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

DETEKTOR VAD S VYUŽITÍM CIS SENZORU
DEFECT DETECTOR USING CIS SENSOR

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR
Bc. Daniel Komzák

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

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. Daniel Komzák ***ID:*** 208293

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

NÁZEV TÉMATU:

Detektor vad s využitím CIS senzoru

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je otestovat využití CIS senzoru pro detekci vad na vyráběném materiálu, typicky obalový materiál (papír, fólie) nebo válcovaný plech.

1. Seznamte se s dostupnými CIS (Contact Image Sensor) a jejich parametry.
2. Vytipujte vhodný senzor pro použití jako detektor děr na běžícím pásu.
3. Navrhněte vhodnou výpočetní jednotku (PC, Raspberry, jednočipový počítač…) a celkové uspořádání.
4. Vytvořte modelové pracoviště pro využití CIS senzoru na běžícím páse.
5. Vytvořte a implementujte software na získání dat z CIS senzoru.
6. Vytvořte a implementujte algoritmy pro detekci vad v materiálu na základě dat z CIS senzoru.
7. 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

Diplomová práce se zabývá zkoumáním CIS senzorů, jejich parametrů a průzkumem trhu. Obsahuje také srovnání senzorů s řádkovými kamerami, které se používají k podobným účelům, tedy v tomto případě ke skenování obalového materiálu. Diplomová práce obsahuje popsanou konstrukci zařízení, včetně osazení součástek a návrhu desek. V práci je detailně popsáno zpracování obrazu z CIS senzoru a různé předzpracování obrazu. Je zde popsán způsob detekce defektů, včetně jejich rozdělení a individuálního přístupu ke každému z druhů těchto defektů. Práce obsahuje popis GUI, včetně jeho funkcí a napojení na aplikaci zajišťující zpracování obrazu.

Klíčová slova

Kontaktní obrazový snímač, CIS senzor, zpracování obrazu, pohon, konstrukce

Abstract

The diploma thesis deals with the research of CIS sensors, their parameters and market research. It contains a comparison between sensors and line cameras, which are used for similar purposes, therefore in this case for scanning the packaging material. The diploma thesis contains the described construction of the device, including the assembly of components and the design of boards. The work describes in detail the image processing from the CIS sensor and various image preprocessing. There is also described method of defect detection, including their distribution and individual approach to each type of defect. The thesis contains a description of the GUI, including its functions and connection to the application dealing with image processing.

Keywords

Contact image sensor, CIS sensor, image processing, propulsion, construction

Bibliografická citace:

KOMZÁK, Daniel. *Detektor vad s využitím CIS senzoru.* Brno, 2020. Dostupné také z: https://www.vutbr.cz/studenti/zav-prace/detail/127107. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Peter Honec.

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma DETEKTOR VAD S VYUŽITÍM CIS SENZORU 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 jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: **31. května 2020** ……

podpis autora

Poděkování

Rád bych poděkoval svému vedoucímu práce, panu Ing. Peterovi Honcovi, Ph.D., za vedení této diplomové práce a cenné rady při realizaci. Dále bych chtěl poděkovat společnosti Camea za poskytnutý hardware a panu Ing. Pavlovi Čípovi, za poskytnuté rady při oživování tohoto hardwaru. V neposlední řadě bych chtěl poděkovat rodině a přátelům, kteří mě při studiu podporovali.

V Brně dne: **31. května 2020**

.....

podpis autora

Obsah

Seznam obrázků

Seznam tabulek

ÚVOD

Nedílnou součástí dnešního světa jsou obaly. Setkáváme se s nimi denně v obchodech nebo v zaměstnání. Každý výrobek je zabalen do nějakého z druhů obalů. Může se jednat o papírový, dřevěný, plastový, nebo tkaný. Proč vlastně balíme věci do obalů? Příkladů je mnoho: potřebujeme převézt určitý počet předmětů a nechceme, aby se jednotlivé kusy při přepravě rozsypaly, dále z hygienických důvodů, kde určitě nechceme, aby se například jídlo, ať už cestou do regálu obchodu nebo působením vnějších vlivů kontaminovalo atd. Proto je třeba obalový materiál kontrolovat, jestli nepodléhá nějakému defektu, což může být díra nebo deformace. V takovém případě je nutné upozornit obsluhu na daný defekt, aby zkontrolovala, jestli je tento defekt přípustný pro danou aplikaci obalu nebo se bude muset daný kus obalu vyříznout a vyhodit. Celý tento proces kontroly se skládá z nějakého senzoru (může jít o kameru, nebo jiný obrazový senzor) a mnoha detekčních algoritmů na detekování defektu. V průmyslu se většinou tato problematika řeší u obalového materiálu navinutého na rolích, a ty jsou následně rozmotávány a kontrolovány právě výše zmíněnými obrazovými senzory. Po kontrole jsou následně použity jako obal pro daný výrobek.

V této diplomové práci bych se chtěl věnovat především kontaktnímu obrazovému senzoru, který by měl mít mnoho výhod, ale i nevýhod oproti řádkovým kamerám, jež jsou v průmyslu daleko častěji používané právě pro kontrolu obalového materiálu.

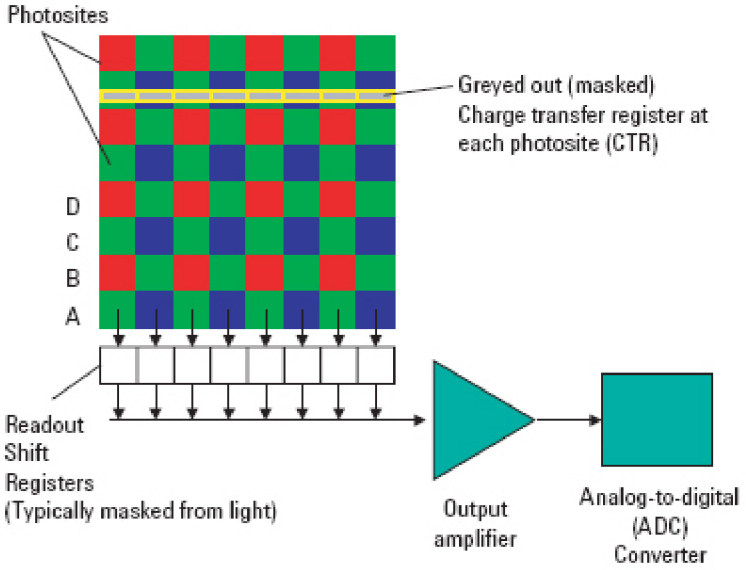
Samozřejmostí byl měl být program komunikující s CIS senzorem a obsahující algoritmy na skládání obrazu z tohoto senzoru. Postaral by se o následnou aplikaci algoritmů používaných při rozpoznání obrazu a rozpoznal obalový defekt při rychlém kontinuálním posuvu obalového materiálu. Na tento defekt by upozornil obsluhu prostřednictvím grafického rozhraní. Samotný model by měl také mít řešený pohon, který bude synchronizovaný právě se senzorem, aby nedocházelo k různým deformacím obrazu. Konstrukci modelu bych chtěl navrhnout pro 3D tisk a následně vytisknout na 3D tiskárně. Poté tuto kostru osadit pohonem a CIS senzorem, a to včetně řešení celého napájení modelu a přenosu obrazu do výpočetní jednotky.

1 SNÍMACÍ ČIPY OBRAZU

Abychom mohli detekovat defekty obalového materiálu, je potřebné převést obraz nějakým způsobem do výpočetní jednotky. K tomu slouží snímače obrazu, jenž jsou obsaženy ve fotoaparátech, kamerách, nebo skenerech. Tyto senzory přenášejí obraz na elektrický signál. V podstatě jde o soustavy jednoduchých fotosenzorů, které pracují na fotoelektrickém jevu. Těmito součástkami mohou být například fotodioda, fotorezistor, nebo fototranzistor. Dříve se místo těchto polovodičů používaly elektronky, avšak tato technologie byla postupně vytlačena křemíkovým mikročipem na bázi CCD nebo nějakým typem MOS technologie například CMOS nebo NMOS. [1]

1.1 CCD

CCD čipy se začaly používat v roce 1975 v televizních kamerách a různých skenerech nebo ve čtečkách čárových kódů. Přestože technologie CCD je z hlediska elektroniky velmi stará, i tak jde o technologii, která je díky propracovanosti použitelná i dnes. [1]
Čip se skládá ze světlocitlivých buněk, které při styku se světlem vytváří náboj. Čím více světla dopadne, tím větší je náboj. Náboj je poté transferován přes soustavu analogových registrů (odtud CCD) na výstupní zesilovač. Čtení se provádí po řádcích, kde je tento náboj zesílen a pomocí A/D převodníku převeden na digitální signál. Čtení probíhá vždy po řádcích, tedy nejdříve se přečte řádek A, poté B, atd. Nevýhodu tohoto čtení je rychlost, kdy je nutné vždy vyčíst celý čip, aby došlo k vyčištění nábojů v registrech. Tato metoda nedovolí číst například pouze střední část obrazu, vždy se musí načíst celý obraz. [1]



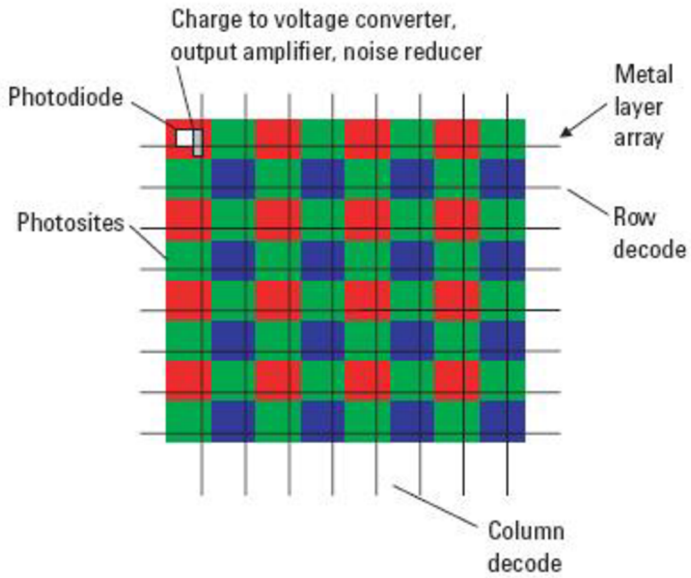
Obrázek 1: Příklad čtení z CCD [1]

Snímač musí být také osazen řadou elektronických součástek, které zajišťují zesílení, A/D převod, což zvětšuje velikost celého čipu, a tudíž i cenu. Spotřeba kompletního čipu je také celkem veliká, a to z důvodu vyššího napětí zařízení

(v řádech několika voltů) a energetické náročnosti při rychlých transferech náboje z analogových registrů. Zpravidla je zde použito vícero velikostí napětí, tudíž je potřeba několik fází o různých velikostech napětí. I přes tyto nevýhody mají CCD čipy velmi dobrou kvalitu obrazu s nízkým šumem oproti CMOS čipům. [1]
Další nespornou výhodou je kvantová efektivita. Tato efektivita udává, jak moc světla je přeměněno na náboj. Samozřejmě jsou zde ovlivňující faktory jako je charakter pixelu nebo vlnová délka světla. Vznikají ztráty (například odrazem světla) a převodové ztráty, kde se většinou u delších a u kratších vlnových délek nevygeneruje náboj. Tato efektivita u CCD může dosahovat až 100 %. [1]

1.2 CMOS

CMOS technologie se zde objevila již v roce 1963. Používá se k výrobě různých polovodičů i třeba procesorů. K využití ve snímací technice došlo ale až o dost později. První experimentování začala zkoumat NASA počátkem 90. let 20. století z důvodu menší spotřeby a menší náchylnosti k záření.
První velkou výhodou je jejich spotřeba. Z tohoto důvodu se začaly hodně využívat u fotomobilů a digitálních zrcadlovek. Hodnota spotřeby je o řád lepší než u CCD čipů. Přestože výrobci museli implementovat množství elektroniky k dosažení dobré zobrazovací kvality, která samozřejmě spotřebovává energii, tak i přesto jsou o dost úspornější. [1]
Další výhoda tkví ve výrobních nákladech. CMOS se vyrábí jednoduchou technologií, která je o dvě třetiny úspornější než výroba CCD. Jelikož není potřeba velké množství součástek, je výroba levnější i rychlejší. Z tohoto důvodu se výroba CMOS počítá v řádech desítek milionů, zatímco u CCD je to v řádech milionů kusů čipů ročně. [1]



Obrázek 2: CMOS rozložení [1]

CMOS čipy můžeme rozdělit na aktivní a pasivní. Pasivní čip je jednoduše tvořen fotodiodami. Aktivní čip má kolem každé buňky zesilovač a filtr k odstraňovaní

šumu. Tímto filtrem lze docílit menšího šumu, než je tomu u pasivního, ale zároveň tím, že každá buňka má svůj vlastní zesilovač, tedy vlivem nedokonalostí výroby, může každý zesilovač zesilovat jinak, tudíž je zde větší šum, než je tomu u CCD, kde je zesilování aplikováno na celý řádek. Dalším důvodem, proč mají CMOS větší šum, je přítomnost zesilovače přímo u fotodiody, kde na něj působí rušení. CMOS čipy jsou náchylné na šum při dlouhých expozicích. Dark current může být u CMOS senzorů až stokrát větší než u CCD. Problémem může být rovněž přetékání náboje do sousedních buněk. [1]

Malá citlivost snímačů je dána také činnou plochou, která je při horších čipech jenom 30 % z důvodu většího množství elektroniky, která musí být kolem každé buňky. V lepších čipech je to 60–70 %, což je výrazně lepší, avšak nedosahuje kvalit CCD, kde plocha může být až 100 %. [1]

Hlavní výhodou je adresace jednotlivých buněk. Nasnímaný obraz lze číst z jakéhokoliv místa, což umožňuje použití například náhledového snímku, nebo čtení pouze části obrazu, kde se očekává zajímavý objekt.

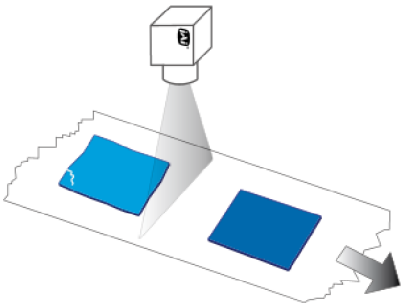
Rychlost CMOS je zpravidla také vyšší – dosahuje rychlosti až 150 MHz, kdežto CCD se pohybuje v maximálních rychlostech 20–30 MHz na kanál.

2 ŘÁDKOVÉ SNÍMAČE

Ke snímání a zpracování kontinuálně pohybujícího se obalového materiálu jsou nejlepší volbou řádkové snímače. Jejich rychlost snímání je daleko vyšší než u běžných kamer, tudíž nedochází k rozmazávání obrazu a mají mnoho dalších výhod, jimiž se budu zabývat v této kapitole.

2.1 Řádkové kamery

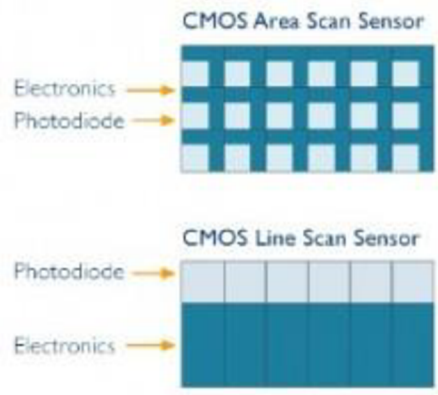
Řádková kamera funguje obdobně jako plošná kamera s tím rozdílem, že řádková kamera snímá pouze jeden obrazový řádek, respektive pixelový. S touto kamerou se můžeme setkat například u kancelářských skenerů. Jeden princip, jak nasnímat celý obraz je takový, že kamera je umístěna nad objekt, kde kamera je nepohybující a pod kamerou je lineárně pohybující se objekt, který chceme nasnímat. Druhým řešením je statický objekt a kamera se lineárně pohybuje nad objektem. Takto nasnímaný obraz je zaslán do nadřazeného systému, který z jednotlivých nasnímaných řádků skládá obraz. Celé toto uspořádání umožňuje vysokou rychlost snímání s poměrně velkými detaily. [2]



Obrázek 3: Princip snímání řádkové kamery [3]

2.1.1 Výhody řádkové kamery oproti plošné kameře

Kdybychom chtěli snímat pohybující se objekt maticovou kamerou, museli bychom zvolit krátký expoziční čas a velmi výkonné světlo pro osvětlení objektu, abychom zabránili rozmazání obrazu vlivem pohybu objektu, který chceme snímat, protože plošný snímač není tolik citlivý. Řádková kamera umožňuje snímat objekt s velmi krátkým expozičním časem, a tedy velmi vysokou rychlostí snímaných řádku za sekundu. Pro osvětlení objektu nám stačí intenzivně osvítit pouze snímaný řádek objektu. Toto může být řešeno například liniovými světly, která soustředí všechen světelný tok do úzké linky. Další výhodou je rozlišení těchto kamer. V současné době lze nasnímat obraz v přímém rozlišení běžně až 16000 bodů a frekvencemi i 200 kHz neboli 200000 lps–(lines per second). Podélné rozlišení pak závisí na velikosti pixelu, rychlosti pohybu materiálu a frekvenci snímání kamery. Takové parametry nyní nenabízí žádná z komerčně dostupných plošných kamer. [2]



Obrázek 4: Porovnání maticové a řádkové kamery [2]

Na obrázku č.4 lze vidět rozdíl mezi maticovou a řádkovou kamerou. Výhoda řádkové kamery je taková, že celá plocha je vyhrazena pro fotodiody a elektronika je umístěna mimo, zatímco u maticové kamery elektronika zaujímá část plochy. Z tohoto důvodu má řádková kamera až 2krát větší citlivost než maticová při stejné velikosti pixelu. [2]

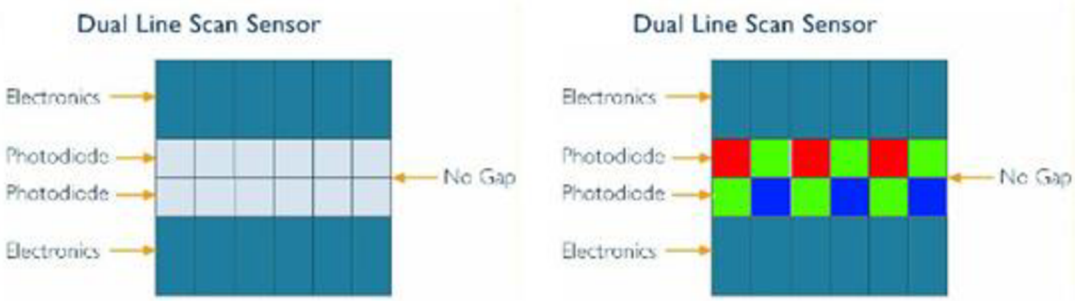
2.1.2 Základní druhy řádkových kamer

Jednořádková monochromatická kamera

Typicky nejjednodušším zástupcem z řádkových kamer je jednořádková černobílá kamera. U této kamery je celá plocha vyhrazena pro snímání k dosažení co největší citlivosti. Tento druh kamery se vyznačuje velmi vysokým rozlišením. V případě použití protokolu camera–link lze dosáhnout frekvence v desítkách tisících kHz. [2]

Dual–line řádková kamera

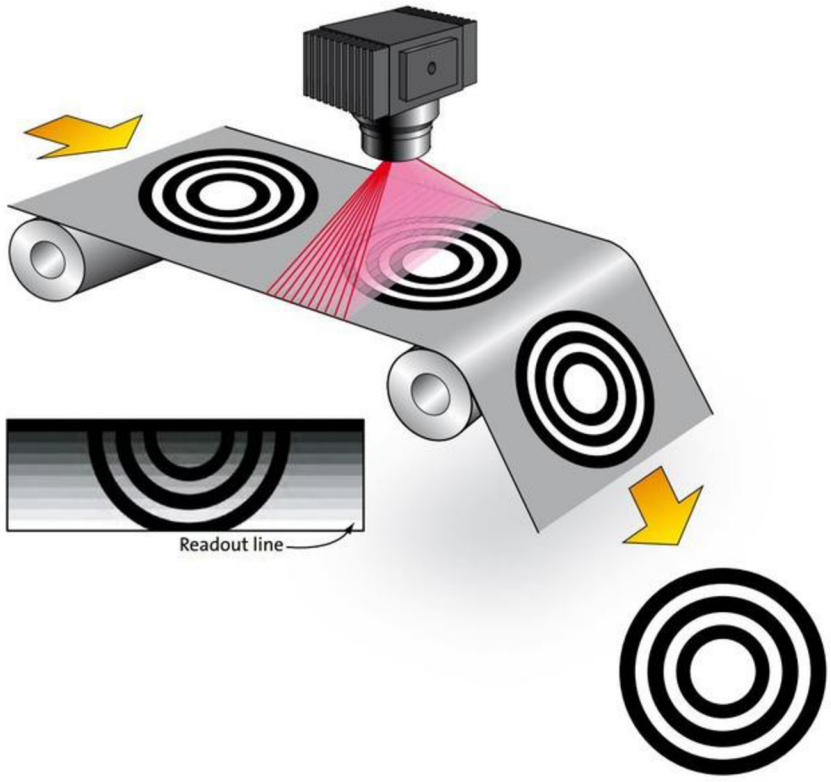
Tento druh kamery se vyznačuje umístěním senzoru do dvou řádků. Toto s sebou v případě jednobarevné kamery přináší mnoho výhod, jako je například zvýšení citlivosti metodou Binning, což je spojování sousedních pixelů tak, že se chovají jako jeden, což zmenší rozlišení, ale zvýší dynamický rozsah. Lze také zdvojnásobit frekvenci snímání, a to střídáním řádků. V případě barevné dvouřádkové kamery se setkáváme s uspořádáním do tzv. Bayerovy masky, kde se v jednom řádku střídají zelené a červené pixely. Ve druhém řádku jsou to modré a zelené pixely. Každé místo je nasnímáno oběma řádky, to znamená, že známe pokaždé dvě ze tří barev. Třetí barva se musí na základě znalostí vypočítat. Zpravidla se o to stará výpočetní jednotka. Výhody jsou vysoká rychlost a nízká cena. Nevýhodou je dopočítávání třetí barvy. [2]



Obrázek 5: Dual–line řádková kamera jednobarevná (vlevo) a barevná (vpravo) [2]

TDI řádková kamera

U standardní řádkové kamery je doba skenování omezena linkovou frekvencí. To znamená, že kmitočty někdy i přes 100kHz jsou omezeny na velmi krátké expoziční časy. Aby bylo možné použít krátké expoziční časy, je třeba aplikovat velmi silná liniová světla, což může být nákladné a nepraktické. Pro aplikace, kde je třeba využívat takto rychlé snímkování, lze použít TDI kameru. TDI technologie je skenování jedné obrazové linky vícekrát. Informace o lince jsou kopírovány synchronně s pohybem objektu a postupně exponovány. Hlavní výhodou je tedy možnost použití méně výkonných světel, tedy ušetření nákladů. Nevýhodou je pak rozmazání obrazu při špatné synchronizaci pohybu materiálu a skenovací rychlosti. TDI skenování je pak netolerantní ke kolísání rychlosti, tedy například při rozběhu, nebo doběhu pásu. [4]



Obrázek 6: TDI skenování [4]

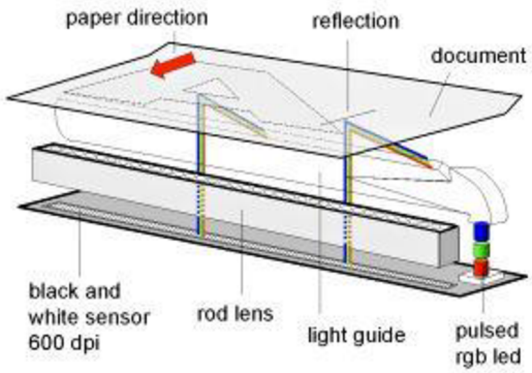
2.2 Kontaktní obrazový senzor

Kontaktní obrazový senzor je CMOS senzor. Má většinou v sobě již integrovaný osvětlovací systém, který může být reprezentován například RGB diodami, optický systém a systém snímání světla. Toto vše je zapouzdřeno do jednoho modulu. Stejně jako u CCD, CIS snímá světlo, které dopadá na jeho křemíkový povrch. Povrch je pak většinou rozdělen na čtvercové buňky. Avšak je zde jeden zásadní rozdíl. Velkost každé buňky na povrchu křemíku odpovídá velkosti buňky zachycené v reálu, tedy pokud je rozlišení senzoru 600dpi, velkost reálné buňky je 1/600 DPI. Jelikož zde není žádné zmenšování objektu, musí být senzor umístěný těsně nad snímaný objekt (cca 13 mm nad snímaným objektem), odtud tedy název kontaktní obrazový senzor. [5]

Optický systém je zde prezentován jako pole tyčových čoček, které přenášejí světlo ze snímaného objektu přímo na povrch křemíku. Vzhledem k faktu, že optická cesta ke křemíku je velmi krátká, je výroba kontaktního obrazového senzoru levnější než výroba CCD, jsou mnohem menší, vyžadují desetinu elektrické energie a vyžadují jen velmi malé, nebo žádné kalibrace. [5][7]

Osvětlení je, jak je již zmíněno, většinou RGB diodami (popřípadě pouze jednou z nich), které jsou namířeny na snímaný objekt. Tyto diody jsou postupně spínány a tím poskytují barevný obsah objektu nebo jsou sepnuty naráz, aby poskytly černobílou složku objektu. V případě přítomnosti pouze jedné diody dostáváme obraz pouze v barvě diody. Tento princip lze použít na detekci černobílého obrazu.

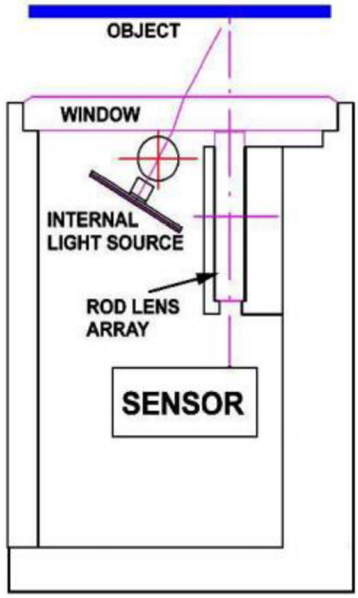
Nevýhodou nejčastěji používaného osvětlení RGB diodami je to, že nemohou poskytnout tak širokou škálu barev nebo dynamický rozsah, jako to dokážou bílé zářivky, nebo bílé LED s barevnými filtry, jako je tomu u CCD. Z toho vyplývá, že CIS senzory nejsou vhodné pro skenování fotografického materiálu nebo dokumentů, kde požadujeme zachování barvy. Je zde také velmi omezená hloubka ostrosti, což představuje problém pro skenovaný objekt, který není dokonale plochý. [5][7]



Obrázek 7: Kontaktní obrazový senzor [6]

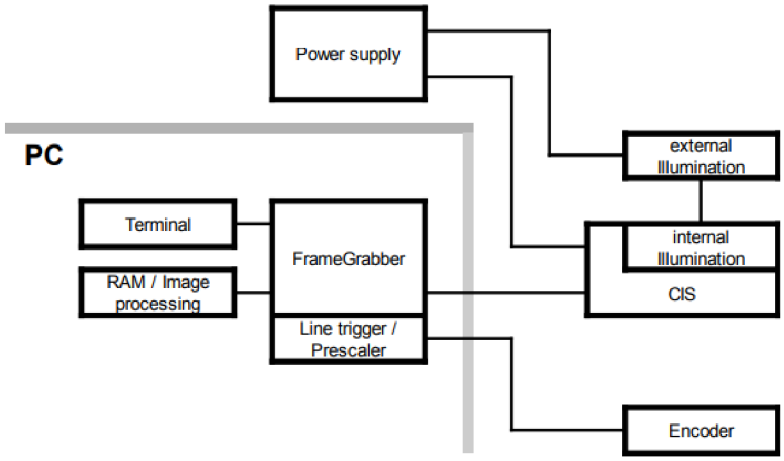
Typický CIS senzor, který je běžně použit ve skenerech, má velikost cca 216 mm, což je velikost formátu DIN A4. Běžné rozlišení těchto senzorů je 200–300 DPI a rychlost skenování je v řádech jednotek kilohertzů pro černobílý obraz. Cena těchto senzorů se pohybuje v jednotkách tisíců korun českých. V průmyslu se také můžeme setkat se senzory větších rozměrů například DIN A3 a větších, kde jejich

rozlišení se může pohybovat až 2400 DPI s rychlostmi až 250 kHz. Cena těchto senzorů je ve stovkách tisíců korun českých. [8]
Na obrázku č.8 lze vidět princip snímání obrazu. V případě použití interního osvětlení, které je nejčastěji natočeno o 45° oproti snímači, aby byl daný bod nasvícený co nejlépe, osvětlí daný bod barvou daného osvětlení. Pro získání červené složky červenou, pro získání zelené složky zelenou, pro získání modré složky modrou. V případě černobílého skenování jakoukoliv jinou barvou, která je zde integrována. Senzor nereaguje na barvu, ale pouze na jas vzniklý odrazem od objektu, proto je zde důležité postupné spínání jednotlivých barev. Po nasvícení bodu je světlo odraženo a přes tyčové čočky dopraveno k senzoru. [8]



Obrázek 8: Princip funkce CIS senzoru [8]

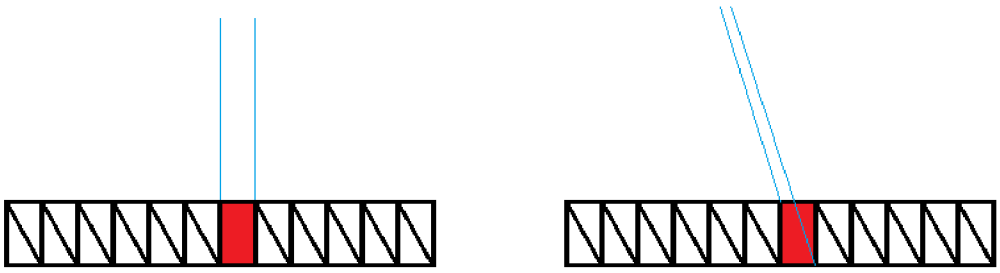
Na obrázku č.9 je znázorněno typické zapojení při použití CIS senzoru. Na tomto zapojení lze vidět i externí osvětlení, které musí být synchronizováno s interním v případě skenování více barev. Propojení mezi PC a CIS senzorem probíhá oboustranně. Počítač pomocí vyčtené hodnoty z enkodéru, který se stará o počítání ujeté vzdálenosti materiálu, zajistí vyslání signálu směrem k senzoru o zachycení jedné linky obrazu. Tato linka obrazu je postupně čtena a dále ukládána do paměti počítače. Poté dále skládána do celého obrazu. Po složení celého (nebo části) obrazu je dále zpracována, nebo již interpretována uživateli. Zdroj se stará o napájení celého senzoru a externího osvětlení. [8]



Obrázek 9: Typické blokové schéma zapojení při použití CIS senzoru [8]

2.3 Porovnání CIS senzoru a řádkové kamery

Nespornou výhodou řádkových kamer oproti kontaktním obrazovým senzorům je jejich umístění. Zatímco CIS senzor musí být umístěný těsně nad snímaným objektem, řádková kamera může fungovat o poznání výše. Řádková kamera také nemusí být kolmo ke skenovanému objektu, to znamená, že její umístění je mnohem víc variabilní. Tato výhoda s sebou přináší ale také jednu velmi velkou nevýhodu. Při použití pro detekování defektu může být defekt vyhodnocen jinde, než se reálně na daném objektu nachází a zároveň s menšími rozměry. Tato chyba se u CIS senzoru projevuje pouze nepatrně, protože se nachází velmi blízko skenovaného objektu, tudíž chyba úhlem je potlačena a zároveň je nastaven kolmo, to znamená, že výsledná chyba, která může nastat, je velmi nepatrná.



Obrázek 10: Chyba daná úhlem

Přítomnost CIS senzoru těsně nad daným objektem zároveň zvyšuje riziko poškrábání ochranného skla, k čemuž může dojít například zvlněním materiálu nebo přítomností nečistot.
Řádkové kamery také nedisponují svým vlastním osvětlením, tudíž je třeba přisvětlování z externího zdroje světla, což přinese řadu problémů s výběrem a nastavením světla, protože je mnoho druhů světel a každé se hodí na jiný materiál. Ačkoliv většina CIS senzorů obsahuje již integrované osvětlení, externí osvětlení by bylo také výhodou.

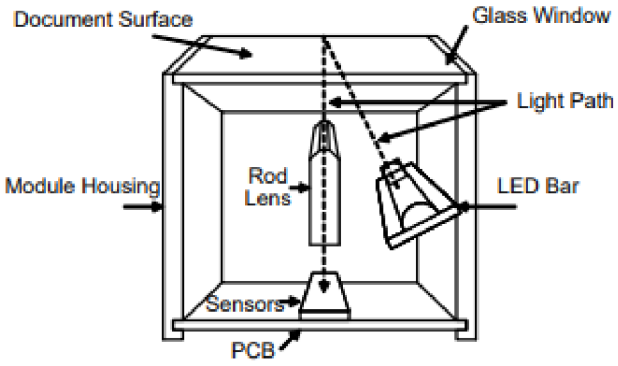
Z důvodu umístění CIS senzoru přímo nad skenovaný objekt není třeba žádného objektivu, přičemž u řádkové kamery potřeba je, což značně zvyšuje pořizovací cenu a opět je třeba zvolit správný objektiv, protože každý objektiv disponuje jinými vlastnostmi.
Co se týče rychlosti CIS senzor je typicky pomalejší než řádková kamera. Řádkové kamery dosahují frekvencí stovek kHz, CIS sensor se typicky pohybuje v řádech jednotek kHz, což znamená, že CIS vyhoví tam, kde není třeba vysoká rychlost posunu materiálu.
V poslední řadě je zde propojení s výpočetní jednotkou. V případě řádkových kamer existuje široká škála komunikačních rozhraní. V současné době jsou tři nejrozšířenější standardy. Gigabytový ethernet, Camera–Link a CoaXpress. Při výběru interface je třeba uvažovat jejich výhody a nevýhody. V případě CIS senzoru zde není žádný standard. Každý výrobce si vytváří svůj interface, což značně ztěžuje implementaci do systémů, ale naštěstí má většina řízení velmi podobný způsob řízení. [2]

3 VÝBĚR CIS SENZORU

Při průzkumu trhu jsem narazil na tři zajímavé typy CIS senzorů. V této kapitole bych se chtěl věnovat rozboru každého z těchto senzorů, popsat jejich parametry a zapouzdření a následně jeden z nich vybrat pro použití k mechanické části modelu.

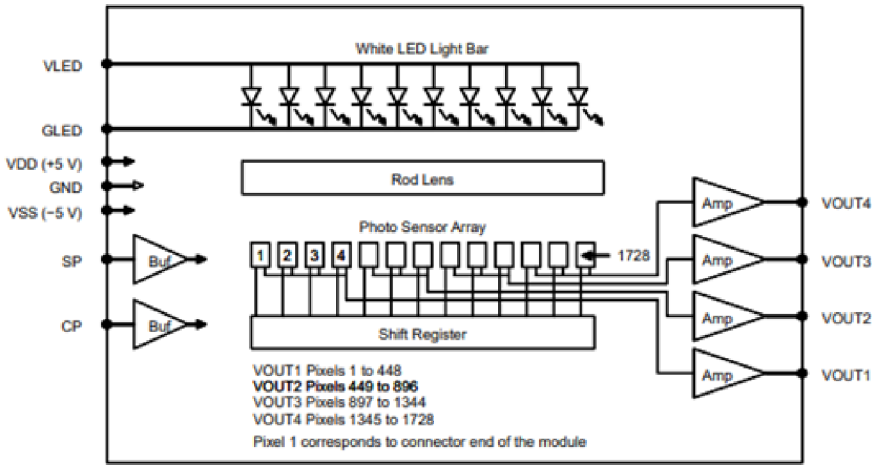
3.1 NOM02A4–MW60G

Jedná se o CIS senzor od výrobce ON Semiconductor. Senzor disponuje rozlišením až 200DPI, jeho velikost je 232.1 mm x 19.2 mm x 13,7 mm, kde 216 mm je skenovací délka. Délka je optimální pro skenování formátu DIN A4. Při této skenovací délce, při rozlišení 200DPI připadá 7,9 pixelů na 1 mm. Tento senzor používá 4 analogové výstupy k co největší přenosové rychlosti. Ke každému výstupu náleží čtvrtina skenovací délky senzoru. Přesné pixelové rozložení na jednotlivé výstupy je možné vidět na obrázku č.12. Na obrázku č.11 můžeme vidět celkové zapouzdření, které je vyrobeno převážně z plastu. Styková plocha s dokumentem je pak pokryta sklem, aby nešlo ke znečištění optických tyček nebo senzoru. [9]



Obrázek 11: Průřez NOM02A4–MW60G [9]

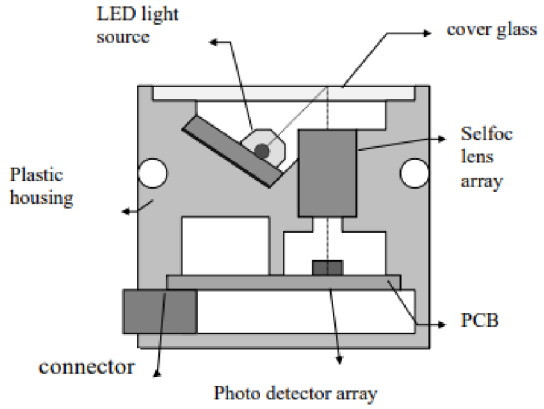
Modul obsahuje 27 senzorů. Každý z těchto senzorů má rozlišení 64 foto detektorů, což odpovídá celkovému počtu 1728 foto detektorů, které jsou připojeny na shift registry. Tyto registry mají dvojitou vyrovnávací paměť, která dopomáhá k sekvenčnímu čtení. Přenosová rychlost může být až 5 MHz, což znamená, že lze dosáhnout až 90 µsec/řádek. O osvětlení se starají bílé LED diody, což znamená, že skenování může probíhat pouze v černobílém režimu. Napájecí napětí Vdd je 5 V a Vss je -5 V. Pro napájení LED diod stačí 5 V, což je velmi velká výhoda, protože s celým modulem můžeme pracovat pouze s 5V napájením. Senzor je osazen 12–pin molex konektorem, který se nachází na spodní straně senzoru. [9]



Obrázek 12: Vnitřní zapojení NOM02A4–MW60G [9]

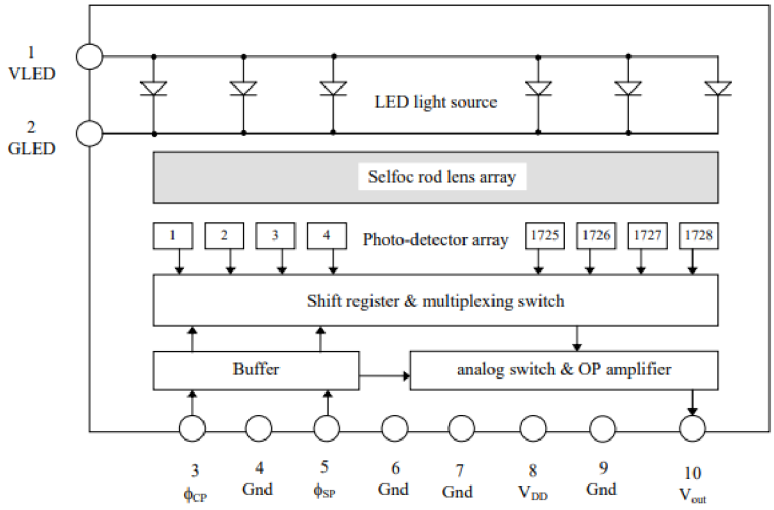
3.2 M106 – A4 – R1

Tento senzor je od firmy CMOS Sensor Inc. Senzor má osvětlení pomocí červených LED diod. Rozměry tohoto senzoru jsou 232 mm x 18 mm x 15 mm, kde 216 mm je skenovací délka. Rozlišení senzoru je 200DPI, čemuž odpovídá 1728 pixelů. Rozlišení je tedy 7,9 pixelů na 1 mm. Skenovací rychlost toho senzoru je při maximální frekvenci 500 kHz 5 ms na řádek. Řízení tohoto senzoru je velmi jednoduché, jelikož obsahuje pouze jeden analogový výstup. Výrobce udává, že nejlepší použití tohoto senzoru je pro skenování dokumentů, značek a použití pro fax zařízení. [10]



Obrázek 13: Řez M106–A4–G1 [10]

Pouzdro tohoto senzoru je z větší části z plastu. Styčná plocha s dokumentem je pokryta sklem, aby nedocházelo ke znečišťování osvětlení, foto čoček a samotného CMOS senzoru, který konvertuje obraz na elektronické signály. Pro napájení tohoto senzoru je zapotřebí 5 V a -5 V. K napájení LED diody je také zapotřebí 5 V. [10]

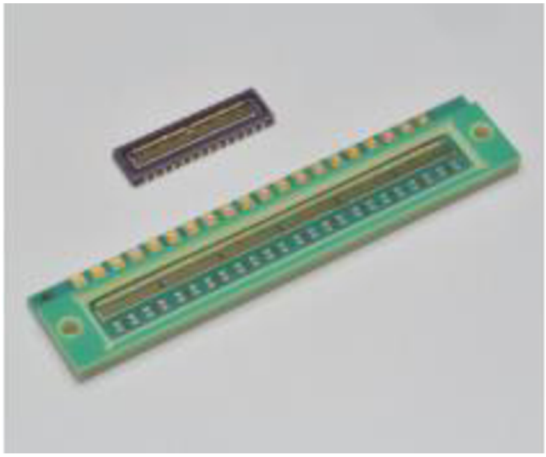


Obrázek 14: Blokové schéma M106–A4–G1 [10]

Na blokovém schématu na obrázku č.14 lze vidět, že LED světla mají oddělenou napájecí kaskádu což znamená, že jsou galvanicky oddělená. Výrobce udává 300 mA jako maximální odběr. Odběr senzoru bez LED diody by neměl přesáhnout 40 mA. [10]

3.3 S14416–06

Jedná se o CIS senzor od firmy HAMAMATSU Photonics. Rozměry senzoru jsou 61,2 mm x 12,7 mm x 3 mm. Skenovací délka tohoto senzoru odpovídá 48,768 mm. Většina konstrukce je z epoxidového skla. Snímací plocha je proti prachu opatřena borosilikátovým sklem. Je nutné podoktnout, že tento senzor nemá v pouzdře přisvětlovací diody, tudíž je nutné přisvětlovat skenovaný materiál, jinak data ze senzoru nebudou použitelná. Rozlišení tohoto senzoru je až 400DPI. Velikost pixelu je čtverec o rozměrech 63,5 µm x 63,5 µm. V případě tohoto senzoru se tedy jedná o 768 pixelů na celý skenovaný řádek. Senzor dokáže pracovat s maximální frekvencí 10MHz a má dva analogové výstupy. Pixely jsou rozdělovány do dvou shift registrů, kde první má první půlku pixelů a druhý má druhou půlku. Tento senzor dokáže pracovat v režimu dvou módů, a to sériovým a paralelním. V případě používání paralelního módu jsou využívány oba analogové výstupy, kde každý shift registr má jeden výstup. V případě sériového módu je používán pouze jeden výstup, každý lichý pixel je z jednoho z shift registrů a každý sudý je ze druhého shift registru. Maximální čtení je tedy 12594 řádků za sekundu Napájení pro tento senzor může být od 3 V do 5 V. [11]



Obrázek 15: Senzor S14416–06 [11]

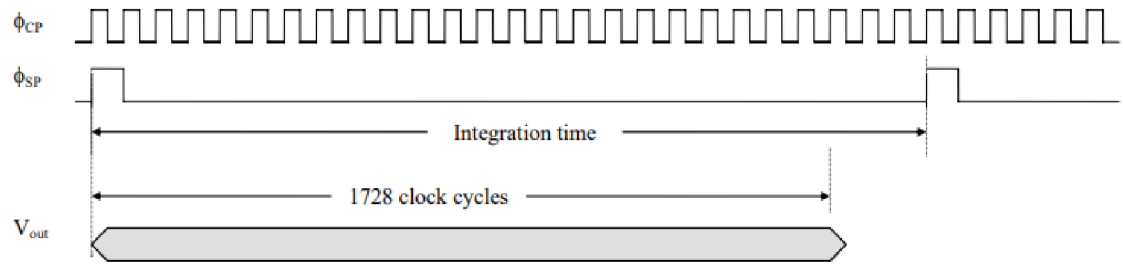
3.4 Výběr CIS senzoru

Všechny popsané senzory, co se týče řízení, se řídí skoro stejně. Liší se pouze malými rozdíly. Senzory mají podobné vlastnosti a dostačující rozlišení pro účely mé práce. Bohužel senzor S14416–06 nedisponuje vlastním osvětlením, což sice není nikterak velký problém, ale musel bych zde řešit externí osvětlení s co největší linearitou, aby nevznikaly stíny a různá světlejší a tmavší místa. Jelikož první dva zmiňované senzory mají integrované osvětlení, bude lepší vybrat jeden z nich.

První ze zmiňovaných senzorů, tedy NOM02A4–MW60G je velmi kvalitní senzor s dobrým osvětlením a dobrým rozlišením. Druhý zmiňovaný senzor M106–A4–G1 má téměř totožné vlastnosti. Ačkoliv je první zmiňovaný senzor mnohem rychlejší, rychlost senzoru M106–A4–G1 je dostatečná pro účely mého zadání. Hlavní výhodou je pouze jeden analogový výstup, kde skládání obrazu a následné zpracování bude mnohem jednoduší. Další výhodu je komerční dostupnost a oproti S14416–06 také jeho velikost, kde senzor dokáže skenovat formát DIN A4.

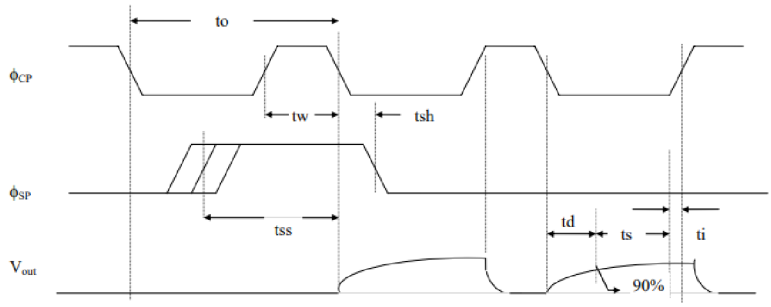
3.4.1 Řízení senzoru

Čtení dat z CIS senzoru probíhá z jednoho analogového výstupu. Celý senzor funguje na následujícím principu. Do senzoru se na ϕCP pouští signál clock pulse o maximální stabilní hodnotě 500 kHz. Výrobce udává, že při dodržení všech ideálních podmínek provozu by senzor mohl pracovat až na 600 kHz, ale takovéto nastavení je třeba testovat. Od rychlosti kmitočtu se odvíjí celková rychlost senzoru, tedy čtení, kterou chceme co největší. Minimální hodnota není uvedena. Je nutno dávat pozor na délku kladného pulzu, kde typicky bývá 0,5us, v případě ideálních podmínek může být až 1us. Při přivedení start signálu ϕSP senzor sejme napětí ze všech fotodetektorů a signál uloží do shift registrů. Každým dalším přijatým signálem z ϕCP se na V_{out} objeví signál z jednoho z fotodetektorů, který byl uložen do shift registru. Po přečtení všech dat z shift registru, je senzor připraven na další pulz na ϕSP. Další pulz zapříčiní opětovné sejmutí všech dat ze senzorů a uložení do shift registru. Celý tento proces se neustále dokola opakuje. Z důvodu nepřítomnosti více LED diod (senzor neumí detekovat barvy) svítí LED diody stále. Není tedy nutné stále je vypínat a zapínat. [10]



Obrázek 16: Timing diagram [10]

Na obrázku č.17 lze vidět přesné průběhy signálů na senzoru. Délka kladného pulzu na ϕSP musí být minimálně 50 ns, což značí *tss*, aby došlo k detekci signálu. Jak bylo již řečeno, kladný pulz ϕCP musí být 500 ns, což zde značí právě *tw*. Značka *to* je délka jednoho pulzu. Jak lze vidět na signálu V_{out}, čtení musí probíhat v signálu *ts*, abychom dostali správné hodnoty ze senzoru. Tento čas musí být minimálně 100 ns. Čas *td* je čas, kdy se dostává na výstup platná hodnota. Tento čas má maximální hodnotu 1 us. Dále je tu čas *ti*, kde ho výrobce značí jako neplatný čas, což je hodnota, kde se nesmí číst. Tato hodnota by neměla přesahovat 300 ns. [10]



Obrázek 17: Průběhy řízení [10]

Parametr	Minimum	Maximum	Jednotka
Analogové napětí na bílém papíře	1,8	2,2	V
Nerovnoměrnost bílého papíru	-30	30	%
Přilehlá pixelová nerovnoměrnost	-25	25	%
Analogové napětí na černém papíře	-200	200	mV
Nerovnoměrnost černého papíru		200	mV
Modulation transfer function	30		%
Linearita	0,85	1,1	

Tabulka 1: Změřené optoelektrické vlastnosti [10]

4 ZPRACOVÁNÍ OBRAZU

Po nasnímání a složení obrazu z přijatých pixelů ze senzoru (nebo jen části obrazu) je nutné defekt nějakým způsobem detekovat a určit, kde se nachází. Abychom byli schopni defekt detekovat a získat jeho hranice, je třeba obraz nejdříve předzpracovat. Existuje mnoho metod, kterými se toto předzpracování provádí. Účelem těchto metod je upravit matici obrazu pro další zpracování, přičemž výběr dané metody závisí na konkrétní problematice dané aplikace.

4.1 Úpravy obrazu

Úpravy obrazu používáme v případě, že chceme z obrázku odstranit šum, najít hrany, vyhladit obraz, odstranit zkreslení, rozmazat obraz, potlačit nebo zvýraznit určitě rysy, které jsou důležité pro další zpracování apod. Existuje velmi velké množství filtrů, detektorů a jiných různých algoritmů. Mezi nejčastěji používané filtry patří například:

- Gaussův filtr
- Filtrace mediánem
- Lineární filtrace
- Sobelův filtr

Na detekci hran můžeme použít například:

- Cannyho hranový detektor
- Houghovu transformaci

K detekci skvrn například:

- Laplaceův Gaussovský algoritmus
- Rozdíl Gaussiánů

Dále se budu zabývat pouze algoritmy, které budu používat pro detekci defektů v mé aplikaci. [12]

4.1.1 Morfologické uzavření

Vzhledem k tomu, že dilatace a eroze se vzájemně nevylučují, vznikají díky tomu nové operátory. Jedním z nich je uzavření, kde je obraz nejdříve dilatován a následně je aplikovaná eroze. Toto uzavření má následující vztah:

$$I \bullet B = (I \oplus B) \ominus B \tag{1.1}$$

Uzavření nám v podstatě spojí blízké objekty a zaplní malé díry. Tato operace mi pomůže v dalším zpracování, jelikož se sníží velikost detailů a zároveň je původní velikost zachována. [13]

4.1.2 Prahování obrazu

Prahování obrazu (anglicky tresholding) je jedna z nejjednodušších metod ve zpracování obrazu. Prahování může být použito například jako transformace obrazu do binární podoby. Je založeno na detekci každého pixelu v obrázku, který je porovnáván s prahem (anglicky treshold) a podle toho, je-li větší než daný práh, je mu přidělena hodnota binární 1, nebo binární 0. Tato operace je prováděna podle vztahu:

$$g(x,y) = \begin{cases} 1 & if \ f(x,y) > T \\ 0 & f \ f(x,y) \leq T \end{cases} \tag{2.1}$$

Kde g je nově vzniklý obrazec a x, y jsou souřadnice výsledného pixelu při porovnávání. Obraz pod odznačením f je původní obrazec a x, y je daný pixel v původním obrazci. T je porovnávací práh. Práh lze nastavit ručně, nebo zde existuje automatický způsob prahování. Tento způsob využívá zpětnovazební smyčku, která optimalizuje prahovou hodnotu a používá se například k rozdělení obrázku do dvou částí. Funguje to následujícím způsobem:

1. Vybere se počáteční prahová hodnota.
2. Pomocí této hodnoty se rozdělí obrázek na dvě části.
3. Naleznou se nové průměrné střední hodnoty obou obrázků.
4. Vypočte se nový práh.
5. Pokud je nový práh ve stanovených limitách, obrázky jsou hotovy.
6. Pokud je nový práh mimo stanovených podmínek aplikuje se nový práh na původní obrázek.

Prahování se dá použít i na barevný obrázek, kde se obraz rozdělí na jednotlivé RGB barvy a aplikuje se práh. Výsledná operace se zkombinuje funkcí AND. [14]

Obrázek 18: Prahování [15]

4.1.3 Semínkové vyplňování

Tento algoritmus vyplňuje oblasti obrazu, kde hledá sousedy stejné barvy, které tvoří nějakou uzavřenou oblast. Algoritmus potřebuje, abychom znali alespoň jeden bod, tzv. semínko, od kterého se začnou hledat sousedé. Algoritmus hledá všechny vnitřní body funkce, aniž by došlo k překročení hranic objektu. Nově najité body se taktéž stávají semínky. Pohyb v mřížce může být realizován ve 4 směrech (nahoru, dolů, doleva, doprava), nebo v 8 směrech, čemuž odpovídají 4 již vyjmenované směry a následně směry, které jsou posunuty od 45 °.
Pro segmentaci je lepší použití 4–okolí, protože při použití 8–okolí považuje algoritmus dvě plochy v rohovém bodu za jeden objekt. [12]

Obrázek 19: Vyplňování. Vlevo mince, uprostřed invertované barvy, vpravo vyplněno [16]

4.1.4 Cannyho hranový detektor

Pro nalezení defektů v obalovém materiálu je nutné tyto nějakým způsobem detekovat. Jedna z metod, jak zjistit, kde se daný defekt nachází, je Cannyho hranový detektor. Jedná se o sadu algoritmů, které jsou schopny postupným aplikováním najít hrany ve dvourozměrném obrazu. Tento algoritmus byl vyvinut Johnem F. Cannym v roce 1986. Algoritmus se skládá z pěti základních kroků. Prvním z pěti kroků je aplikace Gaussova filtru. Tento filtr nám vyhladí obraz a tím pádem odstraní šum. Takto vyfiltrovaný obraz je méně náchylný na chybné detekce hran. Je důležitý správný výběr Gaussovského jádra. Čím větší je velikost jádra, tím menší je citlivost detektoru na šum, ale tím větší je lokalizační chyba pro detekci okraje. [17]
Druhým krokem je určení velikosti a směru pomocí gradientu. Tento gradient může být určen algoritmy, jako jsou: Roberův křížový, Prewittový filtr, nebo Sobelův. Algoritmy vrací první derivaci v horizontálním a vertikálním směru.
Dalším krokem je nalezení lokálních maxim. Tato technika je použita ke ztenčování okrajů. Z předchozích gradientů hledá pouze lokální maxima a má za úkol odstranit body, které nejsou maximem. To nám zajistí, že hrana bude pouze v místě, kde se nachází největší hodnota gradientu. Technika hledá pixely, které mají nižší hodnotu v kladném a záporném směru. Až tento bod nalezne, prohlásí ho za skutečnou hranu. Pokud tato skutečnost nenastane, není tento bod prohlášen za hranu. [17]
Čtvrtým a pátým krokem, které lze spojit dohromady je aplikace prahu s hysterezí. Určí se dva prahy menší T1 a větší T2. Dva prahy se určují z důvodu možného šumu, kde by hodnota mohla kolísat. V případě překročení většího prahu T2 je

pixel označen jako hranový. Pokud se hodnota gradientu bude pohybovat mezi T1 a T2, pixel bude prohlášen za hranový pouze v případě, že sousedí s pixelem, který byl již dříve označen jako hranový. Pokud pixel nepřekročí menší práh, není prohlášen hranou. [17]

Obrázek 20: Cannyho hranový detektor [18]

4.2 OpenCV

Pro zpracování obrazu je nutné použít nějaké knihovny, protože samotné jazyky C, C++, nebo C# nedisponují žádnou funkcí ke zpracování obrazu. Jako výhodné se jeví použití OpenCV, které přináší do jazyka C++ funkce pro zpracování obrazu. OpenCV jsou knihovny, které se zaměřují na real–time počítačové vidění. Původně byly vyvíjeny společností Intel, ke které se poté přidala společnost Willow Garage a později ještě společnost Itseez, jež byla později získána společností Intel. První vydání těchto knihoven proběhlo v roce 1999. Knihovny jsou psány v programovacím jazyce C++ se soustředěním se na vysokou efektivitu, tedy i optimalizaci. K těmto knihovnám lze použít také knihovny IPP (Intel´s Integrated Performance Primitives), ale pouze v případě použití Intel architektury, která přináší low–level optimalizované rutiny. [19]

Tyto knihovny jsou multiplatformní, běží tedy pod Linuxem, Windowsem, Mac OS X a jsou aktivně vyvíjeny do programovacích jazyků Python, Java, MATLAB, atd. Také existují porty na Android a iOS. Knihovny jsou vyvíjeny jako open–source pod BSD licencí. [19]

Jedním z hlavních cílů celé této koncepce je přinést uživatelům knihovny, které budou jednoduché k použití, ale zároveň budou schopné sofistikovaně pomoci uživatelům vytvořit jejich aplikaci co nejrychleji. Knihovny přinášejí přes 500 funkcí, čímž jsou schopny pokrýt velmi široké rozpětí v počítačovém vidění zahrnující inspekci pro výrobní průmysl, lékařské zobrazování, bezpečnost, stereovizi, robotiku, kalibraci kamer apod. [19]

Jelikož je počítačové vidění a strojové učení velmi blízké, OpenCV také obsahuje knihovnu pro strojové učení (ML module). Tato knihovna je zaměřena na statistické patternové rozpoznání a clustering. Tento modul je velmi užitečný pro použití v počítačovém vidění, ale je natolik obecný, aby mohl být použit i ve strojovém učení. [19]

OpenCV je určený pro programátory, co jsou si vědomi aspektů počítačového vidění a pro lidi, kteří vědí, kde je zapotřebí používat počítačové vidění. Díky tomu, že je OpenCV open–source, tak vznikla velká komunita lidí, která dokonce obsahuje i lidi z nadnárodních firem, například lidé z Google, Microsoftu, SONY, IBM apod. kteří tyto knihovny využívají. [19]

5 HARDWARE PRO IMPLEMENTACI CIS SENZORU

K získání obrazu potřebujeme přesně vědět, jakou rychlostí se daný předmět pohybuje. Jedním ze způsobů je zajištění rovnoměrného pohybu materiálu, kde je CIS senzor možné provozovat na stálé řádkové frekvenci. Druhým způsobem je triggerovat CIS senzor na základě posunu materiálu, tedy s využitím například inkrementálního enkodéru.
Prvním nápadem, jak tohoto docílit, bylo přivedení inkrementálního čítače (enkodéru), který by byl roztáčen pohybujícím se objektem. Tento čítač by po určité vzdálenosti dal pulz senzoru a ten by sejmul obraz. Dále by se zde nacházel stejnosměrný motor, který by se staral o rozpohybování objektu. Tento koncept nebyl špatný, ale nakonec jsem zvolil jiné řešení. Spojil jsem řešení dvou problémů v jeden a zvolil jsem krokový motor. V případě krokového motoru je pevně dána rychlost otáčení, pokud nedojde ke ztrátě kroku zapříčiněné poddimenzováním motoru, rychlému náběhu, nebo nevyužitím mikrokroků. Aby byla dosažena synchronizace hodin, změřím s FPS z CIS senzoru a vypočítám danou frekvenci, která bude třeba pro generátor impulzů pro motorový driver.

5.1 Pohon

Pro pohon byl tedy vybrán krokový motor. Krokové motory lze zařadit do speciálního typu synchronního motoru, který se vyznačuje nespojitým pohybem. Nespornou výhodou krokových motorů je jejich vysoká přesnost a velmi vysoká opakovatelnost. Parametry krokových motorů jsou ideální pro použití, kde je třeba vysoká přesnost, například CNC stroje, 3D tiskárny, nebo jenom obyčejné tiskárny. Jednou z mála nevýhod krokového motoru je jeho složitější ovládání. Krokový motor nelze provozovat bez driveru. Tímto driverem a jeho kvalitou jsou ovlivňovány vlastnosti krokového motoru. Profesionální drivery dokážou zajistit velmi velkou rychlost, rozlišení, dokážou tlumit vibrace apod. Řízení krokového motoru tedy probíhá impulzně. Driver musí zajistit správné přepnutí proudu mezi fázemi motoru v definovaných posloupnostech, aby došlo k otočení rotoru motoru. [20]

5.1.1 Krokový motor NEMA 17

V rámci mé práce jsem vybral pro pohon krokový motor NEMA 17, přesněji NEMA JK42HS40–1704. S těmito krokovými motory mám dobré zkušenosti, a to byl hlavní důvod zvolení tohoto motoru. Bral jsem v úvahu i menší verzi tohoto motoru, kde parametry byly také dostačující, ale z důvodu menší dostupnosti byl tento motor dražší. V úvahu jsem bral i motor s označením 28BYJ–48. Tento motor měl velmi jemné krokování, protože je osazen převodovkou, ale má velmi malý krouticí moment. V případě, že bychom dali danému motoru rozmotat rulu obalového materiálu, nerozvinul by ji, protože by na ni neměl dostatečnou sílu. [21]

Obrázek 21: Krokový motor [21]

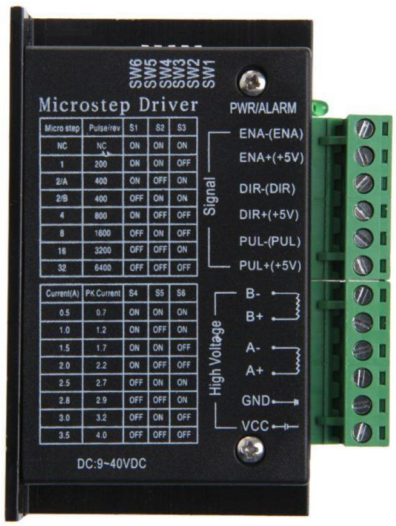
Jedná se o dvoufázový krokový motor. Motor má rozměry 42,3 x 42,3 x 39,8 mm. Z tabulky č.2 se můžeme dozvědět maximální proud jednotlivou fází, který může být až 1,7 A. Tento motor má natočení za jeden krok o 1,8°, čemuž odpovídá 200 kroků na otočení o 360 °. Toto krokování lze změnit pomocí dalšího elektronického zmenšení. Motor také disponuje malým momentem setrvačnosti. [21]

JK42HS40–1704	Specifikace
Natočení za krok	1.8 °
Maximální teplota	80 ℃
Provozní teplota	-20 ℃~+50 ℃
Izolační odpor	100 MΩ Min. ,500 V DC
Proud/fáze	1.7 A
Odpor/fáze	1.65 Ω
Induktance/fáze	3.2 mH
Přídržná síla	0.48 Nm
Průměr osy	5 mm

Tabulka 2: Specifikace krokového motoru [21]

5.1.2 Driver krokového motoru TB6600

K roztočení hřídele krokového motoru je nutné postupně spínat jeho jednotlivé fáze, respektive jeho dva druhy cívek. K tomuto spínání slouží driver krokového motoru. Jak lze vidět na obrázku č.22, do motorového driveru přivedeme obě fáze krokového motoru na pozice A a B, dále zde přivedeme napětí o velikosti 9–40 V stejnosměrného napětí. Tímto přivedeným napětím uvedeme v chod celý driver. Ke spuštění motoru je třeba přivést napětí 5 V na ENA, dále případně na DIR, který nám mění směr a na PUL, kde při každém pulzu driver udělá jeden step, popř. mikro step – záleží, jak je driver nastavený. V tabulce č.3 se můžeme dočíst o různém nastavení tohoto driveru. Nastavováním spínačů 1–3 volíme různé krokování. Driver umí krokovat od 200 pulzů na otáčku po 6400 kroků na otáčku. Pro účely mé práce mi postačí krokování po 1600 pulzech na otáčku. Spínači 4–6 volíme proudové omezení pro daný motor. Nastavení lze zvolit po určitých krocích od 0,5 A po 3,5 A pro stálé zatížení. Toto nastavení se zvolí v závislosti na připojeném krokovém motoru. Pro mou práci jsem si zvolil nastavení s omezením proudu na 1 A. [22]



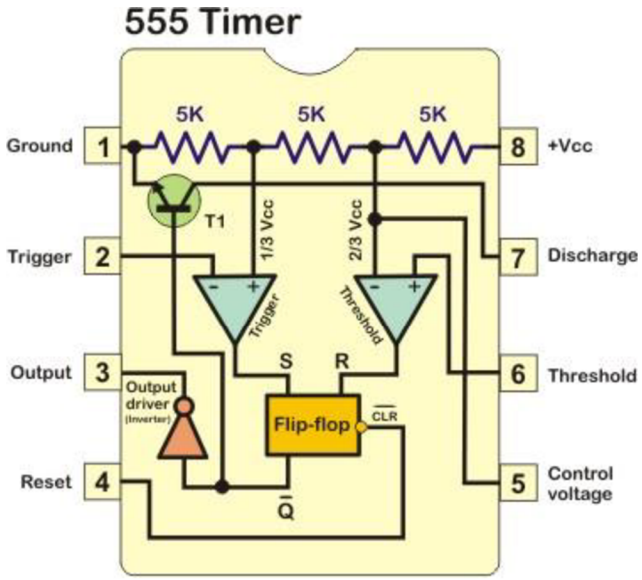
Obrázek 22: Driver krokového motoru [22]

Spínač 1	Spínač 2	Spínač 3	Počet mikrokroků	Pulzů/otáčku
1	1	1	NC	NC
1	1	0	1	200
1	0	1	2/A	400
0	1	1	2/B	400
1	0	0	4	800
0	1	0	8	1600
0	0	1	16	3200
0	0	0	32	6400
Spínač 4	Spínač 5	Spínač 6	Proud (A)	Špičkový proud
1	1	1	0,5	0,7
1	0	1	1	1,2
1	1	0	1,5	1,7
1	0	0	2	2,2
0	1	1	2,5	2,7
0	0	1	2,8	2,9
0	1	0	3	3,2
0	0	0	3,5	4

Tabulka 3: Nastavení krokového motoru [22]

5.1.3 Generátor impulzů NE555

Jde o integrovaný obvod, který nejčastěji slouží jako generátor pravoúhlých signálů. Je velmi populární z důvodu nízké ceny, jednoduchosti použití a dobré stability. Je schopen fungovat jako relaxační multivibrátor, generátor pulzů, blikač, generátor tónů atd. V podstatě jakýkoliv obvod, který potřebuje nějakou formou časového řízení. Jeho základní forma je komerčně vyráběna jako 8–pinový DIP. [23]



Obrázek 23: Vnitřní zapojení NE 555 [24]

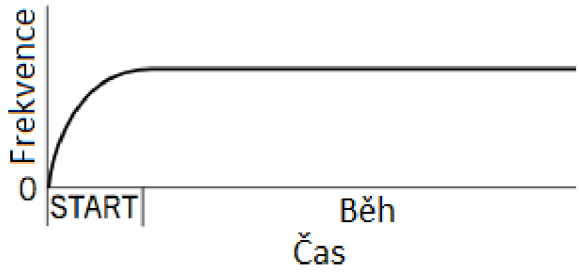
Pro moji práci využiji tento klopný obvod jako astabilní. Jde o obvod, kde se nepřetržitě střídají úrovně napětí. Toto zapojení využívá napětí na kondenzátoru C1, který se periodicky vybíjí a nabíjí. Obvod nemá žádný ze stavů stabilní, proto tedy astabilní klopný obvod. Při přivedení napětí na obvod je na výstupu log. 1 a kondenzátor C1 je vybitý. Začne se tedy přes rezistory R1 a R2 nabíjet. Jakmile dosáhne 2/3 napájecího napětí, zapůsobí treshold a klopný obvod RS překlopí a na výstupu bude log. 0. Také se otevře vybíjecí tranzistor a kondenzátor C1 se přes odpor R2 začne vybíjet. Jakmile napětí na kondenzátoru klesne na 1/3 napájecího napětí, opět zapůsobí trigger a překlopí klopný obvod RS a na výstupu se objeví log. 1. Doba nabíjení je dána odpory R1 a R2 tedy podle rovnice:

$$t = \ln(2) * C1(R1 + R2) \tag{3.1}$$

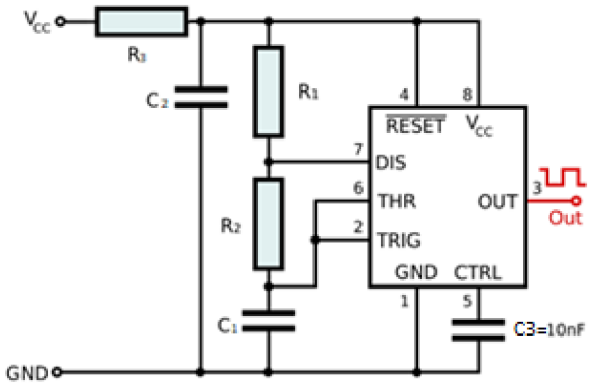
a vybíjení pouze odporem R2 tedy podle rovnice:

$$t = \ln(2) * C1R2 \tag{3.2}$$

Při použití pro krokový motor je třeba také řešit tzv. náběhovou rampu. Tato rampa slouží k tomu, aby krokový motor neztratil synchronizaci s driverem. To znamená, že při přivedení velké frekvence do netočícího se motoru dojde k rozběhu motoru, tedy motor zůstane stát. Aby tento jev nenastal, je zde potřeba použít náběhovou rampu. Tato rampa je řešená kondenzátorem C2, který je nabíjen přes odpor R3. Na obrázku č.24 lze vidět náběhovou rampu. Po nabití kondenzátoru C2 se frekvence ustálí na požadovanou hodnotu. [25]



Obrázek 24: Náběhová rampa



Obrázek 25: Zapojení NE555 jako AKO s náběhovou rampou [26]

5.2 Přenos mezi výpočetní jednotkou a CIS senzorem

V této podkapitole bych se chtěl zabývat způsoby přenosu obrazu mezi CIS senzorem a výpočetní jednotkou. Měl jsem připravené dvě koncepce přenosu dat, bohužel jedna z nich byla z technických důvodů neproveditelná.

5.2.1 Prvotní návrh přenosu

Původní koncepce přenosu mezi CIS senzorem a výpočetní jednotkou byla taková, že by se jako výpočetní jednotka použilo Raspberry Pi 3. Jelikož výstup z CIS senzoru je analogová hodnota, použil bych 8–bitový paralelní A/D převodník s typem paměti Sample and Hold s dostatečně rychlým vzorkováním a zapojil bych je do GPIO portů na Raspberry PI. Vzorkovací frekvence GPIO portů Raspberry PI by měla být sice na hraně možností, ale stále dostačující k bezchybném provozu. Ke zpoždění signálu pro A/D převodníky, které by bylo třeba, kvůli správnému navzorkování signálu z CIS senzoru, bych použil RC článek s komparátorem. Tento koncept bohužel nelze použít z důvodu vnitřního přerušení Raspeberry PI. Jelikož se pohybujeme ve velmi vysokých frekvencích, přerušení procesoru zde hraje velkou roli. V momentě, kdy by nastalo přerušení, nemusela by se navzorkovat data a vznikla by zde pixelová díra, kde by nebylo zřejmé, zda je zde defekt obalového materiálu, nebo jestli je materiál v pořádku.

5.2.2 Přenos přes ethernetové desky

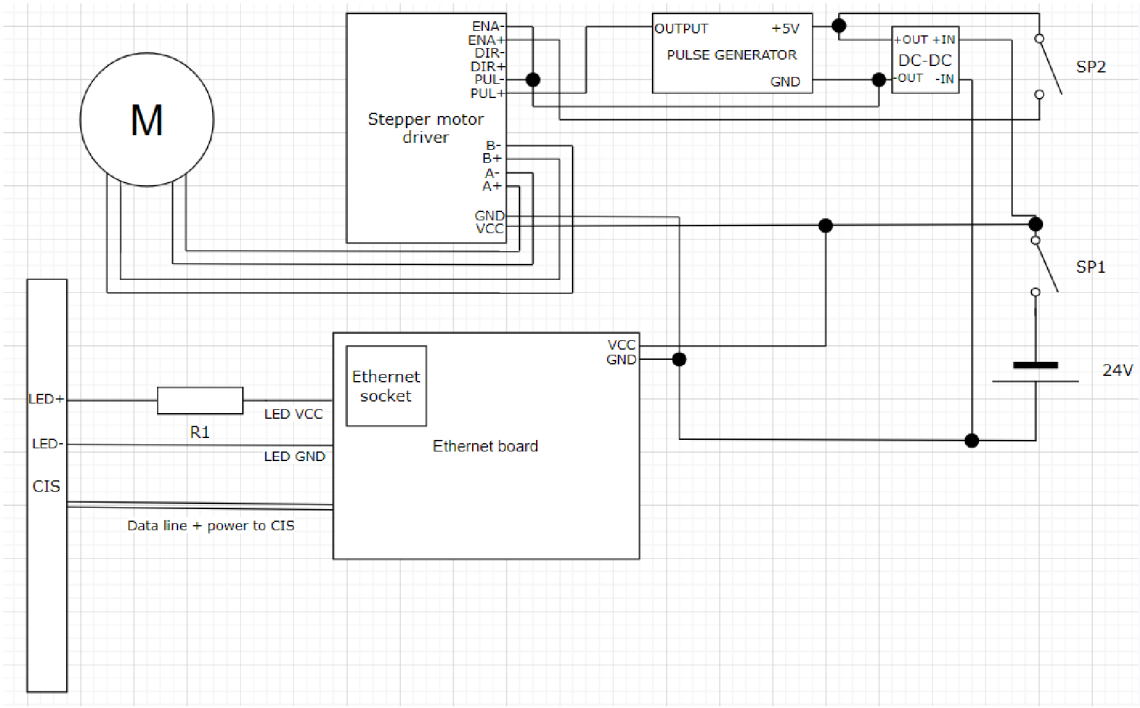
Druhou koncepcí je soubor ethernetových desek, který vyrobila společnost Camea. Tyto desky používá ve svých řádkových kamerách, ale při drobné úpravě softwaru a hardwaru lze tyto desky použít také na CIS senzor, a to z důvodu velmi

podobného způsobu vyčítání. Deska se stará o napájení CIS senzoru včetně přisvětlovacích diod a pořizování obrazu. Je zde jedna hardwarová změna oproti desce, jež se používá k řádkovým kamerám a to, že je zde invertor, který CIS senzor vyžaduje k otočení signálu, aby bylo možné použít současný A/D převodník.
Deska se skládá ze tří poddesek, na nichž se nachází napájecí konektor, ethernetový konektor, identifikační diody, slot na SD kartu, FPGA, MCU, Ethernetový čip apod. Celé zařízení je napájeno 24 V stejnosměrného napětí. Na SD kartě, která je nutná pro běh zařízení se nachází firmware pro FPGA a dále také parametry A/D převodníku a parametry kamery, tedy v tomto případě CIS senzoru. Při přivedení napájení se deska začne startovat. MCU má za úkol nahrát firmware do FPGA, které zajišťuje následné pořizování obrazu a následné bufferování a zasílání dat do ethernetového čipu. Ten se stará o komunikaci mezi PC a těmito deskami. Pro komunikaci je použit UDP protokol. Ačkoliv je tento protokol nepotvrzovaný, je daleko rychlejší než TCP a jelikož je tato deska primárně používána pro řádkové kamery, které mají mnohem větší framerate, než má CIS senzor, je zde tento protokol žádaný, ačkoliv by zde klidně mohl být použit protokol TCP.
IP adresa zařízení je nastavena na 192.168.6.1, proto je nutné při připojení k výpočetní jednotce nastavit adresu jinou, než používá toto zařízení. Ovladače jsou psány v jazyce C++ a je zde použit speciální formát snímku, který používá tato společnost. Naštěstí lze poměrně snadno převést tento formát do OpenCV.
Díky této desce lze jednoduše použít PC jako výpočetní jednotku. Tento fakt je velmi důležitý, protože při použití jednodeskového počítače jako je Raspberry PI by nemusel stačit výpočetní výkon a bylo by zapotřebí stejně využít PC jako výpočetní jednotku. Raspberry PI by sloužil jen jako prostředník, tedy posílal by snímky do PC, což je vlastně tato koncepce akorát s využitím ethernetových desek.

Obrázek 26: Ethernetová deska

5.3 Zapojení

Jak již bylo řečeno, ethernetové desky potřebují ke svému chodu 24 V stejnosměrného napětí. Abych nemusel řešit dvojí napájení, tedy více rozdílných úrovní napětí, snažil jsem se veškerý hardware postavit na těchto 24 V. Jak lze vidět na obrázku, jsou zde 3 druhy zařízení. Ethernetové desky, které potřebují 24 V, dále je potřeba napájet motorový driver a desku s generátorem pulzů. Na motorový driver lze připojit napětí od 9 V do 40 V. Velikost napájení má ale vliv na výkon motoru. Při mém testování byla velikost od 16 V a výše přijatelná k provozu motoru a naopak od 32 V výše se již motor hřál i přes proudové omezení, jež je mnohem nižší, než na jaké je motor konstruován. Připojil jsem tedy napájení napřímo. Poslední deskou je generátor pulzů. Zde je omezení samotného NE555, kde je napájecí rozmezí od 5 V do 16 V. Je nutné podotknout, že motorový driver může být sice napájen vyšším napětím, ale ovládací obvody pracují s 5V logikou. Díky tomuto faktu je nejjednodušším řešením použití těchto 5 V. Na desce se tedy nachází DC–DC měnič, který mění 24 V na 5 V, což zajistí, že i výstup NE555 bude 5 V.



Obrázek 27: Celkové zapojení

Na schématu lze vidět odpor R1. Na desce nelze snížit napětí pro LED diody, ale ke správnému chodu je třeba snížit intenzitu osvětlení z důvodu, že LED svítí moc silně a na snímcích se ztrácí informace o defektech. Jelikož se jedná o odpor u LED diod, chci si pro jistotu spočítat ztrátový výkon na odporu, abych nezvolil špatný ztrátový výkon rezistoru a rezistor se příliš nehřál. Při testování mi vyšlo, že optimální nastavení se pohybuje v rozmezí 5–15 Ω. Zvolil jsem odpor 15 Ω pro výpočet co největšího ztrátového výkonu. Proud osvětlením bez rezistoru je 200 mA a s rezistorem 125 mA. Úbytek napětí je tedy:

$$Úbytek\ napětí = R * I = 15 * 0{,}125 = 1{,}875\ \text{V} \qquad (4.1)$$

Ztrátový výkon je tedy:

$$Ztrátový\ výkon = U * I = 1{,}875 * 0{,}125 = 0{,}234\ \text{W} \qquad (4.2)$$

Zvolil jsem tedy odpor se ztrátovým výkonem 0,6 W.
Spínač SP1 spíná napájení pro celé zařízení. Spínač SP2 povoluje spuštění motoru tak, že je přivedeno napětí na ENA. Toto zapojení tedy nevyužívá náběhové rampy při spouštění motoru. Využije se pouze v případě, že je spínač SP2 zapnutý před zapnutím celého zařízení. Důvod, proč není využita náběhová rampa je ten, že při rychlosti a zatížení, při které je motor provozován, jí není potřeba. Pokud by se vyměnil CIS senzor za rychlejší, lze toto zapojení přepojit tak, aby se využívala i při spuštění přes tlačítko SP1.

5.3.1 Realizace a ladění NE555

Ještě před realizací bylo nutné předem vědět, na jaké frekvenci by měl oscilátor kmitat. Při spuštění programu se po čase objevuje framerate. Tento framerate čítá průměrnou hodnotu 0,545 FPS. Jeden snímek čítá 501 řádků, z čehož lze vypočítat počet řádků za sekundu:

$$\frac{\text{řádků}}{sec} = počet\ \text{řádků} * FPS = 0{,}5456875\ * 501 = 273{,}045\ \text{ř/s} \qquad (5.1)$$

Při rozlišení 200 DPI, čemuž se rovná 7,87 pixelů na 1 mm, je vzdálenost za 1 sekundu:

$$vzdálenost = \left(\frac{\text{řádků}}{\text{sec}}\right) * pixely = \frac{273{,}045}{7{,}87} = 34{,}69\ \text{mm} \qquad (5.2)$$

Tedy za 1 sekundu je třeba, aby obal urazil vzdálenost zhruba 34,69 mm. Průměr hnacího kola na hlavní hnané hřídeli je 17,83 mm. Teď je nutné spočítat obvod kola:

$$otočka = 2 * \pi * \text{r} = 2 * \pi * \left(\frac{17{,}83}{2}\right) = 56{,}01\ \text{mm} \qquad (5.3)$$

Je tedy třeba vypočíst, kolikrát se má hlavní kolo otočit za sekundu, toho docílíme výpočtem:

$$\frac{otáček}{sec} = \frac{vzdálenost}{otočka} = \frac{34{,}69}{56{,}01} = 0{,}619\ \text{ot/s} \qquad (5.4)$$

Nastavení na motorovém driveru je 1600 pulzů na jednu otáčku motoru. Toto nastavení jsem zvolil jako kompromis. Nezvolil jsem 200 pulzů na otáčku z toho důvodu, že kdyby došlo k lehké nestabilitě generátoru, tedy nastavená frekvence by byla v jednu chvíli větší, respektive menší než žádaná, neovlivnilo by to tolik ujetou vzdálenost. Žádaná frekvence se vypočítá:

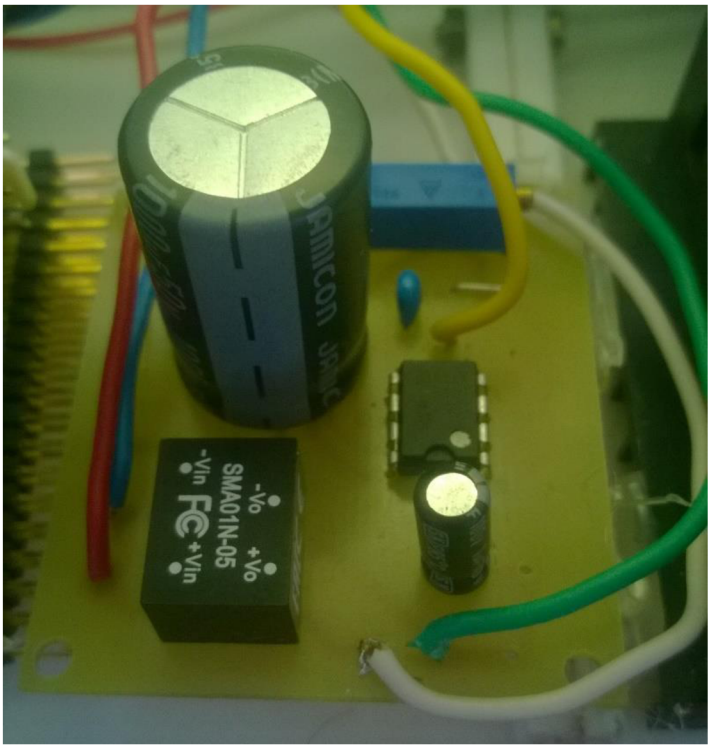
$$frekvence = pulzy * \left(\frac{otáček}{sec}\right) = 1600 * 0{,}619 = 990{,}4\ \text{Hz} \qquad (5.5)$$

Nyní je známo, na kterou frekvenci je třeba nastavit. Jako filtrovací kondenzátor C3 jsem zvolil klasicky 10 nF. Kondenzátor C1 jsem zvolil 1 µF, tedy od tohoto kondenzátoru se budou odvíjet veškeré výpočty:

$$R = \frac{1}{\ln(2) * C * f} = \frac{1}{\ln(2) * 1^{-6} * 990{,}4} = 1457\ \Omega \qquad (5.6)$$

Kde R=R1+2*R2. Není zde použita střída přesně 1:1. V tomto zapojení to nelze udělat z toho důvodu, že odpor R2 zde musí být, jinak by při vybíjení kondenzátoru přes kolektor došlo ke zkratu. Zvolil jsem tedy odpor R2=150 Ω. R1 by měl být 653,5 Ω. V praxi však odpory mají tolerance a vodiče nemají nulový odpor. Z tohoto

důvodu jsem jako rezistor R1 zvolil víceotáčkový cermetový trimr a pomocí osciloskopu jsem naladil frekvenci oscilátoru na 990,4 Hz.
Na náběhovou rampu je použit kondenzátor C2=1 mF a odpor R3 před ním je o velikosti 220 Ω. Pomocí měření jsem zjistil, že začínající frekvence je 538 Hz, a ta se zvětšuje podle nabíjení kondenzátoru až do plné frekvence. Plné frekvence je dosaženo po 75 ms.

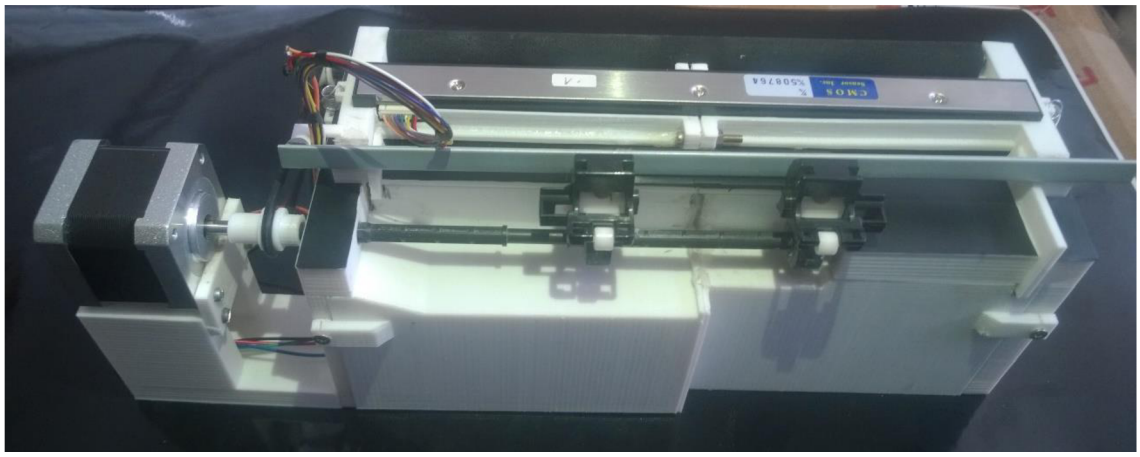


Obrázek 28: Generátor impulzů

5.4 Skelet konstrukce

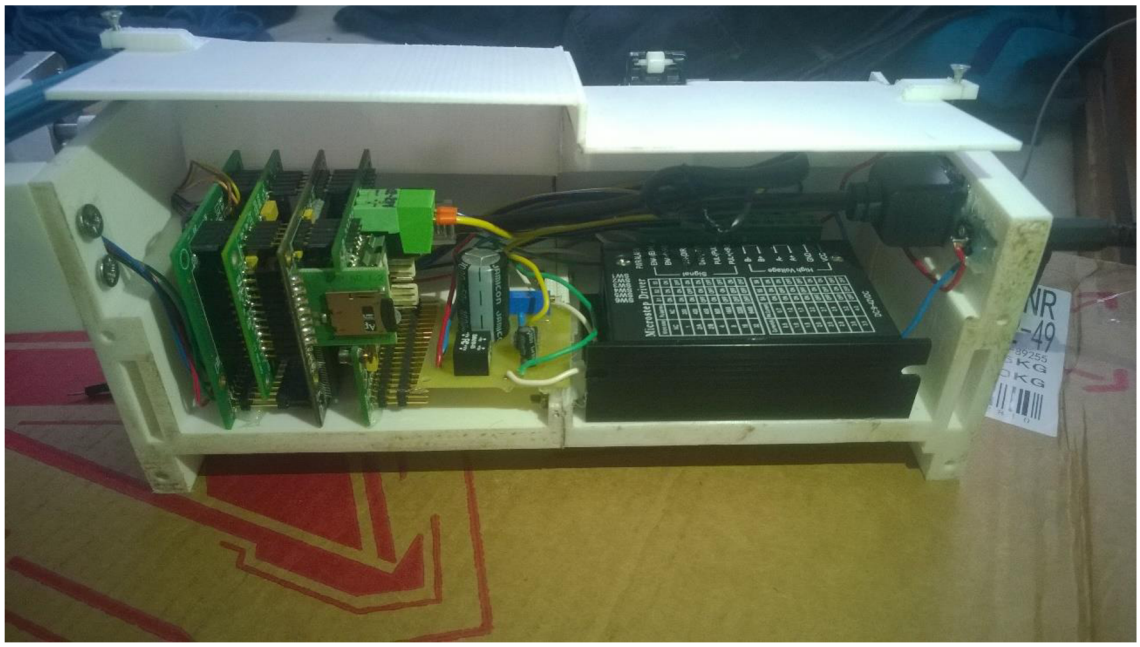
Celý skelet konstrukce jsem navrhoval v programu Tinkercad od společnosti Autodesk. Nejedná se o žádný profesionální program pro tvorbu 3D návrhu, ale je dostačující pro jednoduché konstrukce. Výhodou tohoto programu je jeho jednoduchost při jakémkoliv návrhu designu skeletu. Skelet je vytištěn z bílého PLA materiálu. Důvod výběru materiálu PLA je takový, že na skelet není vyvíjena žádná velká síla, a tudíž není třeba volit materiál s větší adhesí mezi vrstvami. Nespornou výhodou PLA je, že je snadno tisknutelný.
Konstrukci jsem si rozdělil do dvou hlavních částí, a to horní a dolní. Rozměry konstrukce jsou 326x104x110 mm. Výška konstrukce je pak dána výškou obalového materiálu. Zde jsem uvedl nejmenší výšku zařízení. Díky faktu, že zařízení je dlouhé, musel jsem každou část hlavní konstrukce rozdělit v polovině, protože tiskárna, na které byla konstrukce tištěna, nedokáže vytisknout větší objekt než 200 mm. Při tisku větších částí docházelo ke kroucení materiálu, což bylo nutné řešit různými úpravami jako je kytování, vrtání, broušení apod.
V horní části konstrukce se nachází CIS senzor. Tato část je pohyblivá a díky tomu lze do skeneru vložit obal s volitelnou výškou. Jsou zde 2 přítlačné válce, které jsem použil z vyřazené tiskárny a ty tlačí na válce nacházející se na spodní části

konstrukce, čímž vytvářejí styčnou plochu pro obalový materiál. Na bocích konstrukce se nachází tažné pružiny. Tyto pružiny mají za úkol zvýšit tlak mezi válci a tím zajistit, aby nedocházelo k prokluzu obalového materiálu.



Obrázek 29: Celkový pohled na skener

Ve spodní části je umístěn veškerý hardware pro pohon, a to včetně samotného pohonu a dále ovládací deska pro CIS. Vše je uvnitř zařízení. Zvenčí je vidět pouze motor. Toto lze vidět na obrázku č.29 a č.30. Vrchní část je potažena černou matnou folií, a to z důvodu, aby díry vytvářely vysoký kontrast.



Obrázek 30: Pohled na umístění součástek

Z levé strany se nachází motor, jehož hřídel je spojena spojkou s hlavním pohonným válcem. Lze také vidět, že jsem implementoval i pohánění sekundárního válce. Tento pohon jsem dělal z důvodu lepšího zakládání papíru do skeneru. Při testování stačilo mít hnaný pouze hlavní válec, který má dostatečnou styčnou plochu s obalovým materiálem. Jak již bylo řečeno, průměr hlavního hnacího kola je 17,83 mm, ale průměr sekundárního hnacího kola je pouze 12,65 mm. Je zde

tedy nutné udělat převodový poměr v poměru 1:0,71. Nakonec jsem zvolil převodový poměr 1:0,72. Sekundární náhon sice lehce prokluzuje z důvodu, že je pomalejší, ale drží papír napnutý a nestává se, že by se papír kroutil. Na spojce jsem tedy vytiskl kolo s drážkou o průměru 14 mm a v příslušném poměru i kolo s drážkou na sekundární válec. Spojení mezi válci je tvořeno pevnou gumou, která je dostatečně napnutá, aby nedošlo k proklouzávání. Oba hlavní válce jsem také posilikonoval, abych zvýšil eliminaci prokluzu obalového materiálu.
Samotný motor má vlastní stojánek, ke kterému je přichycen šrouby a tento stojánek je pak přišroubován k hlavní dolní části skeneru. Pohled na motor lze vidět na obrázku č.31 vlevo.

Obrázek 31: Pohled z levého (vlevo) a pravého boku (vpravo)

Vpravo lze vidět konektor pro napájení, kam pasuje souosý 5,5 mm konektor. Kladný pól se nachází na středním pinu konektoru. Dále je zde ethernetová zdířka, kde se zapojuje ethernetový kabel pro komunikaci mezi PC a skenerem a 2 tlačítka. Levé tlačítko je pro centrální spuštění celého skeneru a druhé je určené pro spouštění, respektive vypínání pohonu skeneru.
Spodní část konstrukce je otevřená, není zde žádný kryt, a to z důvodu servisních úkonů jako je například přehrání softwaru na SD kartě, která je tak velmi dobře přístupná, nebo naladění frekvence generátoru impulzů.

6 APLIKACE

Aplikace je rozdělena do 2 hlavních částí, kde jedna z částí se zabývá zpracováním obrazu a druhá část se zabývá zobrazováním informací uživateli tedy GUI.

6.1 Aplikace na zpracování obrazu

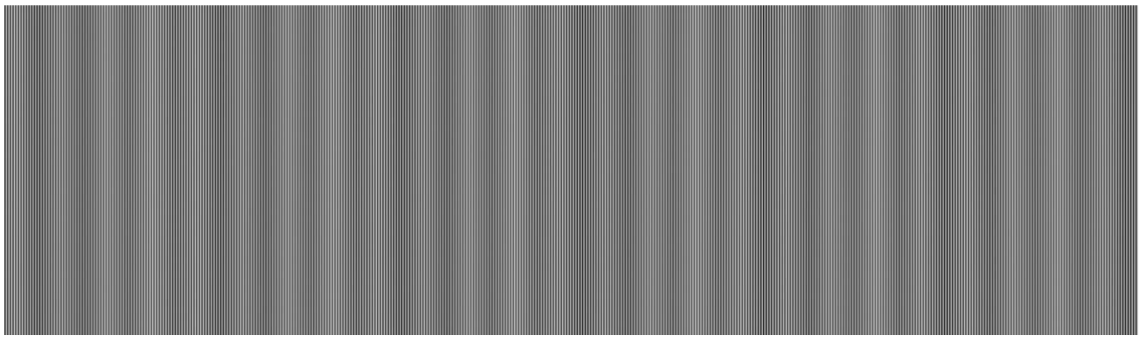
Celá tato část je psána v programovacím jazyce C++, a to ze dvou důvodů. Ovladače k desce, která má na starosti celé skenování a zasílání obrázku, jsou napsány v jazyce C++ a dále knihovny OpenCV mají nativní podporu jazyka C++, čehož jsem se tedy snažil využít.
S využitím těchto ovladačů jsem napsal program, který vyčítá obrázky z bufferu. Vyčítání funguje následujícím způsobem:

1. Vytvoří se instance třídy kamery (CIS sensoru)
2. Nastartuje se kamera
3. Nastaví se IP adresa a port, na kterým zařízení běží
4. Nastaví se timeouty (Hey you perioda, timeout)
5. Nastaví se rozměry snímků, vyplňování chybných sektorů, zahazování špatně přijatých snímků
6. Vytvoří se vyčítací vlákno pro příjem snímků.

Při vytváření instance třídy se zároveň volají parametry snímku, zesílení a offset. Vyčítací vlákno je zacyklené ve whilu. Pokud přijde snímek, spustí se event GetImageFromFifo a vyčte se snímek. Tento snímek je převeden do OpenCV a následně zpracován. Po ukončení zpracování obrazu je snímek z bufferu vymazán.

6.1.1 Zpracování snímku

Přijatý obrázek ze skenovacího zařízení má rozlišení 3480x501 px, tedy senzor naskenuje 501 řádků a ty následně pošle jako jeden obraz. Jak lze vidět na obrázku č.32, tento obraz není absolutně vhodný ke zpracování obrazu. Důvodem je to, že A/D převodník osazený na této desce má minimální rychlost vzorkování 10 MHz. V tomto zařízení se však podařilo dostat tento převodník na rychlost 1 MHz, což byla nejmenší rychlost, na které A/D převodník dokázal stabilně pracovat. Problémem je, že maximální rychlost vyčítání CIS senzoru je 500 kHz. Řešeno je to tak, že každý druhý pixel je černý, což vyřeší moc vysokou vzorkovací rychlost převodníku a umožní používání CIS senzoru při 500 kHz.

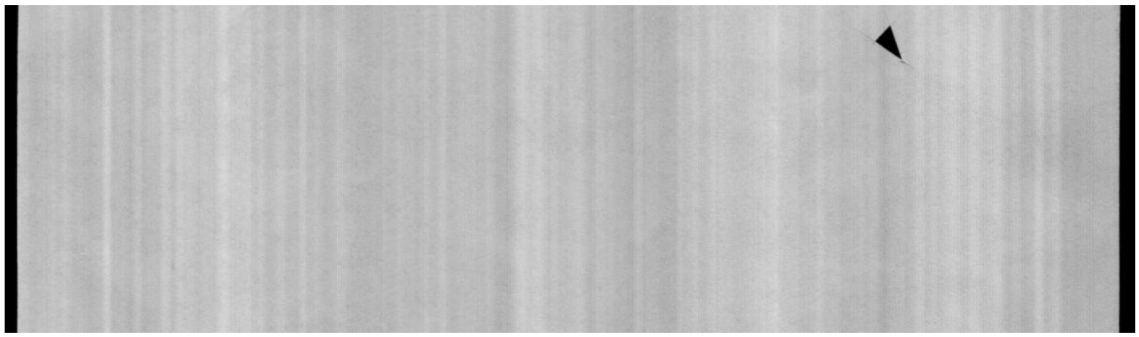


Obrázek 32: Přijatý obrázek (bílý papír)

Velikost obrazu lze tedy zmenšit na rozlišení 1728x501 px, což se rovná téměř polovině šířky obrazu, a vyřadit tedy černé pixely, které jsou do obrazu uměle přidávány. Do přesné poloviny nám schází 12 pixelů. Těchto 12 pixelů nazýváme jako optical black pixels. Tyto pixely jsou černé a slouží k nastavení citlivosti senzoru, respektive nastavení úrovně napětí pro černou barvu, tedy použijí se jako referenční.
Po odstranění těchto černých pixelů dostáváme obraz, který lze použít pro další zpracování obrazu. Je velmi důležité správné osvětlení. Pokud by byl obraz příliš přesvětlený, nebo naopak nedostatečně osvícený, tak se začne ztrácet informace o defektu. Toto je zdokumentováno na obrázku č.33, kde lze vidět, že obrázek dole je příliš světlý, tedy nelze na něm zahlédnout řezy, které náleží díře a obrázek nahoře má nedostatečné osvětlení, tedy není na něm skoro nic vidět.

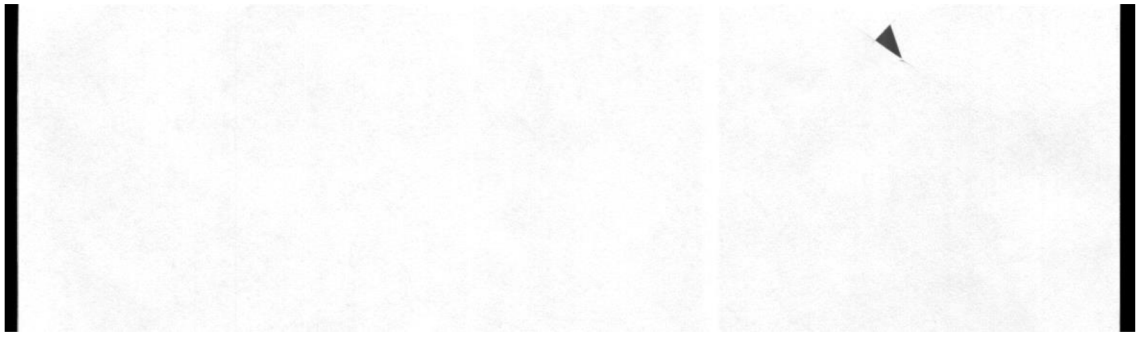
Obrázek 33: Ztrácející se informace (nahoře nedostatečně osvětlené, dole presvětlené)

Na obrázku č.34 lze vidět správně nastavené osvětlení. Při správném osvětlení ovšem vznikají „čáry“ přes obrázek, které jsou pokaždé ve stejném místě. Toto je vlastnost CIS senzoru, kde každý pixel je jinak citlivý na světlo. Tuto nepříjemnou vlastnost lze však softwarově kompenzovat.



Obrázek 34: Správně osvětlený obal

Pokud si vytvoříme jeden řádek bílého podkladu, případně celý snímek bílého obrazu a následně se provede průměr každého sloupce obrazu, dostaneme křivku, respektive pole s citlivostí každého jednoho senzoru na bílou barvu. Pokud tuto křivku zinvertujeme a přičteme ke každému řádku původního obrazu, vyjde nám homogenní barva obrazu, tedy zcela bílý obraz. Bohužel touto kompenzací vzniknou dané pruhy na černé barvě. Jelikož primární podklad, respektive skenovaný materiál bude světlejších barev, převážně bílé, tuto kompenzaci využiji. Na okrajích obrazu vidíme černé pruhy. Tyto pruhy jsou dány z důvodu délky senzoru, jehož šířka je větší, jak šířka papíru DIN A4, tedy senzor zde skenuje pouze podklad. Jelikož počítaná kompenzace zasahuje i na tyto pruhy a skenovaný obraz by nevypadal dobře, pro tyto pruhy jsem kompenzaci vypnul. Obraz tedy působí esteticky lépe a vypnutá kompenzace na těchto pruzích neovlivní následující zpracování obrazu. Nepříjemným faktem je, že se tato kompenzace projeví na detekovaných dírách, což nelze nijak ovlivnit, ale následující zpracování obrazu to také nijak neovlivní, protože nelineárnost jednotlivých senzorů není až tak velká, takže defekt není rozdělen na více malých defektů.



Obrázek 35: Kompenzace obrazu

Po kompenzaci je obraz připravený na detekování děr. Jako nejvhodnější se jeví použití Cannyho hranového detektoru, který by měl na homogenně bílém papíru najít hrany bez větších problémů. Jak lze vidět na obrázku č.36, všechny hrany byly bez problémů nalezeny. Můžeme zde zpozorovat i „šum“, který lze později bez problémů odfiltrovat. Daleko větším problémem zůstávají hrany, které nejsou spojité. Při hledání defektů by tyto hrany byly označeny jako více malých defektů, což by vedlo například ke špatnému rozpoznání okrajů papíru, nebo špatnému rozpoznání defektu.

Obrázek 36: Cannyho hranový detektor

Tento jev lze odstranit například aplikací morfologického uzavření, který vyplní malé díry, což v tomto případě nevadí, zjednoduší tvar objektu, tedy vyrovná různé malé výstupky a v neposlední řadě také spojí ohraničující čáry, které nebyly po aplikaci Cannyho hranového detektoru celistvé.

Obrázek 37: Morfologické uzavření

Po všech těchto operacích lze použít OpenCV funkci findContours. Tuto funkci lze jednoduše vysvětlit jako křivku kolem obvodu objektu, který má stejnou barevnou intenzitu. Díky tomu, že po užití Cannyho hranového detektoru funkce pracuje pouze se dvěma barvami, je tato funkce velmi přesná. Tato funkce dokáže vrátit pozici každého defektu, a to včetně jeho obvodu, čehož se dá využít. Jak lze vidět na obrázku, funkce detekovala dva okraje, šum a díru.
Nyní je potřeba zbavit se okrajů, aby zůstal pouze obraz defektů, které se nacházejí v oblasti papíru. Pro lepší detekci použiji semínkové vyplnění, kde počáteční semínko se bude nacházet v levém horním rohu a druhé v pravém horním rohu. Toto zaplnění kompletně vyplní levý a pravý okraj.

Obrázek 38: Semínkové vyplnění

Na nalezené kontury lze aplikovat funkci nalezení maxim v osách x a y. Pokud se tedy u každé kontury nalezne její levé maximum, respektive její nejmenší bod v ose x, a tento bod se bude rovnat okraji, tedy 1, tak lze jednoduše zjistit, že se jedná o levý okraj. S pravým okrajem je to velmi podobné. Nalezne se pravé maximum, tedy největší bod v ose x, a ten se rovná velikosti obrázku, tedy 1728, jedná se o pravý okraj.
Tuto funkci by samozřejmě bylo možné použít i bez semínkového zaplavení, ale mezi obalovým materiálem, tedy papírem, je malá mezera. Pokud by se nepoužilo semínkové zaplavení a detekce okraje by se vyhodnocovala na základě nejmenšího, respektive největšího bodu na ose y, a mezi okraje by se dostala nečistota, způsobilo by to nalezení okraje, který by byl nečistotou a opravdový okraj by zde zůstal.
Odstraněním okrajů se rozumí překreslení všech kontur, které nebyly detekovány jako okraj nebo šum, na nový černý obraz, protože z obrazu nelze vyjmout objekt, aniž by nedošlo k přetvoření obrazu. Pro toto překreslení lze použít funkci drawContours, která překresluje objekty a zároveň je vyplňuje, čehož lze později využít k různým statistickým účelům jako je například obsah, obvod, velikost v ose x, velikost v ose y apod.

6.2 Detekce složitějších případů defektu

Popsaná detekce funguje pouze na ideální detekci defektu, kde se defekt nachází pouze na jednom z obrazů, nenachází se na okraji papíru a není větší jak 501 px tedy při použitém senzoru, který má 200 DPI v obou osách větší jak 62,625 mm.

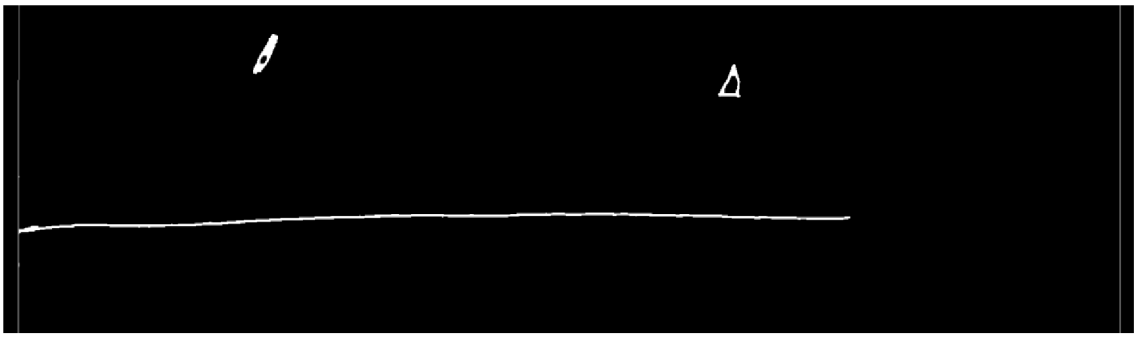
Defekty lze rozdělit do 4 kategorií:

1. Defekt nacházející se na samotném snímku nezasahující do žádného z okrajů.
2. Defekt zasahující do levého, pravého nebo do obou okrajů snímku.
3. Defekt zasahující do horní a zároveň spodní hrany snímku.
4. Defekt nacházející se na snímku, zasahující do vrchní nebo spodní hrany snímku.

První typ defektu byl již popsán, tedy jedná se o nejjednodušší typ, kde není třeba žádných speciálních úprav celkového obrazu, pouze se na něj aplikují různé základní algoritmy zpracování obrazu.

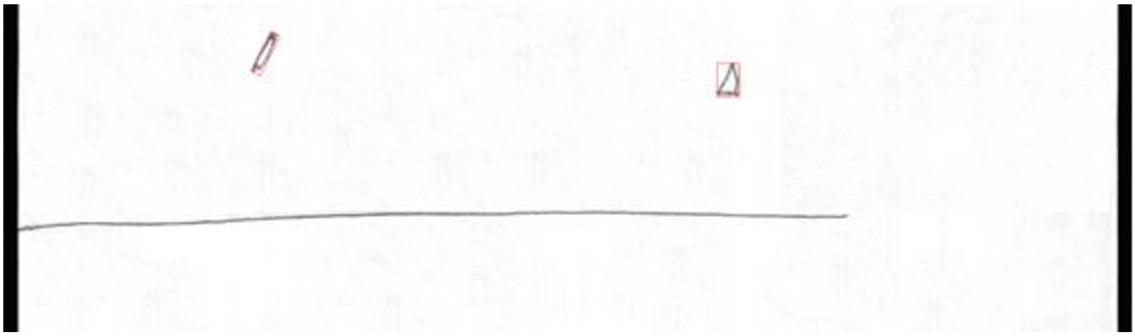
6.2.1 Spojení defektu s levým, pravým nebo oběma okraji

Jednou z problémových detekcí je detekce defektu, kdy je defekt spojen s okrajem, ať už se jedná o levý nebo pravý okraj. Jak bylo již řečeno, okraje jsou odstraňovány pomocí kontur. Jak lze vidět na obrázku č.39, defekt se nachází v celé šíři snímku a je spojen s jedním okrajem, což zapříčiní, že daný defekt není detekován, protože tato kontura byla odstraněna s jedním z okrajů.



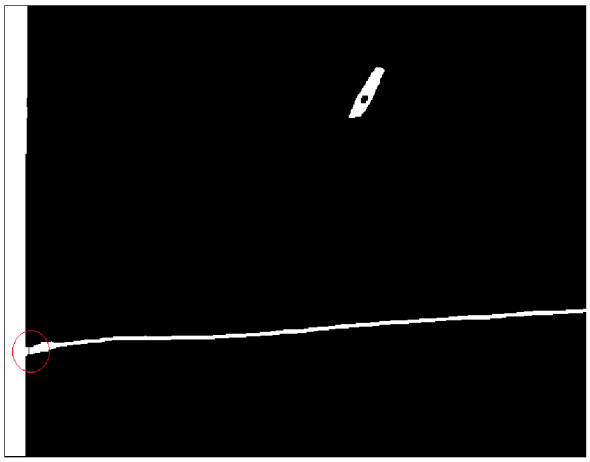
Obrázek 39: Spojení defektu s levým krajem

Na obrázku č.40 lze vidět výslednou detekci obrazu označující všechny defekty. Jak je patrno, defekt, který byl spojen s okrajem, není označen.



Obrázek 40: Snímek zachycující detekci defektů po odstranění okrajů

Tomuto odmazání lze zabránit více způsoby. Jedním z nich je umělé přidání pixelového pruhu. Okraje papíru jsou vždy na téměř stejném místě v obrázku. Po použití Cannyho hranového detektoru a následném semínkovém zaplavení lze přidat hned za tuto hranu do prostoru obrázku sloupec černého pruhu. Pokud se tento pruh přidá na obě strany snímku, oddělí se okraje od defektu, který je spojen ať už s levým, pravým, nebo dokonce oběma okraji. Tuto skutečnost lze vidět na obrázku č.41. Aby nedošlo ke změně velikosti obrazu, a tedy i změně souřadnic daných defektů umělým přidáním oddělovacích sloupců, lze smazat sloupce z obou okrajů, tedy sloupec 1 a sloupec 1728, a to proto, že na výše zmíněných okrajích se nic nenachází. Tento způsob jsem si tedy vybral.
Druhým způsobem by bylo prosté oříznutí okrajů a zmenšení obrázku pouze na formát papíru. Z důvodu dalších zpracování obrazu v OpenCV je pro mě dobré ponechávat původní velikost, ať už je to z důvodu, že není třeba přepočítávat souřadnice defektu do původního obrazu s okraji, nebo i z důvodu opakovaného použití stejného objektu bez nutnosti měnit velikost tohoto objektu.



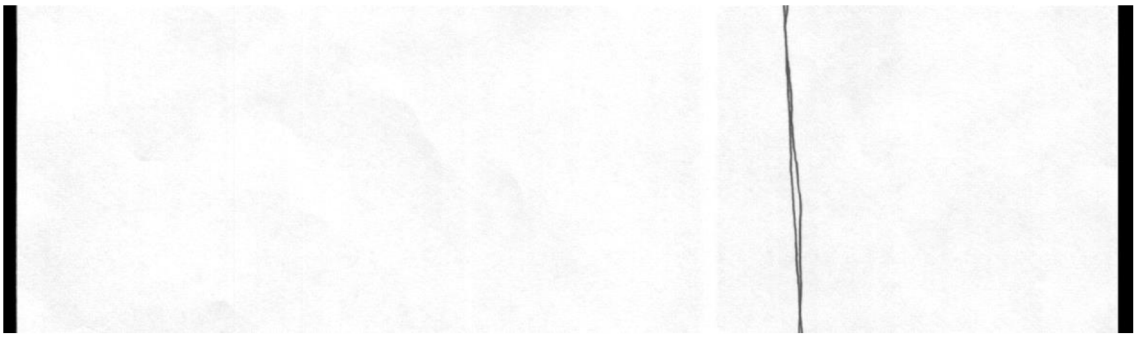
Obrázek 41: Oddělení defektu od okrajů (výřez ze snímku zachycující 1px oddělení od okraje, zakroužkováno červeně)

6.2.2 Defekty nacházející se mezi snímky

Zřejmě nejobtížnějším problémem jsou defekty, které se nacházejí mezi dvěma po sobě jdoucími snímky, tedy jedna z částí se nachází na prvním snímku a druhá část na druhém snímku, nebo dokonce defekt tak velký, že přesahuje do více snímků. Za normálních okolností se tyto snímky spojí do jednoho a vyhodnotí se oba snímky. U kontinuálního obrazu však tuto skutečnost nelze aplikovat, protože pokud by se defekty nacházející se na přechodech snímku vyskytovaly periodicky, šlo by o zacyklený problém. Tento problém by v krajním případě mohl způsobit i kolaps programu z důvodu nedostatku RAM, ale spíše by způsobil to, že do doby, než by tento opakující se jev skončil, nevyhodnotil by žádný z defektů.

Defekt zasahující do vrchního a zároveň spodního okraje snímku

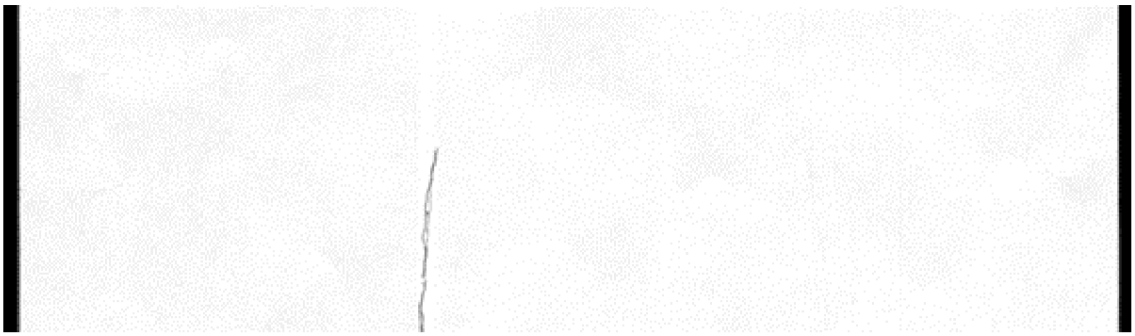
Třetím typem defektu je defekt zasahující do vrchního a zároveň spodního okraje snímku. Tento defekt lze řešit více způsoby.
Prvním již zmíněným způsobem je použití bufferování obrazu do doby skončení jevu. Aby se zabránilo přetečení paměti programu, muselo by se udělat opatření proti přetečení, například stanovením maximální délky bufferu. Toto řešení se nejeví jako vhodné, a to z důvodu velmi zpožděného detekování malých defektů nacházejících se na snímku spolu s tímto velkým defektem. Pokud by byl překročen buffer, musel by snímek projít detekcí, i když by nebylo dosaženo konce tohoto velkého defektu, tedy došlo by k rozdělení celého defektu na více malých částí. V zásadě by se nejednalo o problém, ale popíralo by to koncept tohoto využití.
Druhým způsobem je tento defekt detekovat jako jeden defekt na daném snímku, tedy celý defekt, který je delší než jeden snímek, by byl detekován dílčím způsobem. Na fakt, že tento defekt je delší a pokračuje na následujících snímcích nebo v přechozích snímcích, lze poté upozornit obsluhu.
V mém programu je použito druhé řešení, tedy detekování po dílčích částech, a obsluha je upozorněna na tento fakt pomocí grafického rozhraní.



Obrázek 42: Defekt delší než jeden snímek

Defekt nacházející se na snímku zasahující do horního nebo spodního okraje

Asi nejsložitějším typem defektu je defekt, který se nachází na jedné z hran snímku, tedy jedna jeho část se nachází na prvním snímku a druhá část na druhém snímku. Pokud by se jednalo o nekontinuální obraz, detekce by byla poměrně snadná, a to spojením dvou snímků dohromady a následnou detekcí defektů. Jelikož se zde jedná o kontinuální obraz, situace je složitější.
Tento úkon musí být detekován po částech, aby došlo k zamezení dvojitého detekování. Pokud by se jeden snímek použil dvakrát, tedy pro detekci defektu, který se nachází na spodním kraji snímku a následně pro detekci defektu nacházejícího se na horní hraně snímku, dostali bychom pro defekty, které nejsou na hranách, dvojitou detekci. Zároveň by algoritmus detekoval i defekty nacházející se na opačné hraně, než chceme detekovat.



Obrázek 43: Přechodový defekt

Vymyslel jsem způsob, jak tento efekt zcela eliminovat. Ještě před objasněním problematiky, jak výše zmíněný nepříjemný efekt eliminovat, je třeba rozdělit defekty do dvou kategorií:

1. Jeden přechodový defekt na hraně snímku
2. Více defektů na hraně snímku

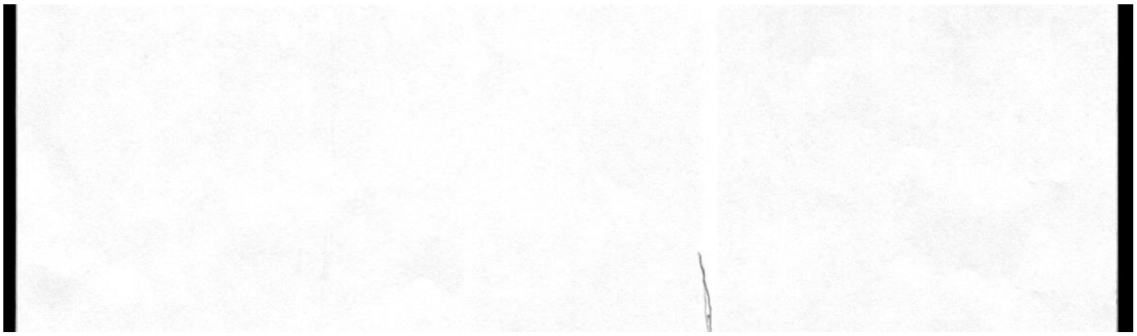
Každou kategorii lze dle umístění defektu rozčlenit na další tři druhy:

1. Defekt nacházející se na hraně prvního snímku nezasahující do druhého snímku.
2. Defekt nacházející se na hraně druhého snímku nezasahující do prvního snímku.

3. Defekt nacházející se na prvním a zároveň druhém snímku.

Jeden přechodový defekt na hraně snímku

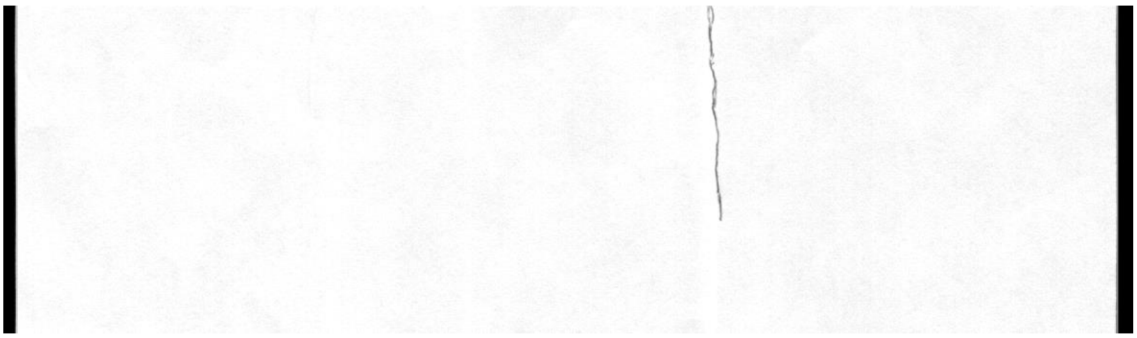
V případě jednoho nalezeného defektu je situace celkem jednoduchá. Pokud algoritmus označí defekt jako možný přechodový, zjistí jeho nejvyšší hodnotu v ose y. Po zjištění nejvyšší hodnoty algoritmus vezme snímek a k němu přičte, respektive odečte, toleranci a vyřízne z daného snímku část, kde se nachází daný defekt. Celý tento jev se děje na spodní hraně snímku.



Obrázek 44: Snímek s jedním přechodovým defektem

Obrázek 45: Spodní výřez z prvního snímku

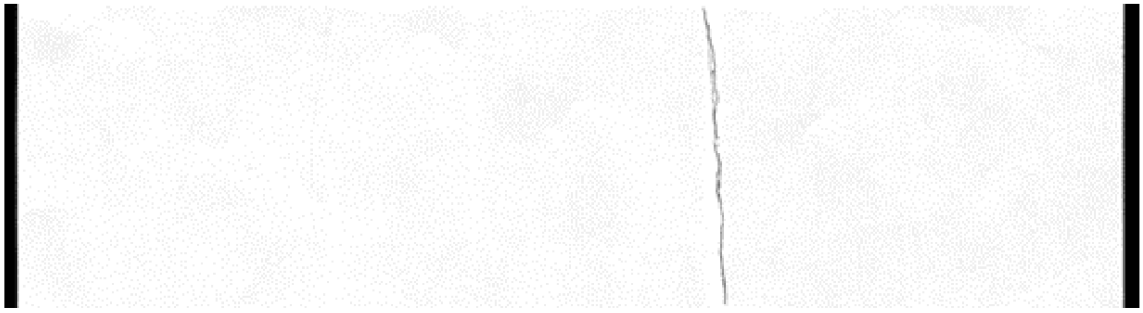
Po vyříznutí se čeká na následující snímek. Po přijetí se rozhoduje o tom, o který druh defektu se jedná. Pokud na se na horní straně nenachází žádný z defektů, je tento defekt označen jako defekt nacházející se pouze na spodní straně snímku. Pokud je však nalezen nějaký z defektů i na horní hraně snímku, je obdobný proces opakován pro horní část. Pro defekt, který je označen jako přechodový, je nalezeno jeho minimum a přičtena tolerance. Následně je vyříznut tento pruh a je spojen s pruhem, který se nacházel na přechozím snímku.



Obrázek 46: Následující snímek

Obrázek 47: Horní výřez z druhého snímku

Pokud se jedná o dva samostatné defekty, které se nacházejí na hranách snímků, jsou označeny dva samostatné defekty. Pokud se jedná o jeden defekt, který se nacházel na obou hranách snímku, je tento defekt označen jako jeden.



Obrázek 48: Spojení obou výřezů v případě jednoho přechodového defektu

Pokud není detekován defekt na spodní hraně prvního snímku, neuloží se žádný výřez. Jestliže je defekt detekován na vrchní hraně snímku a z přechozího snímku není žádná detekce, tedy žádný výřez, je detekován pouze výřez z horní strany snímku a defekt je detekován jako defekt nacházející se na hraně snímku.

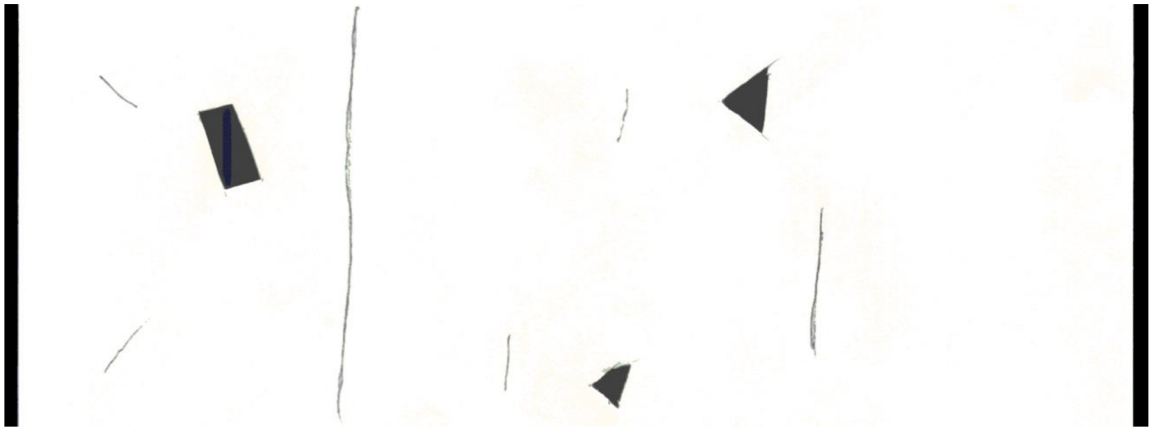
Více defektů na hraně snímku

Pokud se na snímku nachází více defektů, je problém o něco složitější, ale algoritmus vyhodnocování je velmi podobný. Pokud se nachází více defektů na hraně, je nalezeno jejich globální maximum, tedy defekt, který je největší a k němu je přičtena tolerance, poté se snímek ořízne. Tento jev je vidět na obrázku č.49. Na snímku jsem také naznačil modrou čarou řez, který ukazuje, kde bude řez veden pro získání jedné z částí snímku pro detekování přechodových defektů.

Obrázek 49: Snímek s více přechodovými defekty

Již na tomto oříznutém snímku lze vidět 3 druhy defektů. Přechodový defekt, který je žádaný pro detekci, defekt, který se nachází uprostřed a nedotýká se žádné

z hran a defekt, který se nachází na horním ořezu snímku. Je třeba mít na paměti, že druhé dva případy již byly detekovány, a proto je nutné je do detekce již nezahrnovat.
V případě, že se bude jednat o snímek, který nebude mít pokračování na dalším snímku, tedy druhý snímek nebude mít na horním okraji žádný defekt, lze vyhodnotit všechny defekty, které jsou spojené s dolní hranou jako defekty nacházející se na hraně.
V případě, kdy snímek bude mít jenom vyříznutou část z druhého snímku, tedy defekty se budou nacházet na horní hraně druhého snímku a v prvním snímku nebude detekován žádný defekt, je situace obdobná. Všechny defekty, které nejsou spojeny s horní hranou, již byly detekovány a defekt spojný s hranou je detekován jako defekt nacházející se na hraně.
Poslední a zároveň nejsložitější případ nastává, kdy jsou dva výřezy spojeny k sobě.



Obrázek 50: Spojení obou výřezu v případě více přechodových defektů

U tohoto typu je známo, že všechny defekty, které se nachází na hranách, jsou již detekovány, a tudíž je lze odstranit. Problém nastává u ostatních, u kterých nelze jednoznačně určit, o jaký typ defektu se jedná.
Díky tomu, že jsou známy rozměry obou výřezů snímku, lze jednoznačně určit, kde je obraz spojen. Defekt, který se nachází na přechodu těchto snímků, (tedy jeho horní bod v ose y je větší než spoj a jeho dolní bod v ose y se nachází pod touto hranou), lze označit jako přechodový.
Pro detekci defektu, který se nachází na hraně, tedy jeho horní i dolní bod v ose y se nachází nad nebo pod čárou, se musí použít jiný způsob detekce.
Pro detekci takového objektu používám fiktivní zvětšení, to znamená, že objekt fiktivně natáhnu o několik pixelů.
Jelikož není rozeznáváno, jestli se objekt nachází na spodním nebo horním snímku, což není důležité, natahuji objekt na obě strany. To zapříčiní, že pokud se defekt nachází na spodní nebo horní hraně, přesáhne spoj a tím je detekováno, že objekt se nachází na hraně.
Všechny ostatní defekty, které ještě nebyly označeny, tedy neprošly žádnou z těchto podmínek, jsou označeny jako již detekované a dále se s nimi nepracuje.

6.3 Ukládání obrazu a zaznamenávání defektu

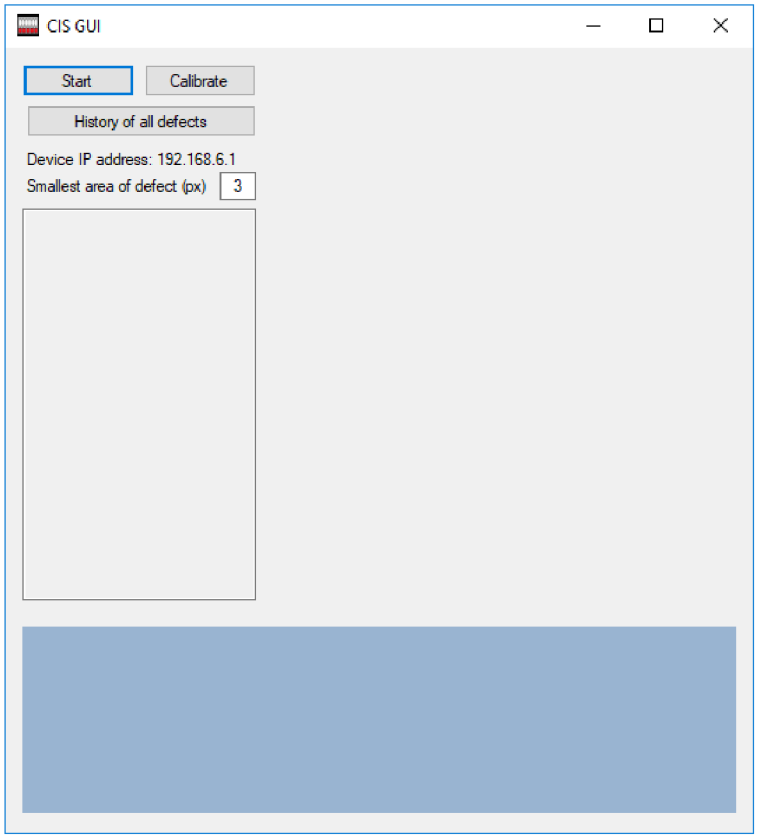
Od startu zaznamenávání obrazu je každý přijatý snímek zpracován pomocí popsaných algoritmů. Pokud se na daném snímku nenachází žádný nebo přechodový defekt, snímek není ukládán z důvodu nezajímavosti snímku.
Pokud se na snímku nachází defekt, který není přechodový, tedy i defekt, který se nachází přes celý snímek, je na takovýto snímek zakresleno barevné ohraničení a snímek je uložen. Je třeba zdůraznit, že na tomto snímku nejsou ohraničeny přechodové defekty. Přechodové defekty jsou ukládány zvlášť, kde je ukládán jejich výřez a do tohoto výřezu je zaznačeno jejich barevné ohraničení. Všechny snímky, na kterých byl detekován defekt, se nachází ve speciální složce aplikace spolu s CSV souborem, ve kterém jsou uloženy veškeré záznamy o všech nalezených defektech, tedy informace o poloze rohů ohraničení, jejich středu, číslo snímku, na kterém byl defekt nalezen apod.
Dále je ukládána historie posledních deseti po sobě jdoucích snímků, kde jsou zaznačeny všechny druhy defektů. U této historie nejsou ukládány informace ohledně defektů. V tomto případě jsou přímo posílány do GUI. Skládání a zaznamenávání defektů do této historie se děje také v aplikaci zpracovávající obraz. Všechny snímky, jak již bylo řečeno, nejsou ukládány, tedy aplikace si drží v paměti posledních deset snímků a ty jsou postupně přepisovány.

6.4 Grafické rozhraní aplikace

Grafické rozhraní obsahuje dvě grafická okna. První, tzv. základní okno, kde se nachází veškerý přehled komunikace mezi aplikacemi a rozšiřující okno, kde se lze lépe podívat na defekty a zjistit o nich informace. V následujících kapitolách popisuji funkce jednotlivých oken a vysvětluji jejich funkčnost.

6.4.1 Základní okno

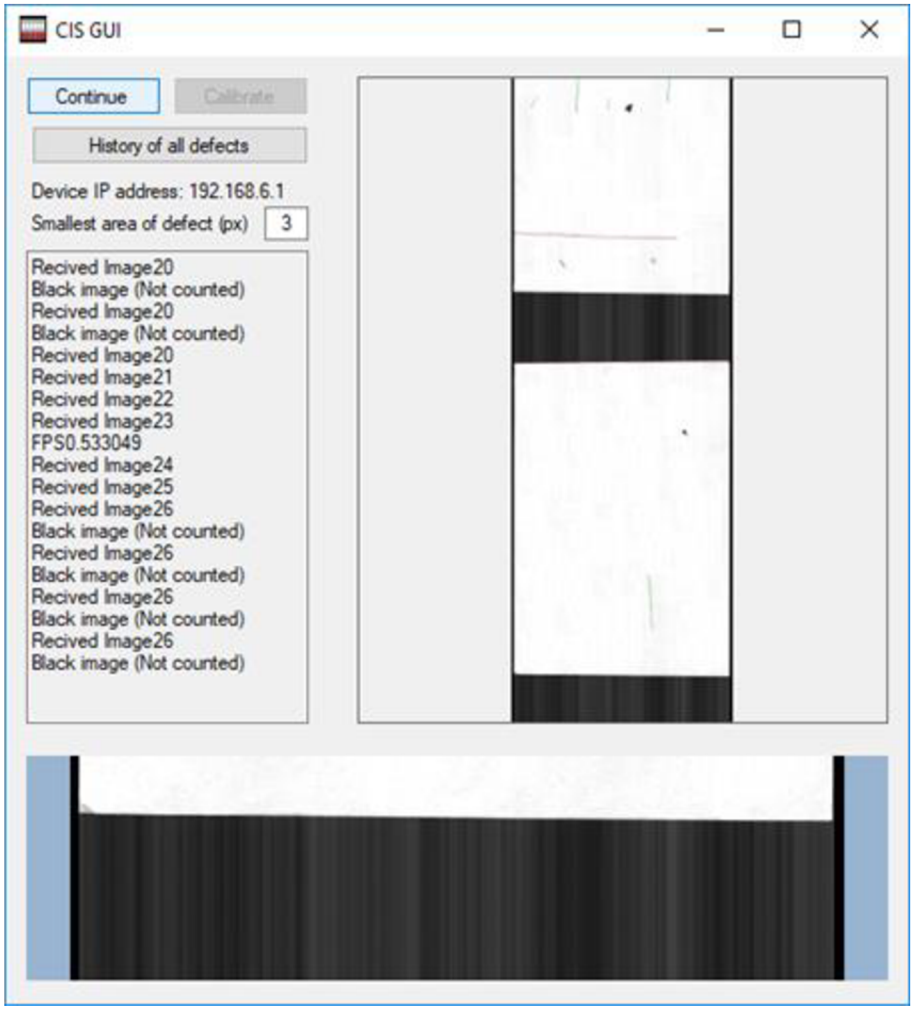
Po startu aplikace se zobrazí základní okno aplikace, které je při spuštění nevyplněné.



Obrázek 51: Základní okno aplikace

Na levé straně lze vidět tři tlačítka a log. V logu se zaznamenávají veškeré informace, které jsou posílány z druhé části aplikace, kde je zpracováván obraz a také veškeré chybové hlášky z tohoto okna aplikace. V logu se lze dozvědět například pořadí snímku, FPS, zda se pod senzorem nachází obalový materiál, zda dojde ke kalibraci apod.
Tlačítko s názvem „Start“ funguje jako spouštěcí. Toto tlačítko má 3 stádia podle toho, v jaké situaci se nachází program zpracování obrazu. Pokud neběží, je na tlačítku nápis „Start“, pokud aplikace aktivně běží, tedy je spuštěn záznam a detekce obrazu, je na tlačítku nápis „Pause“. Tlačítkem „Pause“ je možné program pozastavit, tedy program zpracování obrazu stále běží, ale nedochází ke zpracování obrazu. Po stisknutí tlačítka „Pause“ se nápis tlačítka změní na „Continue“. Tímto tlačítkem lze program opětovně spustit.
Dále se zde nachází tlačítko „Calibrate“. Toto tlačítko umožní pořídit kalibrační snímek, který je v programu zpracování obrazu použit jako referenční k odstranění problému s nelineárností jednotlivých senzorů. Přesněji první snímek, který je pořízen, je uložen jako referenční a následně program běží jako při standardním spuštění programu. Důležité je zmínit, že tuto funkci je možno spustit pouze před startem aplikace a nelze tedy snímek pořídit za běhu programu pro zpracování obrazu.
Dále se zde nachází tlačítko pro zobrazení veškerých nalezených defektů. Funkci tohoto tlačítka blíže vysvětlím v následující kapitole.

Uživatel si také může nastavit filtrování defektů. Lze zde nastavit nejmenší defekt, který má být detekován. Jako výchozí hodnotu jsem zde nastavil 3 px, tedy aplikace zahazuje všechny defekty, které mají obsah 3 px a méně.
Ve spodní části tohoto okna se nachází modře podbarvená oblast. V této oblasti se zobrazují aktuálně přijaté snímky. V těchto snímcích jsou již zaznačené detekované defekty, které lze detekovat na jednom snímku. Vpravo nahoře se nachází historie posledních deseti snímků. Pokud není dostupných deset snímků, je zobrazeno takové množství, které je dostupné. Zpravidla při stratu aplikace nemůže být zobrazeno posledních deset snímků, jelikož neexistují. V takovém případě je zobrazeno dostupné množství čítající i jeden snímek.
Okno je plně responzivní, čímž se rozumí, že uživatel si může okno zvětšovat, respektive zmenšovat a prvky uvnitř okna se zvětšují, respektive zmenšují dle potřeb. Při spuštění aplikace je okno nastaveno na velikost 600x550 pixelů, což je dle mého názoru minimální velikost pro dobrou viditelnost veškerých snímků.
Aplikace se zavírá křížkem, který způsobí ukončení aplikace pro zpracování programu a smaže veškerou historii snímků, na kterých byl nalezen nějaký defekt včetně souboru, kde jsou uloženy informace o defektech tak, aby aplikace byla připravena k dalšímu spuštění.



Obrázek 52: Snímek běžící aplikace

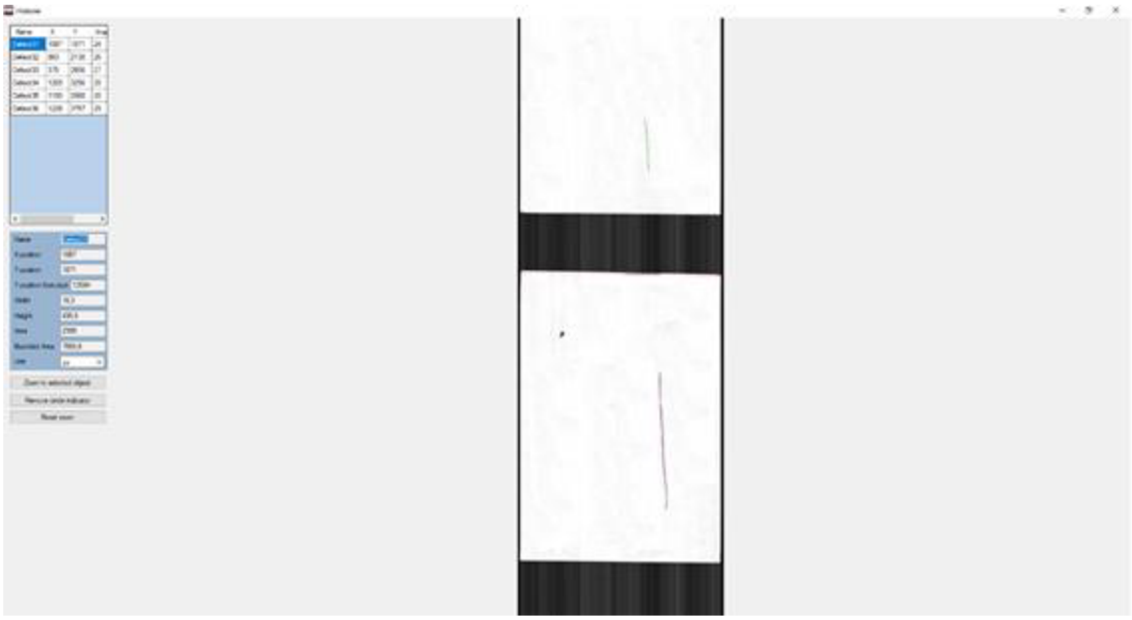
6.4.2 Grafické rozhraní prohlížení snímků

Ve druhém okně lze zjistit informace o defektech a přiblížit nebo oddálit na jakékoliv místo na snímku. Do tohoto okna se lze dostat dvojím způsobem a každý ze způsobů zobrazuje jiné informace o defektech, respektive jiným způsobem.

Způsob zobrazení lze klasifikovat na dvě kategorie:

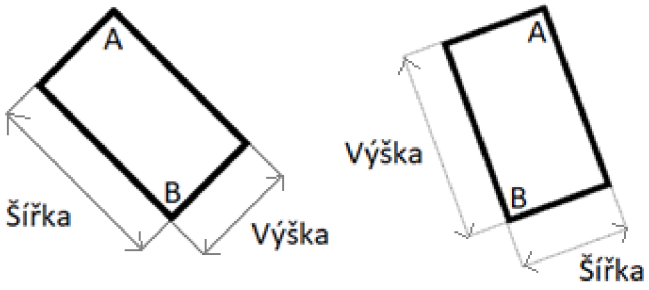
1. Historie všech defektů
2. Historie posledních snímků

Jak již bylo zmíněno, toto rozšiřující okno je pro obě kategorie stejné. Nejdříve popíšu, co je pro obě okna totožné a dále upřesním, v čem se tato okna liší, tedy co je pro každé okno specifické.



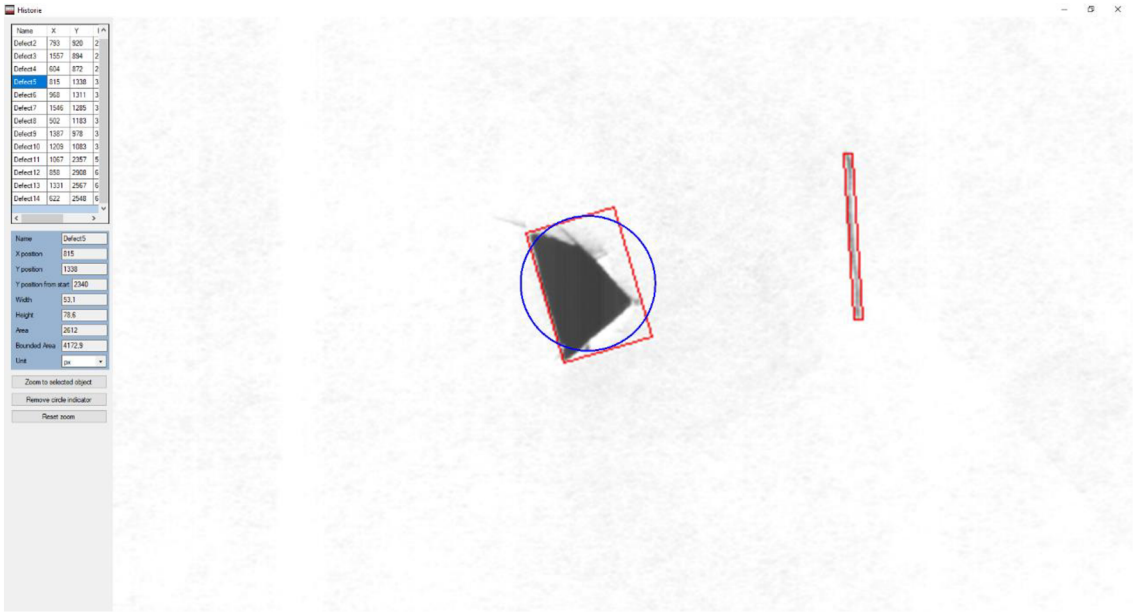
Obrázek 53: Rozšiřující okno při spuštění historie posledních snímků

Na levé straně se nachází okno, kde jsou informace o nalezených defektech, jejich název, který je ve formátu „Defect“ + číslo. Toto číslo je číslo nalezeného defektu, tedy je zaznamenáván celkový počet defektů. Dále pak střed defektu v ose x a y na právě zobrazovaném snímku a číslo snímku, na kterém byl defekt nalezen.
Při kliknutí na jakýkoli řádek nebo buňku na řádku se v přehledové tabulce pod touto tabulkou zobrazí informace o defektu. Jsou zde informace jako je název, poloha defektu v ose x a y na zobrazeném snímku, poloha defektu v ose y od startu zařízení k lepšímu dohledání defektu na obalu. Dále šířka a výška ohraničení defektu. Tyto hodnoty jsou dopočítávány na základě natočení ohraničení, kde záleží na vzájemné pozici rohu A a rohu B. Pro lepší pochopení je na obrázku č.54 naznačeno, jakým způsobem je označena výška a šířka defektu.



Obrázek 54: Výpočet šířky a výšky

Dále se zde nachází plocha, což je reálná plocha objektu a dále ohraničená plocha, což je plocha, která je vypočítávána z šířky a výšky ohraničení. Přidal jsem zde možnost přepínání mezi pixely a milimetry. Defaultně jsou nastaveny pixely z toho důvodu, že veškeré výpočty probíhají v pixelech, tedy jsou přesnější vzhledem k nezaokrouhlovaní čísel. Přesto se domnívám, že pro přesnější představu o reálné velikosti defektu je vhodnější použití zobrazování v milimetrech.
Pod touto tabulkou se nachází tři tlačítka. První z nich s názvem „Zoom to selected object“ způsobí přiblížení a následné zakroužkování daného defektu. Tuto funkci lze také vyvolat dvojklikem v tabulce všech defektů.
Druhé tlačítko s názvem „Remove circle indicator“ způsobí odstranění modrého kroužku kolem objektu.
Posledním tlačítkem v okně je tlačítko s názvem „Reset zoom“. Toto tlačítko oddálí zoom na výchozí hodnotu.
Na celém snímku lze libovolně zoomovat na jakýkoli objekt. Při použití kolečka myši lze objekt přibližovat, respektive oddalovat. Kurzor uživatelské myši představuje střed zoomování, tedy tam kde je ukazatel myši, tam se bude při otáčení kolečkem přibližovat, respektive oddalovat. Pokud to situace dovoluje, tedy obrázek v dané ose je menší než zobrazovací plocha, je centrován na střed, a to z estetických důvodů. Na obrázku č.55 lze vidět přiblížení jednoho z nalezených defektů.



Obrázek 55: Přiblížení na defekt

Jak již bylo řečeno, defekty jsou rozděleny do čtyř kategorií podle toho, kde se nachází. První a druhou kategorií je defekt, který se nedotýká žádných z hran a defekt, který se dotýká pravého nebo levého kraje. Takový defekt je v aplikaci zaznačen červenou barvou. Třetí kategorií je defekt, který je větší než jeden snímek, tedy dotýká se jak honí části, tak spodní části snímku. Tento defekt je zaznačen růžovou barvou. Poslední kategorií je přechodový defekt a ten je zaznačen zelenou barvou. Všechny tyto kategorie defektů lze vidět na obrázku č.52.

Historie posledních snímků

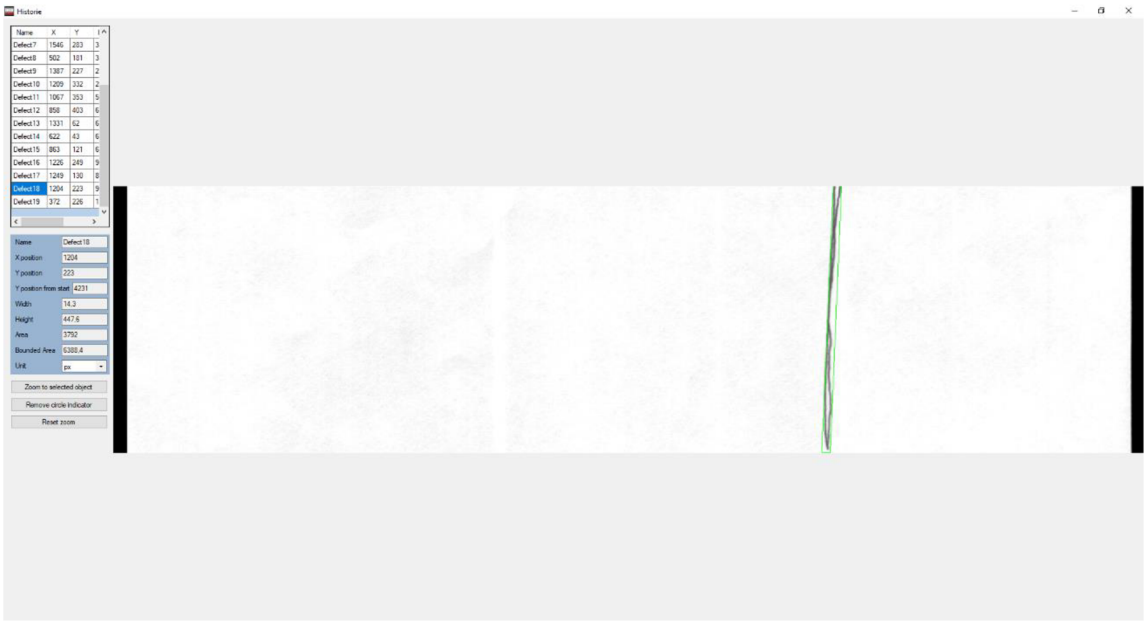
Do tohoto nastavení se lze dostat kliknutím na vyznačenou plochu v základním okně, kde se nachází přehled posledních snímků ve zmenšené podobě. Při najetí myší do tohoto okna se změní ukazatel myši na ručičku a po kliknutí tedy lze vyvolat okno s historií posledních snímků. Ukázku okna je možné vidět na obrázku č.52.

Při kliknutí dojde k zafixování posledního snímku na obrazovce, tedy druhé rozšiřující okno již není aktualizováno novými snímky, a to z důvodu uživatelské analýzy. Pro eliminaci posuvu či smazání přiblížené části snímku se jeví fixace jako elegantní řešení. Dále je zde jinak řešeno načítání tabulky defektů. Tuto skutečnost detailněji popisuji v kapitole věnující se spojení a komunikaci těchto dvou programů.

Historie všech defektů

Do tohoto okna se lze dostat tlačítkem na hlavní stránce pod názvem „History of all defects“. Na rozdíl od historie posledních snímků se také v přehledu defektů nachází označení snímku s příponou 0,5 (např. 2,5). Tímto faktem se rozumí, že se na snímku nachází přechodový defekt a tento snímek je složen z výřezu dolního okraje prvního snímku, nebo horního výřezu následujícího snímku, nebo z obou okrajů. Při kliknutí na defekt se také na rozdíl od historie posledních snímků načtou snímky, na kterých se defekt nachází. V případě označení dalšího defektu,

nacházejícího se na snímku, který je zobrazen, nedojde k opětovnému načtení snímku, aby nedocházelo k problikávání. Je důležité zmínit, že ne všechny defekty jsou na snímku označeny. Pokud se jedná o snímek, který neobsahuje příponu 0,5, jsou zde zobrazeny pouze defekty, které byly detekovány na jednom snímku a defekty, které jsou větší než jeden snímek. V případě, že snímek obsahuje příponu 0,5, jsou zde zaznačeny pouze defekty, které obsahují přechodový defekt. Toto opatření bylo realizováno proto, aby byl pokaždé vidět celý defekt, tedy aby každý ohraničený předmět byl, pokud možno, vyobrazen celý. Informace o defektech jsou vyčítány z externího CSV souboru, kde jsou uloženy veškeré informace o všech defektech, jež byly zaznamenány od startu programu.



Obrázek 56: Okno při spuštění historie všech defektů

6.5 Spojení aplikace pro zpracování obrazu a GUI

Důležité je, aby při ovládání aplikace měl uživatel pohodlí, tedy aby nemusel spustit GUI a aplikaci pro zpracování obrazu zvlášť. Před vlastním programováním se nabízelo několik možností, jak tohoto cíle dosáhnout. V následující kapitole popisuji každou z nich, včetně výhod a nevýhod a důvod výběru právě vybrané koncepce. Následuje popis implementované komunikace mezi aplikacemi.

6.5.1 Koncepce spojení

V první řadě je nutné zohlednit výchozí stav a sice, že ovladače napsané pro desku, která má na starosti bufferování a následné odesílaní snímku do počítače jsou psané v jazyce C++.

Jako první řešení se nabízí napsat GUI v jazyce C++. Grafické rozraní jsem chtěl vytvořit jako WinForms z důvodu přehledného tvoření designu. Bohužel Microsoft přestal s příchodem Visual Studia 2012 tuto možnost nativně přidávat a začal ji poskytovat pouze v programovacím jazyce C#.

Druhou možností se jeví využít WinForms v jazyce C# a přepsat ovladače desky do jazyka C#. Touto možností jsem se zabýval a zkoušel jsem ovladače přepsat.

Bohužel přepsání kódu, který čítá přes 6000 řádků, by zabralo nadměrné množství času, a tak jsem od této koncepce upustil.
Poslední variantou je využití kombinace jazyka C# a jazyka C++. Ačkoliv jsem se nikdy touto problematikou nezabýval, zvolil jsem právě tuto možnost. Tato koncepce má množství výhod ale zároveň jednu významnou nevýhodu.
První velkou výhodou je, že lze využít ovladače desky psané v C++ a dále využívat tento jazyk k používání OpenCV, které nemá nativní podporu jazyka C#. Dále pak využití jazyka C# pro naprogramování GUI, kde tento jazyk má množství výhod oproti C++, jako jsou například již předdefinované třídy Image, Bitmap apod. a k nim příslušné metody, nebo The garbage collector, který se stará o „uklízení“ nepotřebných objektů, jež nebyly ručně odstraněny, aby zamezil pádu aplikace.
Jedinou ale zásadní nevýhodou je, že nelze vzít C++ kód a nakopírovat ho do kódu, kde je použit jazyk C#.

6.5.2 Implementace spojení a komunikace

První návrh byl takový, že GUI pouze spustí exe soubor aplikace pro zpracování obrazu. Bohužel by pak bylo velmi složité a velmi neefektivní předávání informací mezi těmito aplikacemi.
Druhou a zároveň zvolenou možností je použití DLL knihoven, které by obsahovaly funkce k ovládání této aplikace a tyto funkce by bylo možné volat z GUI. Aby bylo možné použít funkce z jazyku C++ v C#, je nutné vytvořit wrapper, který zajistí, aby bylo možné volat funkce z C++ v C#.

```
extern "C" __declspec(dllexport) void StartProcedure();
extern "C" __declspec(dllexport) void End();
extern "C" __declspec(dllexport) void Pause();
extern "C" __declspec(dllexport) void Calibrate();
extern "C" __declspec(dllexport) void passArea(const char* ar);

//typedef void(*callback_function)(string msg);
typedef void(*callback_function)(const char* Mesg);
extern "C" __declspec(dllexport) void SetCallback(callback_function aCallback);
typedef void(*callback_function3)(const char* name, int x, int y, float imagenumber, float point11, float point21, float point31, float point41, float point12,
extern "C" __declspec(dllexport) void Pictureinfo(callback_function3 aCallback3);
typedef void(*callback_function2)(int picture);
extern "C" __declspec(dllexport) void SetCallPicture(callback_function2 aCallback2);

callback_function SendToGUI;
callback_function2 SendPictureToGUI;
callback_function3 Pictureinfosend;

void passArea(const char* ar){

    prah = (int)ar;
}

void SetCallback(callback_function aCallback)
{
    SendToGUI = aCallback;
}
```

Obrázek 57: Ukázka přenesení funkcí v jazyce C++

```
public static string path = Directory.GetCurrentDirectory();
[UnmanagedFunctionPointer(CallingConvention.Cdecl)]
public delegate void CDelegate([MarshalAs(UnmanagedType.LPStr)]string Mesg);
[DllImport("ImageGraber.dll", CallingConvention = CallingConvention.Cdecl)]
public static extern int SetCallback(IntPtr aCallback);

[DllImport("ImageGraber.dll")]
public static extern void End();
[DllImport("ImageGraber.dll", CallingConvention = CallingConvention.Cdecl)]
public static extern void StartProcedure();
[DllImport("ImageGraber.dll", CallingConvention = CallingConvention.Cdecl)]
public static extern void Pause();
[DllImport("ImageGraber.dll", CallingConvention = CallingConvention.Cdecl)]
public static extern void Calibrate();
[DllImport("ImageGraber.dll", CallingConvention = CallingConvention.Cdecl)]
public static extern void passArea(string ar);


[UnmanagedFunctionPointer(CallingConvention.Cdecl)]
public delegate void CPDelegate(int picture);
[DllImport("ImageGraber.dll", CallingConvention = CallingConvention.Cdecl)]
public static extern int SetCallPicture(IntPtr aCallback2);

[UnmanagedFunctionPointer(CallingConvention.Cdecl)]
public delegate void PIDelegate([MarshalAs(UnmanagedType.LPStr)]string Name, int X, int Y, float ImageNumber, float point11, float point21, float point31
[DllImport("ImageGraber.dll", CallingConvention = CallingConvention.Cdecl)]
public static extern int Pictureinfo(IntPtr aCallback3);
```

Obrázek 58: Ukázka přenesených funkcí v jazyce C#

V práci je použito několik různých typů funkcí. Jednodušší typ funkce je typ, který nepředává žádný parametr, tedy pouze spustí danou funkci. Tohoto využívám například při startu, pauzách, nebo ukončení programu. Druhým typem je funkce, která obsahuje parametry. Tohoto využívám například při nastavení filtru pro nejmenší detekovaný objekt. Posledním typem jsou callback funkce. Tyto funkce detekují různé eventy v jazyce C++ a posílají tuto skutečnost do jazyka C#. Všechny callback funkce mají nějaký parametr a předávají informace GUI. Jsou zde 3 callback funkce a každá z nich je zde vysvětlena.

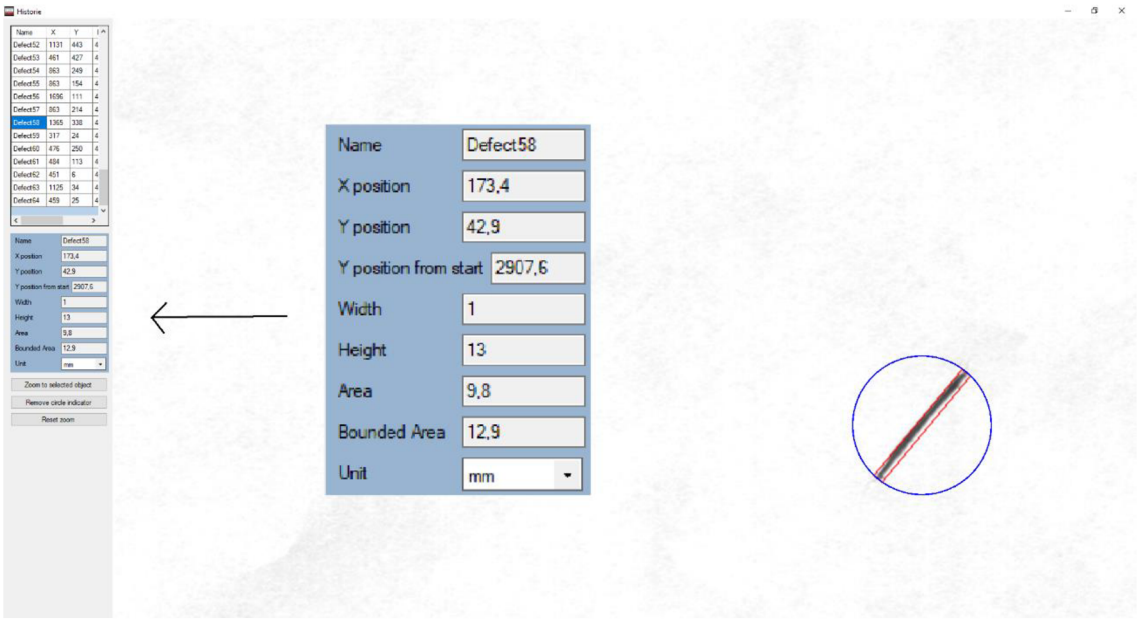
První callback funkcí je SetCallBack, která zasílá text do GUI. Toho je využíváno v logu, kde je možno vidět, co se právě provádí, jaký snímek přišel, chybové hlášky apod.

Druhou callback funkcí je SetCallPicture, která zasílá infomace o tom, že byl právě uložen snímek a dále číslo tohoto snímku. Původně jsem chtěl poslat do GUI celý snímek, bohužel se mi nepodařilo napsat wrapper pro jeho odeslání, takže přenášení snímku je implementováno tak, že je uložen do složky, kde se nachází exe soubor aplikace a následně GUI vyčten a smazán. Toto řešení je pro tuto implementaci nelimitující z důvodu nízkého frameratu CIS senzoru, ale při použití vyšších frameratů by mohl nastat problém.

Třetí callback funkcí je posílání informací o defektu. Může se zdát, že je tato informace posílána duplicitně, respektive je posílána do GUI přes callback funkci a také je ukládána do CSV souboru. Tato duplicita má však velmi důležité opodstatnění. Je totiž používána pro historii posledních x defektů. Callback je volán pokaždé, když je nalezen defekt a v GUI části se plní list. Jelikož je vidět pouze posledních 10 snímků a od začátku startu zařízení může být detekováno i několik stovek defektů, musel by se pokaždé prohledávat celý CSV soubor a hledat pouze defekty, které se nachází na posledních deseti snímcích, je načítání z CSV v tomto případě značně časově náročné. Tato duplicita však zaručuje to, že pokud je snímek odmazán, je v listu také odmazán záznam o defektu, takže list obsahuje pouze defekty na posledních deseti snímcích a načítání je značně rychlejší než při prohledávání celého CSV souboru.

7 TESTOVÁNÍ A ZHODNOCENÍ

Pro testování jsem si vytvořil na několika papírech defekty. Snažil jsem se nasimulovat veškeré druhy defektů tak, aby bylo možné prověřit detekovací funkčnost zařízení na každém z defektů. Tyto papíry jsem postupně vkládal a kontroloval úspěšnost nalezení defektu.
Na každém z papírů byl defekt správně detekován i zaznačen. Tuto skutečnost lze vidět v přílohách, kde jsou fotky tří papírů a tři snímky papírů po detekci.
Dále jsem kontroloval, zda defekty odpovídají rozměrům, které udává GUI.



Obrázek 59: Programové míry defektu

Obrázek 60: Reálná míra defektu

Provedl jsem několik měření a zjistil jsem, že odchylka naměřených hodnot od reálných se pohybuje +-0,3 mm. Na obrázcích č.59 a č.60 lze vidět, že uvedené míry defektu jsou v souladu s reálnými mírami.

Na většině obrázků z aplikace lze vidět, že korekce nerovnoměrnosti jednotlivých senzorů není úplně stoprocentní, tedy jsou viditelné černé pruhy přes obrázek. Toto je způsobeno tím, že papír není vzdálen stejně, jako při kalibraci. Cannyho hranový detektor je nastaven tak, aby tuto skutečnost ignoroval a nedetekoval tyto čáry jako hrany, tudíž na detekci nemají tyto čáry vliv.
Veškerý obalový materiál, který byl vkládán, byl světlé barvy, tak aby tvořil kontrast s černou plochou skeneru. Pokud by byl použit ke skenování tmavý materiál, bylo by nutné vyměnit podklad za světlý a tím zajistit vysoký kontrast mezi barvami.
Skelet konstrukce rovněž považuji za zdařilý, a to i přes skutečnost, že při tisku docházelo ke kroucení materiálu, tedy i nutné úpravy na konstrukci i přes 3D návrh, který by seděl naprosto přesně.

7.1 Možnosti zlepšení

Dle mého názoru je systém velmi dobře zpracovaný a celé zařízení funguje bez problémů, přesto je vždy prostor pro jistá vylepšení.
Konkrétní návrhy pro zkvalitnění systému:

- Lepší detekování defektů, které jsou větší než jeden snímek. Tento defekt by mohl být detekován jako jeden, což by usnadnilo práci obsluze při prohlížení snímků, nebo při vypracovávání statistických údajů, avšak za předpokladu, že by detekce ostatních defektů probíhala bez zpoždění.

- Zlepšení rychlosti skenování. Pokud by nedostačovala rychlost skenování, bylo by vhodné pořídit rychlejší CIS senzor. Hardware pohonu je dimenzovaný tak, aby k této výměně mohlo dojít a nemuselo se nic měnit, pouze by se naladila větší frekvence na generátoru pulzů, popřípadě by se zmenšil počet mikrokroků pro jednu otáčku krokového motoru.

- Skenování užšího formátu než jen DIN A4. Aktuální skener umí detekovat rozměr šířky DIN A4, a to z důvodu, že při tisku konstrukce skeletu došlo ke kroucení materiálu, takže skener musel být osazen vymezovacími podložkami, aby pohyb papíru byl rovnoměrný. Pokud by došlo k absolutnímu srovnání válců, tedy papír by se pohyboval dokonale rovně, bylo by možné použít i menší formát šířky.

- Lepší efektivita přenosu snímku. Aktuální předávání snímku probíhá ukládáním na disk a následným vyčítáním druhé aplikace. Lepším způsobem by bylo, aby se snímky předávaly pomocí wrapperu.

8 ZÁVĚR

Cílem diplomové práce je seznámení se s problematikou CIS senzoru pro pozdější využití v průmyslu, v praktické části pak návrh konstrukce a algoritmů detekčního systému pro vyhodnocování defektů obalového materiálu.
V první kapitole se podrobněji zabývám snímacími čipy, konkrétně CMOS a CCD, které se používají pro zachycení obrazu a jejich výhodami a nevýhodami. Ve druhé kapitole se věnuji srovnání a použití řádkových kamer a CIS senzoru, výhodám a nevýhodám jednotlivých zařízení, a to z důvodu, že tyto dvě technologie jsou nejčastěji používány ke stejným účelům. V další kapitole je popsán průzkum trhu, kde porovnávám tři nejzajímavější senzory a zmiňuji parametry pro výběr jednoho z nich k realizaci fyzického modelu konstrukce. Ve čtvrté kapitole se věnuji teorii zpracování obrazu a podrobnému rozboru algoritmů, které v práci používám právě pro detekci defektů. Nedílnou součástí této kapitoly je i popis OpenCV, jakožto představitele těchto implementovaných algoritmů v jazyce C++. Pátá kapitola je zaměřena na výběr komponentů a sestavení modelu. Vybraný motor spolu s motorovým driverem pracují bezchybně a přítomný generátor pulzů pro motorový driver je navrhnut tak, aby nedocházelo ke změně frekvence v průběhu skenování. Díky tomu je také dosaženo minimálního rozkmitu, a tedy celý pohon nezpůsobuje viditelnou chybu při snímání. Samotný skelet konstrukce je vytištěn na 3D tiskárně. Díky tomu bylo možné navrhnout kostru naprosto volně, bez jakéhokoliv omezení, které by nastalo v případě skládání skeletu z komerčních předmětů, a tím vytvořit lepší funkčnost celého zařízení. Horní částí konstrukce lze nastavovat výšku skenovaného materiálu. Toto umožňuje skenovat téměř jakkoliv vysoký materiál, který vyhovuje šířce formátu DIN A4. V šesté kapitole podrobně popisuji program, který se skládá ze dvou hlavních částí a vysvětluji komunikaci těchto částí mezi sebou. Dále uvádím důvody implementace určitých algoritmů pro předzpracování obrazu, popisuji rozdělení defektů do různých tříd a následně vysvětluji použití jednotlivých algoritmů pro detekování těchto defektů na základě jejich klasifikace. Navrhnuté GUI je intuitivní a velmi uživatelsky přívětivé. Obsahuje algoritmy pro přibližování, respektive oddalování nalezených defektů a různé statistické údaje o nalezených defektech. Každý snímek, na kterém se nachází defekt, je uložen, a to včetně statistických informací pro pozdější inspekci.
V práci byly splněny všechny vytyčené cíle. Největším přínosem je srovnání CIS senzoru a řádkových kamer pro detekování defektů na kontinuálním obrazu, tedy nekonečně dlouhém běžícím páse s možností podrobně nahlédnout na jednotlivé nalezené defekty.

Literatura

[1] *Fotomobily: snímací čipy CMOS vs. CCD*. Digimanie.cz [online]. [cit. 2019-12-06]. Dostupné z: https://www.digimanie.cz/fotomobily-snimaci-cipy-cmos-vs-ccd/2885

[2] *DÍL 1: ŘÁDKOVÉ KAMERY – TYPY A TECHNOLOGIE*. Kamery.atesystem.cz/ [online]. 2013 [cit. 2019-11-25]. Dostupné z: http://kamery.atesystem.cz/know-how/line-scan-velky-pruvodce-radkovymi-kamerami/dil-1-radkove-kamery-typy-a-technologie/

[3] *DÍL 1: ŘÁDKOVÉ KAMERY – TYPY A TECHNOLOGIE*. In: Kamery.atesystem.cz [online]. 2013 [cit. 2019-11-25]. Dostupné z: http://kamery.atesystem.cz//site/assets/files/ 1251/2013-11-18_174916.png

[4] *TDI line scan*. Stemmer-imaging.com [online]. 2020 [cit. 2020-05-18]. Dostupné z: https://www.stemmer-imaging.com/en-pl/knowledge-base/cameras-tdi-line-scan/

[5] *CIS (Contact Image Sensor).* colortrac.com [online]. 2012 [cit. 2019-11-25]. Dostupné z: https://www.colortrac.com/glossary/cis-contact-image-sensor/

[6] *CIS (Contact Image Sensor)*. In: Colortrac.com [online]. 2012 [cit. 2019-11-25]. Dostupné z: https://www.colortrac.com/wp-content/uploads/Cx-CIS-cross-section-300.jpg

[7] *Contact image sensor*. Canon-compo.co.jp [online]. 2012 [cit. 2019-11-25]. Dostupné z: http://www.canon-compo.co.jp/e/technology/cis.html

[8] *Industrial Contact Image Sensor*. Tichawa-vision.de [online]. 2018 [cit. 2019-12-27]. Dostupné z: https://tichawa-vision.de/pdf/manual_en.pdf

[9] *NOM02A4-MW60G*. Onsemi.com [online]. 2019 [cit. 2019-12-01]. Dostupné z: https://www.onsemi.com/pub/Collateral/NOM02A4-MW60G-D.PDF

[10] *Contact Image Sensor (CIS) Module*. Csensor.com [online]. 2008 [cit. 2019-12-01]. Dostupné z: https://www.datasheet4u.com/datasheet-parts/M106-A4-G1-datasheet.php?id=870500

[11] *CMOS linear image sensors*. Hamamatsu.com [online]. 2019 [cit. 2019-12-03]. Dostupné z: https://www.hamamatsu.com/resources/pdf/ssd/s14416-02_etc_kmpd1191e.pdf

[12] *ZPRACOVÁNÍ OBRAZU PŘI URČOVÁNÍ TOPOGRAFICKÝCH PARAMETRŮ POVRCHŮ* [online]. Brno, 2009 [cit. 2019-12-29]. Dostupné z: https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=17582. Diplomová práce. VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ. Vedoucí práce DOC. RNDR. ING. JIŘÍ ŠŤASTNÝ, CSC.

[13] *Matematická morfologie*. Midas.uamt.feec.vutbr.cz [online]. 2019 [cit. 2019-12-29]. Dostupné z: http://midas.uamt.feec.vutbr.cz/ZVS/Lectures/11_Matematicka_morfologie.pdf

[14] UMBAUGH, Scott E. Digital Image Processing and Analysis. 2017. CRC Press, 2018. ISBN 9781498766074.

[15] *Thresholding*. Scikit-image.org [online]. 2019 [cit. 2019-12-29]. Dostupné z: https://scikit-image.org/docs/0.13.x/_images/sphx_glr_plot_thresholding_0021.png

[16] *Filling holes in an image using OpenCV (Python / C++)*. Learnopencv.com [online]. 2015 [cit. 2019-12-29]. Dostupné z: https://www.learnopencv.com/wp-content/uploads/2015/11/imfill.jpg

[17] *A Computational Approach to Edge Detection*. Citeseerx.ist.psu.edu [online]. 1986 [cit. 2019-12-29]. Dostupné z: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.420.3300&rep=rep1&type=pdf

[18] *CANNY EDGE DETECTION*. bogotobogo.com [online]. 2017 [cit. 2019-12-29]. Dostupné z: https://www.bogotobogo.com/python/OpenCV_Python/python_opencv3_Image_Canny_Edge_Detection.php

[19] KAEHLER, Adrian a Gary BRADSKI. Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library. United States of America: O'Reilly Media, 2016. ISBN 978-1-4919-3800-3.

[20] *Řízení krokového motoru pomocí platformy Arduino* [online]. Praha, 2014 [cit. 2019-12-01]. Dostupné z: https://dspace.cvut.cz/bitstream/handle/10467/24362/F3-BP-2014-Manak-Petr-prace.pdf?sequence=3&isAllowed=y.

Bakalářská práce. ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE. Vedoucí práce Ing. Hlinovský Vít CSc.

[21] *Krokový motor Nema 17*. In: Gme.cz [online]. 2019 [cit. 2019-11-25]. Dostupné z: https://www.gme.cz/data/product/480_480/pctdetail.775-138.1.jpg?ts=1554373602

[22] *Driver pro krokové motory 4A TB6600*. In: Arduino-shop.cz [online]. 2019 [cit. 2019-11-25]. Dostupné z: https://arduino-shop.cz/arduino/7899-driver-pro-krokove-motory-4a-tb67s109aftg.html

[23] *555 Timer basics*. In: Circuitlib.com [online]. 2019 [cit. 2019-11-25]. Dostupné z: https://www.circuitlib.com/index.php/lessons/111-555-timer-basics

[24] *555 Timer basics*. In: Circuitlib.com [online]. [cit. 2019-11-25]. Dostupné z: https://www.circuitlib.com/images/theory/inside_the_555_ timer.jpg

[25] HÁJEK, Jan. Časovač 555: praktická zapojení s jedním časovačem. Dot. 2. čes. vyd. Praha: BEN - technická literatura, 1999. 555. ISBN isbn80-901984-1-4.

[26] *555 Astable Diagram*. In: Wikimedia.org [online]. 2019 [cit. 2019-11-25]. Dostupné z: https://upload.wikimedia.org /wikipedia/commons/thumb/3/3d/555_Astable_Diagram.svg/1024px-555_Astable_Diagram.svg.png

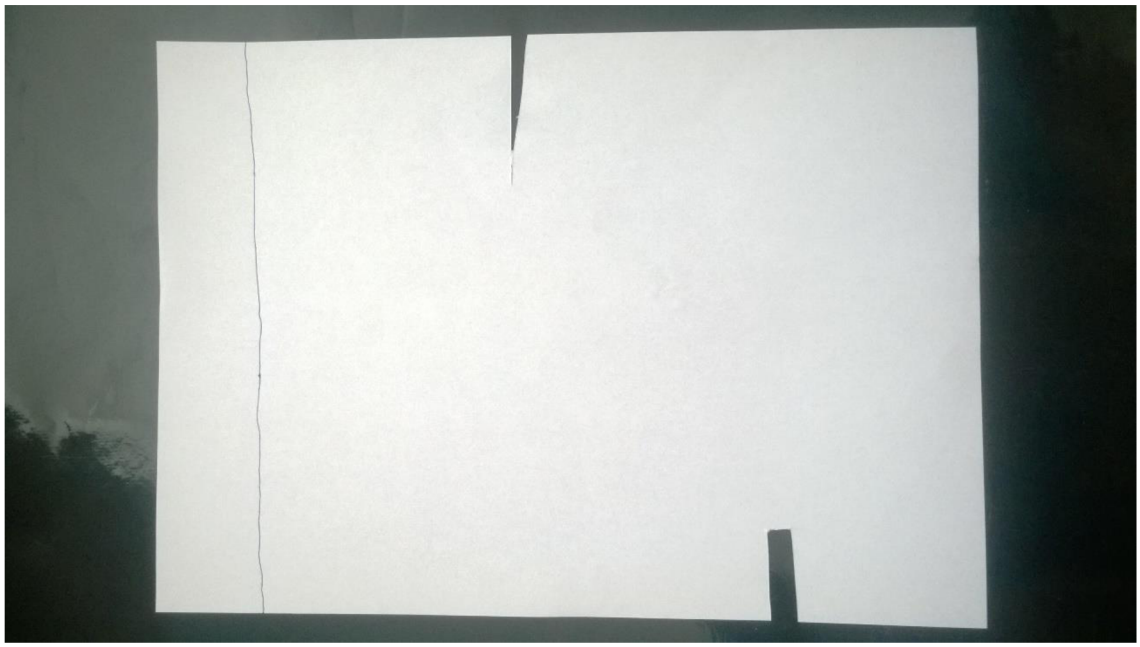
Seznam symbolů, veličin a zkratek

1. FEKT – Fakulta elektrotechniky a komunikačních technologií
2. VUT – Vysoké učení technické v Brně
3. CIS – Contact image sensor
4. RGB – Red, Green, Blue
5. DPI – Dot per inch
6. DIP – Dual in-line package
7. GPIO – General-purpose input/output
8. RC – Rezistor, kondenzátor
9. A/D – Analogově digitální
10. LED – Light Emitting Diode
11. CMOS – Complementary Metal Oxide Semiconductor
12. CCD – Charge coupled device
13. NMOS – N-type metal-oxide-semiconductor
14. GUI – Graphic User Interface
15. CSV – Comma-separated values
16. DLL – Dynamic-link library
17. RAM – Random access memory
18. PLA – polylactic acid
19. DC – Direct current
20. FPS – Frame per second
21. SD – Secure Digital
22. FPGA – Field Programmable Gate Array
23. MCU – Microcontroller unit
24. UDP – User Datagram Protocol
25. TCP – Transmission Control Protocol
26. CNC – Computer Numerical Control
27. ENA – Enable
28. DIR – Direction
29. PUL – Pulse
30. BDS – Berkeley Software Distribution
31. DIN – Standard pro velikosti papíru
32. MTF – Modulation Transfer Function
33. ML – Machine Learning
34. IP – Internet Protocol
35. TDI – Time Delay Integration

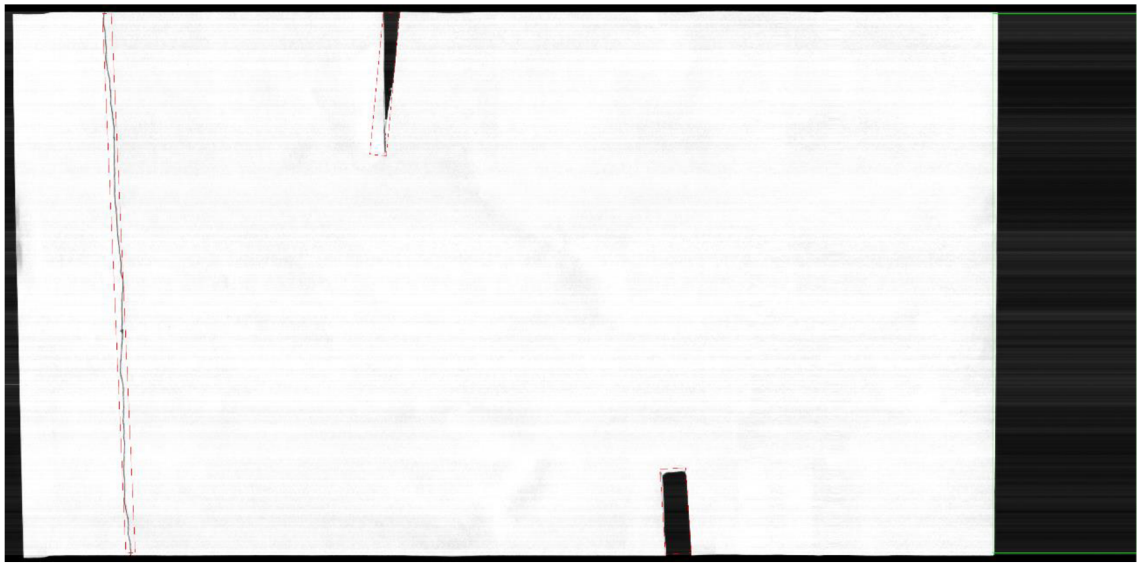
Přílohy

Příloha 1. Papír s defekty 1
Příloha 2. Papír s detekovanými defekty 1
Příloha 3. Papír s defekty 2
Příloha 4. Papír s detekovanými defekty 2
Příloha 5. Papír s defekty 3
Příloha 6. Papír s detekovanými defekty 3
Příloha 7. Osazovací plán plošného spoje generátoru pulzů
Příloha 8. Spodní část plošného spoje generátoru pulzů

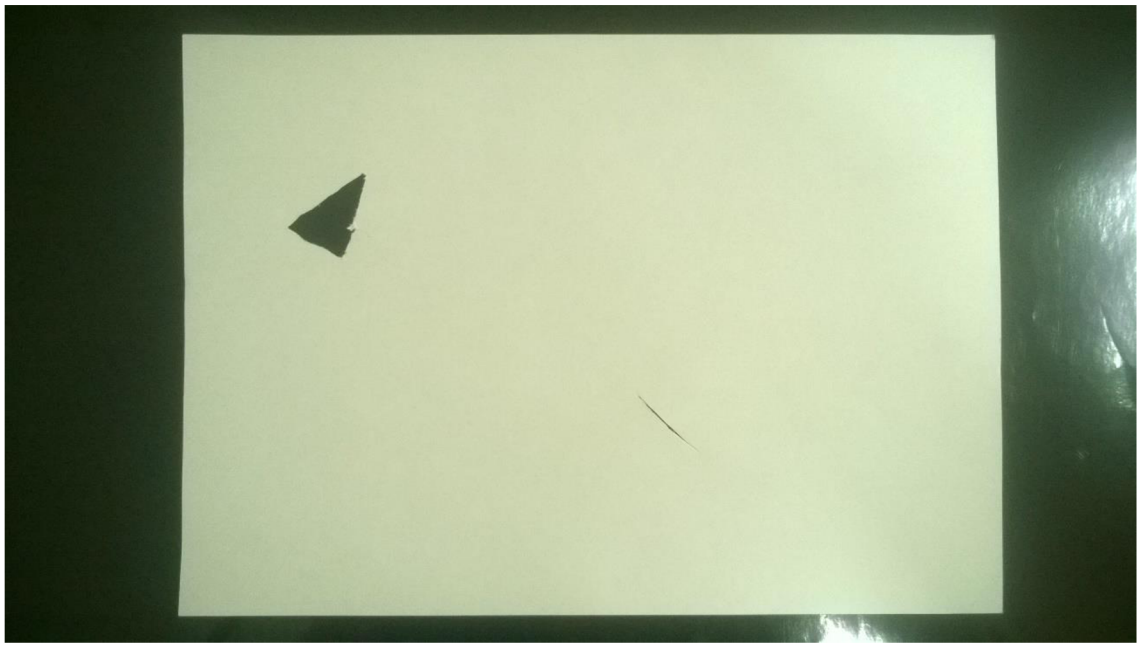
Příloha 1: Papír s defekty 1



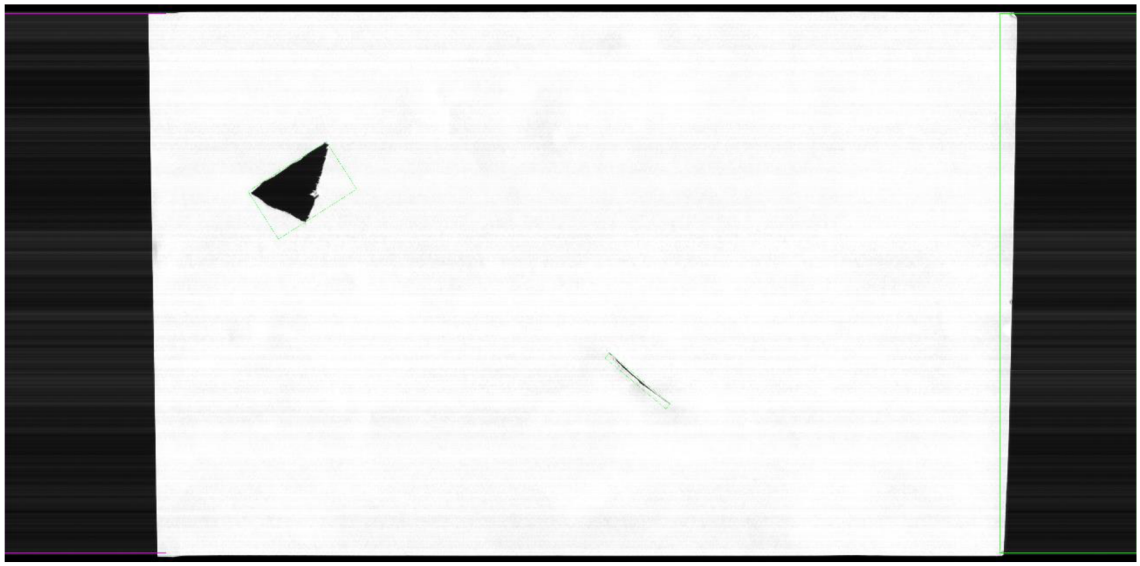
Přílohy 2: Papír s detekovanými defekty 1



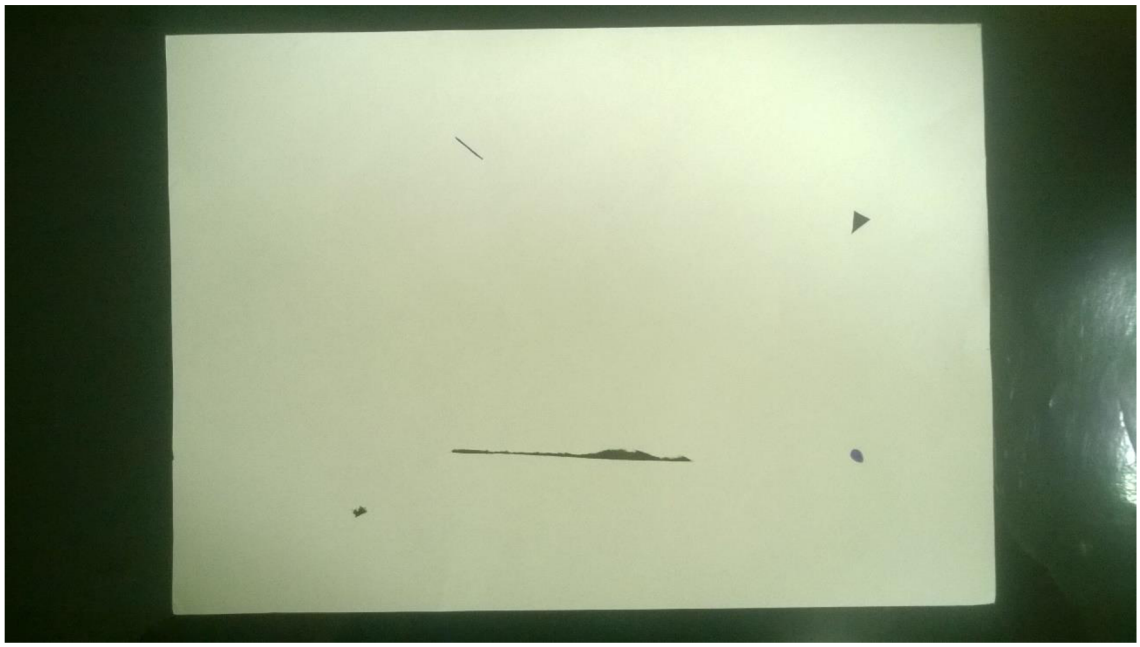
Příloha 3: Papír s defekty 2



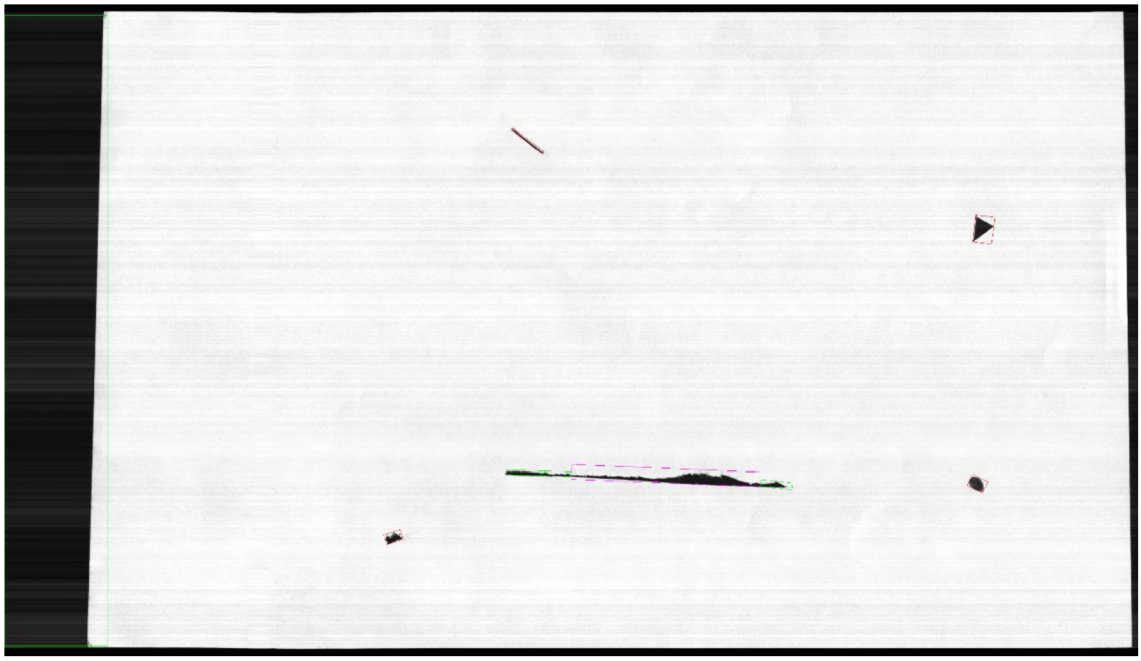
Příloha 4: Papír s detekovanými defekty 2



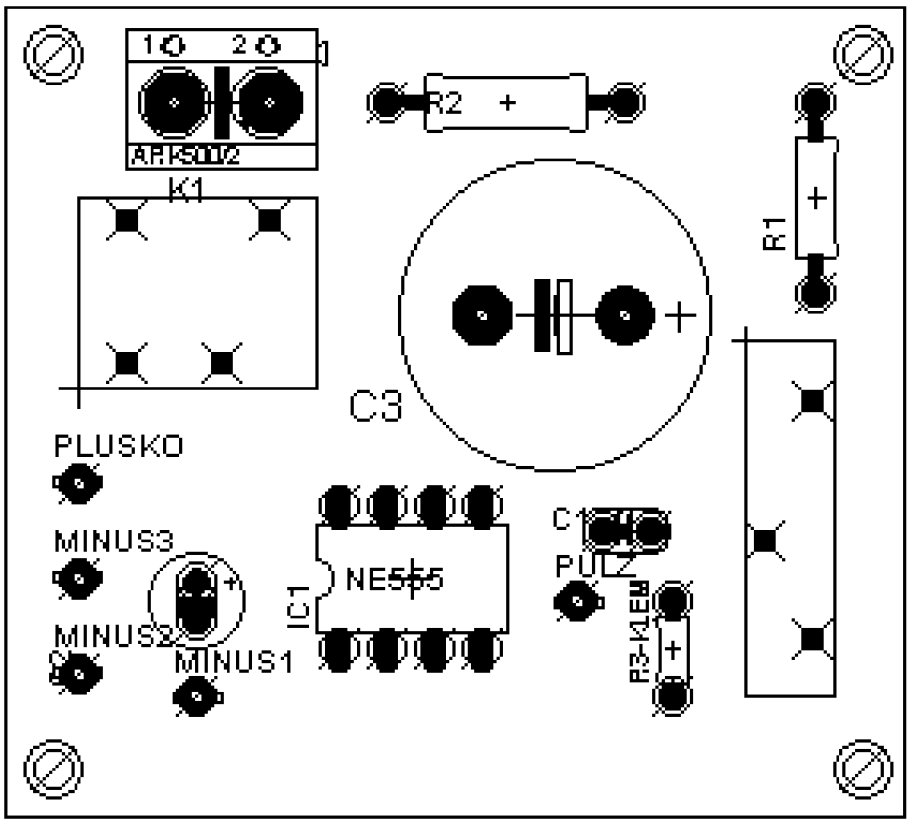
Příloha 5: Papír s defekty 3



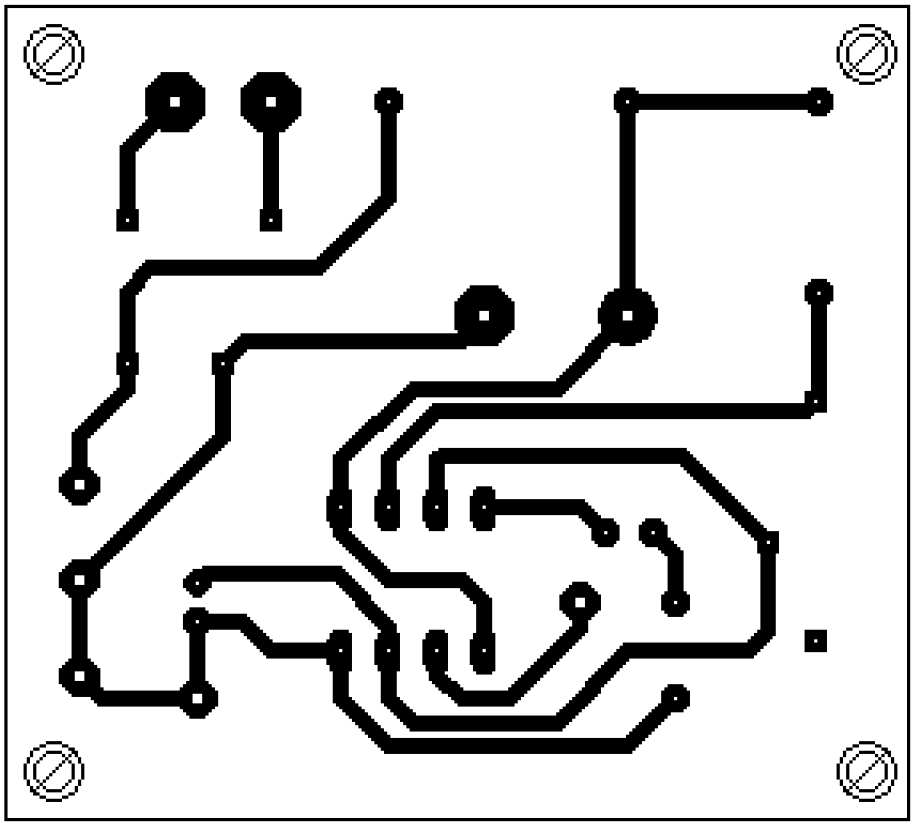
Příloha 6: Papír s detekovanými defekty 3



Přílohy 7: Osazovací plán plošného spoje generátoru pulzů



Přílohy 8: Spodní část plošného spoje generátoru pulzů



Přílohy na CD

Příloha 1. Text diplomové práce
Příloha 2. Obrázky s detekcí a návrh plošného spoje
Příloha 3. Soubory pro tisk konstrukce
Příloha 4. Program
Příloha 5. Zdrojový kód programu