ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická

Katedra mikroelektroniky

Optická inspekce na pracovišti výstupní kontroly

Květen 2018

Student: **Bc. Jakub Demjan**

Vedoucí práce: **Ing. Stanislav Vítek, Ph.D.**

Čestné prohlášení

Prohlašuji, že jsem zadanou diplomovou práci zpracoval sám s přispěním vedoucího práce a konzultanta a používal jsem pouze literaturu v práci uvedenou. Dále prohlašuji, že nemám námitek proti půjčování nebo zveřejňování mé diplomové práce nebo její části se souhlasem katedry.

Datum: 25. 5. 2018

.....
Podpis studenta

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Demjan** Jméno: **Jakub** Osobní číslo: **420022**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra mikroelektroniky**

Studijní program: **Elektronika a komunikace**

Studijní obor: **Elektronika**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Optická inspekce na pracovišti výstupní kontroly

Název diplomové práce anglicky:

Optical Inspection at the Output Control Workplace

Pokyny pro vypracování:

Navrhněte a implementujte systém automatické optické inspekce na pracovišti výstupní kontroly. Řiďte se těmito pokyny:
1) Prostudujte problematiku výstupní kontroly a zhodnoťte možnosti automatické příp. poloautomatické optické inspekce.
2) Mezi úkoly optické inspekce patří zejména:
a) určení pozice a natočení výrobku v obraze,
b) kontrola lícování konektorů s krytem,
c) středová souměrnost nastavovacího elementu vzhledem k otvoru v krytu,
d) přítomnost jumperu ve zdířce, e) přítomnost šroubků a jejich dotažení, f) nepřítomnost škrábanců na krytu přístroje,
g) další úkoly po konzultaci s vedoucím práce.
3) Proveďte rešerši dostupných algoritmů, vhodných pro daný úkol.
4) Na reálných snímcích získaných v průmyslovém provozu ověřte robustrnost algoritmů. Diskutujte vliv okolního prostředí

Seznam doporučené literatury:

[1] Rafael C. Gonzales, Richard E. Woods. Digital Image Processing. Pearson, 2007. ISBN 978-0131687288
[2] Rafael C. Gonzales et al. Digital Image Processing using Matlab. Pearson Prentice Hall, 2003. ISBN 978-0130085191
[3] Cognex InSight software user's manual. Dokument firmy Cognex.

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Stanislav Vítek, Ph.D., 13137

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **12.02.2018** Termín odevzdání diplomové práce: _____

Platnost zadání diplomové práce: **30.09.2019**

Ing. Stanislav Vítek, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Ing. Pavel Ripka, CSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

. _____ Datum převzetí zadání

_____ Podpis studenta

Anotace:

Tato práce se zabývá problematikou optické inspekce výstupní kvality výrobků ve finálních fázích výroby včetně detekce a vyhodnocení vadných kusů. Cílem práce je vyvinout poloautomatický detekční systém uzpůsobený pro kontrolu kritických částí výrobku z hlediska funkčnosti, ale také je zde snaha detekovat i estetické vady z důvodu omezení reklamačních řízení, úspoře materiálu a prostředků.

Klíčová slova:

optická inspekce

Cognex InSight

Matlab

Summary:

This work take care of output quality of industrially manufactured products in final phases of manufacturing. Manufacturing is complicated proces which consist of many steps and it means there is a plenty space for mistakes. The aim of this work is to develop semi-automatic optical inspection system which can be able to detect functional and esthetic defects for saving material and money.

Index Terms:

optical inspection

Cognex InSight

Matlab

Poděkování
Rád bych poděkoval Ing. Stanislavu Vítkovi, Ph.D. za cenné rady, věcné připomínky a vstřícnost při konzultacích a vypracování diplomové práce.

Obsah

Seznam použitých symbolů a zkratek

Explorer - Cognex InSight Explorer Software
Job - sousled funkcí v Exploreru (program)
HOG - Histogram Orientovaných gradientů
OPC - Open Platform Communicetions
FTP - File Transfer Protocol
LED - Light Emitting Diode
ROI - Region of Interest
DIN - Deutsches Institut für Normung
DNA - Deoxyribonukleová kyselina
DPS - Deska plošného spoje
RAM - Random Access Memory

Seznam obrázků

Seznam tabulek

Úvod

V průmyslovém výrobním závodu je několik faktorů ovlivňujících výsledný hospodářský výsledek. Některé z nich jsou ovlivnitelné jednoduchými úpravami výrobního procesu, jako kvalita výrobků, chybovost nebo objem výroby za určitý čas. Některé faktory naopak nejsou ovlivnitelné jednoduchou změnou, například poptávka po výrobku nebo doba života výrobku. Snahou každého výrobního závodu by mělo být zlepšování aspektů výroby pro vyšší zisky, přičemž snadno sledovanou skutečností je chybový stav výrobků v různých fázích výroby.

Tento stav je důležitý pro optimalizaci celého výrobního procesu a zamezení expedice vadných výrobků, současně pak při sledování výroby, s jejím objemem roste i úspora materiálu a výrobního času na produkci zmetků. Základní myšlenkou sledování výrobní sekvence od začátku do konce je periodická kontrola předpokládaného stavu polotovaru a v případě neshody pak vyhodnocení a akce zamezující spotřebu dalšího materiálu. Výše zmíněné potřebě kontroly vyhovuje pro průmysl systém Cognex InSight, který obsahuje zařízení pro snímání výrobků, vyhodnocení a ovládání akčních členů pro vytvoření automatického kontrolního systému na celé výrobní lince.

Použití kontrolních mechanismů v průmyslu má několik podmínek, bez kterých se dobrý systém neobejde. Hlavním ukazatelem využitelnosti systému v průmyslu je jeho implementace do již zavedeného systému, rychlost inspekce a komunikace s ostatními systémy na výrobní lince. Právě proto se výrobci snaží udělat systém všestranný s přihlédnutím k jistým omezením. Hlavním smyslem této práce by měla být definice světlých a stinných stránek při využití systému Cognex InSight ve výrobním procesu v porovnání s jinými metodami řešení inspekčního systému.

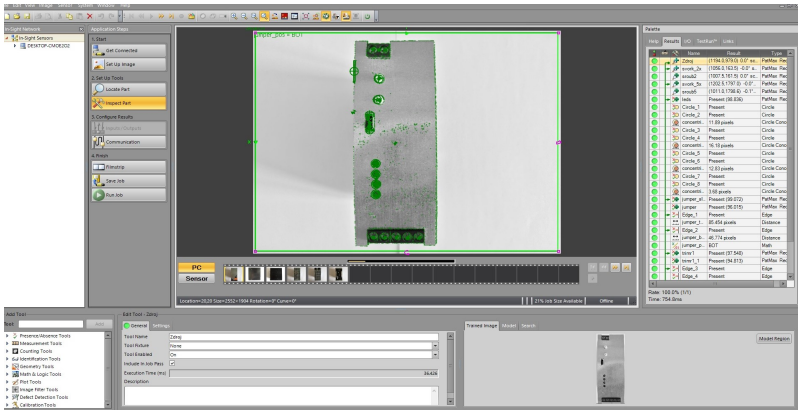
Uvažuji inspekční systém dle zadání diplomové práce jako konečnou kontrolu výrobku z hlediska estetiky a funkčnosti pro zamezení případných reklamací. Systém by měl být náhradou za manuální inspekci a měl by umět rozlišit výrobek od zmetku, stejně tak jako detekovat případné minimální estetické vady. První část práce je věnována návrhu konvenčního řešení používáného v průmyslu a definování silných a slabých stránek takového řešení testováním. Druhou část práce obsadilo řešení založené na neuronové síti s podpůrnými algoritmy v Matlabu, které by mělo demonstrovat schopnost konkurence ostatních řešení vůči běžně používaným postupům s ohledem na dobu návrhu, robustnost algoritmů, rychlost inspekce a další stěžejní faktory ovlivňující možnosti využití inspekčního systému v praxi.

1 InSight Explorer

1.1 Představení

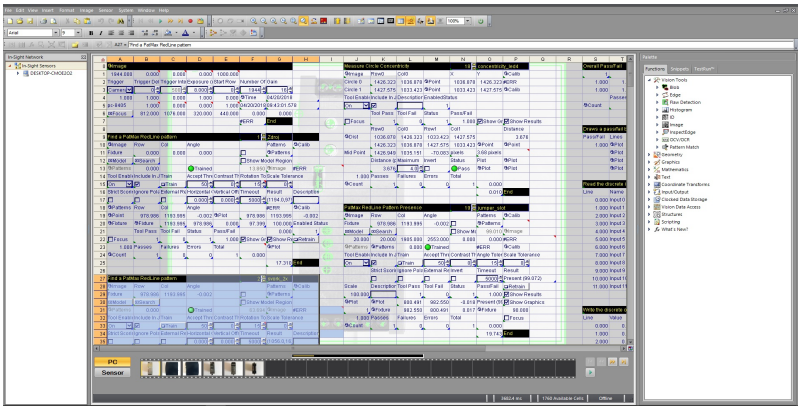
InSight Explorer, dále jen Explorer, je proprietární prostředí pro návrh algoritmů úzce orientovaných na hardware z řady Cognex InSight výrobce Cognex Corporation. Jako takový je velmi dobře optimalizován pro veškerý hardware od tohoto výrobce. Vzhledem k rozdílným požadavkům se dá Explorer používat dvojím způsobem.

První způsob je určený převážně pro rychlý návrh detekčního algoritmu stejně tak, jako použití minimálního počtu detekčních kroků, jakožto i minimální počet použitého hardwaru. Právě těmto skutečnostem výborně nahrává základní mód spuštění Exploreru s názvem Easy Builder.



Obr. 1.1: Cognex InSight Explorer - Easy Builder

Na obrázku 1.1 je vyobrazen funkční job s výsledky (vpravo), paletou dostupných funkcí (vlevo) s konfiguračním oknem funkce (ve spodní části) a obraz z kamery (střední část). Samotná paleta Easy Builderu leží v levé části, kde můžeme jednoduše klikem přepínat jednotlivé esenciální kroky detekčního systému.



Obr. 1.2: Cognex InSight Explorer - Spreadsheet

Druhou možností jak v programu pracovat je zobrazení Spreadsheet, které nám umožní přistupovat k jednotlivým proměnným přes adresovací systém tabulkového procesoru. Tento mód Exploreru, viditelný na obrázku 1.2, je velmi výhodný při provázání systému InSight a

dalších detekčních systémů, zvláště pak schopnost komunikace přes telnet velmi výrazně ovlivňuje použitelnost celého systému.

Spreadsheet můžeme vidět na obrázku 1.2, kde zabírá většinovou plochu okna Exploreru. V pravé části se nacházejí pak použitelné funkce systémem "Drag and Drop". Velmi důležitá je skutečnost, že provedeme-li úpravu vyvinuté sekvence v Easy Builderu pomocí Spreadsheetu, pak již musíme používat pouze toto zobrazení z důvodu nekonzistence obou zobrazení. Z tohoto důvodu, jsem se rozhodl programovat celou sekvenci pomocí Easy Builderu až do doby, kdy bude potřebná funkce provázání jednotlivých jobů pomocí komunikace s další platformou.

1.1.1 Připojení

Pod touto ikonou se skrývá jednoduchá utilita, která umožní nastavit zařízení ze kterého bude snímán obraz. Po připojení zobrazí veškeré dostupné parametry zařízení usnadňující definici zařízení, za účelem nastavení veškerých potřebných parametrů v následujících krocích. To samé platí i u emulátoru zařízení, kde je možné nastavit libovolnou kameru z nabídky výrobce a následně testovat vyvinuté sekvence.

1.1.2 Nastavení obrazu

Hlavní podmínku pro úspěšnou optickou inspekci obrazu, asice kvalitní snímek, docílíme vhodným nastavením snímacího zařízení. V této sekci je možné konfigurovat veškeré parametry týkající se výstupního obrazu jako: doba expozice, rychlost závěrky, rozlišení, vyvážení bílé barvy, spouštění, korekce zkreslení objektivu a další. Tyto parametry je vhodné definovat určitými hodnotami až při testování v reálném provozu na místě, kde systém bude pracovat. V přívětivějších podmínkách okolního prostředí poslouží i všudypřítomné automatické nastavení parametrů dle aktuálního snímku, avšak toto v praxi nefunguje vždy.

1.1.3 Lokalizace vzorku

Před samotnou inspekcí je nutné lokalizovat testovací vzorek a generovat tzv. Fixture. Fixture je upínací funkce, která pozičně svazuje jednotlivé detekční metody a usnadňuje jim práci. V praxi dobrá lokalizace znamená výrazné urychlení jakýchkoliv pozdějších detekcí při navázání na prvotní funkci. Naprostá většina funkcí v tomto kroku pracuje jako nadstavba detekce hran, jejíž princip je popsán v následující kapitole (2.Matlab).

1.1.4 Vyšetření vzorku

Zde už přichází hlavní část programu s maximálním počtem výstupních dat. V nabídce funkcí nalezeneme především tzv. Pattern Matching, kdy je porovnávána předloha určité části snímku s aktuálním snímkem a s následným vyhodnocením pravděpodobnost podoby. Tato funkce čerpá data k porovnání především z detekce hran, ale také z různých deskriptorů, což jsou identifikátory vytvořené různými způsoby. Principy fungování jsou opět popsány v druhé kapitole. Dále zde nalezneme mnoho užitečných funkcí, které buď přímo operují s obrazem nebo extrahovanými daty, avšak vyskytují se zde i funkce čistě matematické nebo logické.

1.1.5 Vstupy a výstupy

Systém optické inspekce InSight Vision si zakládá na jednoduchosti řešení, obsahující většinou jedno snímací zařízení na inspekční pracoviště, které komunikuje s ostatními prvky na výrobní lince. Pro malé linky s minimálním počtem aktuátorů může být výhodné nastavit ovládání inspekčního procesu tak, aby vše řídil snímací hardware (kamera). K tomuto účelu disponují kamerové systémy firmy Cognex Corporation inteligentními kamerami s vnitřním vyhodnocením a digitálními vstupy/výstupy, které je možno nakonfigurovat například na regulaci osvětlení, řízení robota a ovládání pásového dopravníku.

1.1.6 Komunikace

Velmi podobná funkce se skrývá i pod další tlačítkem s popisem komunikace. Z hlediska komunikačních protokolů záleží na podpoře snímacího zařízení, ale můžeme zde najít FTP, proprietární EasyView protokol, OPC protokol, CAN sběrnici a mnohé další. Podporou všemožných komunikačních standardů výrobce usnadní začlenění svého hardwaru do již stávající infrastruktury zařízení ve výrobním provozu.

1.1.7 Filmový pás

Filmový pás umožňuje ukládat předdefinované obrázky v závislosti na detekovaných chybách pro provázání systému s dalším vyhodnocením popřípadě komunikací.

1.1.8 Job

Poslední tlačítka umožňují uložit sestavenou sekvenci jako soubor pro pozdější použití nebo je možné kameru přepnout do online módu a začít ihned používat. Je nutno říci, že velikost sekvence je omezená od výrobce a je nutné pro rozsáhlejší inspekční systémy rozložit sekvenci do více jobů, a poté ji postupně načítat do paměti kamery. Toho se dá docílit různými způsoby, avšak nejjednodušší se zdá být komunikace s kamerou přes telnet a vytvoření automatizované komunikační části, která bude reagovat na výstupy z kamery a přizpůsobí sekvenci požadavkům.

1.2 Realizace optického inspekčního systému

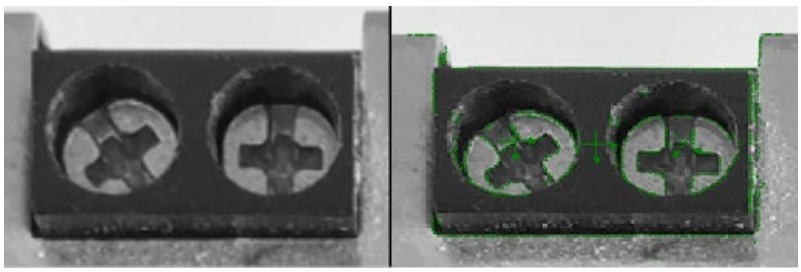
Vzhledem k omezené velikosti jednotlivých jobů je rozdělen testovací program do šesti částí, z nichž každá část jest uložena jako samostatný job pro kontrolu specifické části testovacího vzorku. K tomuto účelu byl vyfocen každý testovací vzorek ze šesti směrů ekvivaletně kolmo ke stranám kvádru. Každý job je poté nastaven na jeden směr pohledu na vzorek tj. jednu stranu kvádru. Algoritmy jsou přizpůsobené na detekci defektů, které vznikají při výrobním procesu, přičemž každý defekt bude popsán v části kde se vyskytuje a bude navrženo řešení. Představím zde úplnou sekvenci pro inspekci jednoho konkrétního testovacího vzorku s tím, že ostatní algoritmy jsou analogické a proto jim nebude věnována vyšší pozornost.

1.2.1 Čelní panel

Čelní panel je jedna ze zásadních stran zdroje, protože obsahuje veškeré nastavovací a indikační prvky potřebné pro správnou funkci celého zdroje. Přítomnost jednotlivých komponent definuje nároky na testovací sekvenci a její přesnost. Po prvotní detekci polohy čelního panelu je nutné zkontrolovat přítomnost veškerých utahovacích šroubků, stejně tak jako správné usazení svorkovnic vzhledem ke krytu. Další kroky algoritmu by se měly zaměřit na středovou souměrnost nastavovacích trimrů a indikačních led vzhledem k otvorům v krytu. Posledním krokem je pak detekce přítomnosti jumperu, popřípadě správné zasazení patice pro jumper. Návdavkem by mohla být ještě funkce kontroly stejnorodosti povrchu pro detekci estetických vad, bohužel krycí plech a jeho porchová úprava značně znesnadňuje detekci, a proto je zde tento krok vynechán.

Obr. 1.3: Čelní panel testovaného vzorku

Ke kontrole svorkovnic s fixačními šroubky je využita funkce Pattmax Redline, která je vyvinuta a patentována firmou Cognex. Vzhledem k této skutečnosti nelze zjisti přesnou podstatu vnitřního fungování a proto zde bude funkce hodnocena a nastavována tak, jak výrobce umožnil. V oblastech detekce předlohy je naprostá většina funkcí založena na detekci hran, popřípadě kombinována s takzvanými deskriptory, které jsem taktéž použil a popsal v části Matlab.



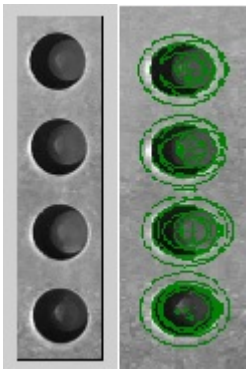
Obr. 1.4: Svorkovnice - Pattmax Redline

Dle obrázku 1.4 můžeme vlevo vidět předlohu vystřihnutou ze snímku, vpravo poté lokalizovanou předlohu. Zelenou barvou je pak vyznačen hlavní obrys předlohy s vnitřní strukturou, což podporuje tvrzení o použité detekci hran. Funkce určí polohu, natočení a pravděpodobnost určení předlohy v testovacím snímku. Dále ve středu lokalizované předlohy

vykreslí orientovaný osový kříž, který je možné použít pro koordinaci dalších funkcí. V tomto případě využívá koordináty další funkce Pattmax Redline, která lokalizuje přítomnost šroubků ve svorkovnici.

Požadavek na detekci správného usazení svorkovnice vzhledem ke krytu je naplněn označením předlohy překrývající hranice svorkovnice/kryt.

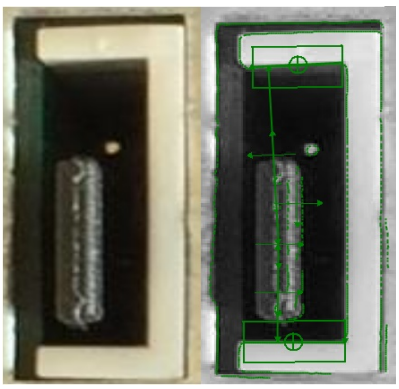
Kontrola indikačních led bude z části analogicky stejná jako jakákoliv jiná detekce předlohy v obraze pomocí funkce Pattmax Redline, avšak zde je nutné kvůli požadavkům na kontrolu pozice diod vzhledem k otvorům v krytu přidat další funkci schopnou toto obstarat.



Obr. 1.5: Indikační led

Vybral jsem si funkci Circle Concentricity, která umí určit dvě překrývající se kruhové hrany, rekonstruovat kruhy a určit vzdálenost mezi středy těchto kruhů. Poté se dá vyhodnotit zda je ještě únosná nebo není. Dle obrázku 1.5 junkce naprosto bez problému nalezla kruhové otvory v krytu, ovšem rekonstrukce kruhů reprezentujících led nebyla úspěšná bez zásahu vlivem kvality vstupních dat, přičemž i tak sama funkce v defaultním nastavení detekovala polovinu led.

Prostředí umožňuje ruční nastavení prohledávané oblasti ve velmi jemných krocích, a proto se mi podařilo detekovat všech osm kruhů. Aby tato funkce vykazovala menší chyby, tak jsou všechny úkony prováděné v oblasti s indikačními led závislé na koordinátech na výstupu funkce Pattmax Redline.



Obr. 1.6: Jumper

V případě jumperu je potřeba využít více kroků, které na sebe navazují a vzájemně se ovlivňují. Jde především o funkci Pattmax Redline, detekci hran, funkci měřící vzdálenost mezi hranami a k vyhodnocení matematicko – logickou funkci.

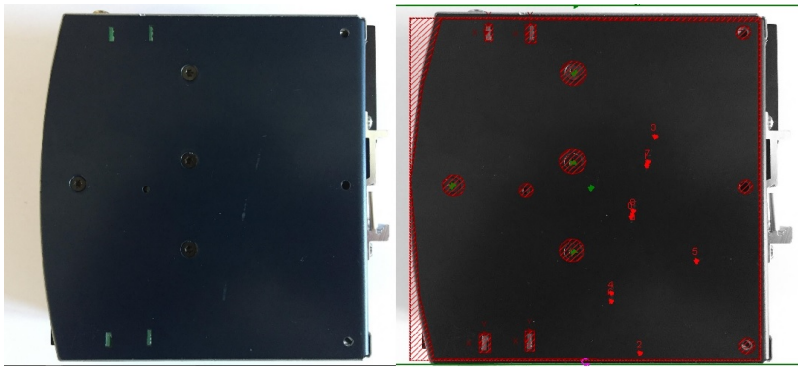
Na obrázku 1.6 je vidět postup inspekce jumperu. Jako první se aplikuje funkce Pattmax Redline na slot jumperu, která vytvoří koordináty potřebné pro přesnější lokalizaci samotného jumperu. Po lokalizaci jumperu toutéž funkcí je aplikována detekce hran na vnitřní část slotu, která slouží k vymezení měřitelných vzdáleností středu koordinátů generovaných funkcí Pattmax na jumper. Obrázek 1.6 ukazuje původní obraz nebo také vzor pro slot jumperu vlevo a obraz s detekovanými hranami a ROI aplikovaných funkcí.

Dalším krokem je detekování směru a úhlu natočení trimrů, realizovaný na stejných základech jako předchozí inspekce, avšak s rozdílným nastavením uvažovaných hodnot mezí, zejména pak pro toleranci natočení předlohy.

Poslední částí inspekčního algoritmu je poté ověření osové souměrnosti trimrů vůči krycímu plechu čelního panelu. Vzhledem k použití trimrů s mechanismem procházejícím určeným otvorem není dle požadavků zadavatele nutné pro prezentovaný vzorek realizovat kontrolu, protože správnost sesazení je definována fyzickými rozměry trimru a otvoru, přičemž pokud trimr detekujeme, pak je osová souměrnost v akceptovatelných mezích.

1.2.2 Boční strana

Kontrola bočních panelů je o mnoho jednodušší vzhledem k požadavkům, které jsou na ní kladeny. Hlavním problémem vyskytujícím se na velkých plochách, při uvažování manipulace, jsou různá mechanická poškození. Z toho důvodu je nutné pouze detekovat polohu boční strany testovaného vzorku a dále zajistit důležitá místa obsahující předem známé nespojitosti povrchu jako jsou různé montážní prvky, větrací otvory a další nezbytná narušení celistvosti povrchu.



Obr. 1.7: Inspekce bočních stran vzorku

Po přípravě v podobě vymaskování ROI je aplikována funkce Surface Flaws, která umí detekovat nespojitosti na povrchu formou sdružených pixelů jiné intenzity než je zbytek povrchu. Vnitřní struktura funkce je opět neznámá, avšak dle nastavovacích parametrů bude tato funkce nejspíše fungovat formou maskování, generované masky na míru, obrázku a zjišťování nenulových rozdílových pixelů. Funkce umožňuje nastavit citlivost formou velikosti detekční oblasti, minimální velikosti detekované chyby a kontrastu.

Na obrázku 1.7 je nalevo vidět původní snímek s montážními prvky, větracími otvory a testovacími vadami povrchu. Vpravo je poté snímek zobrazující sytě rudými koordináty polohy chyb povrchu, šrafovanými mřížkami neprověřované oblasti a zeleným koordinátem počáteční kontrolu pozice boční strany testovaného vzorku. Postup inspekce obou bočních stran je zcela stejný vyjma zrcadlového otočení hledaného vzoru a jiné masky neprohledávaných oblastí, proto je zde uvedena boční strana v jednom vyhotovení.

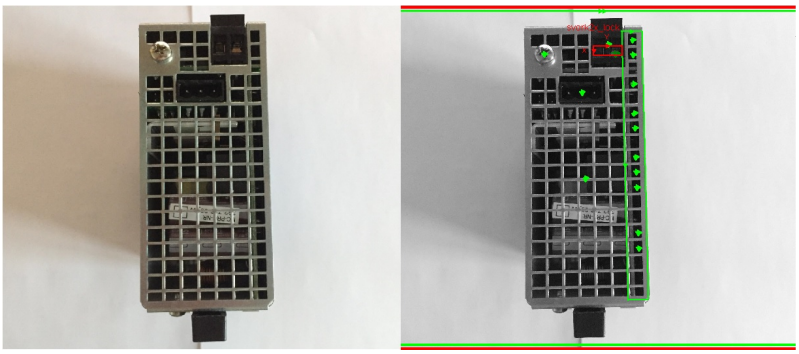
1.2.3 Vrchní strana

Na vrchní i spodní straně jde při inspekci především o zjištění otevřenosti svorkovnic ať již se šroubky nebo pružinovou čelistí, dále pak o kontrolu všech montážních prvků a samozřejmě o správné usazení veškerých konektorů a svorkovnic vzhledem ke krytu.

Na obrázku 1.8 je zcela analogicky jako v předchozích případech původní snímek vlevo a již zpracovaný snímek na pravé straně. Použité funkce jsou dle stejného vzoru jako v předchozích případech, tedy pro určení pozice vrchní strany testovaného zdroje použita funkce Pattmax Redline, stejně tak je tato funkce použita pro určení pozic a správného usazení svorkovnice a konektoru. Navíc je zde oproti předchozím případům tato funkce nastavena na detekci vzorů a jejich počítání.

Na obrázku 1.8 vpravo nahoře svítí červeně oblast, o kterou se funkce zajímá a hledá v tomto případě spodní část čelisti svorkovnice, která se liší velmi výrazně od zadní stěny otvoru, a proto je možné ji detekovat pomocí funkce hledání vzorů. Jediný problém spočíval ve falešné detekci vzorů z důvodu nižšího kontrastu vlivem špatného osvětlení otvorů, a proto jsem zúžením upravil oblast kde daná funkce operuje a dosáhl tak kýženého výsledku.

Masivní počet sytě zelených koordinátů ve sloupci značí část jobu detekující izolační přepážku opět založenou na funkci Pattmax Redline zjišťující počet zadaných vzorů. Spodní strana vypadá většinou velmi podobně, a proto je její inspekce prakticky totožná, z toho důvodu je zde uvedena pouze jedna z nich.



Obr. 1.8: Kontrola otevřenosti svorek a přítomnosti izolační přepážky

V případě inspekce zdroje se svorkovnicemi bez šroubků je nutné kontrolovat i úhel ovládajících páček kvůli zamezení průchodnosti deformovaných vzorků inspekčním systémem. Tato inspekce je založena opět na funkci Pattmax Redline s určením jasným mezí natočení předlohy a nutností najít určitý počet páček ve správné poloze.

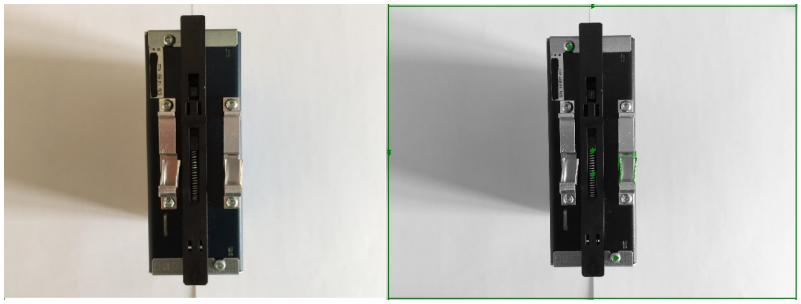
1.2.4 Zadní strana

Po inspekčně úsporných stranách přichází na řadu o něco složitější strana zadní, jejíž kompletnost je skoro tak stejně závažná jako u čelního panelu. Vyskytuje se zde mnoho důležitých součástí, které jsou potřebné pro uchycení zdroje na montážní lištu DIN, popřípadě fixační prvky krycích vrstev.

Hlavní částí fixačního mechanismu je tzv. t-halter, což je profil přesně navržený tak, aby zapadl do montážní lišty. Na zadní straně se vyskytují vždy dva a každý z nich je připevněn k tělu

zdroje dvěma šroubky, dále pak každý z nich obsahuje distanční plechovou vložku. Pro správnou funkci mechanismu je nutná druhá strana čelisti svírající DIN lištu, která je tvořena tzv. šíbrem, což je kluzný plastový fixační profil podpořený pružinou. Celý tento mechanismus musí být kontrolován, a proto je zde opět použita s výhodou funkce Patmax Redline, která kontroluje přítomnost obou t-halterů, šíbru s pružinou a veškerých šroubků.

Vzhledem k volnějšímu uchycení distančních plechových vložek je nutné kontrolovat jejich vůli. Toto je dosaženo opět použitím funkce pro detekci vzorů, ale omezením pozitivní detekce na velmi malé spektrum úhlů odklonu, ve kterých se může daná předloha nacházet. Tato úprava způsobí zvýšenou citlivost v případě většího vybočení plechové vložky a systém pak vyhodnotí absenci. Zadní strana je na obrázku 1.9, zcela analogicky jako v předchozích případech na levé straně snímek původní a na pravé straně pak výsledek většího natočení plechové vložky s následkem ztráty detekce.



Obr. 1.9: Kontrola prvků pro fixaci zdroje na din lištu

Dvojité koordináty na středu šíbru jsou výsledkem funkcí hledání předloh samotného šíbru, stejně tak jako hledání pružiny v šíbru. Šroubky je možné kontrolovat jako součást nějaké předlohy, avšak pro prioritní nálezy je vhodnější použít funkci zvlášť na každý druh předlohy.

1.2.5 Komunikace přes Telnet

Naprogramované joby pro kontroly jednotlivých částí vzorků jsou pouze jednou z hlavních částí celého inspekčního systému. Při snaze vyvinout automatický inspekční systém, potřebujeme využít některý z komunikačních standardů pro koordinaci jednotlivých částí systému a jeho začlenění do výrobního procesu. Možnou volbou je protokol Telnet.

Telnet standardně funguje jako spojení klient server, kdy server naslouchá příchozím připojením na portu 23. Kamery firmy Cognex obsahují tento server, a proto je možné se k nim pomocí Telnetu připojit. Při programování jobů jsem používal emulátor, avšak princip komunikace s kamerami je zcela stejný. Zde přichází na světlo důvod použití zobrazení "Spreadsheet" v programu InSight Explorer, protože přístup k datům jednotlivých jobů je adresován pomocí koordinátů tabulek, jak je vidno na obrázku 1.2.

```
Welcome to In-Sight(tm)  PC-8405 Session 0
User: admin
Password:
User Logged In
GVC20
1
1195.493
GVD20
1
980.488
```

Obr. 1.10: Vyčtení hodnoty proměnné jobu přes Telnet

Na obrázku 1.10 je výstup z komunikátoru při spojení na server emulovaný počítačem, přičemž po zadání přihlašovacích údajů je možné ihned posílat příkazy. Vyobrazená sekvence vyčítá pouze souřadnice specifickými příkazy [4] nalezené předlohy dle koordinátů zjištěných přes InSight Explorer v zobrazení "Spreadsheet" a získává zpět požadované souřadnice se stavovými kódy provedení [4].

Vyčítání hodnot proměnných z jobu není jedinou vlastností komunikace přes Telnet. Existují příkazy pro ovládání základních parametrů kamery, stejně tak jako příkazy pro práci s joby. Kamera obsahuje vnitřní non-volatilní paměť, kde je možno uložit větší množství úzce zaměřených jobů, které je nutné příkazem načíst do RAM paměti, vykonat a extrahovat výsledky. Tento postup musíme realizovat kvůli omezené velikosti jednotlivých jobů. Zcela analogicky a za použití správných příkazů můžeme data z tabulky nejen vyčítat, ale i ukládat nová a měnit. Tím docílíme schopnosti použít jakýkoliv externí program podporující komunikaci přes Telnet i v případě další inspekce.

1.3 Robustnost detekčních algoritmů

Schopnost detekčních algoritmů rozlišit vzorek v pořádku a zmetek výrazně kolísá vlivem kvality vstupních dat. Toto kolísání se dá omezit používání sofistikovaných návrhových pravidel pro veškeré používané obrazové operace. Nejdříve je potřeba si definovat testovací podmínky, které mohou nastat při používání inspekčního systému v reálném provozu. Z experimentů s dodanou kamerou InSight 7010C jsem zjistil působení základních rušivých vlivů, které by mohly ovlivnit schopnost inspekce. Pokud uvažujeme stálou vzdálenost testovacích vzorků od objektivu, pak můžeme z testování vyloučit možnost rozostření objektivu a další zkreslení obrazu objektivem, právě z důvodu nastavení systému na fixní vzdálenost a jeho přizpůsobení. Vlivy který je bohužel proměnlivý v průběhu dne je světlo.

1.3.1 Vlivy na funkčnost detekčních algoritmů

Jedním z rušivých vlivů majících negativní dopad na funkčnost systému by mohla být nekonzistentní intenzita ozáření v průběhu dne, způsobená například východem a západem slunce. Následky tohoto jevu by mohly být vcelku snadno odstraněny zvýšením clonové čísla na objektivu a též modifikací osvětlovacího systému inspekčního stanoviště na maximální vyzařování, přičemž změna okolního osvětlení by poté měla pramalý dopad na odraz záření z testovaného vzorku zpět do objektivu. Tento postup je bohužel jen částečně použitelný, protože vyšší clonění objektivu sice způsobí zvětšení hloubky ostrosti, s tím ovšem přijde na řadu ztráta velmi vysokých frekvencí v obrazu, což způsobí ztrátu obrazové informace v oblastech hustějších na hrany, díky čemuž ztratíme schopnost detekovat prostorově menší útvary.

Dalším z rušivých vlivů, které výrazným způsobem přispějí k nefunkčnosti inspekčních systému obecně jsou kombinace tmavých a světlých míst v obraze, myšleno jako vržený stín přes část sledované plochy. Při nastavení systému na automatickou expozici hledáme kompromis v hodnotách jasové funkce a dle nastavení měníme dobu expozice a rychlost elektronické závěrky. Tento postup je bez problému funkční, pokud obraz vykazuje konstantní jas nebo konstantní změnu jasu v celé zkoumané oblasti a v systému vždy dojde ke správnému nastavení těchto dvou kritických parametrů, kdy nedochází ve snímači k saturaci některých pixelů a naopak ke tmavým bodům. Pokud ovšem před systém postavíme již dříve zmíněnou kombinaci v obrazu světlo/stín, pak můžeme očekávat vývoj inspekce dvojím způsobem. Může dojít k automatické regulaci jasové funkce, která způsobí odstranění saturovaných míst, s tím ale tmavší místa ještě ztmavnou, což snižuje úspěšnost detekce běžných algoritmů bez předchozích úprav obrazu na nulu. Při nastavení automatické expozice na vyšší hodnoty, dojde v případě rozumného

postranního světla a vrženého stínu k omezení vlivu a přijatelnému ztmavení, avšak dodatečnými obrazovými operacemi můžeme obraz upravit do stavu hodného úspěšné inspekce. Jednou z funkcí která se umí vypořádat s tímto problémem je histogramová ekvalizace.

1.3.2 Zvýšení kontrastu

Metod zvyšování kontrastu je několik, já pro svou práci využívám histogramovou ekvalizaci, jejíž definice následuje. Pokud budeme mít černobílý snímek o L úrovních šedé barvy, pak pro pravděpodobnost $p(x_i)$ výskytu pixelů s úrovní šedé barvy právě i platí vztah

$$p(x_i)=\frac{k_i}{k} \qquad 0 \leq i < L \tag{1}$$

kde k_i je počet pixelů právě s úrovní i a k je celkový počet pixelů snímku. Vykreslením pravděpodobnosti získáme normalizoavný sloupcový diagram pravděpodobností výskytů všech úrovní šedé obsažených ve snímku. Dále je nutné definovat kumulativní distribuční funkci $c(i)$, která je tvořena součtem pravděpodobností výskytů úrovní šedi od nejnižší až po zkoumanou, v našem případě se jedná o vztah

$$c(i)=\sum_{j=0}^{i} p(x_j) \tag{2}$$

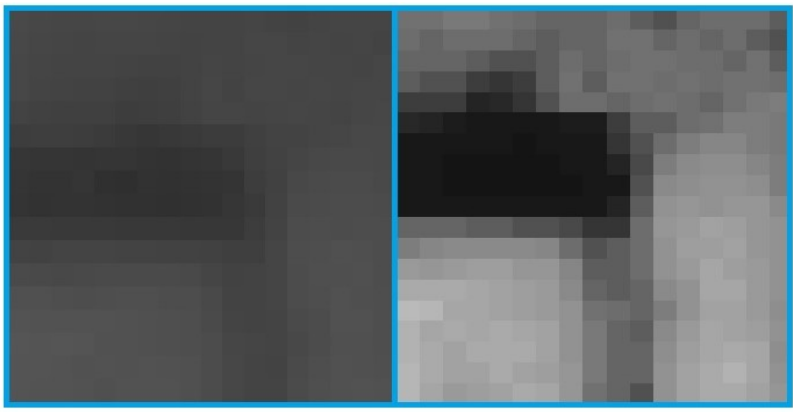
s pomocí distribuční funkce vytvoříme tranformační funkci f hodnot pixelů jednoduchým přepsáním do rovnosti, která respektuje linearizace distribuční funkce

$$y_i=f(x_i)=c(i) \tag{3}$$

výsledek funkce je snímek s hodnotami pixelů od nuly do jedné, a proto je nutné aplikovat další transformaci, která opět respektuje lineární charakter distribuční funkce

$$y'_i=y_i*R+x_{min} \tag{4}$$

kde R je rozsah hodnot pixelů původního snímku a x_{min} pak pixel s minimální hodnotou.



Obr. 1.11: Histogramová ekvalizace (vpravo)

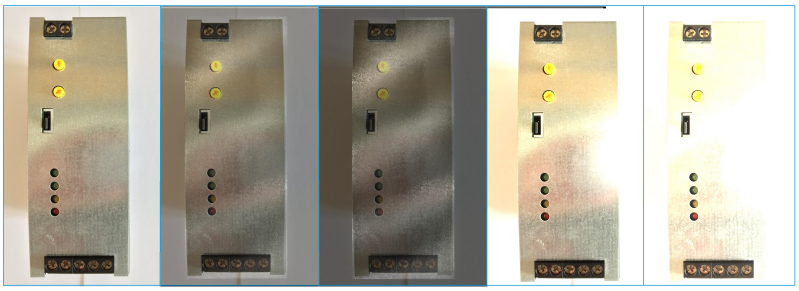
Na obrázků 1.11 můžeme vidět vliv histogramové ekvalizace na kontrast, právě z toho důvodu je velmi výhodné ekvalizaci používat, ovšem dle již dříve zmíněných důvodů s jistým omezením.

1.3.3 Simulace vnějších vlivů

Simulace vnějších vlivů je navržena tak, aby otestovala funkčnost algoritmů pro dva nejzávažnější vlivy: světlo/stín a postranní osvětlení. Testovací snímky jsou označeny vždy popisem rušivého jevu a číslovkou popisující úroveň vlivu tohoto jevu na ideální snímek. Zobrazení výsledků simulace ukazuje dvojsnímky s nižší úrovní vlivu rušivého jevu na levé straně. Snímky byly vytvořeny pomocí programu GIMP, kdy jednoduchou metodou překryvu vrstev byl regulován vliv. Pro první úroveň vlivu se jednalo o 30% překryv vrstvy, přičemž pro úroveň číslo dva to bylo již 70%.

Přední panel

Dle již v předchozí kapitole zmíněných rušivých vlivů jsem připravil sérii testovacích snímků na prověření odolnosti jednotlivých inspekčních algoritmů pro každou stranu vzorku. Vzorové snímky pro testování čelního panelu jsou na obrázku, přičemž ve všech dalších krocích bude testování zcela analogické, a tak nebudou další testovací snímky explicitně znovu vyobrazeny. Z levé strany jsou to pak konkrétně originální snímek, světlo stín ve dvou intenzitách, boční světlo ve dvou intenzitách. Předpokládám, že systém si lépe poradí s bočním světlem než s kombinací světlo/stín.

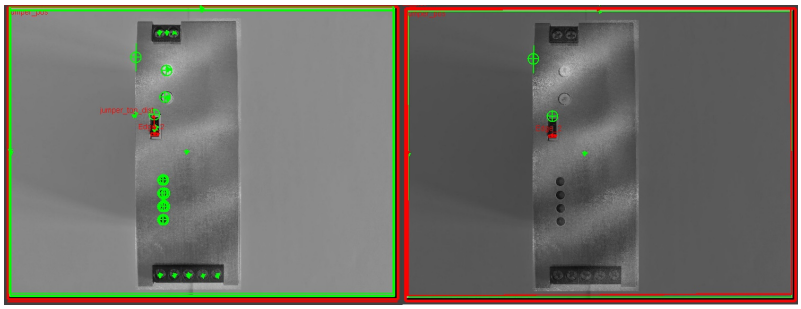


Obr. 1.12: Testovací sada snímků

Testování probíhá v prostředí InSight Cognex Explorer vždy načtením originálního snímku a kontrolou stoprocentní funkčnosti inspekčního programu. V dalším kroku je postupně testována schopnost programu přizpůsobit se simulovaným vnějším vlivům a vyhodnocení počtu chyb a abnormalit hlášených programem.

Obr. 1.13: Výchozí stav výstupu testovacího algoritmu pro čelní panel

Na obrázku 1.13 visí čelní panel testovaného vzorku s pozitivní inspekcí všech kroků algoritmu vyobrazených zelenou barvou v místech identifikace svorkovnic, fixačních částí a nastavovacích komponent. Dle následujícího obrázku 1.14 pak můžeme odhadnout zhoršenou schopnost sekvence identifikovat kritické části testovacího vzorku a v případě druhé úrovně kombinace světlo/stín pak schopnost velmi zhoršenou.



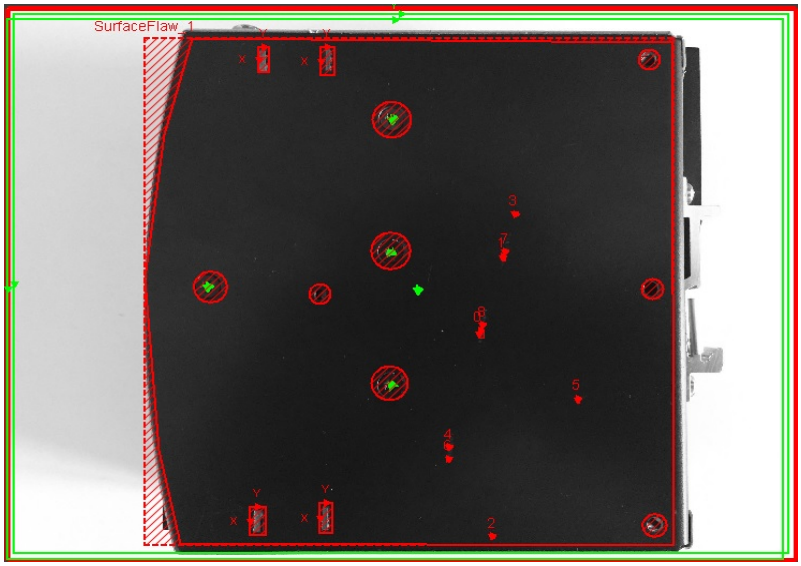
Obr. 1.14: Identifikace komponent vzorku pro světlo/stín

Dle předpokladů naprostá metla pro inspekční funkce se zdá býti střídání světlých a tmavých míst. Testování vlivu postranního osvětlení na práci programu dopadlo dle očekávání o mnoho lépe, dle výstupů na obrázku 1.15 z programu InSight Explorer usuzuji, že funkce byly účelově navrženy tak, aby eliminovaly vady expozice.

Obr. 1.15: Identifikace komponent vzorku pro postranní osvětlení

Boční kryt

U boků je situace s inspekcí s proměnlivými podmínkami poněkud složitější, protože hlavní výkonnou funkcí je zde dříve popsaná funkce Surface Flaws, která detekuje nespojitosti určené

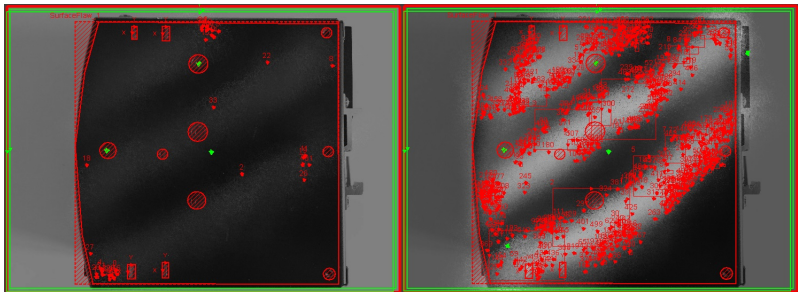


Obr. 1.16: Výchozí stav výstupu testovacího algoritmu pro boční panel

plochy s ohledem na jas, změnu kontrastu a barvy. Právě z toho důvodu předpokládám mizivou funkčnost detekčních algoritmů pro obě boční strany.

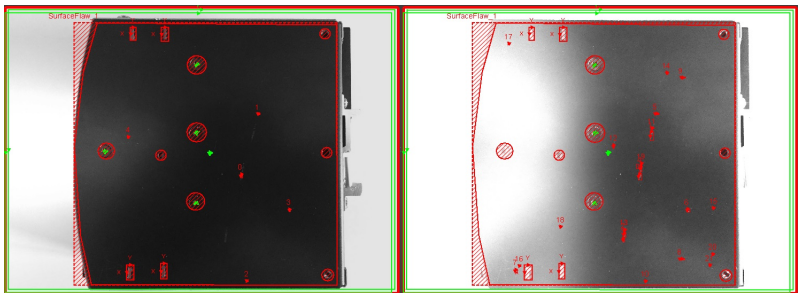
Na obrázku 1.16 vidíme detekci chyb povrchu jako jsou škrábance, zbytky lepidla a jiné nedokonalosti. Ve výchozím stavu algoritmus detekuje přítomnost boční strany a fixačních prvků, dále maskuje fixační prvky a další opticky nestejné části povrchu, aby mohla proběhnout již zmíněná kontrola. Dle předpokladů budeme zde vyhodnocovat schopnost třech funkcí a vyvodíme z toho procentuální úspěšnost této části algoritmu.

Dle předpokladu došlo k výraznému nárůstu detekovaných chyb, což v případě aditivní kontroly nevadí, protože vzorek bude stejně vyřazen. V případě snahy o stoprocentní určení vad povrchu touto metodou je jedinou možností uzavřená stanice s kontrolovaným osvitem, popřípadě velmi silný zdroj záření poblíž kamery.



Obr. 1.17: Inspekce vad povrchu pro kombinaci světlo/stín

Při pohledu na výsledky pod vlivem bočního osvitu na obrázku 1.18 můžeme konstatovat opět podobné hodnocení jako v předchozím případě, dokonce i funkce Surface Flaws nezaznamenala nárůst chyb ve stovkách procent, což je při dané kvalitě snímku obdivuhodné. Zřetelně je pak vidět adaptabilita funkce v případě plošné změny jasu, kdy si funkce zachovává takzvanou „chladnou hlavu“ a nevyhodnocuje celý povrch jako vadu o velké ploše.

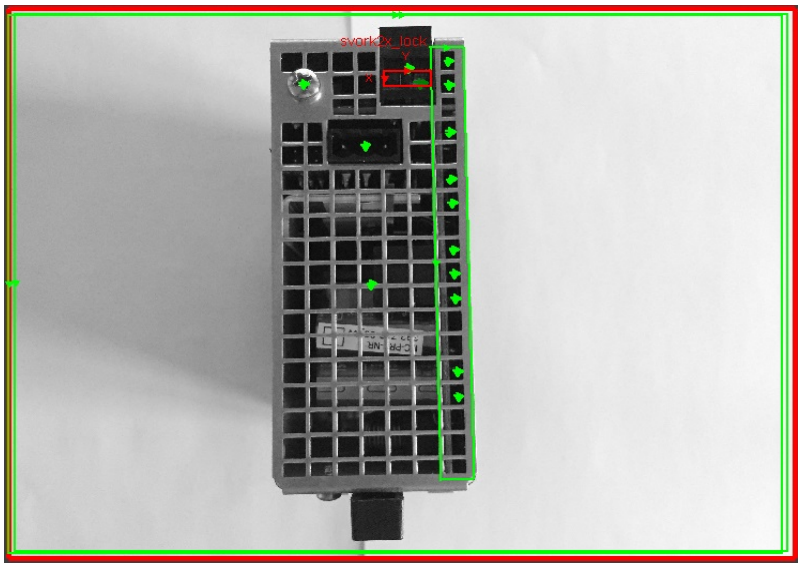


Obr. 1.18: Inspekce vad povrchu pro postranní osvětlení

Samozřejmě pokud by saturovaná oblast pokrývala výraznou část testovaného povrchu, pak lze prohlásit výsledky inspekčního algoritmu jako nesprávné.

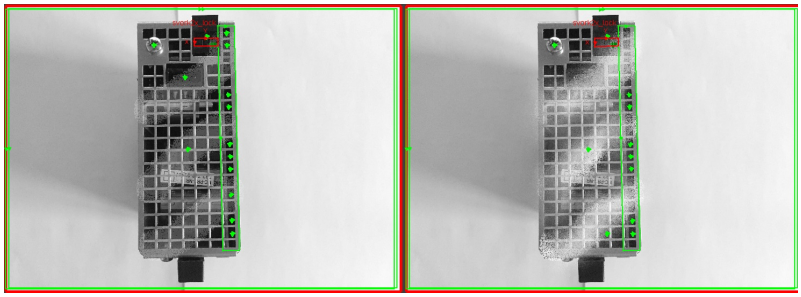
Spodní/vrchní kryt

V případě testování inspekčních algoritmů spodního nebo vrchní krytu je situace opět jednodušší, v zásadě jde pouze o kontrolu funkce hledání předloh, popřípadě jejich počítání. Funkčnost jsem testoval opět v procentuálním měřítku při průběhu algoritmu, s ohledem na všechny použité kroky.



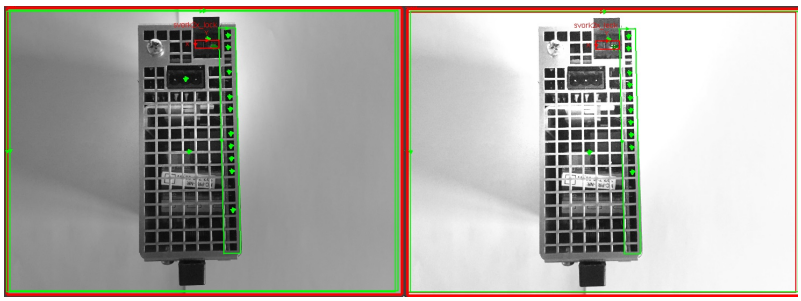
Obr. 1.19: Výchozí stav výstupu testovacího algoritmu pro spodní/vrchní kryt

V této části jde o kontrolu přítomnosti spodního nebo vrchního krytu, fixačního prvku, svorkovnice a jejího stavu, konektoru a izolační podložky pod DPS.



Obr. 1.20: Identifikace komponent vzorku pro kombinaci světlo/stín

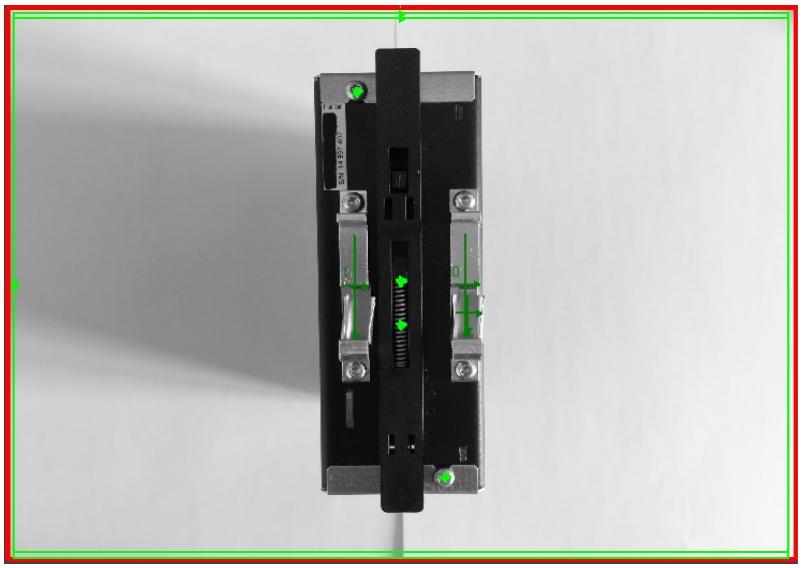
Odolnost tohoto algoritmu se zdá být o mnoho vyšší v případech testování kombinace světlo/stín než v předchozích případech. Nejspíše je to kvůli preciznímu nastavení oblastí pro inspekci, ve kterých se zásadním způsobem neprojeví dramatická změna jasu, nicméně stále algoritmy nedisponují ideální odolností vůči těmto jevům. V případě posuzování vlivu postrannního osvětlení na spodní nebo vrchní část jsem dospěl ke zjištění o výrazném ovlivnění funkce Pattmax Redline v závislosti na intenzitě osvitu.



Obr. 1.21: Identifikace komponent vzorku pro postranní osvětlení

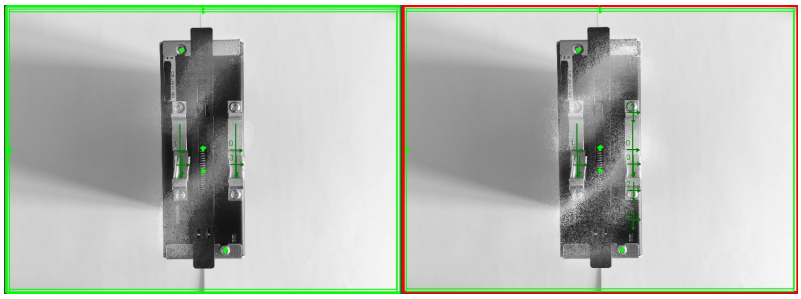
Zadní montáž

Při kontrole odolnosti jobu pro inspekci zadní strany testovacích vzorků dochází opět k ověřování robustnosti funkce Patmax Redline. V základu jsou identifikovány části: přítomnost zadní montáže, všechny fixační prvky a uchycení na DIN lištu s veškerým příslušenstvím. Podmínky testování jsou stále stejné, výsledky ovšem nikoliv.



Obr. 1.22: Výchozí stav výstupu testovacího algoritmu pro zadní montáž

Dle obrázku 1.23 můžeme vidět nezanedbatelný vliv kombinace světlo/stín na funkčnost algoritmů, kdy začalo docházet k falešným detekcím předloh v místech, kde nebyly a na obrázku 1.24 dokonce velmi výrazný vliv postranního osvětlení při vyšších intenzitách.



Obr. 1.23: Identifikace komponent vzorku pro kombinaci světlo/stín

Bohužel simulace postranního osvětlení se pro druhou intenzitu vymkla kontrole, provedl jsem proto mírnou úpravu limitů funkce hledání předlohy zadní montáže v domnění obnovení funkčnosti. Výsledek byl takový, že funkce sice dopočítala chybějící předlohu a správným způsobem ji označila jako nález, avšak u dalších zřetelně vyobrazených částí selhala.

Obr. 1.24: Identifikace komponent vzorku pro postranní osvětlení

Ze simulací jsem vygeneroval tabulku celkové úspěšnosti veškerých testovaných funkcí pro celkem 5 testovacích snímků na job. V celkové úspěšnosti uvažuji funkčnost všek důležitých kroků optické inspekce s přihlédnutím na částečnou funkčnost při extrémním vlivu rušení.

Tabulka 1: Celková úspěšnost algoritmů Cognex

Část	**Světlo/stín 1**	**Světlo/stín 2**	**Boční světlo 1**	**Boční světlo 2**
Přední panel	95,00 %	13,00 %	99,00 %	83,00 %
Boční panel	47,00 %	47,00 %	83,00 %	74,00 %
Spodní/vrchní kryt	95,00 %	90,00 %	82,00 %	65,00 %
Zadní montáž	93,00 %	85,00 %	99,00 %	57,00 %

Při pohledu na tabulku 1 vidíme výsledky působení testovacích vlivů na schopnost algoritmů správně detekovat všechny části vzorku. Komentář k návrhu a optimalizaci algoritmů pro správnou funkci je zapsán v závěru práce, jakožto i postřehy návrhových pravidel z experimentů.

2 Matlab

2.1 Představení

Matlab je výkonné programovací prostředí primárně určené pro práci s maticemi a poli, avšak v současné době existuje velké množství nástrojů a funkcí pro zpracování obrazu přímo naprogramovaných v tomto prostředí, a proto jsem se rozhodl založit konkurenční algoritmus právě na programování v Matlabu s využitím sofistikovaných prostředků.

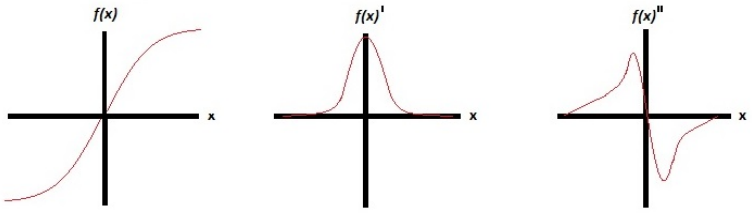
V jednoduchosti lze napsat, že postupnou úvahou a konzultacemi s vedoucím práce byl vyvinut postup operace s obrazovými daty pro dosažení optimálních výsledků. Celý algoritmus je založen na rozpoznávání předloh a jejich lokalizaci podobně jako v předchozí kapitole, avšak s použitím neuronové sítě. Pro jednoznačné určení předlohy ve zkoumaném snímku je nutné místo obrazových dat použít takzvaný deskriptor. Ten nám zaručí jednoznačnost identifikace a omezí vliv šumu a dalších nechtěných vlivů okolního prostředí. Jako deskriptor je použit HOG. Primární snahou se zdá být naučení neuronové sítě na právě jednoznačné deskriptory jednotlivých zkoumaných vzorů. Pro správnou funkci neuronové sítě jako klasifikátoru, která se rozhoduje za pomoci porovnávání deskriptorů, je nutné založit pro každý specifický deskriptor vlastní třídu. Výstupem naučené neuronové sítě při reakci na vstupní podnět v podobě části obrázku, která obsahuje deskriptor shodný s minimálně jednou třídou vzorů, je pravděpodobnost určující přiřazení podnětu právě do této třídy. Situace se zdá být jednoznačná a jednoduchá. Neuronová síť pracuje spojitě, a proto je velmi obtížné získat zcela stoprocentní pravděpodobnost přiřazení. V následujících podkapitolách popíši princip návrhu a strukturu neuronové sítě, všechny potřebné podpůrné algoritmy a doložím dosažené výsledky.

2.2 Hranové detektory

Jedním ze základních stavebních prvků mnoha obrazových operací stále zůstává detekce hran, a to především kvůli množství informace, které jsme schopni z hranového obrázku vyčíst. Hrany vytyčují oblasti, ve kterých se vyskytují předměty stejně tak, jako mění svou četnost v závislosti na vzdálenosti předmětů ve snímku. Pomocí hranového obrázku lze zkoumat různé nespojitosti na předpokládaně spojitém povrchu a vyhodnocovat tak jeho defekty. Detekci hran, stejně tak jako ostatní obrazové operace, je možno provádět více způsoby dle nároků na výsledek, parametrů okolního prostředí, požadavků na výpočetní čas a mnoho dalších faktorů. Proto zde představím nejpoužívanější z detektorů založených na první derivaci jasové funkce obrázku v Matlabu a popíši jejich principy.

2.2.1 Hrana

Nejdříve je nasnadě definice hrany a jejich vlastností. Pokud uvažujeme černobílý jasový snímek, pak hrana je místo se skokovou změnou jasu neboli změnou hodnoty jasové funkce f(x). Pro určení pozice hrany v obraze se v Matlabu používají hranové detektory založené na první nebo druhé derivaci jasové funkce, přičemž pro eliminaci falešných detekovaných hran je vhodné použít metodiku odstranění šumu, popřípadě jiný postup. Zde prezentované detektory spolupracují s šumovým Gaussovým filtrem a jsou založené na principu konvoluce s konvolučními jádry jednotlivých typů detektorů.



Obr. 2.1: Zleva hrana v jasové funkci f(x) , první derivace a druhá derivace

Dle obrázku 2.1 je zřejmé, že detektory pracující s první derivací jasové funkce budou hledat lokální maxima a dále dle nastaveného prahu rozhodování určí hranu. U detektorů založených na druhé derivaci je situace poněkud složitější, protože nelze nastavit rozhodovací mechanismus na detekci druhé derivace při průchodu nulou vzhledem k tomu, že výsledek druhé derivace je roven nule i pro veškeré nehranové oblasti zkoumaného snímku. Je tedy nutností zkoumat okolí domnělé hrany a zajistit, aby mechanismus určil za hranu místo se skokovou změnou hodnoty druhé derivace z kladné do záporné hodnoty nebo naopak.

Dalším problémem může být nejasná nebo několikanásobná hrana projevující se v první derivaci jako vícero lokálních maxim nebo minim. Pro eliminaci nechtěných hran můžeme použít šumový filtr na původní obrázek a v případě velkého množství hran i funkci prahování obrázku, která normuje hodnoty jasové funkce v intervalu <0,1>, což výrazným způsobem omezuje rozlišovací schopnost následné hranové detekce, avšak zároveň eliminuje nechtěné hrany.

V Matlabu se k všemožným operacím se snímky, ať již právě detekce hran nebo použití nějakého filtru využívá konvoluce s konvolučními jádry dotyčné funkce. Pro snímky převedené Matlabem do matic poté platí pro konvoluci vztah

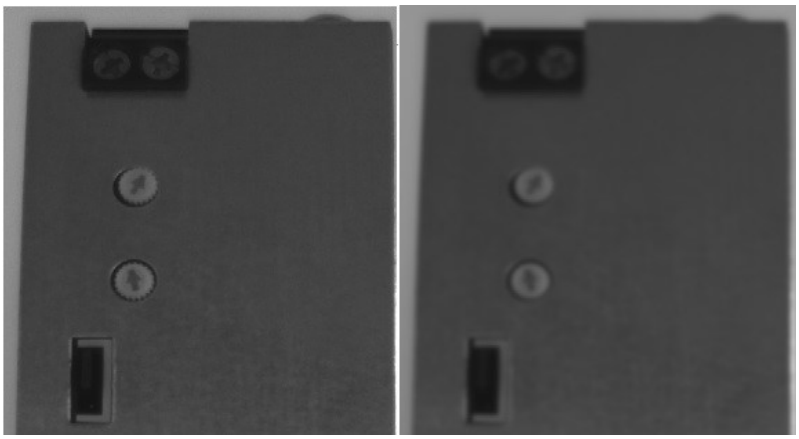
$$f_x * g = \sum_{i=-\infty}^{+\infty} f_{(x-i)} \cdot g_i \tag{5}$$

kde f(x) je hodnota jasové funkce v bodě x a g pak konvoluční jádro. Ze vztahu je zřejmé, že parametry konvolučního jádra zcela definují konečný vliv na výslednou hodnotu v bodě x po konvoluci. Při snímkových operacích jsou použity jako základní struktury matice, proto konvoluční jádro i jasová funkce budou označeny velkým písmenem.

Ve většině hranových detektorů se používá nějaká filtrace šumu pro omezení detekce falešných hran. Jedním z nejjednodušších šumových filtrů je Gaussův filtr založený na konvoluci vhodného konvolučního jádra ať už se zkoumaným snímkem nebo již některým z jeho polotovarů jdoucích k hranovému obrázku. Jednoduché konvoluční jádro může vypadat následovně

$$G_{Gauss} = k * \begin{matrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{matrix}$$

kde k je konstanta udávající míru filtrace v absolutním měřítku a samotná velikost matice poté určuje filtrací ovlivněné okolní pixely. Pro zde prezentovanou Gaussovu matici bývá konstanta k typicky mnohem menší než jedna, protože by jinak nedocházelo k omezení lokálních maxim jasové funkce ve snímku. Výsledný filtrovaný snímek k porovnání na obrázku 2.2.



Obr. 2.2: Vliv Gaussova filtru na snímek (vpravo)

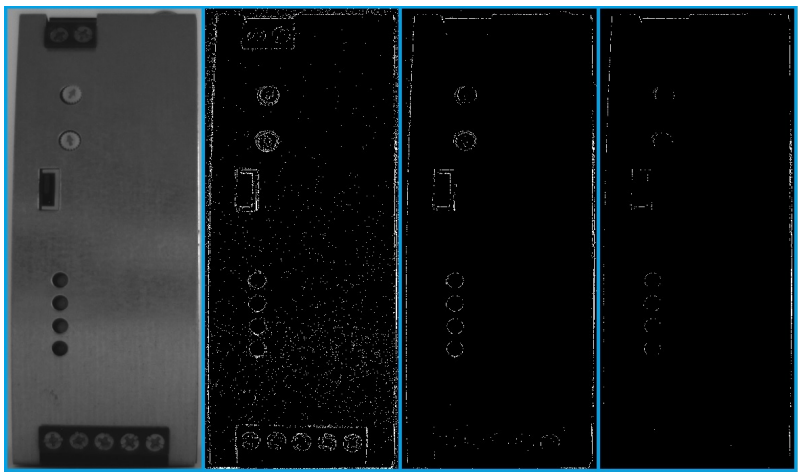
2.2.2 Roberts

Jedním z nejjednodušších detektorů operujících na první derivaci jasové funkce snímku je právě Robertsův detektor, jehož detekce je založená na konvoluci jasové matice snímku s maticí jádra detektoru

$$G_{ver} = \begin{pmatrix} -1 & 1 \end{pmatrix} \qquad G_{hor} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

kde G_{ver} slouží pro detekci vertikálních hran a G_{hor} k detekci horizontálních hran. Jisté vlastnosti tohoto detektoru se dají odvodit již z jádra, které je prakticky nejmenší možné, aby mohla konvoluce přinést nějaké výsledky. Minimální zkoumané okolí předpokládané hrany způsobí přecitlivělost detektoru na hrany způsobené šumem a nedostatečnou citlivost na neostré hrany, například při použití Gaussova šumového filtru.

Na obrázku 2.3 je vidět ohromná šumová citlivost Robertsova detektoru a její změna s volbou prahovací konstanty. Čím menší prah je zvolen, tím více je výsledný hranový obrázek ovlivněn šumem a nevýraznými hranami.



Obr. 2.3: Hranový obrázek Robertsova detektoru pro různé prahové hodnoty

2.2.3 Sobel

Pokročilejší Sobelův detektor využívá větší okolí zkoumané hrany a proto dosahuje lepších výsledků při stejných podmínkách oproti Robertsovu detektoru. Princip provádění detekce hran je analogicky stejný, ovšem díky rozšířenému zkoumanému okolí lze na zkoumaný snímek použít šumový filtr bez větších problémů při detekci hrany. Konvoluční jádra detektoru jsou pak matice

$$G_{ver}=\begin{matrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{matrix} \qquad G_{hor}=\begin{matrix}-1 & -2 & -1\\ 0 & 0 & 0\\ 1 & 2 & 1\end{matrix}$$

kdy dle indexů se opět jedná o konvoluční jádro pro hledání vertikální a horizontální hrany, zde ovšem jádrem definujeme smysl hrany (sestupná, vzestupná). Svými rozměry umožňuje toto jádro zcela jednoduchou úpravou dosáhnout toho, že detektor bude detekovat hrany v osmi směrech, avšak pro každý z těchto směrů je potřebné "jiné" jádro. Slovíčko jiné v uvozovkách je z důvodu toho, že jádro projde zcela minimální změnou hodnot, kdy dojde k netradiční rotaci matice dle středové hodnoty viz.: jádro konvoluce pro vertikální hranu vzestupnou zleva a jádro konvoluce pro horizontálná hranu vzestupnou shora. Hledaná hrana má vždy stejný smysl jaké zobrazují nulové hodnoty v konvolučním jádru.

Obrázek 2.4 zobrazuje vliv prahové hodnoty na detekci šumu a nevýrazných hran, přičemž Sobelův detektoru vychází ve srovnání s předchozím detektorem lépe, právě pro volbu konvolučního jádra s větším rozměrem.

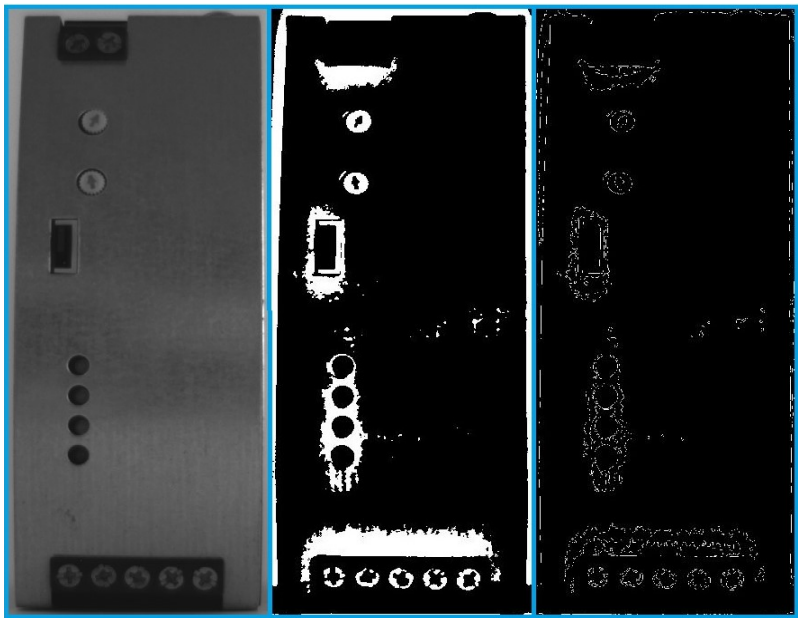
Obr. 2.4: Hranový obrázek Sobelova detektoru pro různé prahové hodnoty

2.2.4 Canny

Zcela specifickým případem detektoru hran založeném na první derivaci je Cannyho detektor, kde se při detekci hran postupuje odlišně. Prvním krokem detekce bývá zpravidla odstranění šumu konvolucí s Gaussovou maticí, dále se aplikuje funkce tresholding neboli prahování, která pomáhá potlačit oblasti bez hrany v závislosti na nastavené prahové hodnotě. Výsledkem zpravidla bývá binární snímek dle přepisu

$$f(x)<L \rightarrow f_{bin}(x)=0$$
$$f(x)>L \rightarrow f_{bin}(x)=1$$

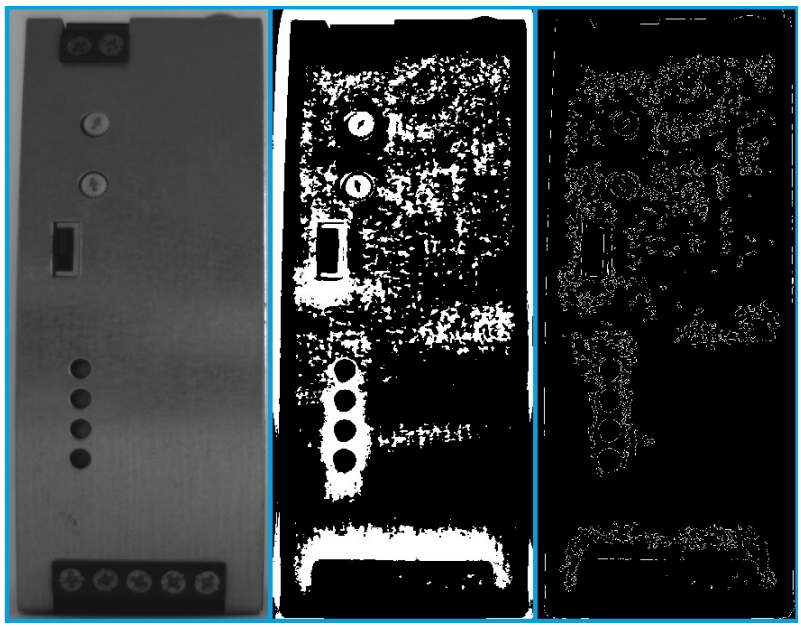
kde $f(x)$ reprezentuje hodnotu jasové funkce pixelu x, L je prahová hodnota a $f_{bin}(x)$ pak výsledná hodnota pixelu x po binarizaci.



Obr. 2.5: Cannyho detektor

Posledním krokem je poté zkoumání první derivace prahovaného binárního snímku a rozhodnutí zda je přítomna hrana nebo ne. Díky filtraci šumu velmi vzroste odolnost tohoto detektoru na falešné detekce, a proto jsem se jej rozhodl využít při konstrukci inspekčního systému.

Dle obrázků 2.5 vidíme celý postup detekce hran Cannyho detektorem: vlevo původní snímek po průchodu šumovým filtrem, uprostřed binární snímek a vpravo hranový snímek. Postup je nutné optimalizovat pro definované vlivy působící na vstupní data, v ideálním případě dosáhnout již při binarizaci odstranění nedůležitých vzorů ve snímku.



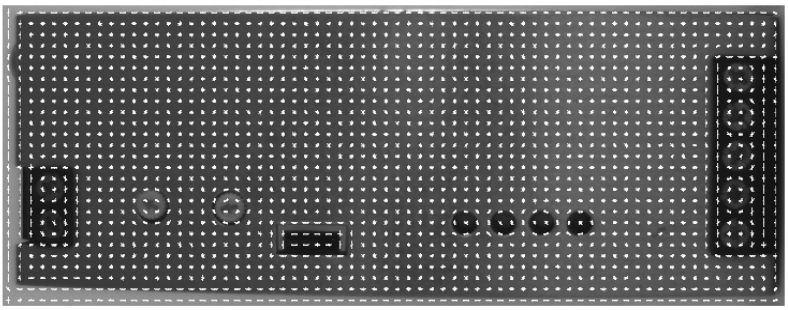
Obr. 2.6: Cannyho detektor s využitím histogramové ekvalizace

Velký rozdíl při detekci hran hraje i jakákoliv předchozí úprava, které se zdá se na pohled správná, ale s katastrofálním výsledkem. Příkladem může být hojně využívaná histogramová ekvalizace kontrastu, kdy dochází k rovnoměrnému rozprostření jasové intenzity přes všechny pixely nezávisle na stupni šedi a individuálního původního jasu. Tento postup je výhodný v případě že původní snímek je přesvícený v určitých místech nebo naopak tmavší než by měl být. Bohužel pro mojí aplikaci přináší spoustu problémů, zvláště pak nadbytečný šum, které jsem odstranil použitím Gaussova filtru již v původním snímku. Dopad histogramové ekvalizace na detekci hran je na obrázku 2.6.

2.3 Deskriptor – HOG

Při operacích s velkým množství obrazových dat vznikají ohromné nároky na výpočetní výkon a dobu výpočtu, která může v případě nasazení v průmyslovém provozu zapříčinit zpomalení celé výrobní linky. Právě proto je nutné optimalizovat inspekční algoritmy pro zkrácení doby běhu, ale také omezením zpracovávaných dat. V případě snímku o rozměrech 2187x870 pixelů dochází při průchodu celého obrázku výpočetním mechanismem k zpracování 1,9 milionu hodnot.

Optimalizace vyhodnocování obrovského počtu hodnot za účelem urychlení běhu programu je možná pomocí deskripce snímku HOG. Celý postup deskripce snímku je následující. Snímek je nejříve rozdělen do takzvaných buněk v nichž je dle hran a jejich gradientů určen histogram směrů těchto gradientů. Buňky se dále skládají do bloků, které se v obraze vždy určitou plochou překrývají, takže některé buňky jsou součástí až čtyř bloků. Po blocích dochází taktéž k normalizaci hodnot kvůli odolnosti deskriptoru vůči rušivým vlivům jako jsou stíny nebo přesvětlená místa. Výstupem deskripce je pak vektor popisující všechny hranové gradienty bloků.



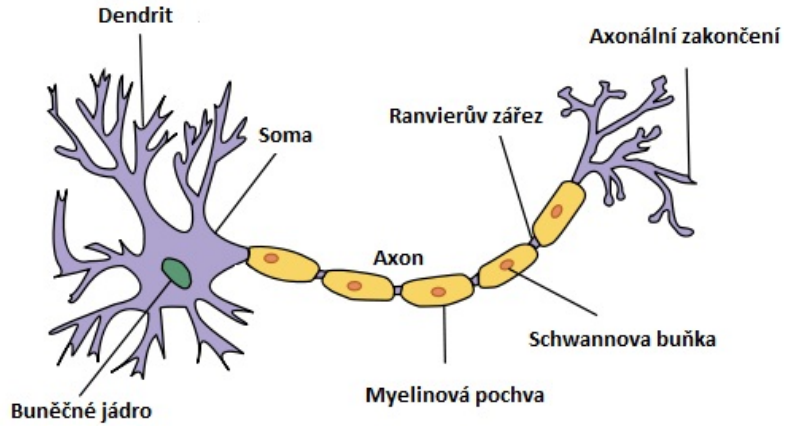
Obr. 2.7: Vizualizace deskriptoru HOG

Na obrázku 2.7 je vyobrazena vizualizace deskriptoru HOG, který je popsán vektorem délky 62716 míst generovaným ze struktury buněk 32x32 pixelů. Pokud tedy použiji při popisu obrázku s vysokým rozlišením deskriptor, omezím velmi výrazně množství zpracovávaných dat při vyhodnocování a správnou konfigurací velikostí buněk taktéž zamezím výrazné ztrátě informace. Vždy platí rovnost: menší bloky k vyhodnocení = více detailů a vyšší časová náročnost.

2.4 Klasifikátor – neuronová síť

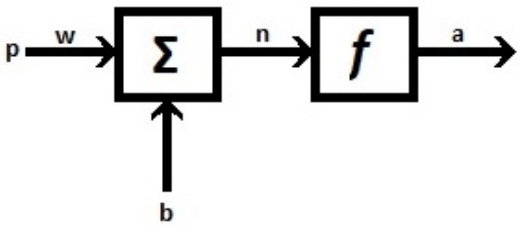
2.4.1 Neuron

Neuron je speciální druh buňky nervové soustavy, která je schopná reagovat na vnější podněty a dle vnitřních stavů sdílet informaci o vjemu s dalšími buňkami stejného typu. Hlavními částmi těla (soma) neuronu jsou pak: jádro obsahující DNA, dendrity slouží jako vstupy vzruchu a axon neboli dlouhý výběžek šířící vzruchy směrem ven z neuronu. Struktura axonu definuje rychlost šíření vzruchů, především elektricky buzené vzruchy profitují při šíření axonem Schwannovou buňkou z vytvořeného izolantu zvaného Myelinová pochva, kdy na tenkém přechodu mezi jednotlivými pochvami dochází k obnovení intenzity vzruchu. Toto místo se nazývá Ranvierův zářez. Axonální zakončení dále distribuují vzruchy mezi jednotlivými neurony přes dendrity dalších neuronů. Dle propojení neuronů dokáže celá neuronová síť fungovat všemožnými způsoby, proto je snaha napodobit matku přírodu vytvořením výpočetní jednotky s podobnými vlastnostmi a zkoumání vlivu propojení těchto základních stavebních jednotek na způsob řešení problémů.



Obr. 2.8: Neuron [7- upraveno]

2.4.2 Model neuronu v Matlabu



Obr. 2.9: Jednoduchý model neuronu v prostředí matlab dle [8]

Dle biologické předlohy tvoří model neuronu v Matlabu základní jednotku při budování neuronových sítí. Obsahuje vstupní část, kde je každá vstupní hodnota p ovlivněna váhou w a

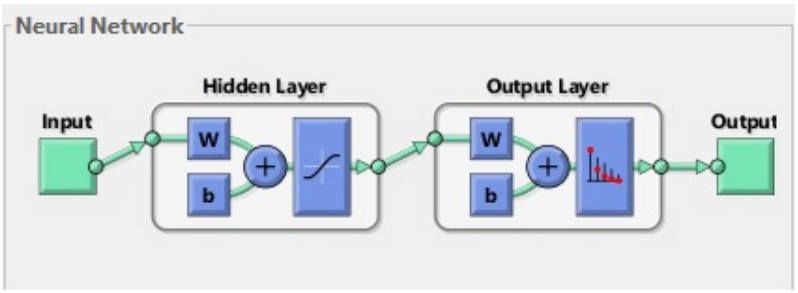
sečtena s vnitřním stavem neuronu reprezentující boční vstup b. Výstup a je poté ovlivněn nastavenou přenosovou funkcí f. Model pak realizuje změnu vstupních dat na výstupní dle vztahu:

$$a = f(w * p + b) \tag{6}$$

Zcela analogicky funguje model s více vstupními hodnotami, vždy dochází k váhování každé vstupní hodnoty zvlášť a následný součet výsledků za přispění vnitřních stavů. Přenosová funkce na výstupu neuronu bývá nejčastěji lineární, pro speciální případy sítí pak může být přenosová funkce na výstupu logaritmický sigmoid nebo jiná.

2.4.3 Struktura neuronové sítě v Matlabu

Při návrhu neuronové sítě v Matlabu můžeme využít návrhového průvodce pro specifickou síť s požadavky na určitou funkci. Pro rozpoznávání naučených předloh průvodce nabízí „Feed-forward pattern net“ což je topologie sítě určená primárně pro zpracování a porovnávání předloh. Struktura sítě je na obrázku 2.10, kdy vstupní a výstupní bránu zastává vrstva neuronů. Výstupní data jsou ovlivňována hodnotami vnitřních stavů ve skryté neuronové vrstvě a křivkou výstupní přenosové funkce.



Obr. 2.10: Feed-forward neuronová síť v Matlabu [9]

Takto jednoduchá topologie sítě je vhodná pro práci s menšími objemy dat, a proto dle doporučení vedoucího diplomové práce využívám pro rozpoznávání předloh ve snímcích s vysokým rozlišením neuronovou síť s pěti vnitřními vrstvami přičemž každá z nich obsahuje několik stovek neuronů. Taková struktura výrazným způsobem přispěje ke spolehlivosti klasifikace předloh ve snímku, bohužel ale bude mít za následek prodloužení doby potřebné k provedení procesu ovlivnění vstupních dat.

2.5 Podpůrné algoritmy

Při používání specifických funkcí je potřeba pracovat se specifickými daty a právě z toho důvodu existují podpůrné funkce, kterými jsou ve většině případů vstupní data, u funkcí které to požadují, ovlivněna. Nejčastěji se jedná o omezení vstupní množiny dat, popřípadě vyloučení funkcí nezpracovatelných výskytů.

2.5.1 Otsu metoda

Pro jakékoliv následující obrazové operace jako je binarizace nebo detekce hran, je potřeba zjistit prahovou hodnotu pro konkrétní vstupní data. Hodnota prahu může být volena jako

globální pro celý snímek nebo lokální pro různé ROI, avšak v praxi nelze vždy použít pevně navolenou hodnotu prahu kvůli možnému odprahování důležitých bodů. Pro tyto případy se používá metoda Otsu založená na zkoumání histogramu.

Pokud uvažujeme šedotónový snímek ve kterém se vyskytuje jasný objekt a tmavé pozadí, pak můžeme v histogramu nalézt dvě křivky Gaussova rozdělení pixelů do jednotlivých úrovní šedé barvy, přičemž jedna reprezentuje bod v úrovních sestavující objekt a druhá poté body z pozadí. Ideální práh by se dal jednoduše určit jako lokální minimum histogramu, bohužel téměř vždy se jednotlivé křivky překrývají a spojují, a proto se ideální práh nachází někde ve společné části.

Řešení tohoto problému je přenecháno statistické metodě optimalizace rozptylů a to jak vnitřního rozptylu (internal) úrovní pro popředí i pozadí, tak i mezi-rozptylu (among) úrovní popředí od pozadí. Optimalizace probíhá pomocí následujících vztahů

$$\sigma^2_{internal}(L) = n_B(L) * \sigma^2_B(L) + n_F(L) * \sigma^2_F(L) \tag{7}$$

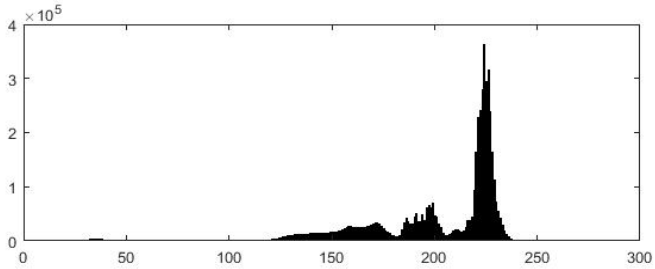
$$\sigma^2_{among}(L) = n_B(L) * n_F(L) * [\mu_B(L) - \mu_F(L)]^2 \tag{8}$$

kde $\sigma^2_B(L)$ je rozptyl pozadí, $\sigma^2_F(L)$ rozptyl popředí, $\mu_B(L)$ průměrnou hodnotu intenzity popředí, $\mu_F(L)$ průměrnou hodnotu intenzity pozadí, $n_B(L)$ a $n_F(L)$ jsou pak váhovací koeficienty popředí a pozadí rovné

$$n_B(L) = \sum_{i=0}^{L-1} p(i) \tag{9}$$

$$n_F(L) = \sum_{i=L}^{N} p(i) \tag{10}$$

pro světlejší popředí a tmavší pozadí, kde N je počet úrovní šedotónového snímku a $p(i)$ pravděpodobnost výskytu bodu úrovně i.



Obr. 2.11: Histogram testovacího snímku

Cílem optimalizace je nalézt co nejmenší vnitřní rozptyl jednotlivých úrovní intenzity v histogramu tak, aby bylo možné spravedlivě rozlišit popředí a pozadí. Na obrázku 2.11 je skutečný histogram testovacího šedotónového snímku, kde lze jasně definovat popředí jako křivku nacházející se v co nejvyšších hodnotách jasu, vše ostatní je pozadí popřípadě šum.

2.5.2 Binarizace

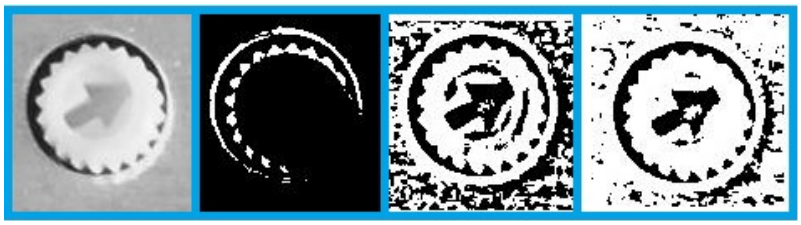
Jak již název napovídá, funkce binarizace převádí data do množiny z intervalu <0,1> a to právě v případě zpracování obrazových dat přináší mnoho pozitiv i negativ. Provedením této operace

dosáhneme výrazné čitelnosti obrazových dat pro program, avšak záleží na jejím nastavení. Naštěstí v Matlabu můžeme provést binarizaci pomocí funkce imbinarize(), která je velmi konfigurovatelná narozdíl od obyčejného prahování.

Jak již bylo zmíněno, funkce pracuje dle předpisu

$$f(x) < L \rightarrow f_{bin}(x) = 0$$
$$f(x) > L \rightarrow f_{bin}(x) = 1$$

kde L je nastavená prahová hodnota, která se počítá použitím Otsu metody. Ohromná výhoda počítání prahu právě touto metodou spočívá v možnosti nastavit funkci prostorovou citlivost, respektive co má funkce požadovat za popředí a co za pozadí. Díky této vlastnosti dosahuje metoda zvláště v nemnoho kontrastních datech velmi dobrých výsledků.



Obr. 2.12: Vliv nastavení výpočtu prahové hodnoty na výstupní binární snímek

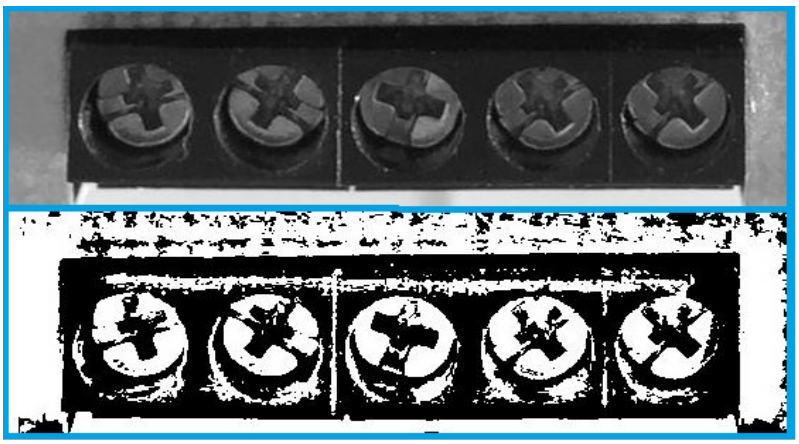
Dle nastavení citlivosti, funkce operuje na různě velkých oblastech při výpočtu prahové hodnoty a určuje dále popředí a pozadí snímku pro lepší výsledek. Funkce imbinarize umožňuje nastavit zda popředí je v zásadě světlejší než pozadí a naopak, včetně případu kdy to není možno jistě určit a toho lze využít například při zpracovávání tištěného textu, protože v základu jsou brány při binarizaci jako popřední vždy světlé oblasti. Tvrzení vychází z logiky prahovací funkce, na které je binarizace založena.

Obrázek 2.12 ukazuje experimentální nastavení citlivosti v průběhu testování funkce imbinarize, kdy zleva je hodnota velmi vysoká, poté je snižována až na hodnotu 58%. Popředí je zde nastaveno jako tmavší než pozadí, proto dochází při vyšší citlivosti k odprahování důležité tmavé šipky na jinak světlé čepičce trimru. Snaha byla zachytit a úspěšně binarizací zvýraznit šipku ve středu nastavovacího trimru z důvodu následující shlukové analýzy pro určení natočení trimru.

2.5.3 Maskování

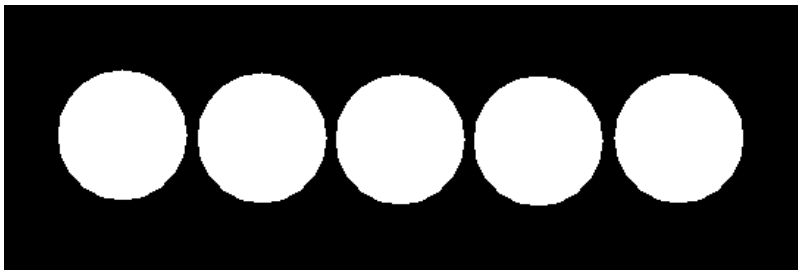
Velmi důležitou roli při zpracování datových polí a vynášení výsledků funkcí pro zájmové body pole hraje i jejich okolí. Bohužel pokud nechceme ručně nastavovat souřadnice dat pro analýzu každý běh programu a udržet zpracovávaná data v celistvém poli, tak nemáme mnoho jiných možností než právě maskování popřípadě volbu ROI pevně zadanými souřadnicemi.

Maskování v Matlabu lze jednoduše realizovat pomocí matice o stejných rozměrech jako mají obrazová data a s vhodně zvolenými hodnotami nul a jedniček. Následný postup k vymaskování splňuje funkce logického součinu polí, kdy dochází k vynulování veškerých pozic v poli, které korespondují s nulovými pozicemi masky. Samozřejmě tato operace je možná pouze s výsadně binárními daty, a proto je nutné snímek pro maskování nejdříve binarizovat.



Obr. 2.13: Výřez zkoumané oblastí a její binární snímek

O problémech s kritickým nastavením binarizace bylo pojednáváno v předchozí kapitole, proto zde již nebude rozebíráno. Po úspěšné binarizaci přestoupíme ke generaci masky. Při použití funkce maskování pomocí logického součinu generujeme masku s jedničkami v ROI. Pro správnou funkci je nezbytné, aby maska a maskovaný snímek měly stejné rozměry.



Obr. 2.14: Maska pro operaci maskování logickým součinem polí

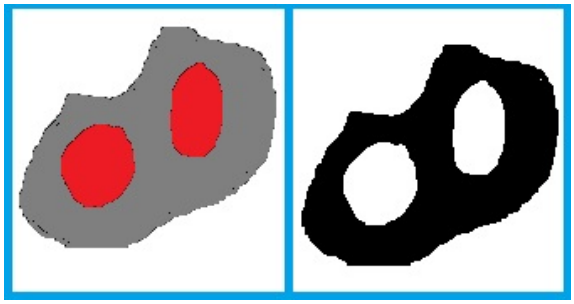
Bílá místa na masce reprezentují datově hodnotu jedna, z toho lze při překryvu masky s binárním snímkem odvodit jaký bude výsledek maskování. Dle obrázku 2.15 vidíme výrazné zvýšení omezení rušivých vlivů reprezentované čistým binárním snímkem s daty, která požadujeme.

Pokud známe polohu oblasti s kritickými daty pro běh programu ve snímku, můžeme s ohromnou výhodou použít funkci maskování. Je třeba si dát pozor na to, abychom si nevymaskovali část dat důležitou pro definici jednoznačného výsledku. Matlab umožňuje více druhů maskování i jednoduchou funkcí a výběrem ROI kliknutím. Dokonce některé funkce podporují i negativní maskování mimo zvolenou masku nebo různé další kombinace masek a použitých funkcí.

Obr. 2.15: Výsledek maskování

2.5.4 Shluková analýza

Shluková analýza je statistická metoda rozřazování jednotlivých výstkytů čehokoliv dle stejných nebo rozdílných vlastností do skupin a analyzováním zástupce každé skupiny pro vyhodnocení dat pro celou skupinu. V mém případě se jedná o analýzu shluků pixelů a určování jejich základních vlastností při vnějším pohledu. Zpravidla se jedná o velikost plochy, kterou shluk pixelů zabírá, dále pak orientace plochy v prostoru, středový bod, rozptyl a další vlastnosti jednotlivých bodů tvořících shluk.



Obr. 2.16: Binarizace testovacího snímku shluků

K analýze v Matlabu s výhodami používám funkci regionprops, která realizuje veškeré potřebné operace pro analýzu snímků. Jak už bývá zvykem, je nutné omezit rozptyl předkládaných dat binarizací, kvůli omezení množství zpracovávaných a výsledných dat pro lepší orientaci. Zcela jasný případ ušetřeného výpočetního času a zjednodušení vyhodnocování výstupních dat jsem realizoval na testovacím snímku shluků na obrázku 2.16, kdy bez binarizace funkce regionprops vygenerovala přes dvě sta nálezů a po binarizaci již jen čtyři. Příklad postupu analýzy jednoduchého shluku pixelů můžete vidět na obrázku 2.17. Především je zde snaha určit centrální bod shluku, dále pak plochu kterou shluk zabírá a v posledním bodě orientaci této plochy vzhledem k vodorovné ose. Funkce dle nastavení analyzuje okolní body zkoumaných pixelů a hledá sousedy s podobnými vlastnostmi a vytvořené shluky dále analyzuje různými funkcemi.

Uplatní se zde funkce jako watershed (zaplavování), kdy uvažujeme hodnoty jasové funkce snímku jako úrovně výšky pevniny. Postupným navyšováním prahu zaplavovací funkce, která přepisuje hodnoty pixelů nižší než je samotný práh, získáme výškové regiony a ty dále analyzujeme.

Obr. 2.17: Analýza regionprops

Dále zde pracují třídící a popisovací funkce pro správné zařazení výskytů do skupin, se kterými se dá dále pracovat v případě nutného přístupu k jednotlivým členům každé skupiny. Umožňují třídění, řazení výskytu ve skupině dle velikosti plochy nebo i dalších parametrů. To je výhodné v případě, že víme co hledáme. Při dobrém výběru z tabulky nalezených výskytů provádíme taktéž filtraci nežádoucích výskytů způsobených šumem nebo špatnou metodou předzpracování dat.

2.5.5 Houghova transformace

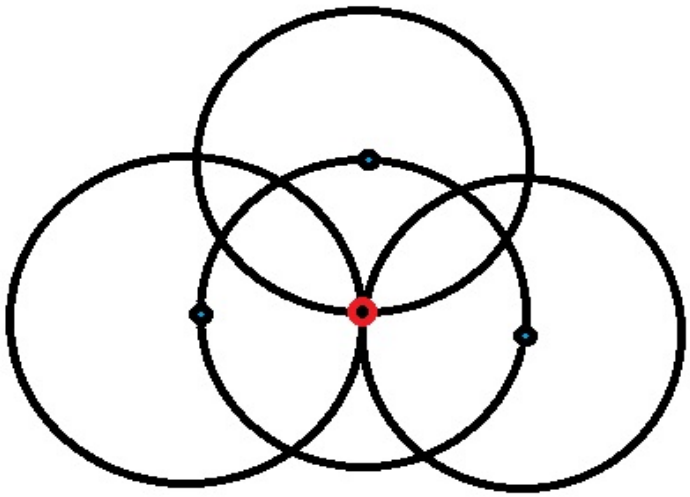
Houghova transformace je ideální transformací schopnou detekovat základní geometrické tvary v obraze. Prvním krokem je vždy definice geometrického tvaru, spolu s jeho popisovou rovnicí. V případě úvahy o detekci kružnic v obraze, musíme nejdříve určit rovnici kružnice platnou pro jakýkoliv souřadnicový bod určený koordináty x, y. Základní rovnice kružnice je

$$(x-x_o)^2+(y-y_o)^2=r^2 \tag{11}$$

kde x_0 a y_0 jsou souřadnice středu a r pak její poloměr. Pro použití transformace s fixními osami by bylo možné využít již takto jednoduchou rovnici, avšak standardně se používá rovnice parametrická umožňující různě orientované osy a operující s úhlem φ jako parametrem nabývajícím hodnot $(0,2\pi)$ radiánů.

$$\begin{aligned} x&=x_0+r*\cos\varphi \\ y&=y_0+r*\sin\varphi \end{aligned} \tag{12}$$

Vstupem tranformace je pak souřadnice bodu x,y, případně poloměr r se snahou nalézt střed kružnice.



Obr. 2.18: Princip hledání kruhů v Matlabu (imfindcircles)

V Matlabu funguje Houghova transformace na intervalu zadaných poloměrů kružnic a s dalšími možnostmi nastavení citlivosti. Princip funkce transformace není složitý, jde pouze o předpřípravu dat ve formě převedení snímku do šedotónového obrazu, dále pak použití dodatečných algoritmů pro ideální detekci hran s vytvořením hranového obrázku. Na samotném

hranovém obrázku je poté aplikována metoda hledání středů pomocí zadaných rovnic geometrických tvarů s mechanismem akumulace v bodech ležících na domnělých kružnicích. Po průběhu celého snímku jsou detekována maxima v předem určených bodech a pokud body leží na jedné kružnici, pak je dle rovnice (12) dopočítán střed.

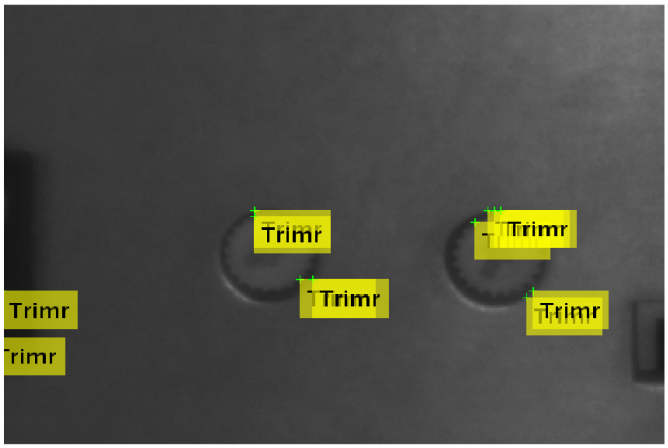
Na obrázku 2.18 je princip funkce imfindcircles založené na Houghově transformaci, kdy dochází k akumulaci hranových pixelů vyhovujícím rovnici kružnice a akumulačním bodům (modrá) pro definici středu kružnice (červená) vykreslením kruhů se známým poloměrem r, kdy poté bod s nejvyšší intenzitou je námi hledaný střed kružnice. V případě zvolení intervalu poloměrů je prováděn tento cyklus pro každý poloměr zvlášť s použitím trojrozměrného snímku, kdy dochází ke tvorbě kuželů, které se zcela stejným způsobem protínají a tím definují střed hledané koule.

2.6 Návrh algoritmu

Základní myšlenkou při inspekci nějakých obrazových dat, je jasná klasifikace prověřovaných oblastí na testovaném vzorku a schopnost systému definovat stav, ve kterém se oblast nachází. V průmyslové výrobě není situace černobílá, kdy by buď vzorek vyhoví nebo ne, ale počítá se zde s jistými tolerancemi, které jsou avšak pro různé fáze výroby rozdílné. Ano, na konci výrobního procesu při finální inspekci musíme vynést rozsudek nad testovaným vzorkem, ale do toho bodu je prováděno mnoho dalších kontrol stavu výrobku s případnými opravami. Pro navázání na systém jsem zvolil využití možných tolerancí i při finální inspekci, kdy algoritmus pracuje s hodnotami a rozhodnutí OK/NOK je až v konečné fázi.

2.6.1 Klasifikace

Základní přípravná fáze lokalizace zájmových oblastí a jejich popsání je realizována neuronovou sítí naučenou na rozpoznávání deskriptoru HOG a následná klasifikace do tříd předloh hledaných ve zkoumaném snímku. Samotné vyhledávání předloh je realizováno jako plovoucí okno velikosti předlohy, které je posouváno v horizontálním i vertikálním směru přes celý obrázek.



Obr. 2.19: Pozitivní nálezy pro třídu "trimr"

Tento postup je velmi zdlouhavý, a proto algoritmus obsahuje kroky urychlující jeho činnost. Dle předchozího tvrzení o hranách kdy v zásadě jediné co nás zajímá jsou právě hrany,

algoritmus generuje hranový obrázek, který používá jako naváděcí systém pro samotné posuvné okno, v němž vždy probíhá extrakce deskriptoru. Další fintou pro urychlení skenování obrázku pak může být přeskakování některých bodů, avšak to má bohužel za následek pokles výsledné pravděpodobnosti přiřazení předlohy do jedné z naučených tříd.

Ve finálních fázích návrhu a experimentálního ladění dochází stále k mnohanásobné detekci předloh, což nepovažuji za chybu ale za zcela očekávanou vlastnost klasifikace pomocí neuronové sítě a stavového automatu. Tento jev je dalšími kroky využit ke kontrole naměřených a vypočtených hodnot a k tak u proměnlivých dat důležité redundance. Taktéž zkoumání nadbytečných detekcí usnadňuje ladění celého algoritmu pro optimální poměr mezi přesností a dobou inspekce.

2.6.2 Inspekce zájmových oblastí

Pomocí histogramu orientovaných gradientů jako deskriptoru a neuronové sítě jako klasifikátorů byly vygenerovány zájmové oblasti označené záměrnými body. Navázání na tento program je poté nasnadě. Snahou je získat výřezy z hlavního zkoumaného snímku a ty nadále zpracovat. Pro vymezení zájmové oblasti označené zaměřovacím křížkem využijeme znalosti velikostí zkoumaných prvků a experimentálního zhodnocení chování programu. Dalším krokem je vytvoření výřezu celé oblasti. Použití algoritmů bude předvedeno na jednom vzorku, kde budou představeny metody rozličné inspekce specifických zájmových oblastí.

Inspekce trimru
Dle materiálů popisujících časté defekty na vzorcích není třeba zkoumat nějaký specifický defekt v případě nastavovacího trimru. Jedinou kritickou vlastností je jeho přítomnost a správné umístění, což kontroluje již nadřazený program založený na neuronové síti. V případě pozitivního výsledku je polovina inspekce dokonána, dalším postup je poté určení natočení trimru.

U konkrétně testovaného kusu disponuje trimr vzorem šipky, která se dá k tomuto účelu použít, avšak kontrast mezi samotnou šipkou a další hmotou kolem ní je malý. Jediný rozdíl na pohled způsobuje vržený stín na vnitřní straně šipky. Matlab umožňuje nastavení preferencí binarizační funkce, a proto jsem experimentálním způsobem dospěl do stádia, kdy disponuji binárním snímkem vhodným ke zkoumání na obrázku 2.12. Po vygenerování binárního snímku jej analyzuji pomocí funkce regionprops a z ní extrahuji otočení šipky parametrem „orientation“. Postup je prakticky totožný s ukázkovou analýzou v kapitole Shluková analýza.

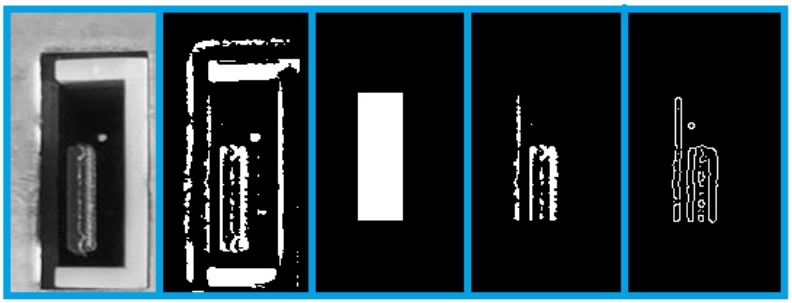
Testovací data nejsou ideální, a proto i při sebelepším zpracování výsledky ovlivňuje šum, nepravidelný povrch a další běžné aspekty. Tyto skutečnosti výrazným způsobem ovlivňují měřením, čímž způsobují řadu chyb ve výpočtech a následných vynášených závěrech. Eliminace chyb provádím pomocí analýzy většího množství zdánlivě stejných výsledků a následné korekce.

Inspekce jumperu
Zcela analogicky jako v předchozím případě spoléhám na určení pozice jumperu neuronovou sítí a dále dle znalosti velikosti prověřovaného prvku generuji obrázek zájmové oblasti. Požadavky na výstupy z inspekce jumperu jsou prakticky stejné jako v předchozím případě, avšak zde je množina stavů vstupních dat omezená na tři prvky: přítomen, nepřítomen a poloha. V případě inspekce přítomnosti jumperu ve slotu mohu celý inspekční algoritmus založit na jednoduchém postupu.

První krok je zcela stejný jako v předchozím případě, pakliže výsledek porovnávání předloh nalezne v testovacím snímku objekt třídy jumper, pak je daná oblast dle nastavených rozměrů

ořezána. Následuje příprava dat na zpracování odfiltrováním šumu a binarizace s dodatkovou funkcí vyplnění nalezených děr pro získání menší členitosti zkoumaného výřezu. Doplňková metoda dodatečné úpravy dat je pak maskování funkcí logického součinu s předvolenou binární maskou.

Druhým krokem je již z předchozích kapitol hojně zmiňovaná detekce hran, konkrétně použitá funkce edge z knihovny algoritmů pro zpracování obrazu s předvolbou detektoru Canny. Na obrázku 2.20 zleva je pak původní výřez, binární snímek, binární maska, maskovaný snímek a detekce hran



Obr. 2.20: Postup vyhodnocení pozice jumperu

Porovnání hran provádí skenovací funkce for cyklu, kdy dochází ve dvou rovinách ke sčítání nenulových pixelů z hranového obrázku, přičemž roviny jsou voleny dle zkoumaného výřezu tak, aby hrany způsobené přítomností jumperu zasahovaly v každé poloze pouze do jedné roviny. Na základě nastavených prahových hodnot rozdílů počtů je vynesen verdikt, kdy z logiky algoritmu vychází pozice jumperu protínající rovinu s vyšším počtem nenulových pixelů. Pro případ kdy by nebyl jumper přítomen je nastavena konstanta minimálního rozdílu a při jejím nepřekročení bude verdikt přítomnosti jumperu negativní.

Inspekce LED

Jedním z jednodušších úkolů inspekce poskytnutých vzorků vzhledem k požadavkům je právě kontrola LED. Co se týče přítomnosti diody, tak jako v předchozích případech základní krok inspekce je proveden neuronovou sítí. Další požadavek vyplývá z umístění samotné diody za otvorem krytu čelního panelu, kdy jde o kontrolu osové souměrnosti diody a otvoru jímž světlo prochází.

Obr. 2.21: Postup vyhodnocení souměrnosti LED s krytem

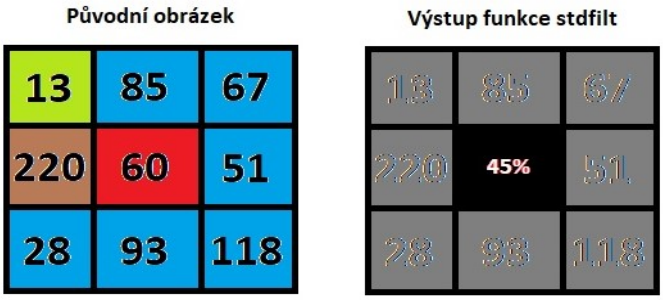
V zásadě jde o zjištění přítomnosti překryvných kružnic a kontrolu vzdálenosti jejich středů. Při známém rozměru kružnic a omezeném prostoru vyhledávání můžeme s výhodou použít opět funkci z knihovny zpracování obrazu s názvem imfindcircles, která je založena na Houghově transformaci. Obrázek 2.21 popisující postup vyhodnocení zvýrazňuje červenou barvou nálezy kruhů se zelenými středy a výpočet vzdálenosti středů vyobrazuje jejich spojnice. Dle logiky věci platí, čím menší vzdálenost středů nalezených kružnic, tím lepší osová souměrnost.

Samozřejmostí je úprava množství dat pomocí binarizace s vhodně zvolenou citlivostí a následné zpracování výstupních dat z funkce přepočítávající z koordinátů souřadnic středů jejich vzdálenosti.

Inspekce vad povrchu

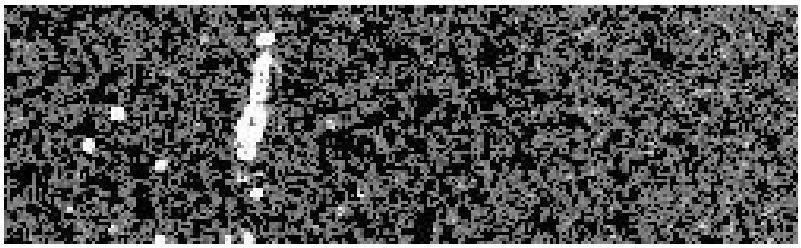
Dalším z kritických vlastností jakéhokoliv výrobku je jeho vzhled a s ním spojené estetické vady nepředstavující žádný problém pro funkčnost výrobku, avšak z pohledu zákazníka jde mnohdy o daleko větší problém, zvláště pokud výrobky slouží pro reprezentaci distributora nebo operují na místech, kde estetické vady ruší jinak bezvadný vzhled celku. Pokud mluvíme o lakovaném kovovém krytu zařízení, pak jako vady povrchu můžeme brát již samotné chyby vznikající při lakování, jako jsou třeba vzduchové kapsy mezi jednotlivými vrstvami barvy nebo nehomogenní tloušťka laku. Jednou z nejhorších vad jsou škrábance různých úrovních hloubky a rozsahu na místech, kde bývá s výrobkem manipulováno. Vzniku jemných vlásečnicových škrábanců se nedá nikdy zcela zabránit, lze pouze omezit jejich výskyt velmi šetrnou manipulací a ochranou výrobků ve všech fázích výroby. Odhalování vad povrchů s definovanou strukturou není pomocí kamerového systému jednoduchý úkol, protože odezva vad na okolní osvětlení není vždy přesně definována. Z experimentů a dle zákonů geometrické optiky pro šíření světla vychází najevo podstata detekce vad jakožto lokální změna jasu povrchu, kvůli závislosti odrazu světelných paprsků od povrchu na jeho členitosti a koeficientech odrazu a lomu. V praxi tyto skutečnosti mohou znamenat pro kamerový systém při různém osvětlení různé výsledky inspekce, v závislosti na poloze snímacího a osvětlovacího zařízení a směru působení rušivého záření.

Rozhodl jsem se založit inspekci vad povrchu na funkci zjišťující standardní odchylku hodnoty jasové funkce sledovaného bodu od hodnot jasové funkce okolních bodů právě tohoto středobodu. Princip funkce algoritmu stdfilt z knihovny Image Processing Toolbox je znázorněn níže.



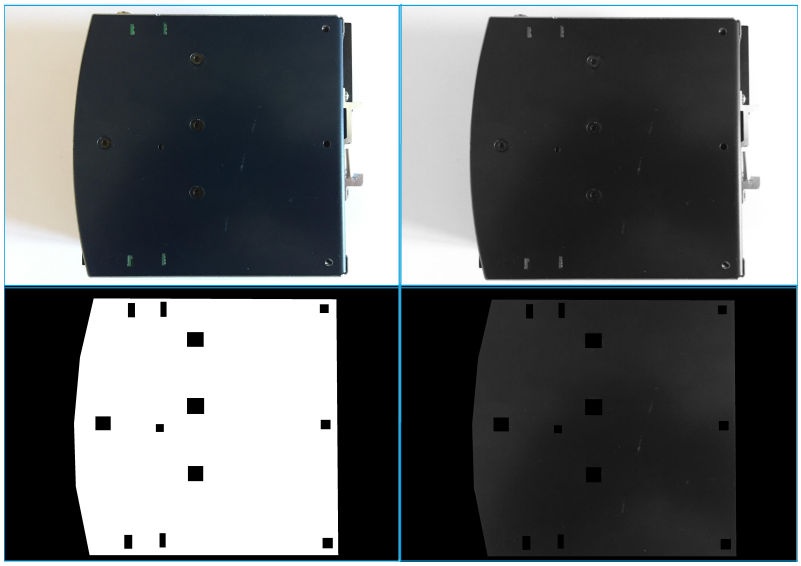
Obr. 2.22: Princip funkce stdfilt

Funkce stdfilt vytváří z původních dat šedotónového snímku s rozsahem hodnot pixelů 0-255 pro osmibitový integer snímek daný odchylkami hodnot zkoumaných bodů od jejich okolí. Pro výpočet odchylky středobodu od průměrné hodnoty dané okolím, musíme uvažovat minimální (zelená) a maximální (hnědá) hodnotu okolních bodů, jak ukazuje obrázek. Funkce poté realizuje výpočet aritmetického průměru a porovná hodnotu zkoumaného (červená) bodu s hodnotou vypočtenou. Výsledkem je poměr skutečné hodnoty a průměrné hodnoty v procentech reprezentovaný proměnou double vloženou na každou zkoumanou pozici.



Obr. 2.23: Výstup funkce stdfilt

Samotné využití této funkce má svá úskalí, především pak vyhodnocení vad povrchu na základě snímku odchylek. Na obrázku 2.23 zcela jasně rozpoznáme vliv vady povrchu na výstup funkce, avšak můžeme zde také pozorovat členitost povrchu a menší vady. V tomto případě velký světlý útvar představuje samotnou vadu, menší světlé tečky drobnější vady, tmavší místa odkazují na nerovnosti povrchu a černá místa pak zobrazují výpočetně ideální povrch. Vyhodnocení povrchových vad lze pro tento případ řešit snadno posuvným okénkem, kde bude vždy ze snímku odchylek generována průměrná hodnota okna, která bude porovnávána s thresholdem funkce určení chyba ano/ne. Tento způsob je nadmíru snadný a výpočetně nepříliš náročný, ale bohužel při experimentech se ukázalo, že při vyšší členitosti povrchu dochází k ignorování domnělých vad nebo k přehnané detekci, a proto jsem si zvolil pokročilejí algoritmus funkce regionprops.

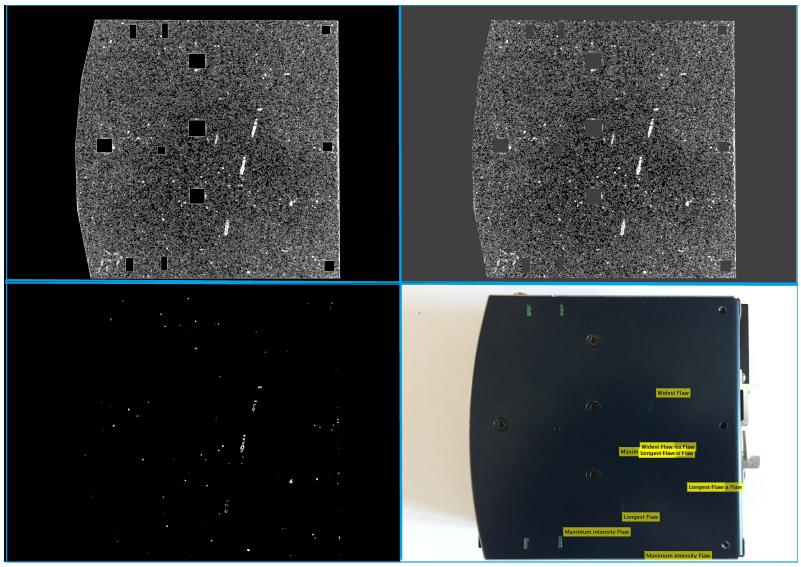


Obr. 2.24: Předzpracování dat pro inspekci povrchu

Celý proces probíhá následujícím způsobem. Vstupní data projdou předpřípravou pro zpracování, kdy po převodu do šedotonového snímku dochází k filtraci šumu Gaussovým filtrem a vymaskováním kritických oblastí určených pro zjištění stavu povrchu. Výsledkem je snímek s nenulovými hodnotami pixelů v případě, že jde o oblast kterou chceme prověřovat, avšak všude jinde disponují předpřipravená data nulovou hodnotou jasové funkce.

Obrázek 2.25 výstižně mapuje jednotlivé kroky inspekce povrchu, přičemž základem je funkce stdfilt, jejíž produkt vidíme vlevo nahoře. Vystupující černé obdélníky jsou pozůstatek maskování původních nezájmových regionů pro inspekci a je tedy nutné je odstranit. Využil jsem vlastní smyčkovou strukturu, která nahrazuje hodnoty pixelů předem definovanou maticí průměrných hodnot a způsobuje nepřehlednost těchto míst pro hodnotící algoritmus. Výsledek operace uzavírá horní řádek obrázku.

Dalším základním algoritmem přípravy dat pro vyhodnocování pokročilými funkcemi jako je právě regionprops je binarizace. Jednoduchost tohoto algoritmu způsobuje další komplikace k necitlivost celého systému na špatné nastavení expozice a další jevy způsobující výraznou změnu hodnot pixelů. Pro zajištění alespoň minimální spolehlivosti je třeba definovat hodnotu thresholdu pokročilou metodou Otsu, kterou jsem upravil vynásobením konstantou tak, aby pro testovací snímky počítaná hodnota thresholdu odpovídala rozumným mezím. Výsledek binarizace otevírá spodní řádek obrázku.



Obr. 2.25: Kroky inspekce povrchu

Při pohledu na binarizovaný snímek a jeho porovnání se vstupním snímkem můžeme říci, že se nám podařilo zpřehlednit data pro použití pokročilé funkce regionprops s několika stěžejními parametry. Pro budoucí zhodnocení nalezených výskytů uvažuji použití parametrů: Centroid, MaxIntensity, Area, MinorAxisLength a MajorAxisLength. Samotné názvy parametrů funkce regionprops definují strukturu výsledného výstupu a její samotnou použitelnost při inspekci povrchu. Základní myšlenka při definování všech požadovaných parametrů byla co nejpřesnější určení jakýchkoliv nespojitostí prověřovaného povrchu, avšak dle výrobních požadavků vím, že estetické vady se hodnotí třemi základními parametry. Kromě rozměrů definovaných jako součet všech délek vad (škrábanců) v určitém sektoru, jakožto i jejich šířka, jde samozřejmě o výši poškození hodnocenou pomocí hloubky. Bez použití velmi přesných laserových měřících aparátů není možné jednoduchým způsobem určit změnu vzdálenosti při měření povrchu v pořádku a škrábance, a proto uvažuji závislost hloubky každého ze škrábanců na intenzitě jasové funkce v šedotónovém snímku. Funkce regionprops operuje v případě zjišťování parametrů „skvrn“ na pouze binarizovaných datech, kdežto při analýze maximální intenzity bodů pro definici nejvýznamnější vady, co se týče hloubky, zkoumá i samotný šedotónový snímek před detekcí odchylek.

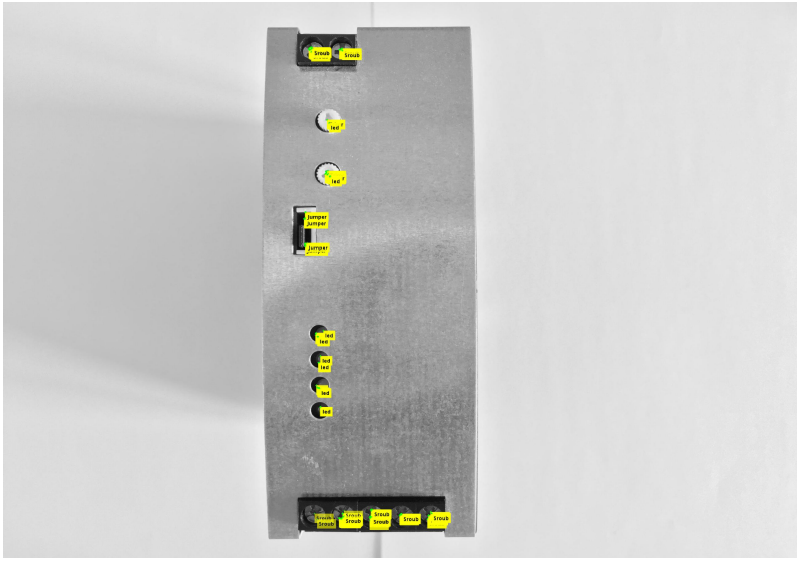
Obr. 2.26: Vyhodnocení parametrů: Area, MaxIntesity, Major/MinorAxisLength

Díky této konfiguraci je možné detekční algoritmus přizpůsobit rozličným požadavků z hlediska tolerance vad ve výrobním procesu. Na obrázku 2.26 je výchozí konfigurace s označením jedné nejvýznamnější vady dle každého kritéria. Můžeme vidět seskupení kritérií pro plochu, vodorovný a svislý rozptyl v místě významné vady, avšak nejvýznamnější vada co se týče intenzity se nachází jinde. S těmito daty se dá s výhodami dále pracovat, právě při vícenásobné detekci jedné vady nebo naopak rozprostření nejlepších kandidátů od každého kritéria na jiné místo. Dochází k výraznému zpřesnění inspekčního systému a takto navržený systém nám umožňuje lépe nastavit tolerance k nevýrazným vadám. Zároveň lze pomocí vícenásobné detekce eliminovat nesprávná rozhodnutí o škrábanci.

2.7 robustnost detekčních algoritmů

2.7.1 Klasifikace

Dle již dříve zmiňovaných nástrojů a postupů byl vyvinut inspekční systém založený na prostředí Matlab s využitím deskriptoru HOG a neuronové sítě jako klasifikátoru. Úkolem této kapitoly je vyhodnotit úroveň vyvinutého systému, což znamená jednoduchý test funkčnosti všech částí algoritmu v závislosti na kvalitě dodávaných dat. První podkapitola je zaměřena na hodnocení schopnosti neuronové sítě rozpoznávat při naučených deskriptorech vzorových snímků skutnečné výskyty v nepříliš kvalitních datech ovlivněných rušivými jevy.



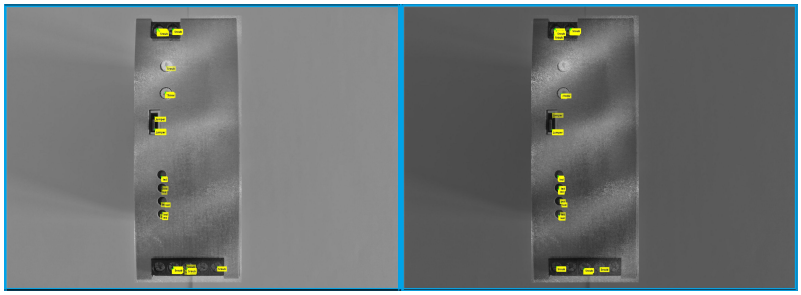
Obr. 2.27: Výchozí stav výstupu klasifikátoru neuronové sítě

V případě klasifikace vzorků do tříd dle naučení neuronové sítě, jde v algoritmu o sledování výstupní hodnoty sítě po vložení deskriptoru udávané pravděpodobností správné klasifikace a následného označení místa nálezu. Při realizaci jsem narážel na problémy podobnosti některých vzorů patřících do rozdílných tříd. Při nastavení rozhodovacího algoritmu na vysokou úroveň pravděpodobnosti příslušnosti dochází k ignoraci potencinálních nálezů, které mohou i nemusejí do dané třídy patřit. Naopak při příliš nízkých hodnotách pravděpodobností a velkému kroku skenování dochází k výraznému nárůstu falešných detekcí. Před testováním robustnosti algoritmu byl rozhodovací mechanismus vyvážen tak, aby každý právem platný nález byl označen popiskem, třebaže se jedná o nález mnohonásobný. Výchozí stav výstupu klasifikačního algoritmu je na obrázku a další vývoj správných i nesprávných klasifikací je

vyobrazen v následujích snímcích.

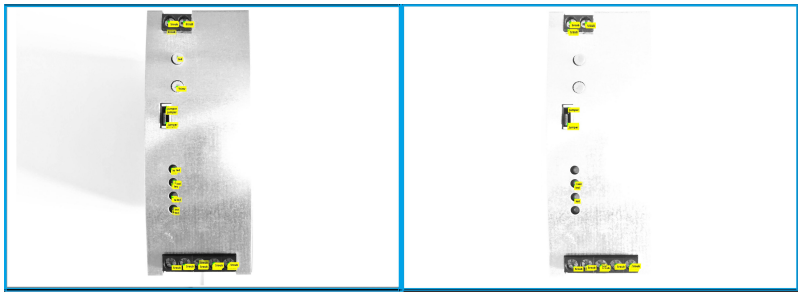
Algoritmy byly prověřovány analogickým způsobem pomocí testovacích obrázků vygenerovaných v programu GIMP s využitím překryvné vstvy pro kombinaci světlo stín s poměrem překryvu opět 30% a 70%. Postranní osvětlení bylo vytvořeno simulačním skriptem filtru světlo, s analogickou hranicí překryvu pro jednotlivé intenzity. Celková úspěšnost algoritmů je hodnocena jako podíl úspěšných klasifikací ku celkovému počtu klasifikovatelných elementů. Neuvažuji ztrátu mnohonásobné klasifikace jako chybu, až do doby kdy neexistuje jediná správná klasifikace zkoumaného komponentu. V případném výskytu chybných klasifikací u jinak správně klasifikovaného komponentu, uvažuji odfiltrování minoritního množství klasifikací. Pokud je většina klasifikací správná, neuvažuji výsledek algoritmu jako chybu.

Na obrázku 2.27 ve výchozím stavu se nachází na první pohled chybná klasifikace trimru jako objekt třídy led. Dle předchozích úvah toto nepovažuji za chybu, protože výrazným způsobem neovlivňuje výsledek celého programu.



Obr. 2.28: Zhoršení klasifikace komponent vzorku pro světlo/stín

Pokud začneme klasifikační algoritmus trápit nekontrastními vstupními daty, začne docházet k postupné ztrátě rozpoznávací schopnosti, nikoliv vinnou neuronové sítě, ale částí programu optimalizující rychlost běhu. Tato část pomáhá, dle předchozí kapitoly detekce hran, programu najít zájmové oblasti, kde má klasifikovat.



Obr. 2.29: Zhoršení klasifikace komponent vzorku pro postranní osvětlení

Při prvotních zkouškách odolnosti algoritmu jsem dospěl k závěru, že není možné algoritmus používat v takto rozličných podmínkách bez úpravy optimalizační části, proto jsem se rozhodl pro hodnocení odolnosti upravit vstupní podmínky hranového obrázku a zvýšit kontrast určením zájmových oblastí, kde se nachází komponenty vhodné pro klasifikaci. Toto bylo možné postupnou generací hranového obrázku jednotlivých oblastí a jeho následným složením. S vysokou pravděpodobností jsem dospěl k vhodnému řešení pro testování robustnosti tak, abych neznevýhodňoval jednu část programu před další, popřípadě zbytečně nezhoršil robustnost testované části. Co se týče běhu programu v případě absence optimalizační části, tak se jedná o jednotky hodin s přiměřeně dobrými výsledky.

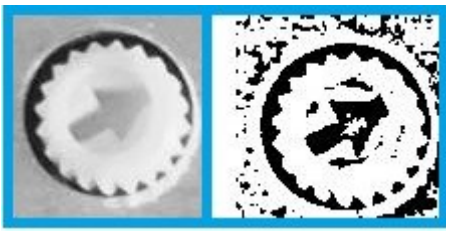
Na předchozích obrázcích 2.28 a 2.29 je vidět slušná odolnost klasifikačního algoritmu po předchozí úpravě hranového obrázku coby ukazatele. V případě vlivu světlo/stín nedochází ke klasifikaci u málo kontrastních komponentů, které dělaly problémy již při optimalizaci programu. Typicky šlo o krajní komponenty třídy šroub, avšak při vyšší úrovni vlivu docházelo k omezení klasifikační činnosti i u dalších komponent. V případě vlivu postranního osvětlení byla situace o něco lepší, právě z důvodu použití deskriptoru relativně necitlivého na absolutní hodnoty jasové funkce. Při testování došlo pouze k přehlédnutí komponent s velmi malým kontrastem (trimr) nebo komponent s podobnými deskriptory (led).

2.7.2 Inspekce zájmových oblastí

Na klasifikátor navazující algoritmy vyhodnocují dle klasifikovaných oblastí samotný stav komponent. V případě robustnosti je srovnán výchozí stav se stavem inspekce při poskytnutí testovacích snímků robustnosti. V případě trimru vyhodnocuji odchylku změřeného úhlu vzhledem k původním hodnotám. Pokud jde o jumper a určení jeho pozice ve slotu, tak uvažuji mnohonásobnou klasifikaci s mnohonásobnými výskyty jedné polohy, vše ostatní je bráno jako chyba a poměrově počítáno. Osová souměrnost LED s krytem je závislá na Houghově transformaci a hledání kružnic v obraze. Právě z toho důvodu je celkem jasná mez určení funkčnosti programu opět brána v poměrem správných a nesprávných detekcí. S vadami povrchu je situace o něco složitější, pokud jsou založeny na odchylkách jasové funkce. Rozhodl jsem se neřešit nadbytečné určené chyby vlivem saturovaných nebo tmavých oblastí způsobených testovacími daty, ale přehled algoritmu pro hodnocení vad i s takto špatnými daty. Ve všech testech inspekčních algoritmů pro zájmové oblasti uvažuji správnou klasifikaci těchto zájmových oblastí.

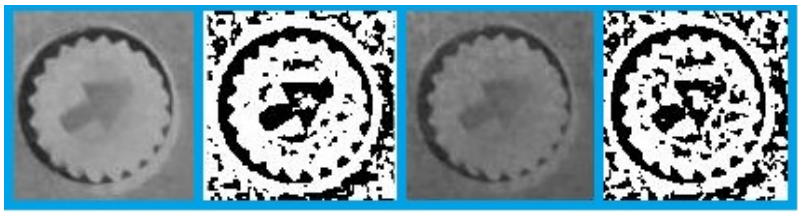
Inspekce trimru

Pokud uvažuji úspěšnou klasifikaci vzorku a následné označení ROI pro funkci analyzující natočení trimru, tak mohu pozorovat samotnou robustnost pouze a jen této funkce. Na obrázku 2.30 vidíme výchozí stav výstupu funkce počítající natočení trimru. Vlevo je skutečný výřez ROI z původního snímku a vpravo pak jeho binární obraz. Ten je dále využívaný pro inspekci a defacto na něm leží veškerá odpovědnost za nedokonalosti určení natočení. I ve výchozím stavu je vidět nedokonalost vyobrazení středového symbolu trimru, a proto i zde dochází k odchylce vypočtené hodnoty od skutečného natočení trimru. Tato skutečnost je mi ovšem známa již od počátků experimentů, a proto je ve vyhodnocení kompenzována množstvím výsledků a početní úpravou.



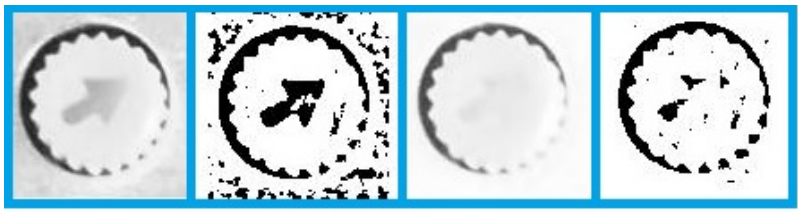
Obr. 2.30: Výchozí stav pro kontrolu natočení trimru

V případě podstrčení nerovnoměrně osvětlených dat dochází k nepřesnostem při určování natočení trimru. To je způsobenou více skutečnostmi. Při pohledu na binární snímky na obrázku 2.31 pro kombinaci světlo/stín u obou úrovní, můžeme pozorovat významný nárůst objektů ve snímku. Z logiky fungování algoritmu můžeme říci, že pokud nebudou nechtěné objekty větší než žádaný znak, pak nikdy nedojde k fatálnímu selhání funkce. K nepřesnostem určení úhlu natočení trimru dochází již dlouho před selháním funkce, proto hodnocení robustnosti zakládám na přesnosti určení úhlu natočení.



Obr. 2.31: Zhoršení kontroly natočení trimru pro světlo/stín

Na obrázcích 2.31 a 2.32 se nacházejí výsledky ořezů ROI a jejich binární obrazy potřebné pro běh funkce, přičemž vliv světlo/stín má neblahý následek právě na binární snímek ve smyslu vzniku vedlejších „skvrn“. Tento efekt není až tak závažný, na rozdíl od efektu postranního světla, kdy pro vyšší intenzity dochází k selhání samotné detekce hran, na kterou je navázána binarizace. Z toho důvodu dochází k selhání celého podprogramu pro kontrolu natočení trimru.

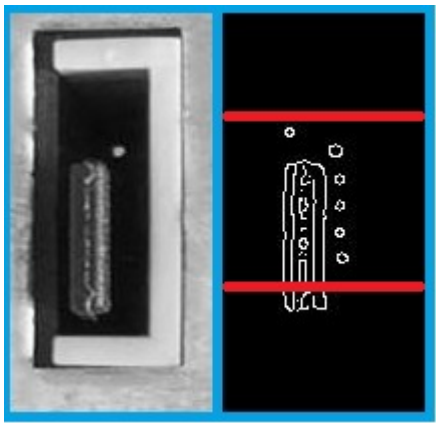


Obr. 2.32: Zhoršení kontroly natočení trimru pro postranní osvětlení

Výsledná odolnost je vynesena na konci kapitoly do přehledné tabulky zohledňující chybu v určení úhlu a selhání programu.

Inspekce jumperu

Inspekce přítomnosti jumperu je založená na počítání hran v hranovém obrázku maskovaném tak, aby vynikla pouze část patice s piny.

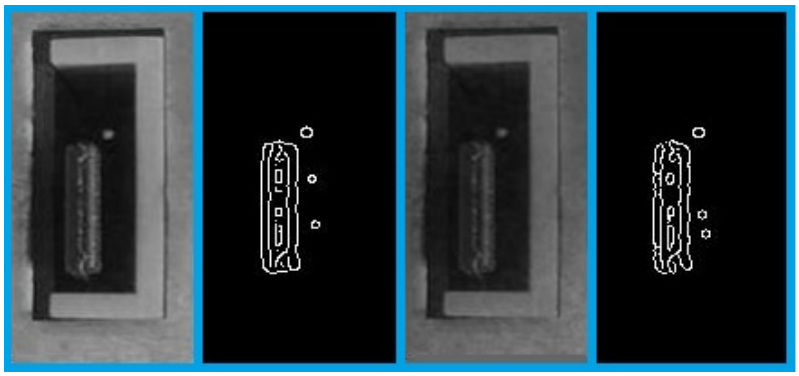


Obr. 2.33: Výchozí stav kontroly pozice jumperu

Leč se tento způsob vyhodnocování polohy jumperu zdá příliš jednoduchý a neodolný, tak opak je pravdou. Z experimentů vyplynula při správně klasifikaci téměř stoprocentní úspěšnost určení polohy jumperu, bohužel ale klasifikační algoritmus při současném stavu naučení reaguje i na části patice pro jumper, a tak se může stát, že se do ROI nedostane celá potřebná část pro určení pozice jumperu.

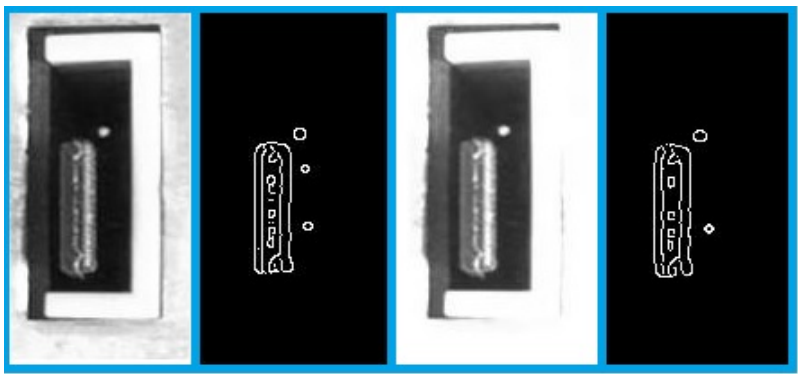
Obrázek 2.33 výchozího stavu při kontrole pozice jumperu vyobrazuje na levé straně výřez ROI a na straně pravé pak jeho binární obrázek. Červené linky dělící hranový obrázek na přibližné

třetiny demonstrují řádky hranového obrázku, kde je prováděno sčítání nenulových bodů a jejich následné vyhodnocení.



Obr. 2.34: Zhoršení kontroly pozice jumperu pro světlo/stín

Z výstupů funkce pro určování pozice jumperu při použití simulačních dat dostaneme vcelku uspokojivé výsledky. Při pohledu na binární snímky na výstupu pro kombinaci světlo/stín ve dvou intenzitách, můžeme pozorovat minimální úbytek hran a s tím spojené relativně spolehlivé vyhodnocování pozice. Jediným problémem zde zůstává velká závislost na klasifikátoru a jeho přesné určení ROI, kvůli použitému maskování.



Obr. 2.35: Zhoršení kontroly pozice jumperu pro postranní osvětlení

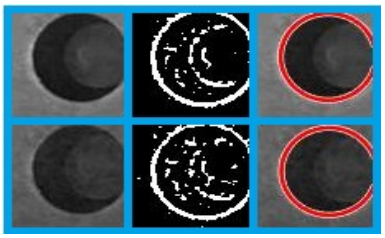
V případě působení postranního osvětlení je výsledek prakticky totožný, opět dochází k úbytku hran bez větších příčin. Velmi zásadně se zde projevuje závislost na klasifikátoru, avšak samotná funkce programu je dobrá.

Inspekce LED

Po klasifikaci objektu třídy „led“, dojde k ořezání původního snímku na okno vhodné pro skenování. Pro všechny kontroly byl systém navržen při vypnutém stádiu, kdy diody nesvítí a to způsobuje řadu problému při změnách osvětlení. Po nastavení systému na bezchybný provoz jsem při experimentech dospěl k nemilému zjištění. Funkce používaná na hledání kruhů v obraze při nastavení vyšší citlivosti generuje neexistující kruhy a výsledky jsou pak zcestné, proto jsem se rozhodl ověřovat funkčnost tohoto podprogramu na jednom konkrétním nálezu, kdy za plnou funkčnost považuji vyhledání kruhového otvoru pro průsvit a diody. Adekvátním způsobem zohledňuji zda vykreslené kruhy jsou na správných místech, stejně tak jako výpočet vzdálenosti jejich středů, o který jde primárně.

Obr. 2.36: Výchozí stav pro inspekci osové souměrnosti LED s krytem

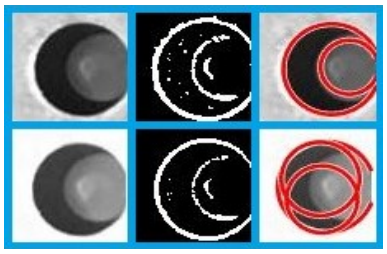
Na obrázku 2.36 jsou znázorněny důležité kroky analýzy osové souměrnosti diody s krytem, zleva jde pak o vstupní výřez, binární snímek a nalezené kruhy. V případě pozitivního nálezu (dle vykreslení) kruhů můžeme s jistotou říci, že následné výpočty budou mít správné výsledky.



Obr. 2.37: Zhoršení detekce kruhů pro světlo/stín

Představa bezproblémové funkce této části programu je při vlivu stínu z poloviny zničena, protože led jsou u vzorku poměrně zapuštěny a při snímání kamerou dochází s vyrovnáváním expozice ke ztmavení právě těch částí, které jsou nesmírně důležité pro detekci polohy kruhu reprezentující pouzdro led. První tři snímky shora na obrázku reprezentují vliv světlo/stín na první úrovni (nižší) a druhý řádek poté reprezentuje vliv jevu úrovně druhé (vyšší).

Zcela analogicky je na obrázku 2.38 výstup při ovlivnění dat postranním osvitem, kdy ale dochází při menší intenzitě k podpoře činnosti algoritmu a ten zůstává funkční (horní řada obrázků). Při vyšší úrovni postranního osvětlení dochází vlivem vyšší citlivosti funkce k mnohonásobnému určení falešných kruhů, které o ničem nevypovídají. Jedinou jistotou v případě detekce kruhů je pak určení otvoru v čelním panelu, protože ten poskytuje ve většině případů dostatečný kontrast pro široké spektrum vnějších vlivů.

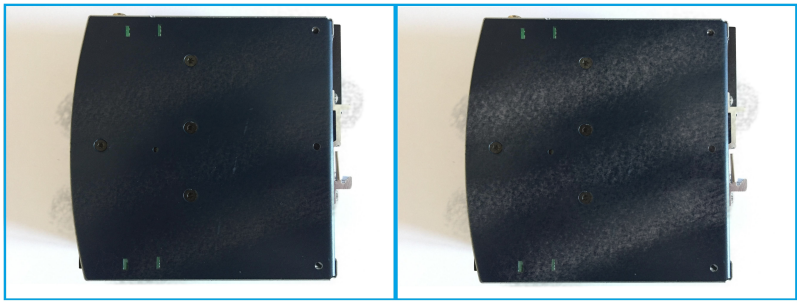


Obr. 2.38: Zhoršení detekce kruhů pro postranní osvětlení

Na předešlých obrázcích 2.37 a 2.38 je vidět závislost binárního snímku na šumu s nízkou hodnotou jasové funkce, což může být při jistém nastavení kamerového systému i stín. Bohužel při velkých rozdílech jasu v jednom snímku dochází při zpracování ke ztrátě detailů a s tím spojenému selhání programů operujících právě s nimi. Řešením by mohlo být stavění programů na neměnné veličině, ale to v praxi není vždy možné.

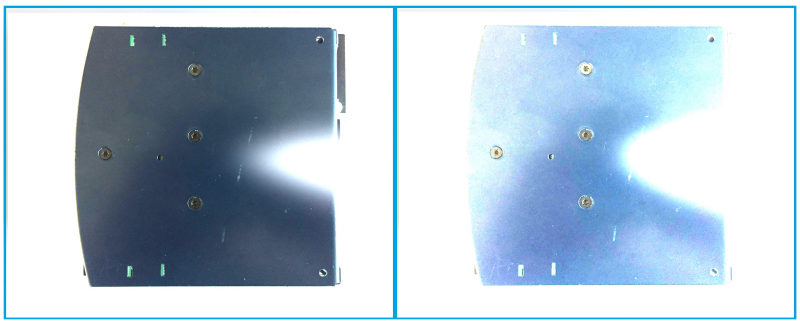
Inspekce vad povrchu

Pokud jde o inspekci povrchu založenou na vyhodnocování jasové funkce a data nabízená této funkci budou upravena manipulací s jasovou funkcí, pak se dá předpokládat solidní problém při vyhodnocování povrchových vad, proto nelze brát jako chybu programu detekci falešných vad.



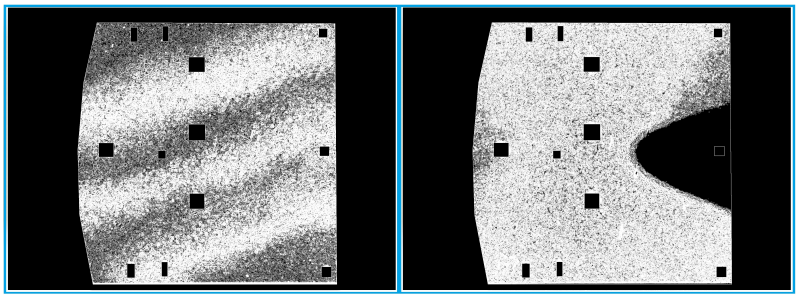
Obr. 2.39: Testovací snímky inspekce vad pro vliv světlo/stín

Testovací snímky 2.39 a 2.40 pro vliv světlo/stín a postranní osvětlení jsou analogicky upraveny jako v předchozích případech a opět ve dvou úrovních intenzity, kdy nižší je vždy na levé straně snímku a vyšší pak na straně pravé. Očekávám výrazné ovlivnění výstupních dat z funkce stdfilt a případné problémy s vyhodnocováním již dříve určených vad povrchu, na výchozích snímcích, v předchozí kapitole. Při pohledu na testovací snímky můžeme říci, že je vysoká pravděpodobnost detekce dalších vad na hranicích přechodů mezi oblastmi se změnou jasu a u postranního osvětlení na přechodu z oblasti saturace ven a naopak.



Obr. 2.40: Testovací snímky inspekce vad pro postranní osvětlení

Výstupní data funkce stdfilt po vymaskování nespojitých oblastí dávají smysl v tom, že v homogenních oblastech zatížených vlivy dochází k poklesu odchylky, což značí tmavší barva snímku. V případě místní saturace vzniká pak černá oblast, což naznačuje možnou lepší odolnost vůči lokálnímu masivnímu zisku jasu, pokud je alespoň přiměřeně v celé oblasti stejný. Bohužel toto neplatí pokud dojde k saturaci snímače, který je v ten okamžik ve všech saturovaných místech necitlivý k veškerým vadám.



Obr. 2.41: Projev vlivů odchylek, světlo/stín vlevo, postranní osvětlení vpravo

Dle výstupů filtrační funkce jsem potvrdil odolnost algoritmu pro jevy působící v lokálním i globálním měřítku, pokud jsou po celém poli působení homogenní. Při hodnocení vad povrchu založeném na detekci odchylek jasové funkce musíme počítat s velkou závislostí na její fluktuaci nebo jakýchkoliv změnách působení okolních vlivů. Dopad okolních vlivů je možné potlačit volbou vhodné osvětlovací soustavy s vysokým jasem tak, aby při inspekci povrch vykazoval vyšší jas než kterýkoliv jiný předmět v záběru kamery.

Tabulka 2: Celková úspěšnost algoritmů Matlab

Část	Světlo/stín 1	Světlo/stín 2	Boční světlo 1	Boční světlo 2
Přední panel	79,00 %	79,00 %	93,00 %	71,00 %
Trimr	97,00 %	72,00 %	92,00 %	0,00 %
Led	50,00 %	50,00 %	100,00 %	30,00 %
Jumper	71,00 %	37,00 %	97,00 %	0,00 %
Boční panel	30,00 %	0,00 %	30,00 %	30,00 %

Co se týče celkové úspěsnosti algoritmů v sekci Matlab, tak dle tabulky 2 se ukazuje výrazná závislost veškerých funkcí na detekci hran a potažmo na kontrastu a kvalitě vstupních dat. Celkové zhodnocení poznatků z programování se nachází v poslední kapitole.

3 Závěr

Porovnání systémů

V předchozích kapitolách byly definovány návrhy inspekčních systémů v prostředí InSight Explorer a Matlab pro pracoviště výstupní kontroly, včetně testování jejich robustnosti pod vlivem dvou nejvýznamnějších rušení, které vyplývaly z experimentů. Jedná se o kombinaci světlo/stín, a také vliv rušivého postranního osvětlení.

Z každého testování byly generovány výsledky založené na porovnávání s ideálními stavy algoritmů, které jsem dále koncentroval do tabulek celkové úspěšnosti. K porovnání robustnosti navržených systému navzájem není možné přistoupit pomocí předem definovaných vlivů se stejným dopadem, protože systémy pracují na mírně odlišných principech. Kromě toho je následek rušivých vlivů velmi závislý na prostorovém umístění ovlivněné oblasti. Toto můžeme pozorovat při porovnání tabulek a testovacích snímků zatížených rušivými vlivy.

Z toho důvodu jsem se rozhodl otestovat odolnost algoritmů na pseudonáhodně generovaném šumu v celém vstupním snímku. Použil jsem funkci imnoise z prostředí Matlab a nastavil Gaussovské rozložení šumu. Jako zástupce algoritmů z obou prostředí jsem vybral kompletní inspekci čelního panelu, kde jsou použity všechny zmiňované funkce, a proto můžeme po otestování přesně odhadnout chování funkcí i v ostatních podprogramech.

Tabulka 3: Šumová odolnost algoritmů na Gaussův šum

σ	**0.01**	**0.02**	**0.05**	**0.1**	**0.5**
InSight	85,00 %	85,00 %	73,00 %	73,00 %	69,00 %
Matlab	40,00 %	40,00 %	40,00 %	20,00 %	0,00 %

Dle tabulky 3 vidíme úspěšnost inspekce algoritmů navržených v prostředí InSight Explorer a Matlab v závislosti na rozptylu šumu Gaussova rozložení kolem nuly. Je zřejmé, že průmyslově používané algoritmy budou disponovat vyšší základní odolností vůči rušivým vlivům, avšak tato vlastnost je vyvážena nemožností úplné konfigurace funkcí. Největším problémem byly temné části testovaného snímku, kde byl algoritmus v Matlabu nefunkční buď vůbec nebo ukazoval zcela nesmyslné hodnoty. Například kladnému natočení trimru odpovídaly záporné hodnoty velkého rozptylu, prakticky 90°. Domnívám se, že nejslabším článkem tohoto programu byla právě optimalizace za pomocí generace hranového obrázku, kdy docházelo k detekci hran tak nešťastně, že neuronová síť nebyla schopna identifikovat naprostou většinu zkoumaných komponent, a proto jsem musel přistoupit k analýze podprogramů s použitím ukazatelů generovaných z původních šumem neovlivněných snímků.

Inspekční systém v Matlabu má i přes všechny naměřené hodnoty i svoje výhody. Naprostým kladem je jednoznačně konfigurovatelnost veškerých hlavních i podpůrných funkcí pro dosažení kýženého výsledku, ale za cenu dlouhé doby návrhu každé funkce. Podmínkou pro úspěšné zpracování vstupních dat je pak i jejich kvalita, zejména absence přesvícených nebo tmavých míst.

Inspekční systém v InSight Exploreru je výborná volba pro rychlý návrh jednoduchých sekvencí, s důrazem na kompatibilitu se stávajícími technologiemi na montážní lince. Co se týče vlivu nekvalitních dat na výstupy jednotlivých funkcí, tak dochází k podobným jevům jako v předchozím případě. Algoritmy trpí na nedostatek osvětlení stejně tak, jako na místa se saturovanými pixely a při ladění programu je potřeba dle vstupních dat experimentálně volit parametry jednotlivých funkcí, bez znalosti míry ovlivnění každé z nich.

Osvětlení

Velkou roli při inspekci předmětů v jakémkoliv kamerovém systému hraje světlo, jeho množství a směr ze kterého přichází. Díky správnému nasvícení zkoumaného vzorku eliminujeme většinu vad vznikajících při návrhovém procesu algoritmu. Může jít o strukturní vady algoritmu, které se projevují za určitého osvětlení neošetřenou nefunkční částí programu, a proto se vyplatí definovat alespoň částečně světelné podmínky při simulaci dodáním osvětlovacího zařízení, které následným použitím při ostrém provozu omezí vliv okolního rušení.

Typickým příkladem může být použití osvětlovacího prstence na objektivu kamery, kdy množství světelného toku o mnoho překonává schopnost kamery při odrazu zpět jej zpracovat, a proto je nutné kameru zaclonit. Tímto způsobem se dá relativně snadno omezit vliv odrazů záření do objektivu z okolních zdrojů. Celý systém je potřeba nastavit na konstantní hodnotu světelného toku, stejně tak jako konstantní hodnotu elektronické závěrky a s ní spojenou dobu expozice.

Postranní osvětlení se nám může hodit v případě potřeby detekce povrchu v jiném než kolmém směru od objektivu kamery, avšak tento postup je nutné, pro dobrou funkci použitého algoritmu, dodržet vždy stejný, což není v případě proměnlivého postranního osvětlení zcela možné.

Navázání

Navržený inspekční systém je možno implementovat na různých zařízení díky možnosti generace algoritmů do jazyka C/C++, právě proto uvažuji o možném budoucím rozšíření inspekčního systému a zautomatizování celého procesu.

Pokud jde o samotnou automatickou inspekci s vyhodnocením, tak není při poskytnutí správných dat systému a rozumném využití výstupů systému žádný problém. V případě úvahy automatické výstupní kontroly nahrazující plně lidskou vizuální kontrolu je to ovšem zcela jiná situace. Takový systém musí obsahovat senzory a akční členy usnadňující rozhodovací logice realizovat tak jednoduchou operaci jako je natočení zkoumaného vzorku na požadovanou stranu. Jedním způsobem by mohlo být omezení počtu pozic testovaného vzorku ku inspekčnímu systému použitím všudypřítomných dopravníkových pásů se sloty umožňující přesné usazení. Při úvaze ideální mechanické konstrukce dopravníkového pásu a přídržného mechanismu umožňujícího umístění zkoumaného vzorku do potřebných pozic zbývá k realizaci automatického pozičního systému robotické rameno manipulující se vzorkem dle poskytnutých dat.

Realizace algoritmu umožňující využití dat získaných z prvotního snímku s určením a ovlivněním pozice vzorku a výběr vhodného inspekčního algoritmu (hledání předloh, kontrola zájmových oblastí, inspekce povrchu) by byla jedna z cest navázání na tuto práci. Samotná komunikace s akčními členy je velmi obsáhlá kapitola, avšak při snaze udělat systém co nejuniverzálnější, bych volil některý z využívaných standardů v průmyslu jako CAN, iEthernet nebo GPIB.

Informace o chybovosti zařízení spolu s určením polohy chyby by se dala využít pro tvorbu statistik výskytu chyb a dalšímu rozvoji výrobních postupů tak, aby k chybám nedocházelo. V zásadě by se mohlo jednat o primitivní databázi shromažďující informace ze struktur generovaných mými algoritmy s makry umožňující automatické řazení vad do skupin v závislosti na poloze výskytu, závažnosti, četnosti, druhu a mnoho dalších vlastností. Působení automatického inspekčního systému na výrobní proces by poté vykazovalo znaky strojového učení za pomoci optimalizace výroby dle získaných dat.

Seznam použité literatury

[1] Rafael C. Gonzales, Richard E. Woods. Digital Image Processing. Pearson, 2007. ISBN 978-0131687288

[2] Rafael C.Gonzales et al.Digital Image Processing using Matlab. Pearson Prentice Hall,2003. ISBN 978-0130085191

[3] Cognex InSight software user's manual. Dokument firmy Cognex.

[4] PLC to In-Sight Communications Using EIP. Dokument firmy Cognex.

[5] GREGOR, M. Detektory objektů v obraze a jejich realizace. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 51 s. Vedoucí bakalářské práce Ing. František Kyselý.

[6] ROJAS, Raúl. Neural networks: a systematic introduction. Berlin: Springer, c1996. ISBN 3-540-60505-3.

[7] MATHWORKS, (2018). Neural Network Toolbox: User´s Guide (R2018a). [cit. 2018-04-21]. Dostupné z: http://www.mathworks.com/help/nnet/

[8] Neuron.svg. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation,2001-[cit.2018-04-22].
Dostupné z http://commons.wikimedia.org/wiki/File:Neuron.svg

[9] MATHWORKS, Neuron Model, (2018). Neural Network Toolbox: User´s Guide (R2018a). [cit. 2018-04-21]. Dostupné z: http://www.mathworks.com/help/nnet/ug/neuron-model.html?searchHighlight=neuron%20model&s_tid=doc_srchtitle

[10] MATHWORKS, paternnet, (2018). Neural Network Toolbox: User´s Guide (R2018a). [cit. 2018-04-21]. Dostupné z: http://www.mathworks.com/help/nnet/ref/patternnet.html?searchHighlight=patternnet&s_tid=doc_srchtitle

[11] MATHWORKS, Image Segmentation and Analysis, (2018). Image Processing Toolbox: User´s Guide (R2018a). [cit. 2018-04-21].
Dostupné z: https://www.mathworks.com/help/images/index.html

[12] MATHWORKS, regionprops, (2018). Image Processing Toolbox: User´s Guide (R2018a). [cit. 2018-04-21].
Dostupné z: https://www.mathworks.com/help/images/ref/regionprops.html

[13] SHIRASUKA, Keiichi. Mosaic block detection based on HOG with SVM classifier and template matching. In: 2018 IEEE International Conference on Consumer Electronics (ICCE) [online]. IEEE, 2018, 2018, s. 1-2 [cit. 2018-05-22]. DOI: 10.1109/ICCE.2018.8326081. ISBN 978-1-5386-3025-9. Dostupné z: http://ieeexplore.ieee.org/document/8326081/

[14] REN, Haoyu a Ze-Nian LI. Object detection using edge histogram of oriented gradient. In: 2014 IEEE International Conference on Image Processing (ICIP) [online]. IEEE, 2014, 2014, s. 4057-4061 [cit. 2018-05-22]. DOI: 10.1109/ICIP.2014.7025824. ISBN 978-1-4799-5751-4. Dostupné z: http://ieeexplore.ieee.org/document/7025824/