuje tedy dvě hlavní části, a to vřeteno – aktivní část, která roztáčí snímaný objekt, a koník – pasivní část upevňující hrot fixy.

Vřeteno je tvořeno pravoúhlým hliníkovým profilem, který je upevněn k nosné konstrukci. Na tento profil je ze zadní strany připevněn motor, jehož hřídel prochází skrz profil. Na hřídeli je přišroubovaný vysoustružený hliníkový hrot, jehož povrch je pro lepší přilnavost k válcovému tělu snímané fixy zdrsněn.



Obr. 4.14: Detail vřetene se zdrsněným hrotem

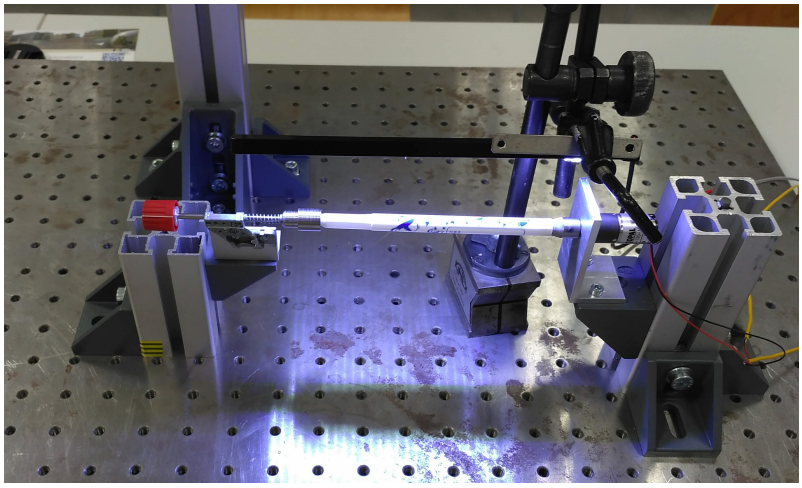
Koník zde má funkci pohodlného upínání fixy v definované poloze. Protože některé fixy mají na své hrotové části kulatý průřez a jiné zase trojúhelníkovitý, rozhodl jsem se upínat fixy nikoli pomocí hrotu jako na druhé části, ale po obvodu z vnějšku.



Obr. 4.15: Vpravo detail zakončení hrotové části fixy, vlevo detail koníku s ložiskem pro uchycení fixy z vnějšku

Tímto unifikujícím krokem není potřeba používat různé nástavce pro různé typy zakončení fix, ale pouze jeden univerzální kalíšek.

Je důležité, aby koníček kladl co možná nejmenší odpor rotující fixe, to je zajištěno ložiskem zalisovaným do vysoustruženého hliníkového kalíšku. Stejný přítlak pro všechny tužky, a tím opět i stejný moment, zajišťuje přítlačná pružinka koníčku. Tento mechanismus kromě zmíněných výhod má výhodu jednoduché a rychlé manipulace při výměně vzorků fix.

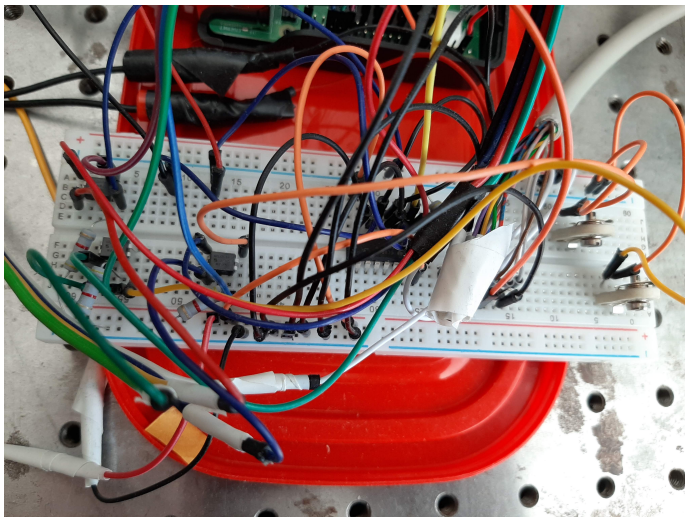


Obr. 4.16: Přehledový snímek otáčecího mechanismu

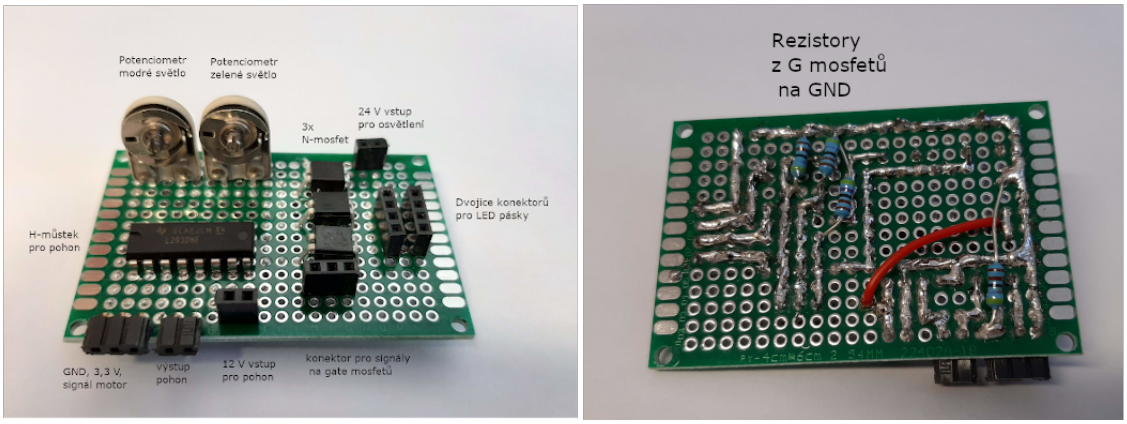
4.7 Elektrická část - řídící obvody

Elektrické obvody byly prvotně navrhovány na nepájivém poli. Po navržení byly obvody osazeny na univerzální pájivé pole.

DPS obsajuje 3 větve, na kterých jsou spínány N-Mosfet tranzistory RGB kanály led pásku. Pro vyrovnání světelné intenzity všech kanálů byly do větve zeleného a modrého osvětlení, které byly silnější než červené, přidány potenciometry. Dále byl na DPS umístěn integrovaný obvod H můstku pro spínání motoru. Deska je následně doplněna dutinkovými lištami pro přivedení 12 V a 24 V napájení, vstupy a výstupy motoru a led pásků a řídících signálů z RPi.



Obr. 4.17: Nepájivé pole v průběhu procesu navrhování řídících obvodů



Obr. 4.18: Deska plošných spojů s řídící elektronikou

5 Softwarové vybavení

5.1 Rasbpian

Pro Raspberry Pi je možnost několika operačních systémů:

1. RASPBIAN
2. Ubuntu 19.10
3. RISC OS
4. Windows IoT Core
5. OSMC
6. OpenELEC
7. RaspBSD

a další méně rozšířené.

Pro tuto práci byl zvolen operační systém Raspbian Buster. Je to systém vycházející z Linux Debian, který je optimalizovaný pro použití na zařízeních Rasberry Pi. Tento systém je poskytovaný samotným výrobcem Rasbperry Pi, tedy Rasberry Foundation, který zaručuje podporu pro všechny řady a verze RPi a který je volně dostupný na webových stránkách této organizace. Tím, že je tento systém primárně určen pro RPi, je zajištěna maximální kompatibilita s dostupným hardwarem a aplikacemi. Další nespornou výhodou je i jeho masivní rozšíření, zvláště na zařízeních RPi, díky němuž je tento systém dobře odladěný a existuje velké množství informací, návodů a rad, které lze na internetu k případnému problému dohledat. Toto všechno jsou výhody, díky nimž byl po řádném zvážení zvolen právě tento operační systém.

Raspbian Buster je možné stáhnout ve dvou verzích, a to Raspbian Buster Desktop anebo Raspbian Buster Lite. Verze Raspbian Lite je odlehčenou verzí plné distribuce Raspbian Desktop. Liší se od plné verze v tom, že neobsahuje některé aplikace jako je například Libre Office a Wolfram Alfa. Postrádá také GUI a lze tedy ovládat pouze přes příkazovou řádku. Pro účely této úlohy byla zvolena plná distribuce.

Zvolenou distribuci je možné instalovat buďto přímo anebo pomocí tzv. NOOBS (New Out Of the Box Software), který obsahuje zavaděč všech dostupných operačních systémů a po nabootování si uživatel operační systém zvolí.

Operační systém je na zařízeních RPi uložen na SD kartách. Na kartě může být v daném okamžiku nainstalovaný vždy pouze jeden operační systém, proto instalací nového se starý operační systém smaže.

Pro nahrání staženého Image operačního systému bylo použito softwaru Balena Etcher, kterým byl flashnut image systému na SD kartu. Následně byl systém stan-

dardně nainstalován dle nabídnutého wizardu a též byla provedena prvotní konfigurace.

5.2 Použité knihovny

V této podkapitole jsou uvedeny nejdůležitější knihovny, které byly pro práci využity.

5.2.1 WiringPi

Knihovna WiringPi je napsaná v jazyku C a zajišťuje přístup k BCM2535 a BCM2836 SoC zařízením, mezi která patří i Raspberry Pi[33]. Zajišťuje přístup k GPIO pinům v jazycích C, C++ a RTB(BASIC). Její instalace se provede následujícími příkazy:

```
> sudo apt-get install git-core
> git clone git://git.drogon.net/wiringPi
> cd wiringPi
> git pull origin
> ./build
```

5.2.2 OpenCV

Knihovna OpenCV (Open Source Computer Vision) je jednou z nejznámějších a nejrozšířenějších knihoven v oblasti zpracování obrazu. Na jejím vývoji začal pracovat Intel v roce 1999 a první release byl v roce 2000. Od té doby vychází se update nové knihovny objevuje každého půl roku.

Knihovna je opensource pod licencí BSD, je podporována v jazycích C, C++, Java a Python a může běžet na platformách operačních systémů Windows, Linux, iOS, OS X, Android, ... Ve vývoji je pro akceleraci výpočtů na grafických kartách aktuálně rozhraní založené na architekturách OpenCL a CUDA.

Knihovna má modulární strukturu a jsou v ní obsažené moduly:

- **core** - Modul definující základní datové struktury, včetně vícedimenzionálních polí Mat a základních funkcí
- **imgproc** - Modul zaměřený na zpracování obrazu včetně například lineární a nelineární obrazové filtrace, geometrických obrazových transformací, konverzí mezi různými barevnými modely, histogramy a další.
- **video** - Modul zaměřený na analýzu videa jako je například estimace pohybu nebo sledování objektů.

- **calib3d** - Modul zaměřený na kalibraci kamer. Obsahuje algoritmy zabývající se geometrickými vztahy při snímání scény více kamerami, kalibrací standardní a stereo kamery, korespondenčními algoritmy, 3D rekonstrukcí.
- **reatures2d** - Modul obsahuje algoritmy detekce význačných prvků, deskriptory a jejich porovnávání.
- **objdetect** - Modul zaměřený na detekci předdefinovaných tříd, jako jsou například obličej, oči, lidé, auta, ...
- **highgui** - Modul zaměřený na vytváření jednoduchých grafických rozhraní.
- **videoio** - Modul obsahující kodeky a rozhraní pro pořízení videa.
- a další pomocné moduly

V této práci byla využita nejnovější možná verze, totiž 4.3.0.

Instalace OpenCV

Před instalací OpenCV je třeba doinstalovat potřebné moduly:

Tímto příkazem se zkontrolují dostupné aktualizace firmwaru RPi a případně se doinstalují.

```
> sudo apt-get update && sudo apt-get upgrade
```

Dalším příkazem se nainstalují developerské nástroje včetně CMake.

```
> sudo apt-get install build-essential cmake pkg-config
```

Budeme potřebovat balíčky, které umožní vstupně výstupní práci s obrázky, jako je například načtení obrázku.

```
> sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev
libpng-dev
```

Dále také doinstaluji balíčky pro vstupně výstupní operace s videem.

```
> sudo apt-get install libavcodec-dev libavformat-dev
libswscale-dev libv4l-dev
> sudo apt-get install libxvidcore-dev libx264-dev
```

Výpočty některých operací, které využívá OpenCV (jako například maticové násobení) můžou být optimalizovány.

```
> sudo apt-get install libatlas-base-dev gfortran
```

Dále pro vytváření GUI.

```
> sudo apt-get install libqtgui4 libqtwebkit4 libqt4-test
python3-pyqt5
```

Nakonec je nainstalován balíček hlaviček Python 3 pro kompilaci OpenCV s Python vazbami.

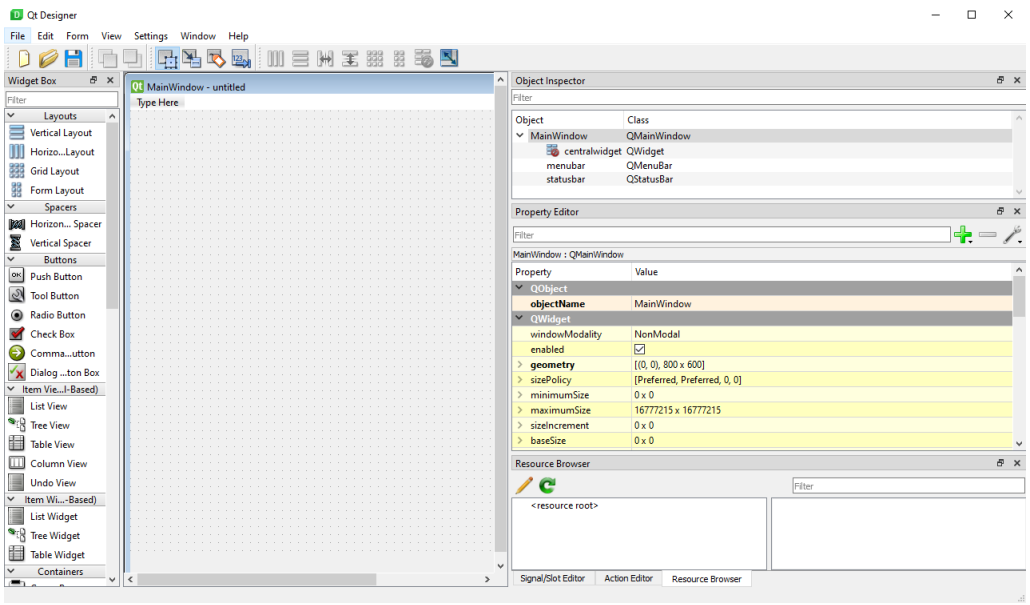
```
> sudo apt-get install python3-dev
```

Po instalaci těchto balíčků jsou připraveny všechny prerekvizity pro instalaci OpenCV. Dalším příkazem nainstalujeme přílušnou verzi. V tomto případě 4.1.0.25.

```
pip install opencv-contrib-python==4.1.0.25
```

5.2.3 PyQt5

Qt je soubor knihoven v jazyku C++ a vývojových nástrojů vyvíjený společností Digia. Knihovny jsou nezávislé na platformě a poskytují abstrakci, která umožňuje vývoj především grafického uživatelského rozhraní, ale do projektu může být zahrnuta i práce se sítěmi, vlákny, regulárními výrazy, SQL databázemi, SVB, OpenGL, XML, nastaveními uživatelských aplikací, krátkodosahovými komunikacemi jako např. NFC a Bluetooth a přístup ke cloudům[34].



Obr. 5.1: Zobrazení okna vývojového editoru QtDesigner.

PyQt5 je soubor rozšiřující framework Qt5 o Python uživatelské rozhraní, který je vyvíjen britskou společností Riverbank Computing. Je dostupný jak pro Python 2.x, tak i pro nový Python 3.x. PyQt5 má modulární architekturu, která obsahuje asi 15 základních modulů zaměřených na danou funkcionalitu. Celkově knihovna

obsahuje přes 600 tříd a 6000 funkcí a metod. Stejně tak jako Qt je multiplatformní a běží na operačních systémech Unix, Windows, Mac OS.

PyQt5 poskuje nástroj „QtDesigner“, který umožňuje navrhnout Front-end aplikace v grafickém editoru. To umožňuje rychlejší a intuitivnější vývoj front-endu. Na obrázku 5.1 je zobrazen tento nástroj.

Výstupní soubor QtDesigneru je uložen jako soubor s příponou .ui. Aby bylo možné dále naprogramovat i backend aplikace je nutné tento soubor transformovat kódu, v našem případě do Python skriptu. To je provedeno příkazem:

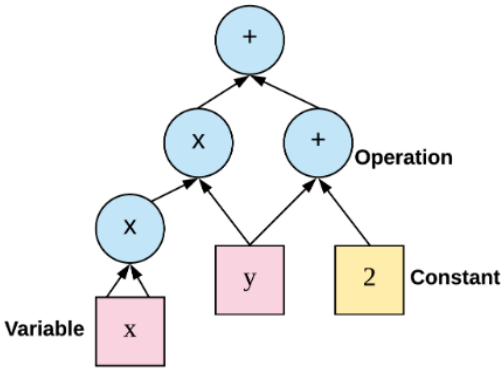
```
pyuic5 -x vstupni_soubor.ui -o vystupni_soubor.py
```

5.2.4 TensorFlow

TensorFlow je multiplatformní open-source softwarová knihovna pro numerické výpočty orientované především na oblast strojového a hlubokého učení.

Knihovna je vyvíjena výzkumínky z Google Brain Team. Její první release byl v listopadu 2015 a od té doby se stále vyvíjí. Důležitým mezníkem byl říjen 2019, kdy vyšla nová verze TensorFlow 2.0, která přinesla řadu vylepšení, např. zjednodušení práce s API, přidání rozšiřujícího modulu TensorFlow Lite, ...

Programy jsou psány formou výpočetního grafu, kde jednotlivé uzly reprezentují dané matematické operace a spojení data, resp. toky dat mezi uzly. Pro ilustraci je na obrázku 5.2 zobrazena rovnice $f(x, y) = x^2y + y + 2$ ve tvaru výpočetního grafu.



Obr. 5.2: Zobrazení rovnice ve tvaru výpočetního grafu[35]

Knihovna má API v jazyku Python, ale kvůli maximální rychlosti vykonávání je knihovna implementována v C++.

Výhodou knihovny je možná volba vhodné úrovně abstrakce. Program lze psát buďto na nejnižší úrovni v samotném TensorFlow nebo lze zvolit vyšší úroveň jako

je „Distribution Strategy API“ nebo ještě vyšší úroveň poskytuje Keras API. Právě úroveň Kerasu byla zvolena pro tuto práci.

5.3 Pylon Camera Software Suite

Pylon Camera Software Suite je softwarový balíček obsahující SDK, ovladače, a nástroje k nastavení a řízení kamer od výrobce Basler. Tento balíček je dostupný pro platformy Windows, Mac, Linux - x86 i ARM. Podle zvolené platformy má SDK aplikační rozhraní programovatelné v některých z jazyků C, C++, C#, Visual Basic, .Net a dle platformy podporuje připojení pomocí CoaXPress 2.0, GigE, USB 3, Camera Link, GenICam 1.5, BCON.

Zároveň nástroj obsahuje i aplikaci Pylon Viewer, která umožňuje jednoduché a rychlé nastavení a vyzkoušení parametrů v grafickém prostředí. Tato aplikace přirozeně nemůže obsáhnout všechny funkcionality kamer, ale je vhodná pro rychlé naladění základních parametrů.

6 Akvizice snímků

6.1 Řízení a synchronizace kamery a osvětlení

6.1.1 Řízení kamery

Výrobce použité řádkové řádkové kamery dává možnost řídit kameru pomocí definovaného API. To je možné zvolit v jazycích C, C++ nebo C#. Vzhledem k tomu, že celý systém je řízen z Linux Rasbian OS, byly možnosti redukovány pouze na první dvě zmíněné, z nichž jsem zvolil C++.

Kromě toho lze nainstalovat i GUI Pylon Viewer, kde lze velice rychle a efektivně vyzkoušet velkou část z možných volitelných nastavení. Výsledky této akvizice lze s výhodou okamžitě vidět ve vizualizačním okně aplikace.

Přes zmíněné aplikační rozhraní lze nastavit parametry jako frekvenci snímání, expoziční dobu, zesílení, rozměry posílaného rámce, režim spouštění expozice nebo slučování sousedních pixelů.

Řádková kamera neposílá jako výstup jednolivé řádky rovnou tak, jak je exponuje. To by nebylo příliš praktické a ani by takový přenos nebyl při vyšších snímacích frekvencích možné, ale posílá tzv. rámce. Rámec (též anglicky "frame") je dvoudimenzionální matice obrazových elementů. Ten je seskládán interně v kameře řádek po řádku jak probíhá samotná expozice a po dosažení nastavených rozměrů je tento rámec jednotně dán na výstup.

Spouštění expozice kamery

Kamera umožňuje následující režimy spouštění:

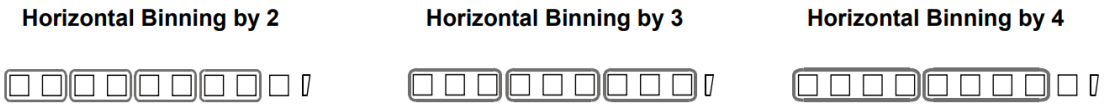
- **Volnoběžný režim**
  Spouštění expozice jednotlivých řádků je realizováno dle nastavené vzorkovací frekvence.
- **Režim externí synchronizace**
  Spouštění expozice (angl. „trigger“) je podmíněno příchodem spouštěcího signálu. A to buď softwarového, který je v kódu realizován formou příkazu a který je komunikován skrze ethernetovou komunikaci, nebo hardwarového signálu formou pulzu na příslušný pin konektoru kamery.

Z podstaty zamýšlené filosofie synchronizace osvětlení a spouštění expozice kamery byl zvolen režim externí synchronizace. Z počátku jsem experimentoval s možností softwarového spouštění pro zdánlivou jednoduchost, ale narážel jsem na problémy se synchronizací. Proto jsem finálně zvolil spouštění hardwarové. Tato možnost nejenže je přímočařejší, ale také nabízí možnost vhledu do reálného dění na

vodičích a kontrole časování skrze osciloskop.

Slučování sousedních pixelů

Kamera má možnost horizontálního sloučení (angl. „binning“) dvou až čtyř pixelů snímače kamery. Tím se zvýší citlivost snímače, což může být zvlášťe vhodné v případě menší míry světla, za cenu nižšího rozlišení kamery.



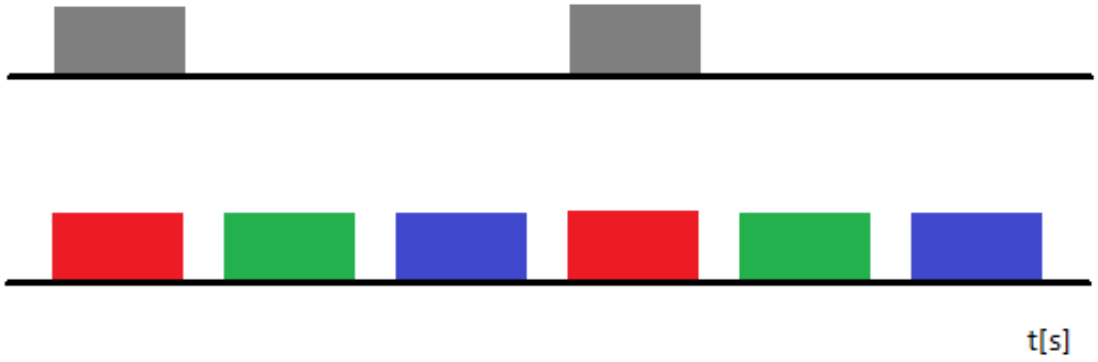
Obr. 6.1: Slučování sousedních pixelů kamery[36].

V této aplikaci bylo využito sloučení tří pixelů jako kompromis mezi navýšením světelné citlivost a velikostí snímku.

6.1.2 Synchronizace kamery s osvětlením

V závislosti na barvě pozadí a potisku konkrétního snímaného předmětu je k pořízení optimálního snímku s maximálním kontrastem zapotřebí jiná vlnová délka světelného záření. Aby se předešlo potřebě zaintegrovat do měřicího pracoviště dodatečný zdroj informace o barvách předmětu, byl navržen postup, kdy dojde k získání hned tří snímků jedné scény za různých světelných podmínek. Tyto tři snímky jsou pořízeny za osvětlení červeným, zeleným a modrým světelným spektrem. Z těchto snímků se následně dle zvoleného kritéria vybere ten nejvhodnější.

V základním módu je vzhledem k rychlosti rotace objektu požadavek na snímací frekvence kamery přibližně 500 Hz. Maximální snímací frekvence kamery je řádově vyšší. K pořízení snímků tedy mohl být navrhnut princip, kdy se zvýší 3x výška grabovacího rámce i frekvence snímání a synchronním přepínáním sekvence RGB složek osvětlení se získá převzorkovaný snímek, ze kterého se následně po jeho dekompozici sestaví tři snímky za červeného, zeleného a modrého osvětlení. Tento proces proběhne v čase pořízení jednoho standardního snímku. Pro vysvětlení je princip zobrazen na obrázku níže.



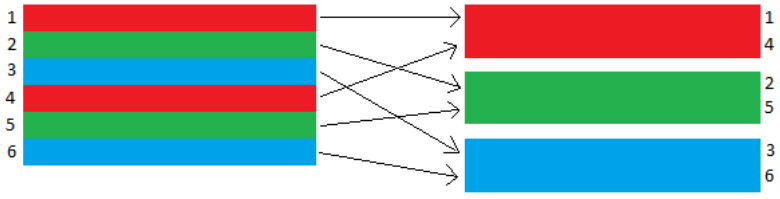
Obr. 6.2: Nahoře je ilustrace časového průběhu expozice řádků s jedním zdrojem světla. Dole zobrazení navrženého principu.

6.2 Kompozice snímků dle vlnové délky osvětlení

Aplikací navrženého akvizičního principu jsou získána surová data, která aby byla smysluplná a dále použitelná je potřeba upravit. Úprava spočívá v dekomponování jednotlivých řádků primárního snímku a vytvoření tří sekundárních snímků se stejným počtem sloupců a třetinovým počtem řádků.

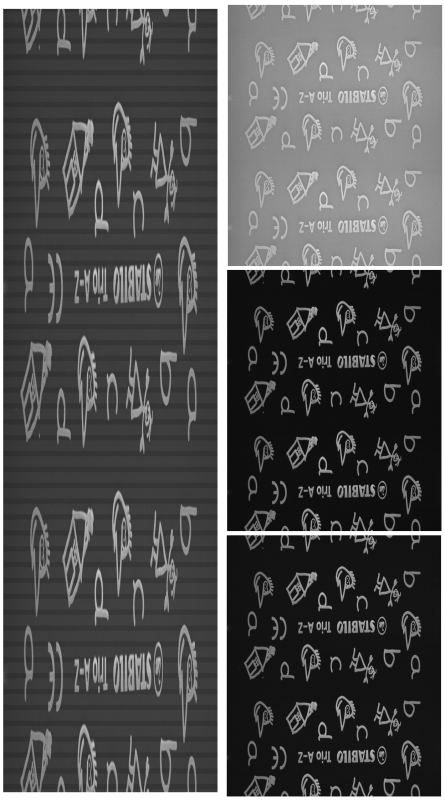
Počet řádků primárního snímku splňuje podmínku dělitelnosti třemi a zároveň v programu je pevně definované pořadí jednotlivých barev osvětlení. Proto lze bezpečně určit příslušnost jednotlivých řádků primárního snímku k danému světelnému spektru a tedy i přidat řádky k odpovídajícímu sekundárnímu snímku.

Princip popisované dekompozice je pro názornost zobrazen na přiloženém obrázku.



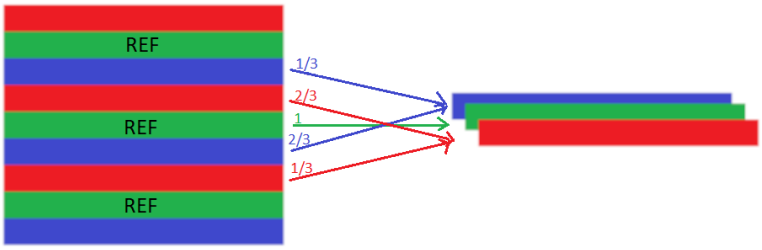
Obr. 6.3: Dekompozice primárního snímku na tři sekundární

Reálný příklad je uveden na obrázku níže.



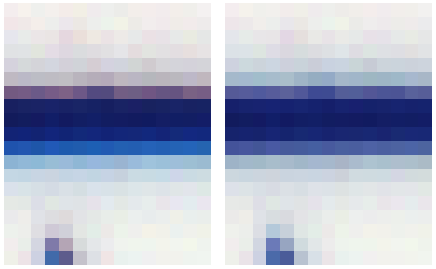
Obr. 6.4: Příklad snímku červené fixy s bílým potiskem - dekompozice primárního snímku na tři sekundární. Vlevo snímek primární, vpravo pak sekundární snímky, zvrchu postupně ve spektru - červeném, zeleném, modrém

Uvedený princip kompozice je vhodný pro pozdější samostatnou práci s jednotlivými kanály. Pokud ale se z jednotlivých kanálů syntetizuje RGB snímek, tak snímek obsahuje na hranách vady chromatické aberace. To je způsobeno tím, že jednotlivé složky RGB pixelu nejsou pořízeny v jednotném čase a tudíž u pohybujícího se objektu neexponují přesně totožnou scénu. Tento jev lze kompenzovat úpravou algoritmu kompozice. Ten byl upraven tak, jako by všechny složky pixelu byly pořízeny v jednom okamžiku. Jako referenční perioda vzorkování byla zvolena perioda expozice za zeleného osvětlení. Řádky exponované touto barvou světla byly převzaty v původních hodnotách. Hodnoty pixelů v řádcích exponovaných za červeného a modrého světla byly spočítány interpolací dvou sousedních řádků pixelů příslušné barvy vzhledem k času expozice referenčního zeleného řádku. Algoritmus je opět pro lepší pochopení zobrazen na obrázku níže.



Obr. 6.5: Princip vylepšené dekompozice primárního snímku na tři sekundární

Na následujícím obrázku je uveden příklad detailu modré vodorovné linky syntetizované pomocí obou přístupů.



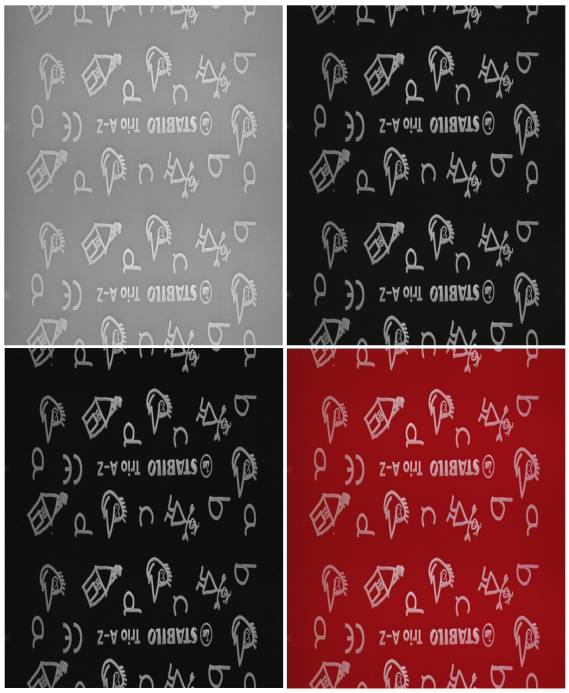
Obr. 6.6: Detailní RGB snímek zobrazující modrou vodorovnou čáru složený klasickou kompozicí - vlevo, a kompozicí s potlačením chromatické aberace - vpravo.

6.3 Rekonstrukce barevného snímku

Barevný snímek lze přirozeně zachytit pomocí kamery se senzorem s RGB pixely. Ovšem stejně tak lze i ze snímků z monochromatické kamery za pomoci drobného postprocessingu barevný obraz zrekonstruovat. Toho bylo s výhodou využito i v této úloze.

K tomu je nutné exponovat monochromatický senzor zabírající jednu scénu postupně s červeným, zeleným a modrým osvětlením a z těchto samostatných obrazových matic o rozměru $m \times n$ vytvořit matici o rozměru $m \times n \times 3$, přičemž jednotlivé složky musí mít indexy v pořadí červená, zelelená, modrá.

V této práci je tato technika s úspěchem využita například ke kontrole pořízených snímků, kalibraci světelného výkonu RGB složek osvětlení, či k reálnému zobrazení vyhodnocovaného objektu.



Obr. 6.7: Příklad kompozice barevného snímku z jednotlivých RGB složek. Tři šedotónové snímky reprezentují RGB složky, ze kterých je zrekonstruován barevný obrázek - vpravo dóle.

6.4 Kalibrace výkonu RGB složek osvětlení

K tomu, aby bylo možné zrekonstruovat barevný snímek v realitě odpovídajících barvách a aby jednotlivé kanály byly mezi sebou kvantitativně porovnatelné pro výběr nejvhodnějšího kanálu s ohledem na barvu snímaného předmětu a jeho potisku, je potřeba zajistit přibližně stejné množství světla dopadajícího na snímaný předmět v rámci expozice všech tří kanálů.

Tato pomínka by byla zajištěna, pokud by led pásek, který byl jako osvětlovač zvolen, vyzařoval stejný světelný výkon ve všech RGB složkách. Bylo zjištěno, že tuto podmínku však nesplňuje.

K vyřešení tohoto problému jde přistoupit dvěma způsoby, a to přidáním sériového odporu k led pásku, pro omezení protékajícího proudu, nebo různou délkou expoziční doby pro jednotlivé RGB kanály osvětlovače.

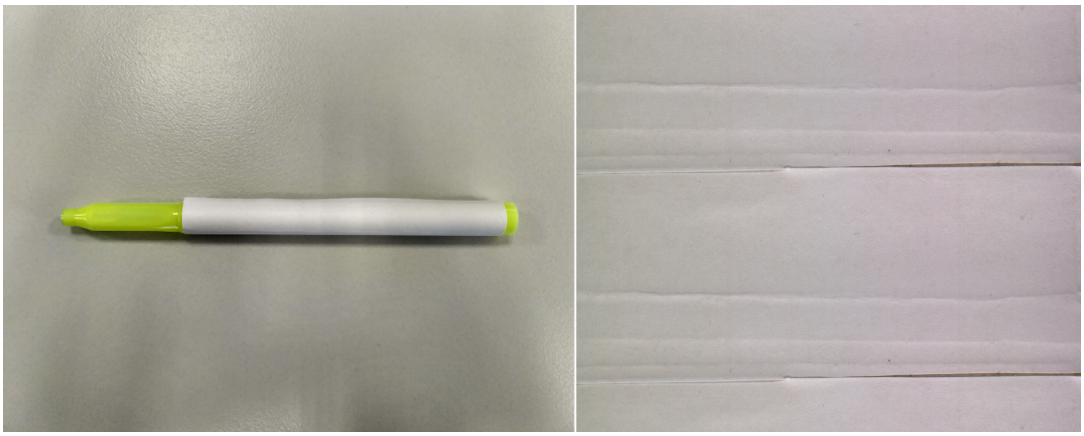
Z počátku byl zvolen druhý způsob, neboť se jevil jako elegantnější pro menší množství potřebných součástek. Výrobce řádkové kamery uvádí, že v režimu externího spouštění expozice kamery lze nastavit délka expozice právě délkou kladné úrovně obdélníkového spouštěcího pulzu. Tato možnost řešení byla pečlivě vyzkoušena, ale z chování kamery se jevilo, jako by expoziční doba byla totožná s periodou

pulzů, nikoliv pouze s kladnou částí expozičního pulzu. Tento problém byl následně konzultován s distributorem kamery, který též po neúspěchu se obrátil přímo na výrobce. Délka tohoto řešení byla ale až příliš neúměrná její důležitosti, proto bylo od tohoto řešení upuštěno.

Problém byl vyřešen přidáním sériového nastavitelného odporu na dva nejsilnější kanály osvětlovače, jejichž světelný výkon byl tímto omezen.

Kalibrace těchto složek byla prováděna na bílém povrchu fixy. Vycházelo se z předpokladu, že u bílého předmětu by měla být energie totožná na všech RGB kanálech. Proto vždy po získání snímku byla provedena sumace jednotlivých pixelů přes celý kanál a jednotlivé kanály porovnány. Tento iterativní kalibrační proces byl tedy ukončen při získání přibližně stejné energie na všech kanálech.

Rekonstrukcí byl pro kontrolu vytvořen barevný snímek bílého povrchu předmětu.



Obr. 6.8: Kalibrace světelného výkonu na bílém povrchu objektu - vstupní vzorek a výsledný snímek objektu.

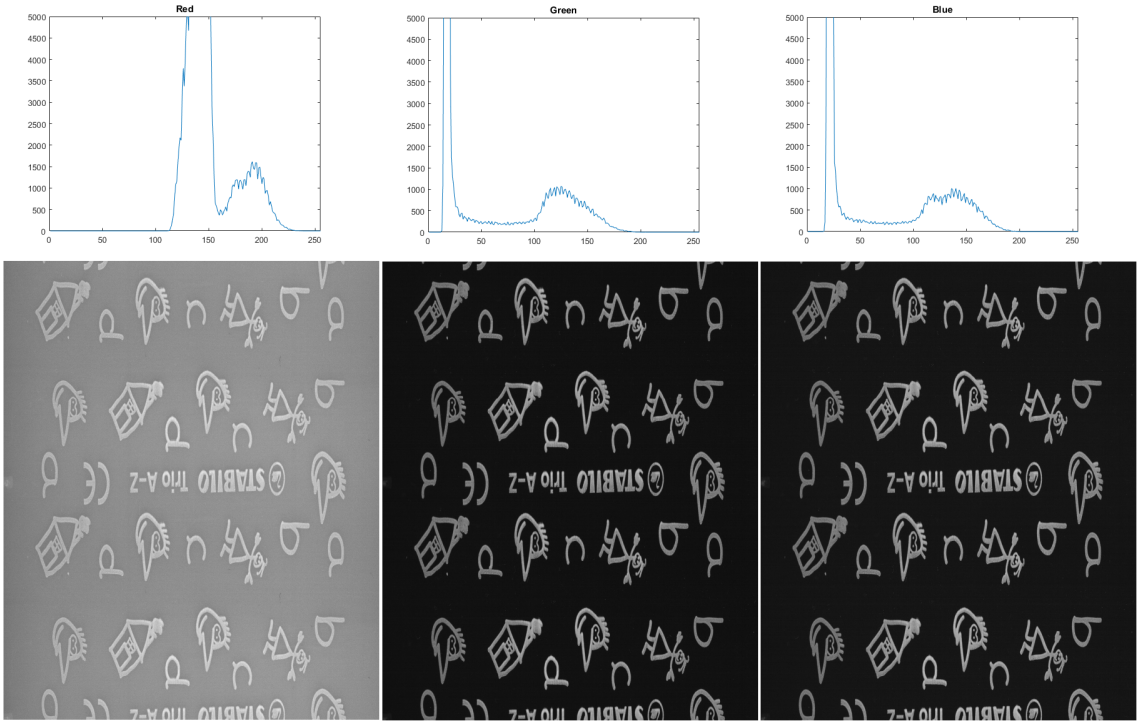
Tzv. „Colour test“ pro kontrolu správnosti barev, byl proveden rekonstrukcí snímku fixy s implantovaným barevným vzorem.

Obr. 6.9: „Colour test“ , originální vzor (v pravo nahoře) dostupný na [37].

7 Předzpracování snímků

7.1 Výběr nejkontrastnějšího světelného spektra

K detekování vady v potisku loga není třeba barevného snímku, bude stačit snímek šedotónový. Jelikož barva fixy a barva potisku mohou být v různých barevných kombinacích, jsou pořízeny snímky za osvětlení o vlnových délkách odpovídajících červenému, zelenému a modrému světlu. Dle barevné kombinace objektu a pozadí je následně zvolen ze tří pořízených ten snímek, který má nejvyšší rozdíl průměrné jasové úrovně potisku a těla fixy, čili snímek s maximálním kontrastem.



Obr. 7.1: Ilustrační obrázek jednotlivých kanálů snímku s příslušnými histogramy. Protože pozadí fixy je červené, nejvíce kontrastní bude buď kanál zelený nebo modrý.

Kanál s maximálním kontrastem je nalezen pomocí histogramu. V histogramu jsou nalezeny dvě největší maxima jasových hodnot v obrazu, přičemž druhé maximum je hledáno se zvoleným pásmem necitlivosti kolem prvního maxima. V aplikaci bylo zvoleno pásmo až 30 jasových úrovní oboustraně od maxima. Vzdálenost těchto dvou maxim odpovídá míře kontrastu. Kanál s největší hodnotou je zvolen k detekci a následně už se vyhodnocuje pouze tento vybraný kanál.

7.2 Kalibrace systému na nerovnoměrné osvětlení

7.2.1 Zdroje a příčiny nerovnoměrnosti osvětlení

Vzhledem k povaze snímané scény, způsobu osvětlení a parametrům kamery s objektivem je nasnímaný obraz zatížen problémem nerovnoměrného osvitu.

Největší měrou na nerovnoměrnost osvitu přispívá faktor zdroje světla. Jako světelný zdroj je využit LED pásek. Tyto pásky jsou osazeny LED čipy, které jsou na pásku rozmístěny v konstantních vzdálenostech, nikoliv ale přímo jeden vedle druhého, neboť část prostoru zabírají i spoje, rezistory a další podpůrné prvky.

Obr. 7.2: Ilustrační obrázek uspořádání prvků na Led pásku[38].

Tento problém osvětlení lze částečně redukovat zvětšením vzdálenosti osvětlovače od osvětlovaného předmětu, změnou vzájemného prostorového uspořádání zdroje světla, snímaného předmětu a kamery nebo vložení difuzního členu do světelné cesty. Tyto možnosti ale výrazně snižují dodávaný světelný výkon.

Dalším zdrojem nerovnoměrného osvětlení způsobený zdrojem světla je jeho drobné vyosení od roviny snímaného objektu, resp. snímačem kamery. Zásadní vliv má také skutečnost, že nejvíce světla bude dopadat do oblasti uprostřed snímaného objektu a méně osvětlené budou okrajové části snímaného objektu z důvodu omezené délky osvětlovače.

K dalším zdrojům nerovnoměrnosti osvětlení patří vinětace čipu a objektivu. Tento efekt se násobí se zmíněným faktorem nerovnoměrného osvitu způsobeným světelným zdrojem.

7.2.2 Kompenzace nerovnoměrného osvětlení

Za předpokladu, že světelné podmínky snímané scény jsou stálé v čase s pozicí objektu, je možno nalézt takovou kompenzační funkci, která by nerovnoměrnosti osvitu eliminovala.[39]

Reálný obraz lze popsat matematickým modelem:

$$f(x,y) = f_0(x,y).g(x,y) \tag{7.1}$$

kde:
$f(x,y)$... reálný obraz,
$f_0(x,y)$... ideální obraz,
$g(x,y)$... funkce popisující nerovnoměrnost osvětlení,
x... pozice v řádku,
y... index řádku.

Kompenzační funkci $k(x,y)$ lze potom definovat:

$$k(x,y) = \frac{P}{g(x,y)} \tag{7.2}$$

kde:
P... posunutí jasové úrovně,
$g(x,y)$... funkce popisující nerovnoměrnost osvětlení,
x... pozice v řádku,
y... index řádku.

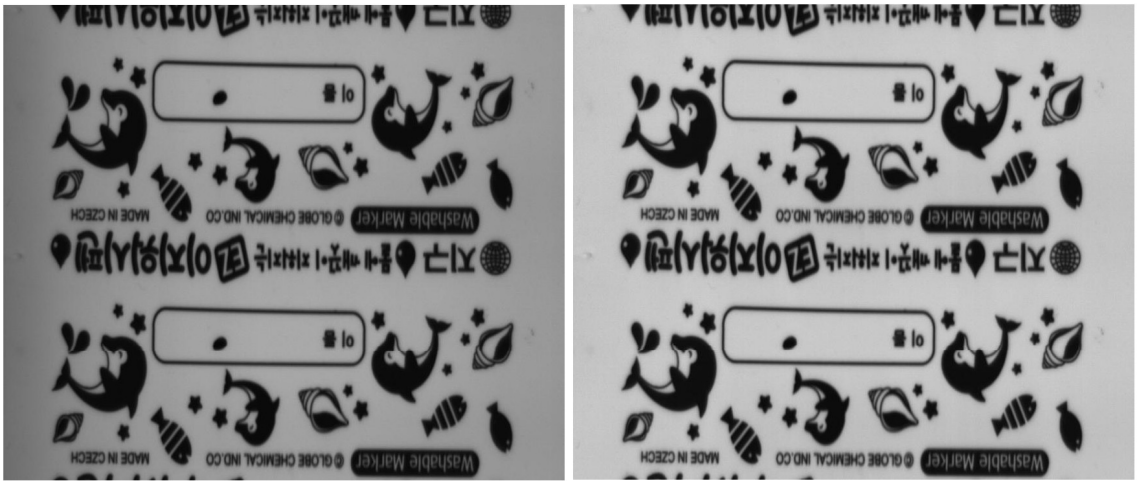
Potom kompenzovaný obraz $f_k(x,y)$ se vypočítá dle rovnice:

$$f_k(x,y) = f_0(x,y).g(x,y).k(x,y) \tag{7.3}$$

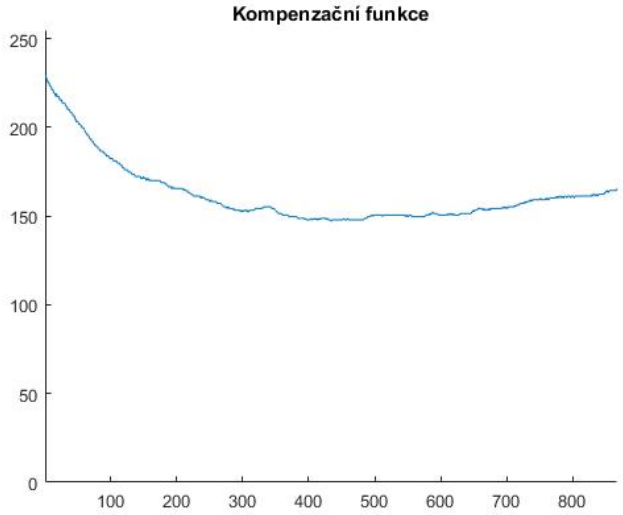
kde:
$f_k(x,y)$... kompenzovaný obraz,
$f_0(x,y)$... ideální obraz,
$g(x,y)$... funkce popisující nerovnoměrnost osvětlení,
x... pozice v řádku,
y... index řádku.

Pro výpočet kompenzační funkce se sestaví matice histogramů $h(x,y)$, kde každý jeden histogram je spočítán z jednoho příslušného sloupce snímku $f(x,y)$. Z matice histogramů $h(x,y)$ byla následně spočítána matice kumulovaných histogramů $c(x,y)$.

Funkcí popisující nerovnoměrnost osvětlení $g(x, y)$ je jasová hodnota vhodně zvoleného kvantilu z kumulovaného histogramu $c(x, y)$. V mém případě byl experimentálně s ohledem na charakter potisků zvolen $90 - kvantil$ histogramu $h(y)$. P složka byla zvolena experimentálně s ohledem na požadované posunutí jasové úrovně.



Obr. 7.3: V levo snímek bez kompenzace nerovnoměrného osvětlení, v pravo snímek s kompenzací nerovnoměrnosti osvětlení



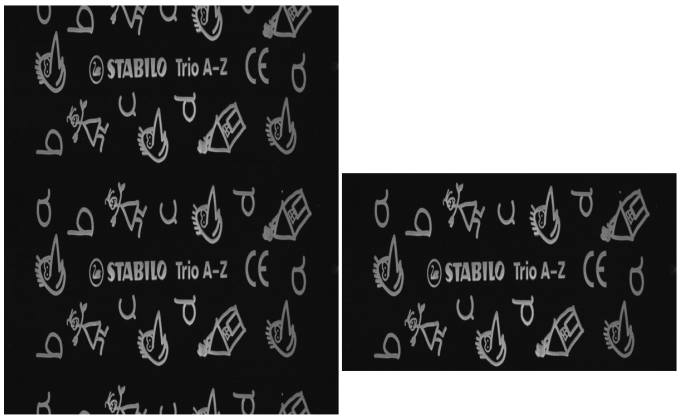
Obr. 7.4: Kompenzační funkce osvětlení

8 Detekce vad textu

8.1 Výřez povrchu objektu

Protože počáteční úhel natočení fixy při skenování je určen jejím upevněním do držáku, proces skenování je započat v nedefinované pozici. Kdybychom pořídili sken pouze jedné otáčky fixy, mohlo by dojít k situaci, že text, který nás zajímá bude neúplný na horním i spodním kraji snímku. To je nechtěné, proto jsou vždy pořízeny dvě otáčky, čímž je zaručena přítomnost alespoň jedné kompletní sady textu. Z tohoto snímku je následně vyříznut požadovaný výřez reprezentující jednu otáčku.

Všechny vyhodnocované typy fix mají jednu společnou vlastnost, a to že alespoň na některých řádcích je potisk vynechán. Algoritmus nalezení vhodného výřezu byl tedy založen na nalezení tohoto prázdného řádku, resp. toho nejširšího řádku bez potisku. Šířka výřezu je známá, je určena počtem pixelů v jedné otáčce, který je vždy konstantní.



Obr. 8.1: Původní (vlevo) a vykrojený (vpravo) snímek

8.2 Segmentace zájmových oblastí

Aby bylo možné určit, zda jsou v natisknutém textu přítomny vady, je na celém vstupním obrazu provedena segmentace výřezů textů. Tento text bude následně klasifikován jako s vadou nebo bez vady. Segmentovány jsou jednotlivá slova potisku, symboly či logické celky.

K segmentaci lze využít různé metody. V této úloze byla použita metoda „template matching“. Obecně řečeno, jedná se o metodu, kde jeden obraz může být použit k

extrakci objektů a vzorů a na následných obrázcích jsou hledány stejné nebo podobné vzory. Porovnávané vzory mohou být velmi malé nebo mohou reprezentovat celý objekt zájmu [40].

Volba této metody je výhodná, neboť hledané vzory díky pevnému stabilnímu uchycení vyhodnocovaného objektu jsou téměř bez úhlového natočení a jsou velice podobné na všech vyhodnocovaných objektech s většími rozdíly pouze ve vlastních vadách.

Shoda se vzorem je vyhodnocena podle zvolené metriky, která se volí dle vlastností hledaného objektu a podle jeho vztahů s okolím. Bylo vyzkoušeno více možných metrik. Nejlepší výsledky byly pozorovány u normované korelace, kterou vyjadřuje rovnice:

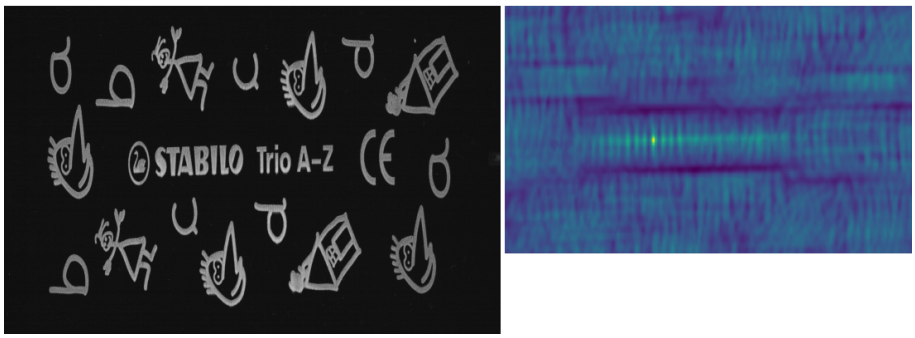
$$R(x,y) = \frac{\sum_{x',y'}(T'(x',y').I'(x+x',y+y'))}{\sqrt{\sum_{x',y'} T'(x',y')^2 . \sum_{x',y'} I'(x+x',y+y')^2}} \tag{8.1}$$

kde R je mapa příslušnosti, T je hledaný vzor, I je obraz, ve kterém je shoda se vzorem zjišťována.

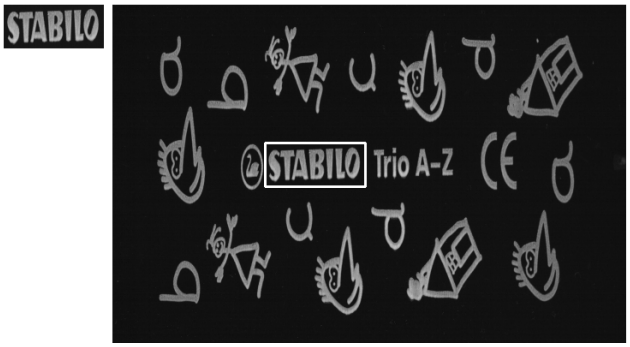
Jako referenční porovnávaný vzor byly vzaty jednotlivé výřezy bez vad. Bylo vyzkoušeno, že i vzory s vadou lze stejně tak dobře segmentovat. Jako lokace hledaného vzoru je považována nejvyšší hodnota výstupní mapy.

Tato technika je obsažena v knihovně OpenCV, odkud byla pro praktickou implementaci využita.

Na obrázku níže vlevo je zobrazena příklad takové mapy.



Obr. 8.2: Mapa korelace vzoru s příslušnou oblastí v obraze - nejvyšší hodnota značí nejvyšší příslušnost.

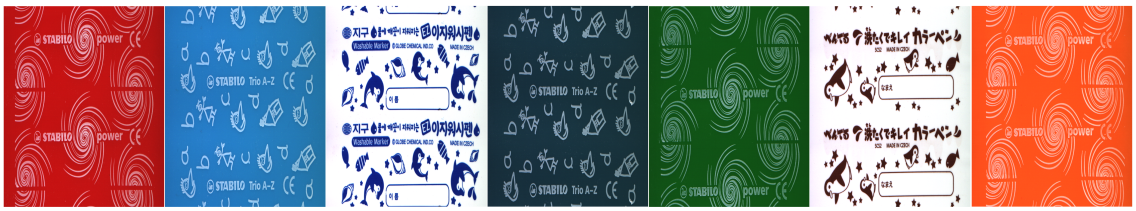


Obr. 8.3: Názorná ukázka nalezeného objektu v obrázku

8.3 Vytvořená galerie dat

Pro trénování konvolučních neuronových sítí, které vyhodnocují přítomnost vad v jednotlivých částech potisknutého textu na fixách, byla vytvořena galerie dat. Na snímcích jsou zachyceny všechny čtyři vzory potisků fix, kde každý vzor je v různých barvách. Celkově bylo pořízeno 607 snímků objektu o rozměrech $734 \times 590 \times 3$ pixelů, z toho snímků s vadou textu je 522 a zbývajících 85 snímků zachycuje potisk fixy s textem bez vad. Příslušnost snímku k anotované kategorii určuje adresář, ve kterém byl daný snímek uložen. Názorná ukázka snímků je zobrazena na obrázku 8.4.

Ze surových snímků byly vytvořeny RGB snímky a byly uloženy ve formátu .png.



Obr. 8.4: Ukázka vybraného vzorku snímků z galerie

Z těchto snímků byly zmíněnou metodou srovnání se vzorem získány oblasti jednotlivých částí nápisů, resp. jednotlivých slov, která byla rovněž uložena dle dané kategorie a příslušnosti k textu. U těchto výřezů jednotlivých slov textu nebyl uložen pouze nejkontrastnější kanál, ale jednotlivě každý kanál zvlášť jako šedotónový snímek. To bylo provedeno z důvodu získání většího množství dat. Příliš nekontrastní snímky byly z galerie následně vyřazeny.

Popsané úkony byly vykonány v poloautomatickém režimu pomocí vytvořeného skriptu dataset_maker.py. Ten, kromě řízení celého pracoviště, vždy zobrazil pořízený snímek, který byl na monitoru operátorem zkontrolován a následně buď přijat

nebo vyřazen. Kontrola probíhala z důvodu možných chyb způsobených mnohými neočekávatelnými faktory, jako třeba dodatečné přichycení nečistot, zvláště pak prachu, na tělo fixy vlivem statické elektřiny nebo nekorektní nasazení fixy do upínacího mechanismu. Pokud byl snímek v pořádku, byly v něm nalezeny metodou srovnání se vzorem příslušné vzory, které byly opět jednotlivě zkontrolovány a případně potvrzeny či vyřazeny. Ty potvrzené byly pak uloženy.

8.3.1 Augmentace dat

Augmentace dat byla provedena za účelem rozšíření galerie dat výřezů textů, na kterých se budou učit modely konvolučních neuronových sítí. Jak bylo řečeno, základní augmentace spočívala v získání všech tří kanálů RGB snímku. Dále byly na původních datech provedeny operace posunutí, zoomu a také rotace. Rotace byla omezena pouze na interval $<-3,0.5)\wedge(0.5,3>^\circ$ a hodnota úhlu natočení z tohoto intervalu byla generována náhodně s rovnoměrným rozložením pravděpodobnosti.

8.4 Klasifikace nalezených oblastí

Klasifikace nalezených oblastí je v procesu detekce vad prováděna poté, co jsou lokalizovány zájmové oblasti obsahující text. Ty je třeba vyhodnotit, zda obsahují vady nebo jsou bez vad. K tomu je v této práci využito konvolučních neuronových sítí.

8.4.1 Příprava datasetu k trénování

Aby bylo možné přejít k fázi učení neuronové sítě, je třeba transformovat trénovací data do formátu, který model vyžaduje. K tomu byl vytvořen program make_dataset.ipynb.

Program byl vytvořen v open-sourcové webové alplikaci Jupyter Notebook, která umožňuje vývoj v jazyku Python. Tato platforma byla zvolena z důvodu snadného vývoje, neboť kombinuje práci v iteraktivním (v příkazy v terminálu) i editačním režimu (příkazy ve skriptu) pomocí tzv. buněk. Buňky jsou tvořeny sérií příkazů a lze je spouštět v libovolném pořadí. Přímo pod tyto buňky se pak zobrazují případné výstupy.

Ve zmíněném programu byla tedy postupně nahrána všechna data, tedy vždy obrázek textu s příslušnou přidruženou požadovanou třídou. Následně byly u obrázků upraveny jejich rozměry dle rozměrů příslušného referenčního obrázku. Tato data pak byla uložena do kontejneru typu seznam. Po zpracování celé množiny dat byla data promíchána, aby za sebou nenásledovaly prvky stejné kategorie tak jak byly

nahrány. Z této galerie poté byla oddělena validační množina, která byla zvolena na 15% z celého objemu dat. Tato množina dat je následně využita během procesu učení k regularizaci tzv. Early stoppingem. Oba kontejnery dat byly následně uloženy pomocí modulu pickle. Ten umožňuje ukládat a načítat python objekty. Kontejnery byly uloženy pod názvy X.pickle a y.pickle, resp. X_val.pickle a y_val.pickle

8.4.2 Klasifikátor

Ke klasifikaci nalezených oblastí je využito souboru modelů konvolučních neuronových sítí. Každá instance byla na příslušné galerii dat trénována pro klasifikaci jednoho vzoru textu. Architektura modelu je společná pro všechny instance.

Topologie a parametry učení

Síť obsahuje tři konvoluční vrstvy s filtry o rozměrech $3 \times 3 \times x$ (kde x je počet kanálů dané vstupní příznakové mapy) v počtech 32, 32 a 64. Konvoluční vrstvy jsou aktivovány funkcí ReLU a následně podvzorkovány 2×2 Max Poolingem. Poté následují dvě dopředné plně propojené neuronové vrstvy s 512 a 32 neurony. Výstupem je One Hot End enkodovaná dvojice neuronů aktivovaná funkcí softmax. Chyba sítě je počítána metrikou Categorical Crossentropy a jako optimalizační algoritmus byl zvolen ADAM. Model byl vytvořen pomocí knihovny TensorFlow 2.1 na úrovni jeho rozhraní Keras.

Jednotlivé vrstvy jsou vytisknuty na obrázku 8.5.

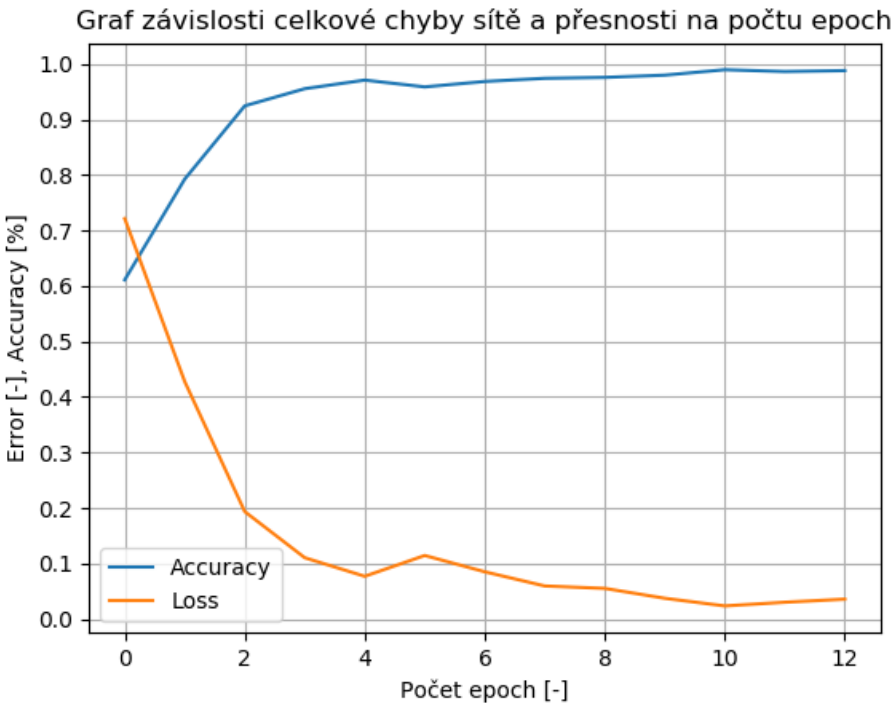
```
Layer (type)
=================================
conv2d (Conv2D)
_________________________________
activation (Activation)
_________________________________
max_pooling2d (MaxPooling2D)
_________________________________
conv2d_1 (Conv2D)
_________________________________
activation_1 (Activation)
_________________________________
max_pooling2d_1 (MaxPooling2
_________________________________
conv2d_2 (Conv2D)
_________________________________
activation_2 (Activation)
_________________________________
max_pooling2d_2 (MaxPooling2
_________________________________
flatten (Flatten)
_________________________________
dense (Dense)
_________________________________
dropout (Dropout)
_________________________________
dense_1 (Dense)
_________________________________
dropout_1 (Dropout)
_________________________________
dense_2 (Dense)
=================================
```

Obr. 8.5: Struktura modelu klasifikační CNN

Učení a nasazení

Učení modelu bylo provedeno s využitím architektury CUDA Toolkit v10.1 a knihovny primitiv pro neuronové sítě cuDNN na grafické kartě NVIDIA Geforce RTX 2070.

Přepočet vah během učení probíhá po dávkách o 32 vzorech. Model je v procesu učení validován na validační galerii dat a pokud na tomto setu nevykazuje snížení chyby v sérii tří epoch, tak je proces učení ukončen.



Obr. 8.6: Ukázka průběhu závislosti celkové chyby a přesnosti sítě na počtu epoch na validační galerii

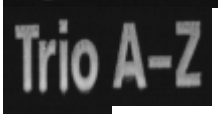
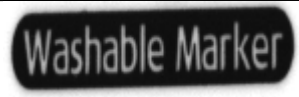
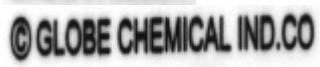
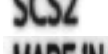
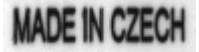
Kvůli podstatnému zrychlení výpočtu predikce na Raspberry Pi 4B je následně ještě před nasazením naučený model konvertován do modelu TensorFlow Lite, který je optimalizovaný pro většinou méně výkonná mobilní zařízení a který ani nevyžaduje instalovat plnou verzi TensorFlow, stačí pouze TensorFlow interpret.

9 Testování a vyhodnocení výkonnosti

Testovány byly jednak samotné konvoluční neuronové sítě co by klasifikátory vad jednotlivých zájmových oblastí a pak také celkový systém, resp. jeho výkonnost na jednotlivých vzorech propisovacích fix.

V následující tabulce je uveden souhrnný orientační přehled klasifikovaných vzorů, jejich označení a příslušnost k fixe.

Tab. 9.1: Přehled klasifikovaných vzorů

Fixa	Označení vzoru	Vzor
F1	V1.1	
	V1.2	
	V1.3	
F2	V2.1	
	V2.2	
	V2.3	
F3	V3.1	
	V3.2	
	V3.3	
F4	V4.1	
	V4.2	

9.1 Testování klasifikátorů

K otestování výkonnosti jednotlivých konvolučních sítí byla použita metoda Cross-validace. A to ve formě tzv. „Ten-Fold“, kdy celková množina dat je rozdělena na deset podmnožin, kde postupně na sjednocení devíti z nich je model trénován a na jedné zbývající testován. Výsledky správnosti klasifikace jednotlivých klasifikátorů spolu s odpovídajícími směrodatnými odchylkami jsou uvedeny v Tab.9.2

Tab. 9.2: Výsledky testování klasifikačních CNN

	F1			F2			F3			F4	
Vzor	V1.1	V1.2	V1.3	V2.1	V2.2	V2.3	V3.1	V3.2	V3.3	V4.1	V4.2
Správnost[%]	96,0	96,9	99,1	95,5	92,6	98,1	97,7	98,1	94,3	95,3	98,1
Směr. od.[%]	2,5	3,1	0,5	3,2	6,0	0,1	2,2	1,9	3,4	7,3	1,9

9.2 Testování celkové

Testování celkového systému bylo provedeno manuálně na celkem 320 variacích propisovacích fix s vadami a bez vad. Tedy každý ze čtyřech možných typů fix byl testován 80 krát.

Test byl proveden sekvencí kroků:

1. Vyběr vzorku (případně výroba vady)
2. Vložení vzorku do zařízení
3. Spuštění akvizice
4. Vyhodnocení kusu
5. Záznam výsledku

Aby byla detekována vada, stačí pouze pozitivní výsledek jednoho z klasifikátorů použitého pro daný typ fixy. Pro výsledek bez vady musí mít všechny klasifikátory negativní výsledek.

Výsledky těchto testů jsou pro všechny vzory fix zobrazeny formou chybové matice v Tab.9.3. Ukázkové příklady zobrazení jednotlivých log a jejich kvalitativního stavu v GUI je zobrazono formou tzv. „print screenů“ na Obr.9.1.

Tab. 9.3: Výsledky testování systému na všech čtyřech vzorech propisovacích fix

F1		Realita	
		Pozitivní	Negativní
Klasifikace	Pozitivní	47	1
	Negativní	2	27

F3		Realita	
		Pozitivní	Negativní
Klasifikace	Pozitivní	40	4
	Negativní	0	36

F2		Realita	
		Pozitivní	Negativní
Klasifikace	Pozitivní	44	1
	Negativní	5	30

F4		Realita	
		Pozitivní	Negativní
Klasifikace	Pozitivní	37	3
	Negativní	3	37

Obr. 9.1: Ukázka příkladů zobrazení jednotlivých log s příslušnými symboly jejich stavu na výřezech „print screenů“ GUI. Celkový stav log měřeného objektu je dán logickým součinem stavů jednotlivých log (pro úsporu místa zde není uveden, k náhledu je na obrázku v následující kapitole)

Z provedených testů byly vypočteny statistické výkonnostní parametry *accuracy*, *recall*, *precision*, *F1-score* systému pro detekci jednotlivých vzorů fix. Jejich vzorce jsou uvedeny níže a výsledky pro jednotlivé typy fix jsou uvedeny v Tab.9.4. Hodnoty parametrů pro jednotlivé vzory fix byly zprůměrovány a jsou uvedeny v posledním řádku téže tabulky jako obecné parametry systému.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{9.1}$$

$$recall = \frac{TP}{TP + FN} \tag{9.2}$$

$$precision = \frac{TP}{TP + FP} \tag{9.3}$$

$$F1 - score = \frac{2.precision.recall}{precision + recall} \tag{9.4}$$

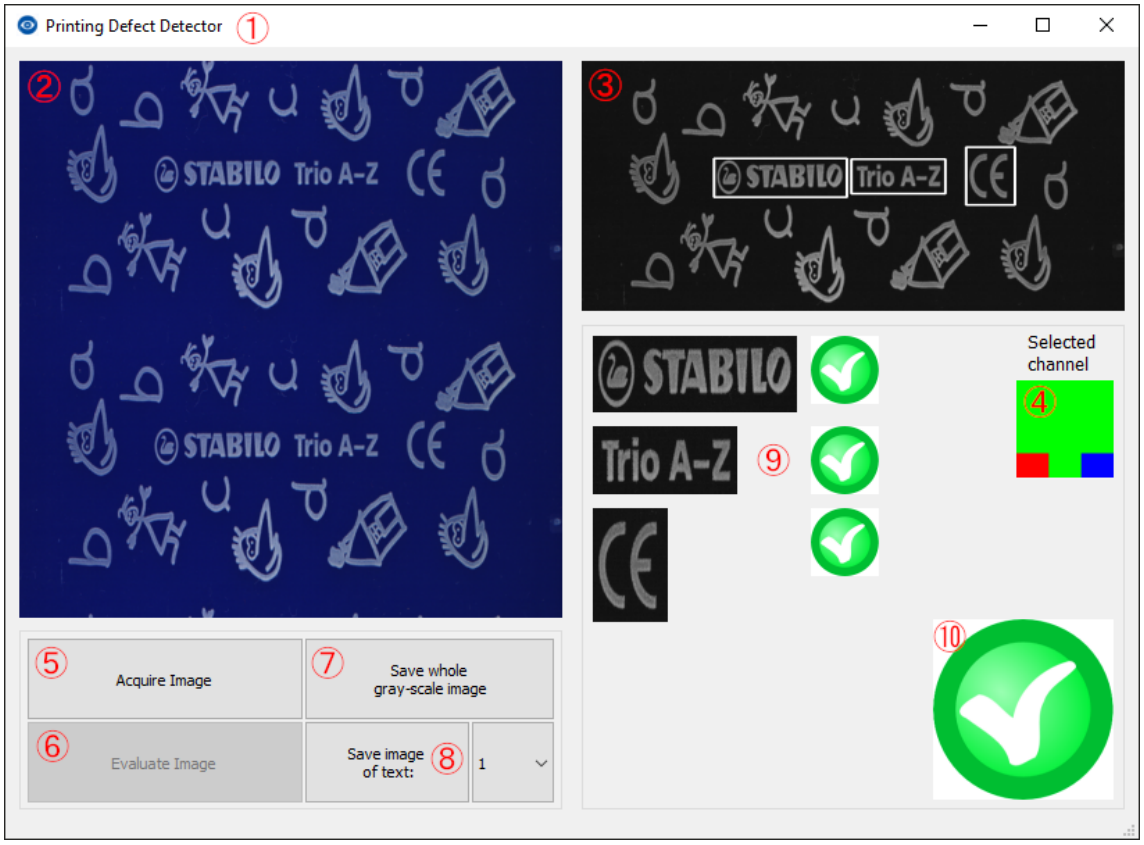
Kde TP - True Positive, TN - True Negative, FP - False Positive, FN - False Negative.

Tab. 9.4: Vyhodnocení systému dle detekce vad na jedtlivých vzorech fix

	accuracy	recall	precision	F1-score
F1	0,961	0,959	0,979	0,969
F2	0,925	0,898	0,978	0,936
F3	0,950	1,000	0,909	0,952
F4	0,925	0,925	0,925	0,925
Systém	0,940	0,946	0,948	0,946

10 Grafické uživatelské rozhraní (GUI)

Pro interakci s operátorem bylo vytvořeno grafické uživatelské rozhraní (GUI). Náhled tohoto rozhraní je zobrazen na obrázku 10.1. Obrázek obsahuje označení jednotlivých prvků, které jsou následně v textu vysvětleny. Toto GUI se automaticky spouští po startu RPi.



Obr. 10.1: Grafické uživatelské rozhraní - popis jednotlivých prvků

Popis prvků GUI:

1. Název GUI
2. Barevný snímek povrchu objektu
3. Šedotónový snímek s vyznačeným textem
4. Vybraný kanál
5. Tlačítko pořízení snímku
6. Tlačítko vyhodnocení
7. Tlačítko pro uložení celého šedotónového snímku
8. Tlačítko uložení dílčího textu dle přidružené výběrové rolety

9. Výřezy textů s příslušnými dílčími výsledky
10. Celkový výsledek kontroly

Barevný snímek povrchu objektu

Po spuštění akvizice je získán surový snímek, ze kterého je ihned tento přehledový barevný snímek zrekonstruován a zobrazen.

Šedotónový snímek s vyznačeným textem

Ve vizualizaci je též zobrazen nejvíce kontrastní kanál RGB snímku, ve kterém je dle jasu objektu a pozadí v černých nebo bílých rámečcích zvýrazněn nalezený text, u kterého je provedena detekce vad.

Vybraný kanál

Tento indikátor barevně zobrazuje nejvíce kontrastní spektrum osvětlení, resp. kanál RGB snímku. Ten je pak využit k následnému vyhodnocení.

Tlačítko pořízení snímku

Tímto tlačítkem se zahájí akvizice snímku a na ní navázané procesy.

Tlačítko vyhodnocení

Toto tlačítko je neaktivní dokud neproběhne akvizice snímku. Stisknutím se spustí proces nalezení jednotlivých vzorů textu a následně i klasifikace a celkové vyhodnocení.

Tlačítko pro uložení celého šedotónového snímku

Tímto tlačítkem je uložen do příslušného adresáře aktuální šedotónový snímek s nejvyšším kontrastem. Tlačítko je neaktivní dokud neproběhne akvizice. Snímek je uložen se vzrůstajícím indexem a každý vzorek lze uložit pouze jedenkrát.

Tlačítko uložení dílčího textu dle přidružené výběrové rolety

Tlačítko je neaktivní dokud nejsou nalezeny a klasifikovány jednotlivé výřezy textu. Následně je z rolety zvolen výřez textu pro uložení v zobrazeném pořadí. Stisknutím je zvolený výřez uložen do příslušného adresáře výřezů. Každý výřez lze uložit pouze jednou.

Výřezy textů s příslušnými dílčími výsledky

V této části jsou pod sebou zobrazeny jednotlivé páry - výřez textu a přidružený indikátor výsledku jeho dílčí klasifikace.

Celkový výsledek kontroly

Tento indikátor zobrazuje celkový výsledek kontroly. Jedná se o konjunkci výsledků klasifikace dílčích výřezů textů. Zelený symbol značí, že potisk fixy je v pořádku. Červený symbol značí, že alespoň v některém z dílčích potisků je vada a tedy celkově je potisk vadný.

Závěr

S automatizací výrobních procesů a s postupným prosazováním konceptu Průmysl 4.0 je neoddělitelně spojena i vizuální kontrola výrobků. Potřeba modernizace a zkvalitnění vizuální kontroly produktů dala vzniknout i této diplomové práci, kterou inicioval požadavek firmy Centropen Dačice na kontrolu potisku loga vyráběných značkovacích propisek.

Práce se zabývá návrhem a vytvořením vizuálního kontrolního systému od návrhu vhodné metody snímání potisku pomocí řádkové kamery, vytvoření snímacího pracoviště, implementace detekčních algoritmů až po jeho otestování.

Bylo vytvořeno měřicí pracoviště, kde upínací mechanismus pro rotaci kontrolovaného značkovače je konstruován obdobně k mechanice soustružného zařízení s vřetenem a přítlačným koníkem. Rotaci objektu vykonává do pomala zpřevodovaný stejnosměrný motor, liniové osvětlení objektu zajišťuje výkonný RGB led pásek. Ke snímání je využita zapůjčená monochromatická řádková kamera od firmy Basler.

Pro implementaci řízení celého pracoviště a vyhodnocení produktu byla zvolena výpočetní jednotka Raspberry Pi 4B, na které je nainstalován operační systém Raspbian. Akviziční program pro řízení pracoviště a synchronizace kamery je napsán v jazyce C++. Ostatní programy pro následné zpracování snímku, jeho vyhodnocení, vytvoření galerie dat, testování a interakci s uživatelem je v jazyku Python 3.7.

Spolehlivost vizuálních systémů nezávisí pouze na robustnosti algoritmů zpracování a vyhodnocení obrazu. Ale je důležitá kvalita všech prvků řetězce zpracování obrazu. Zpravidla k největším ztrátám podstatné informace v obrazu dochází již ve fázi jeho snímání. Proto byla velká část optimalizací soustředěna již do fáze vývoje vhodné metody snímání a předzpracování obrazu. Výstup navržené metody snímání je nezávislý na kombinaci barev potisku a pozadí měřeného objektu. Následnou softwarovou kompenzací nerovnoměrnosti osvětlení byla zvýšena robustnost systému i s ohledem na nehomogenitu intenzity osvětlení a nedokonalé seřízení osvětlovačů.

Na základě provedené rešerše přístupů k vizuální kontrole produktů bylo navrženo řešení detekce potisku loga a jeho vyhodnocení. Pro detekci loga byla využita technika srovnání se vzorem, ke klasifikaci nalezené oblasti pak konvolučních neuronových sítí. Využití tohoto distribuovaného systému je výhodné i po stránce jeho dobré rozšiřitelnosti v případě budoucí potřeby na kontrolu dalších objektů.

Všechny body zadání byly splněny. Systém byl otestován na 320 vzorcích značkovačů s vadami i bez vad a bylo dosaženo hodnoty F1-score 0,946. Toto zařízení lze považovat za funkční základ pro stavbu budoucího průmyslového produktu, byť pro průmyslové nasazení bude vyžadovat ještě mnohé úpravy a rozšíření, na které v této práci již nezbyl prostor. Cílem práce však nebylo vytvořit finální produkt, ale otestovat základní principy a posoudit, jestli použitý hardware, způsob snímání a

použité metody lze s úspěchem použít pro případnou realizaci kamerové kontroly.

Literatura

[1] Wikipedia. *Offset printing* [online]. 2020 [cit. 2020-01-01]. San Francisco. Dostupné z: <https://en.wikipedia.org/wiki/Offset_printing>

[2] Orbo. *Offset printing: your all-round solution for flexible packaging* [online]. Kortemark [cit. 2020-01-03]. Dostupné z: https://www.orbo.be/en/opportunities/eb-offset-printing

[3] KOŘÍNEK, Ota. *Sitotisk/Seriografie.* Spálená 54, Praha 1: Nakladatelství technické literatury, n. p., 1971. ISBN 04-603-71.

[4] Lamiaa Nail. *Project 2: Screen Printing* [online]. Egypt [cit. 2020-01-03]. Dostupné z: http://fabacademy.org/2019/labs/egypt/students/lamiaa-nail/Week17.html

[5] Lenstar. *PCB Screen Printer, Screen Printing Machines* [online]. No.35 Gongye 15th Rd., Taiping Dist., Taichung City (411), Taiwan, 2016 [cit. 2020-01-03]. Dostupné z: https://www.lenstar.com.tw

[6] BAYGIN, M.; KARAKOSE, M.; SARIMADEN, A; AKIN, E: *Machine Vision based Defect Detection Approach using Image Processing* [online] IEEE. 2017 [cit. 2020-03-26]. Dostupné z: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8090292&tag=1

[7] ZHOU, F.; LIU, G; XU, F.; DENG, H. *A Generic Automated Surface Defect Detection Based on a Bilinear Model* [online] ResearchGate. 2019 [cit. 2020-03-26]. Dostupné z: https://www.researchgate.net/publication/334968994_A_Generic_Automated_Surface_Defect_Detection_Based_on_a_Bilinear_Model

[8] VANS, M.; SCHEIN, S.; STAELIN, C.; KISILEV, P.; SIMSKE, S.; DAGAN, R.; HARUSH, S. *Automatic visual inspection and defect detection on Variable Data Prints* [online] hp. 2011 [cit. 2020-03-27]. Dostupné z: https://www.hpl.hp.com/techreports/2008/HPL-2008-163R1.pdf

[9] ZACCONE, Giancarlo, Md. Karim REZAUL a Ahmed MENSHAWY. *Deep Learning with TensorFlow: Take your machine learning knowledge to the next level with the power of TensorFlow 1.x.* Birmingham, United Kingdom: Packt Publishing Limited, 2017. ISBN 1786469782.

[10] CHOLLET, François. *Deep learning v jazyku Python: knihovny Keras, Tensorflow.* Přeložil Rudolf PECINOVSKÝ. Praha: Grada Publishing, 2019. Knihovna programátora (Grada). ISBN 978-80-247-3100-1.

[11] SHARMA, Sagar. *Activation Functions in Neural Networks* [online] 2017 [cit. 2020-04-16]. Dostupné z: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

[12] StackExchange. *Is this linear model overfitting when I add more parameters?* [online]. 2020 [cit. 2020-04-17]. Dostupné z: https://stats.stackexchange.com/questions/360293/is-this-linear-model-overfitting-when-i-add-more-parameters

[13] GeeksforGeeks. *Underfitting and Overfitting in Machine Learning* [online]. 2020 [cit. 2020-04-17]. Dostupné z: https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/

[14] JIRSÍK, Václav. *10 MUIN nn paradigmata 2019* [online] VUT Brno, Ústav Automatizace a měřicí techniky. 2019 [cit. 2020-04-16].

[15] KHAN, Salman; RAHMANI, Hossein; SHAH, Syed; BENNAMOUN, Mohammed. *A Guide to Convolutional Neural Networks for Computer Vision* Morgan & Claypool, 2018. ISBN 9781681730226.

[16] SRIVASTAVA, Nitish; HINTON, Geoffrey; KRIZHEVSKY, Alex; SUTSKEVER, Ilya; SALAKHUTDINOV, Ruslan. *Dropout: A simple way to prevent neural networks from overfitting.* Journal of Machine Learning Research, 2014, 15(1):1929–1958.

[17] GÉRON, Aurélien. *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.* 2nd ed. USA: O'Reilly Media, 2019. ISBN 1492032646.

[18] TILGNER, Martin. *Person Detection in Image by Machine Learning.* Brno, 2019, 92 p. Master's Thesis. Brno University of Technology, Faculty of Electrical Engineering and Communication, Department of Control and Instrumentation. Advised by Ing. Karel Horák, Ph.D.

[19] SHAMRAI, Daria. *What is leaky ReLU?.* [online]. 2020 [cit. 2020-04-22]. Dostupné z: https://www.quora.com/What-is-leaky-ReLU

[20] ØRSTAVIK, Mathilde; MIDTBØ, Terje. *New Era for Feature Extraction in Remotely Sensed Images byThe Use of Machine Learning.* Trondheim, Leden 2017, Norwegian University of Science and Technology, NO-7491, ISSN 0047-3278

[21] Machinecurve. *What are Max Pooling, Average Pooling, Global Max Pooling and Global Average Pooling?*. [online]. 2020 [cit. 2020-04-22]. Dostupné z: https://www.machinecurve.com/index.php/2020/01/30/what-are-max-pooling-average-pooling-global-max-pooling-and-global-average-pooling/

[22] YUNG, Jessica. *Explaining Tensorflow Code for a Convolutional Neural Network* [online]. 2017 [cit. 2020-04-22]. Dostupné z: https://www.jessicayung.com/explaining-tensorflow-code-for-a-convolutional-neural-network/

[23] Aia vision online. *Understanding Line Scan Applications* [online]. 900 Victors Way, Suite 140, Ann Arbor, Michigan, 48108, USA, 2009 [cit. 2020-01-03]. Dostupné z:
`https://www.visiononline.org/vision-resources-details.cfm/vision-resources/Understanding-Line-Scan-Applications/content_id/5338`

[24] Atesystem. *Řádkové kamery - typy a technologie* [online]. Studentská 6202/17,Poruba 708 00, Ostrava 8, 2020 [cit. 2020-01-03]. Dostupné z: http://kamery.atesystem.cz/know-how/line-scan-velky-pruvodce-radkovymi-kamerami/dil-1-radkove-kamery-typy-a-technologie/

[25] Basler. *raL6144-16gm - Basler racer* [online]. Germany, 2020 [cit. 2020-01-03]. Dostupné z: https://www.baslerweb.com/en/products/cameras/line-scan-cameras/racer/ral6144-16gm/

[26] FotakyRecenze.eu *Canon EF 50mm f/1.8 II* [online]. 2020 [cit. 2020-01-03]. Dostupné z: https://fotakyrecenze.eu/canon-ef-50mm-f1-8-ii/

[27] Atesystem. *Světla pro řádkové kamery* [online]. Studentská 6202/17,Poruba 708 00, Ostrava 8, 2020 [cit. 2020-01-03]. Dostupné z: http://kamery.atesystem.cz/know-how/line-scan-velky-pruvodce-radkovymi-kamerami/dil-3-svetla-pro-radkove-kamery/

[28] National Instruments. *A Practical Guide to Machine Vision Lighting* [online]. USA, 2020 [cit. 2020-01-03]. Dostupné z: http://www.ni.com/cs-cz/innovations/white-papers/12/a-practical-guide-to-machine-vision-lighting.html

[29] Vision Doctor. *Illumination techniques for line scan cameras* [online]. 2020 [cit. 2020-01-03]. Dostupné z: https://www.vision-doctor.com/en/illumination-techniques/illumination-line-scan-camera.html

[30] NIDEC COPAL ELECTRONICS [online katalogový list]. *MG16B-120-AB-00* [cit. 2020-01-08]. Dostupné z: https://cz.farnell.com/nidec-copal-electronics/mg16b-120-ab-00/dc-geared-motor-1-120-100rpm-90mn/dp/3010301

[31] Distrelec. *PI4 MODEL B/4GB - Raspberry Pi 4 1.5GHz Quad-Core, 4GB RAM* [online]. 2020 [cit. 2020-03-13]. Dostupné z: https://www.distrelec.cz/cs/raspberry-pi-5ghz-quad-core-4gb-ram-raspberry-pi-pi4-model-4gb/p/30152781

[32] Raspberry Pi Pinout. *Raspberry Pi Pinout* [online]. 2020 [cit. 2020-03-13]. Dostupné z: https://pinout.xyz

[33] @drogon. *Wiring Pi* [online]. 2020 [cit. 2020-04-13]. Dostupné z: `http://wiringpi.com`

[34] Riverbank Computing Limited. *PyQt5 Reference Guide* [online]. 2020 [cit. 2020-04-13]. Dostupné z: `https://doc.bccnsoft.com/docs/PyQt5/introduction.html`

[35] Easy TensorFlow. *Introduction* [online]. 2020 [cit. 2020-04-13]. Dostupné z: `https://www.easy-tensorflow.com/tf-tutorials/basics/graph-and-session`

[36] Basler. *User's manual for GigE vision cameras* [online]. 2020 [cit. 2020-03-19]. Dostupné z: `https://www.baslerweb.com/fp-1556548672/media/downloads/documents/users_manuals/AW00118308000_racer_GigE_User_Manual.pdf`

[37] A set of monitor test images .... [online]. 2020 [cit. 2020-03-18]. Dostupné z: `http://www.footootjes.nl/Various/Monitor_test_charts.html`

[38] Kili. *LED pásek SHB teplá bílá 9,6W 120LED/m* [online]. 2020 [cit. 2020-03-12]. Dostupné z: https://www.kili.cz/cs/led-pasek-shb-tepla-bila-9-6w-120led/m-3202111601

[39] HONEC, Peter. *Spolehlivé systémy zpracování obrazu.* Brno, 2008, 93 s. Dizertační práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce: Ing. Ilona Janáková, Ph.D.

[40] SONKA, Milan; HLAVAC, Vaclav; BOYLE, Roger. *Image Processing, Analysis, and Machine Vision.* 1st ed. Springer-Science+Business Media, B.V., 1993. ISBN 978-0-412-45570-4

Seznam symbolů, veličin a zkratek

ADAM	Adaptive Moment Estimation
API	Application Program Interface
CMOS	Complementary Metal–Oxide–Semiconductor
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
DC	Direct Current
GigE	Gigabit Ethernet
GPIO	General Purpose Input Output
GPU	Graphics Processing Unit
GUI	Graphical User Interface
k-NN	k-Nearest Neighbors algorithm
LED	Light-Emitting Diode
NOOBS	New Out Of the Box Software
OS	Operating System
PCA	Principal Component Analysis
ReLU	Rectified Linear Unit
RGB	Red - Green - Blue
RPM	Revolutions per Minute
SoC	System on Chip
SOM	Self-organizing map
SVM	Support Vector Machines
3D	Three Dimensional

Seznam příloh