VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

JEDNODUŠE KONFIGUROVATELNÝ KONTROLNÍ KAMEROVÝ SYSTÉM PRO PRŮMYSLOVÉ APLIKACE

EASILY CONFIGURABLE VISUAL INSPECTION SYSTEM FOR INDUSTRIAL APPLICATIONS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **ONDŘEJ ANDRLA**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. ŠPANĚL MICHAL, Ph.D.**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2020/2021

Zadání bakalářské práce

Student: **Andrla Ondřej**
Program: Informační technologie
Název: **Jednoduše konfigurovatelný kontrolní kamerový systém pro průmyslové aplikace**
Easily Configurable Visual Inspection System for Industrial Applications
Kategorie: Zpracování obrazu

Zadání:

1. Zorientujte se v problematice zpracování obrazu. Zaměřte se na problematiku kalibrace kamery a metody detekce hran a základních geometrických útvarů (elipsa, kružnice, apod.) v obraze.
2. Seznamte se s problematikou detekce objektů v obraze na základě definovaných vzorů (tzv. template matching).
3. Vyberte vhodné metody a nástroje a navrhněte jednoduše konfigurovatelný systém, který umožní definovat a následně automaticky vyhodnocovat kontrolní místa (body, linie, otvory, apod.) a kontrolní metriky (např. vzdálenost a úhel) v záznamu z kamery.
4. Experimentujte s vaší implementací a případně navrhněte vlastní modifikace metod.
5. Porovnejte dosažené výsledky a diskutujte možnosti budoucího vývoje.
6. Vytvořte stručný plakát nebo video prezentující vaši práci, její cíle a výsledky.

Literatura:

- Dle pokynů vedoucího.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Španěl Michal, Ing., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 12. května 2021
Datum schválení: 30. října 2020

Abstrakt

Cílem této práce je vytvořit prototyp jednoduše konfigurovatelného kamerového systému, sloužícího ke kontrole výrobků na automatizované montážní lince. Práce je zvláště zaměřena na problematiku detekce LED. Detekována je jejich samotná přítomnost ve snímku a dále systém vyhodnocuje, zda LED svítí, a pokud ano, tak jakou barvou. Výsledné řešení umožňuje uživateli jednoduše definovat kontrolní metriky pro vyhledávaní kruhových objektů v obraze, a to pomocí konfigurační aplikace s grafickým uživatelským rozhraním. Příslušná konfigurace se ukládá do souboru ve formátu JSON. Dále byl implementován runtime, který na základě konfigurace samostatně zpracovává záznam z kamery, kontroluje výrobky a podává o nich zprávy dalším součástem systému. K detekci objektů v obraze systém využívá Houghovu kruhovou transformaci. K otestování efektivnosti systému byl vytvořen datový soubor za pomoci nástroje FitKit s maticí LED. Při testování runtime na datovém souboru se podařilo dosáhnout celkové správnosti detekce LED 97,79%.

Abstract

The aim of this thesis is to create a prototype of an easily configurable camera system, used to control products on an automated assembly line. The thesis is especially focused on the issue of LED detection. Their very presence in the image is detected and the system also evaluates whether the LED is lit and, if so, in what color. The final solution allows the user to easily define control metric for searching for circular objects in the image in a configuration application with a graphical user interface. The configuration is saved in a JSON file. Furthermore, a runtime which independently processes the recording from the camera, checks the products and reports on them to other components of the system based on the configuration was implemented. The system uses the Hough circular transformation to detect objects in the image. To test the efficiency of the system, a dataset was created using the FitKit tool with a LED matrix. When testing the runtime on the dataset, the overall accuracy of LED detection was 97,79%.

Klíčová slova

vizuální inspekce, kamerový systém, Houghova kruhová transformace, detekce LED

Keywords

visual inspection, camera system, Hough circle transformation, detection of LED

Citace

ANDRLA, Ondřej. *Jednoduše konfigurovatelný kontrolní kamerový systém pro průmyslové aplikace.* Brno, 2021. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Španěl Michal, Ph.D.

Jednoduše konfigurovatelný kontrolní kamerový systém pro průmyslové aplikace

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Michala Španěla, Ph.D. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Ondřej Andrla
7. května 2021

Poděkování

Děkuji vedoucímu práce panu Ing. Michalu Španělovi, Ph.D., za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci. Také děkuji Ing. Michalu Bidlovi, Ph.D. za zapůjčení hardwaru FitKit3 – Minerva.

Obsah

1 Úvod **2**

2 Automatická vizuální kontrola – shrnutí současného stavu **4**
2.1 Princip automatické vizuální inspekce 6
2.2 Průzkum současných řešení 7

3 Metody zpracování obrazu pro vizuální inspekci **14**
3.1 Detekce kruhových objektů v obraze pomocí Houghovy transformace 14
3.2 Kalibrace kamery 17

4 Návrh řešení jednoduše konfigurovatelného kontrolního kamerového systému **19**
4.1 Konfigurační GUI aplikace 20
4.2 Runtime pro vyhodnocení snímků 22

5 Implementace jednoduše konfigurovatelného kontrolního kamerového systému **23**
5.1 Konfigurační GUI aplikace 23
5.2 Přenos konfigurace a zobrazování výsledků 25
5.3 Runtime pro vyhodnocení snímků 26

6 Testování jednoduše konfigurovatelného kontrolního kamerového systému **29**
6.1 Uživatelské testování GUI konfigurační aplikace 29
6.2 Testování runtime pro vyhodnocení snímků 32

7 Závěr **36**

Literatura **37**

A Plakát **39**

B Dotazník **41**

C Obsah paměťového média **42**

Kapitola 1

Úvod

Cílem této práce je vytvořit jednoduše konfigurovatelný kontrolní kamerový systém, využitelný pro automatickou kontrolu kvality výrobků na robotizované montážní lince.

Automatizace průmyslu klade stále vyšší požadavky na přesnost a rychlost vizuální inspekce produktů. Manuální kontrola prováděná lidským inspektorem je pro tento účel z ekonomického hlediska neefektivní. Automatizace kontroly kvality pomocí kamerového systému umožní zrychlení a zpřesnění kontroly kvality výrobků. Z těchto důvodů panuje v současné době velký zájem o software pro automatizovanou kontrolu kvality výrobku.

Cílem této práce je tedy navrhnout a následně implementovat takovýto kontrolní kamerový systém. Důležitou vlastností výsledného produktu je to, že se bude jednat o jednoduše konfigurovatelný kamerový systém. To znamená, že uživatel bude mít možnost snadno definovat a měnit kontrolní metriky. Proto je kromě samotného nástroje pro analýzu snímků z montážní linky realizována i aplikace pro definici kontrolních úloh s grafickým uživatelským rozhraním (GUI) V ní může uživatel definovat konfiguraci úloh pro detekci objektů v obraze. GUI by mělo být intuitivní a má být navrženo takovým způsobem, aby s ním mohl snadno pracovat i nepříliš zkušený zaměstnanec. Výsledný systém má být jednoduchý a a snadno ovladatelný.

V práci jsem se rozhodl zvláště zaměřit na detekci LED (Light-Emitting Diode). Bude detekován samotný výskyt LED i to, jestli svítí a jakou barvou. Důvod zaměření se na detekci LED je následující. Běžné kontroly pomocí automatické vizuální inspekce se často soustřeďují na odhalení povrchového poškození výrobku. Vadu, skrývající se pod povrchem, však odhalit nedokážou. Díky kontrole signalizačních LED je však možno takový defekt odhalit. Pokud LED nesvítí, nebo svítí jiné LEDky, než by správně měly, může to signalizovat poškození výrobku. Nástroj předává informaci o nestandardním chování LED řídícímu jádru systému, které rozhodne, jakým způsobem bude s takovým produktem naloženo.

Práce se v teoretické části věnuje zhodnocení současného stavu řešení nástrojů pro vizuální inspekci, přičemž se zaměřuje zvláště na porovnaní několika vybraných komerčních nástrojů, z nichž jsem čerpal inspiraci pro implementaci mnou navrženého systému 2.2. Sleduje u nich to, jakým způsobem umožňují uživatelům zadefinovat v obraze kontrolní metriky. Také u těchto systémů vzájemně porovnávám jejich přednosti a nedostatky, jak v oblasti přesnosti detekce, tak v intuitivnosti uživatelského rozhraní, a to zvláště s ohledem na detekci kruhových objektů. Text se následně kapitole 3 věnuje popisu metod pro detekci objektů v obraze. Vzhledem k tomu, že se práce primárně soustřeďuje na detekci LED, je hlavní pozornost věnována detekci kruhových objektů. Zvláštní zřetel pak patří metodě Houghovy kruhové transformace.

Posléze se text soustřeďuje na popis návrhu a realizaci výsledného produktu 4. Navrhovaný systém sestává ze dvou částí, totiž z Grafického uživatelského rozhraní (GUI) a runtime pro detekci objektů v obraze. Díky GUI má uživatel možnost samostatně definovat kontrolní metriky, pro detekci objektů. Uživatel si při práci načte vzorový snímek a v něm poté nastavuje, co bude kontrolováno. Toto se uloží do konfiguračního souboru ve formátu JSON. Ten je po spuštění provozu montážní linky předán runtime. Runtime pracuje samostatně, bez přítomnosti uživatele. Kromě konfiguračního souboru si načte i snímky z přednastavené kamery. V pevně stanoveném časovém intervalu načítá snímky a provádí v nich detekci objektů na základě kontrolních metrik. Výsledky zapisuje do souboru, který je předán dalším částem systému, které na jeho základě rozhodnou, zda je kontrolovaný výrobek poškozen a je jej tedy nutné vyřadit z výrobního procesu. Implementace tohoto systémů je podrobně popsána v kapitole 5, testování pak v kapitole 6.

Kapitola 2

Automatická vizuální kontrola – shrnutí současného stavu

Automatická vizuální inspekce (AVI) je metoda strojového zpracování snímků z kamery, sloužící ke kontrole kvality produktů. Je široce využívána na montážních linkách nejrůznějších průmyslových odvětví. Ve většině výrobních odvětví je cílem dosáhnout 100% kvality výrobků. Kontrola produktů je tudíž velmi důležitým krokem v rámci výrobního procesu. Včasné odhalení a odstranění poškozených produktů z výroby, zajišťuje výrazné snížení nákladů a zvýšení produktivity.

Dosáhnout toho, aby kvalita každého produktu splňovala požadovanou normu, bývá velmi náročný úkol, který je však nezbytný v mnoha odvětvích. Mezi hlavní oblasti využití automatické vizuální kontroly patří kontrola balení výrobků, jako jsou například zdravotnické obaly, plastové lahve od nápojů, nebo plechovky od potravin. Dále se automatická vizuální inspekce běžně používá v oděvním průmyslu, nebo ke kontrole automobilů, elektroniky, polovodičových destiček, čipů, atd. Důležité je taktéž odhalení toho, zda nedošlo k poškození povrchu nějakého produktu, například u kovových dílů, textilu, dřevěných desek atd. Proces kontroly zahrnuje validaci celistvosti sestavy, povrchové úpravy produktu a jeho geometrických rozměrů [15].

Vizuální inspekce a kontrola kvality se často stále ještě řeší lidskými odborníky. I když lidé v mnoha případech zvládnou práci lépe než stroje, bývají zpravidla pomalejší a rychle se unaví. Vizuální kontrola je totiž monotonní práce a po určitém čase u inspektorů klesá pozornost. Ve srovnání se stroji je tedy pracovní doba lidských inspektorů relativně krátká. Kvalifikované odborníky bývá navíc často obtížné najít, nebo udržet na žádané pozici, vyžadují školení a jejich rozvoj může nějakou dobu trvat. Existují také případy produktů, u kterých inspekce bývá velmi zdlouhavá nebo obtížná, a to i pro nejlépe vyškolené experty. Při kontrole rychle se pohybujících výrobků na montážní lince hrozí, že lidský inspektor některý defekt přehlédne. V určitých aplikacích musí být přesné informace rychle nebo opakovaně extrahovány a použity. V některých prostředích (např. Inspekce pod vodou, jaderný průmysl, chemický průmysl atd.) může být inspekce obtížná nebo nebezpečná. Počítačové vidění však může snadno v takovýchto náročných případech nahradit lidskou inspekci [11]. Nasazení automatických systémů pro kontrolu kvality tedy zaměstnavatelům šetří náklady na mzdách a přitom umožňuje v mnoha případech dosáhnout rychlejších a kvalitnějších výsledků.

Pokroky v technologii výrobních zařízeních vedly k vývoji levnějších průmyslových systémů vizuální kontroly, což umožňuje jejich širší nasazení. Díky rychlému rozvoji výpočetní

techniky a zařízení pro snímání digitálního obrazu, ke kterému dochází v posledních desetiletích, je dnes možné, aby byly na montážních linkách realizovány systémy optické kontroly v reálném čase a zároveň byly konzistentní, robustní a spolehlivé. Automatická detekce objektů se v posledních letech značně zpřesnila díky pokroku technologií, jako jsou kamerové systémy a senzory, hardware řídících počítačů, nebo software pro definici konfigurace [10]. Širší nasazení těchto produktů klade nároky na školení nových pracovníků. Poptávka je dnes proto zvláště po systémech, které jsou intuitivní a mají minimální požadavky na školení zaměstnanců, kteří s nimi budou pracovat [15].

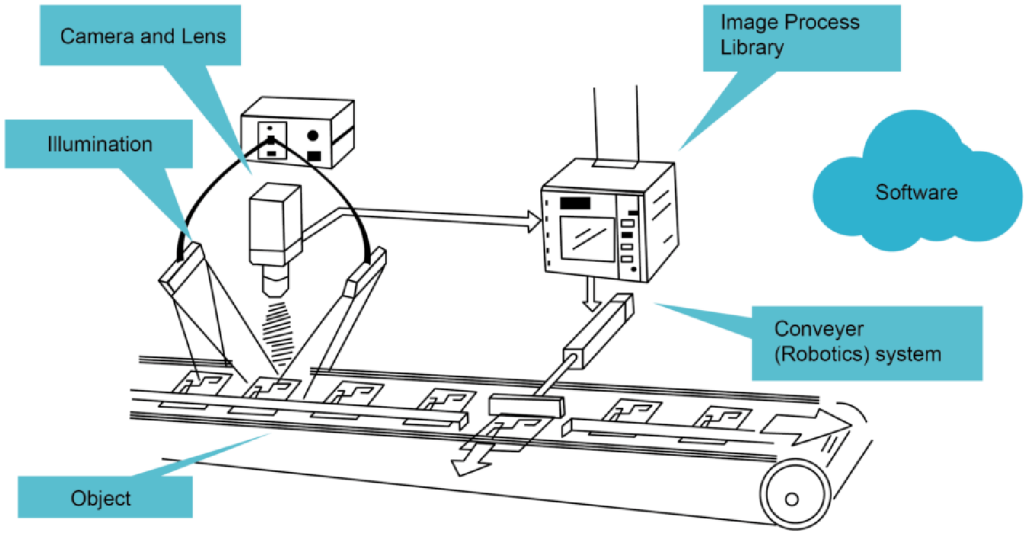
Systém průmyslového vidění pracující v reálném čase musí být dostatečně rychlý, aby stíhal sledovat pohyb produktů na montážní lince. Rychlost závisí na úloze, která má být splněna. Pokud jsou některé konfigurační úlohy příliš složité, je pro jejich zpracování v požadovaném čase zapotřebí speciální hardware. Pokud je aplikace navržena tak, že dokáže splnit rychlostní požadavky linky bez potřeby speciálního hardwaru, přináší to firmě výraznou finanční úsporu [11]. Příklad toho, jak vizuální inspekce v praxi můžeme vidět na obrázku 2.1.

Obrázek 2.1: ukazuje příklad běžného využití automatické vizuální inspekce. Na výrobní lince se v tomto případě nacházejí plastové lahve s nápoji, které jsou snímány kamerou. Vedle linky vidíme uživatele s tabletem na němž běží aplikace s GUI pro definici konfigurace kontrolního systému. Z obrázku vyplývá, že v tomto případě systém kontroluje to, zda jsou zátky lahví správně zašroubovány. Převzato z [4].

2.1 Princip automatické vizuální inspekce

Tato sekce popisuje, jakým způsobem je automatická kontrola kvality v praxi běžně používána. Soustředím se zejména na popis kontroly na montážní lince. Schéma takovéto montážní linky se nachází na obrázku 2.2. Výrobky se pohybují podél dopravníku a digitální obraz každého z nich je zachycen videokamerou. Video je snímáno buď jednou, nebo i více kamerami umístěnými nad linkou. Pozice kamer jsou obvykle pevné. Jejich správné nastavení je velmi důležité, neboť účinnost klasifikace závisí také na kvalitě získaných obrázků [12].

Systém průmyslového vidění musí být dostatečně robustní. Měl by být schopný se automaticky přizpůsobovat okolnostem a dosahovat trvale vysokého výkonu navzdory nepravidelnostem v osvětlení a přizpůsobit tomu, že výrobek může být mírně posunutý, nebo špatně natočený oproti původnímu očekávání. Robustního výkonu bývá obtížné dosáhnout [10]. Ve většině případů jsou totiž systémy průmyslové automatizace navrženy tak, aby kontrolovaly pouze známé objekty na pevných pozicích. Scéna musí být vhodně osvětlena a uspořádána, aby se usnadnil příjem obrazových prvků nezbytných pro zpracování a klasifikaci. Vysoké přesnosti lze dosáhnout pouze za podmínek dobrého osvětlení a nízkého šumu v obraze [11].



Obrázek 2.2: ilustruje strukturu typického systému průmyslové aplikace automatické vizuální inspekce. Vidíme na něm schéma montážní linky osazené kamerovým systémem vybaveným speciálním osvětlením. Snímky jím pořízené se přenášejí do počítače, který v nich provede detekci a na základě ní rozhodne, zda je výrobek defektní a bude z linky odstraněn, nebo bude propuštěn dále. Odstranění výrobku je zajištěno pomocí robotického ramene. Převzato z [1].

Pro zpracování získaných obrazů je určen počítač, který využívá speciální software, navržený k tomuto účelu. Obraz je předzpracován a poté je v něm provedena detekce objektů. Na jejím základě počítač rozhodne, zda je některý z produktů vadný. Rozsah jakékoli vady může být odstupňován. Výrobek může být prohlášen za zmetek, pokud některá ze

závad překročí stanovený stupeň, nebo je na něm objevena kombinace několika menších závad. Počítač následně předává signál robotickému rameni, nebo jiné mechanické součástce automatizované linky, která vadný produkt odstraní z pásu. Poté počítač zpracovává další obrázek a celý proces se opakuje po potenciálně neomezenou dobu. Systém může komunikovat i s dalšími externími zařízeními (např. prostřednictvím sítě, nebo jiného typu rozhraní, jako je FireWire), nebo informace zasílat k vyhodnocení centrálním systémům [7].

Automatickou vizuální inspekci je možno využívat dvěma základními způsoby. Jednak jako prostředek k zajištění kontroly kvality pomocí odstranění detektivních produktů z montážní linky, jak bylo popsáno výše. Druhou možností je, že automatizovaný kontrolní systém bude sloužit jako prostředek pro shromažďování statistických informací o kvalitě výrobků, za účelem poskytnutí zpětné vazby k výrobnímu procesu. Na jejím základě bude následně možno posoudit efektivitu výrobního procesu a případně v něm provést změny [7].

Pro provoz v reálném čase musí být proces kontroly schopen držet krok s výrobním procesem. Toto je často nejnáročnější úkol při navrhování systému automatické vizuální kontroly, protože produkty se mohou na pásu pohybovat velkou rychlostí. Jedním z nejkritičtějších časovacích měření je doba cyklu mezi prezentací následujících obrazů vyžadujících analýzu (obvykle to bývá úměrné rychlosti dopravníku). Toto období poskytuje horní mez přípustného času zpracování pro jeden obrázek. Pokud zpracování trvá déle je možné, že produkt proklouzne aniž by byl zkontrolován [17].

Když je proces velmi časově náročný nebo výpočetně intenzivní a překračuje možnosti zpracování hlavního procesoru, použije se ke zmírnění problému s rychlostí zpracování specifický aplikační hardware (např. DSP, ASIC nebo FPGA) [11].

Aby byl systém spolehlivý, musí co nejvíce snižovat míru úniku (tj. neodhalené vadné výrobky) a falešné poplachy (tj. označení produktu, který je v pořádku, jako vadného). Systém průmyslového vidění musí být také rychlý a rentabilní.

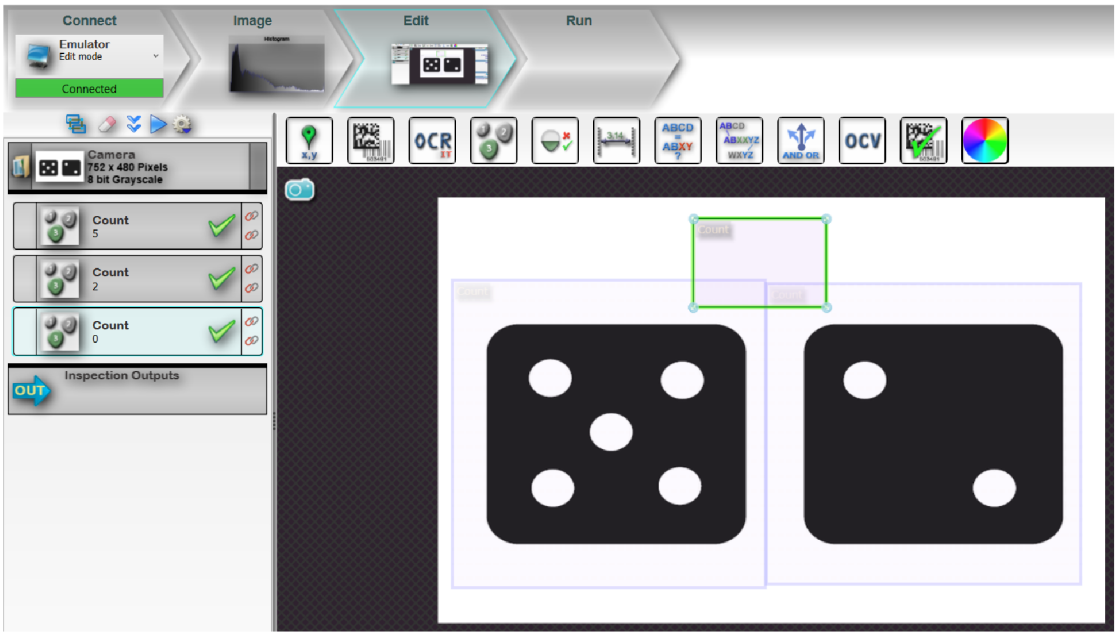
2.2 Průzkum současných řešení

V předcházejících oddílech jsem se soustředil zvláště na popis obecných trendů ve vizuální inspekci. Také byly popsány praktické aspekty jejich nasazení ve výrobním provozu. V této kapitole se zaměřuji na porovnání aktuálně používaných desktopových konfiguračních aplikací vybavených uživatelským rozhraním. Zvláště se soustředím na jejich funkcionalitu a srozumitelnost jejich grafických uživatelských rozhraní. K porovnání jsem si vybral tři komerční systémy, které jsou v praxi používány v průmyslových aplikacích. Cílem tohoto srovnání je získání znalosti pro návrh mé vlastní aplikace. Proto se soustředím také na to, jakým způsobem umožňuji tyto aplikace detekci kruhových objektů. Informace o funkcionalitě těchto systémů jsem čerpal z technických popisů na internetových stránkách firem, které je vyrábí a z instruktážních videí, na nich umístěných. Produkt Microscan AutoVISION si na rozdíl od ostatních bylo možné stáhnout a vyzkoušet si pak jeho používání i prakticky.

Microscan AutoVISION

Microscan AutoVISION je produktem firmy Omron Microscan [4]. Na obrázku 2.3 je zobrazen vzhled daného uživatelského rozhraní. Toto uživatelské rozhraní umožňuje uživateli se přepnout do čtyř módů. První a druhý slouží k připojení se k zařízení, konfiguraci hardwaru a načtení obrázku, se kterým se bude pracovat v editačním módu. Editační mód slouží k samotnému programování úlohy. Ve čtvrtém módu může uživatel spustit úlohu a následně

sledovat a vyhodnocovat výsledky. Firma ve svém popisu nástroje opakovaně poukazuje na to, že je jejich uživatelské rozhraní velmi intuitivní a uživatelsky přívětivé [4].



Obrázek 2.3: popisuje vzhled vývojového prostředí Microscan AutoVISION. V horní části obrázku se nacházejí čtyři tlačítka: dvě pro načtení kamery a snímku, další pro přepnutí do editačního módu a tlačítko ke spuštění detekce. Pod nimi je dvanáct menších ikon, s jejichž pomocí uživatel definuje kontrolní metriky. Seznam již uživatelem zadefinovaných metrik jde vidět v levé části obrázku. Většinu plochy zabírá samotný snímek výrobku. V něm uživatel vidí pozici definovaných metrik a pomocí myši je může upravovat. Převzato z [4].

Prvním krokem při práci s nástrojem Microscan AutoVISION, je nastavení kamery pro příjem obrazu. Připojení k systému PLC lze vytvořit v GUI kliknutím na tlačítko Link. Nástroj po spuštění automaticky detekuje všechna kompatibilní zařízení. Z nich je následně získán obrázek, v němž uživatel může definovat kontrolní metriky. U obrázku je možno upravit expozici a osvětlení. Uživatel může taktéž použít režim Emulátor, který umožňuje načíst obrázky uložené v počítači [4]. V horní části GUI, nad editovaným snímkem, se nachází řada malých ikon, s jejichž pomocí je možno definovat konfigurační úlohy.

Obrázek 2.4: zobrazuje jednotlivé nástroje vývojového prostředí Microscan AutoVISION. Jedná se o zvětšený výřez o obrázku 2.3. Pomocí těchto dvanácti ikon, může uživatel definuje kontrolní metriky. Převzato z [4].

Funkcionalita nástrojů je následující:

- První z nich je nástroj Lokace, který slouží k nalezení definovaného objektu. Je jej možno používat v kombinaci s ostatními nástroji.

- Druhým je nástroj Decode. Ten slouží ke čtení a dekódování čárového kódu. Další nástroj zvaný OCR se používá ke čtení psaného textu. Využívá pokročilé algoritmy IntelliText, díky nimž dokáže detekovat těžko čitelné znaky. V oblasti detekce znaků obsahuje kompletní sada nástrojů AutoVISION mimo jiné výkonné dekódovací funkce X-Mode Omron Microscan pro čtení 1D a 2D symbolů [4].

- Čtvrté z tlačítek volá nástroj pro počítání objektů. Ten detekuje hledané předměty na základě schody pixelů s předem definovaným vzorem. Uživatel může nastavit kolik objektů chce detekovat, přičemž definuje horní a dolní hranici. Tento nástroj také umožňuje měření počtu pixelů v zadané oblasti.

- Další z nástrojů pracuje podobným způsobem. Narozdíl od předchozího se nezaměřuje na zjišťování počtu předmětů, ale dokáže rozpoznat, zda se předmět v uživatelem specifikované oblasti nachází, nebo v ní chybí.

- Nástroj pro měření dokáže změřit vzdálenosti mezi dvěma zadaným objekty. Vzdálenost je počítána v pixelech. Uživatel může nastavit minimální a maximální toleranci pro vzdálenost. Tento nástroj také umožňuje měřit, v jakém úhlu jsou objektu v obraze natočeny.

- Výstupy jednotlivých funkcí je možno vzájemně kombinovat a poté vyhodnocovat. Například jeden z nástrojů dokáže porovnat schodu textových řetězců, které jsou výstupem po aplikaci funkcionality jiných nástrojů, například nástrojů pro dekódování textu a OCR. Nástroj pro formátování řetězců dokáže upravovat texty získané jinými nástroji. Nástroj pro logiku získává výsledky ostatních metrik a na základě jejich zkombinování a zpracování vyhodnocuje, zda produkt splňuje specifikaci či nikoliv.

- Nakonec Microscan AutoVISION poskytuje uživateli dva nástroje pro verifikaci a to jednak OCV, sloužící pro verifikaci kvality psaného textu, a druhý používaný k verifikaci 1D a 2D symbolů. Uživatel zadefinuje oblast kolem kontrolovaného objektu a poté systém zajistí natrénování metriky na rozpoznávání daného objektu.

Systém Microscan se především zaměřuje na dekódovaní čárových kódů, QR kódů a tištěného textu. Pro detekci kruhových objektů je pak možno využít nástrojů pro počítání objektů a pro zjištění výskytu objektu ve vymezené oblasti. Také je možno používat nástroj pro měření vzdálenosti a úhlů.

Práce uživatele s GUI probíhá následujícím způsobem. Uživatel klikne na ikonu nástroje a poté se v obrázku objeví obdélníkové ohraničení oblasti, ve které bude nástroj pracovat. Uživatel může přetažením myši tuto plochu posouvat po obraze a měnit její velikost. V levé části obrazovky může sledovat seznam definovaných úloh a po kliknutí na ně je dodatečně upravovat a zadávat podrobnější specifikaci.

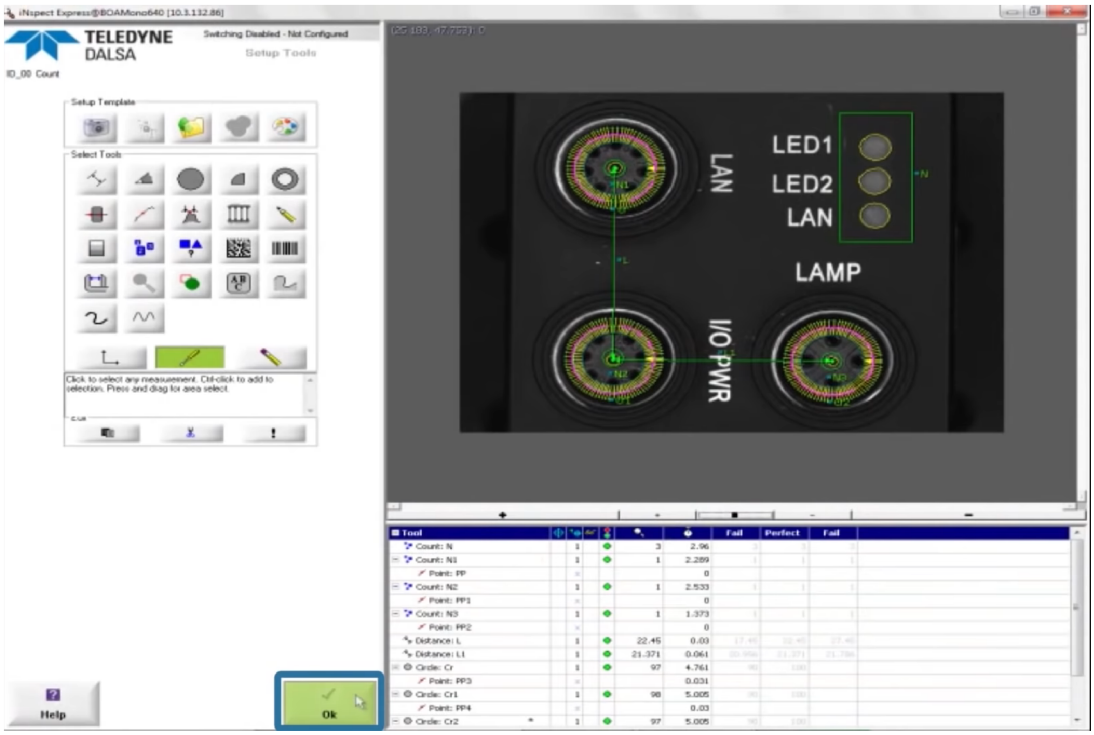
Pro komunikaci s dalšími částmi systému používá aplikace software CloudLink Web HMI. CloudLink poskytuje nástroje pro vizualizaci dat AutoVISION, včetně plně přizpůsobitelného rozhraní CloudLink Dashboard. CloudLink Dashboard poskytuje v reálném čase pohled na hodnoty propojených nástrojů a obrázky z kompatibilních kamer a kamerových systémů AutoVISION. Využívá webový prohlížeč k zobrazování výsledků v přizpůsobitelném okně na jakémkoli zařízení s povoleným prohlížečem, včetně chytrých telefonů a tabletů [4].

iNspect

Nástroj iNspect je produktem společnosti Teledyne Dalsa. Je používán v kombinaci s kamerou BOA Smart Camera, vyráběnou stejnou společností. iNspect je aplikační software, který je speciálně navržen pro zjednodušení návrhu a nasazení automatizované kontroly v továrně. Nabízí novým i zkušeným uživatelům praktický nástroj poskytující funkce, které lze snadno aplikovat na širokou škálu výrobních úkolů. Firma o produktu říká, že jeho jednoduché a přímočaré ovládání umožňuje uživatelům aplikaci rychle konfigurovat a nastavit kontrolní úlohy. Není při tomto procesu nutné žádné programování, ani není potřeba rozsáhlé školení zaměstnanců [2].

Načítání obrázku a spouštění testů probíhá podobně jako u produktu Microscan AutoVISION 2.2. Před samotným spuštěním editačního módu, musí uživatel nastavit adresu vstupní kamery, u které může specifikovat rychlost snímání obrázků. Také zde má možnost nastavit jas a kontrast načteného snímku.

Definice úloh opět probíhá podobným způsobem jako u Microscan AutoVISION 2.2. Úlohy jsou zadávány pomocí kliknutí myší do obrazu a následně může uživatel takto definované okno zvětšovat, nebo zmenšovat a přesouvat po snímku. Na obrázku 2.5 je ukázán vzhled daného uživatelského rozhraní.



Obrázek 2.5: popisuje vzhled vývojového prostředí iNspect. V levé části se nachází pět ikon pro manipulaci s kamerou a s výsledným konfiguračním souborem. Pod nimi je dvaadvacet dalších ikon, s jejichž pomocí uživatel definuje kontrolní metriky. Většinu plochy zabírá obrázek výrobku, do nějž uživatel metriky zanáší. Seznam již definovaných úloh jde vidět v dolní části snímku. Převzato z [2].

iNspect umožňuje detekci různých objektů, jako například:

- Detekci chybějících nebo nesprávných součástek v balení nebo sestavě. Systém například zjišťuje přítomnost a integritu pilulek v blistru. K tomu používá funkcí pro vyhledávání a počítání objektů, nebo prvků jednoho objektu.
- Systém ověřujte korektnost produktů a jejich komponent například, zda čárový kód, nebo tištěný text na výrobku odpovídá specifikaci. K tomu využívá speciálních funkcí pro rozpoznávání QR kódů a pinů. Také disponuje funkcí pro rozpoznání písmen a pro porovnávání textových řetězců.
- Dále umožňuje například zkontrolovat polohu štítku na obalu od potravin. Také sleduje míru naplnění lahví, utěsnění uzávěrů a bezpečnostní pečeti na lahvích.
- Umožňuje též kontrolovat zarovnání desek plošných spojů. K tomu využívá funkcí pro měření úhlů, rozpoznání tvarů, počítání rohů, nebo posouzení tvaru okraje objektu.
- Produkt detekujte krátké výstřely v plastových lisovaných dílech. K tomu využívá nástroje pro rozpoznání barevné intenzity šedotónového obrazu [2].

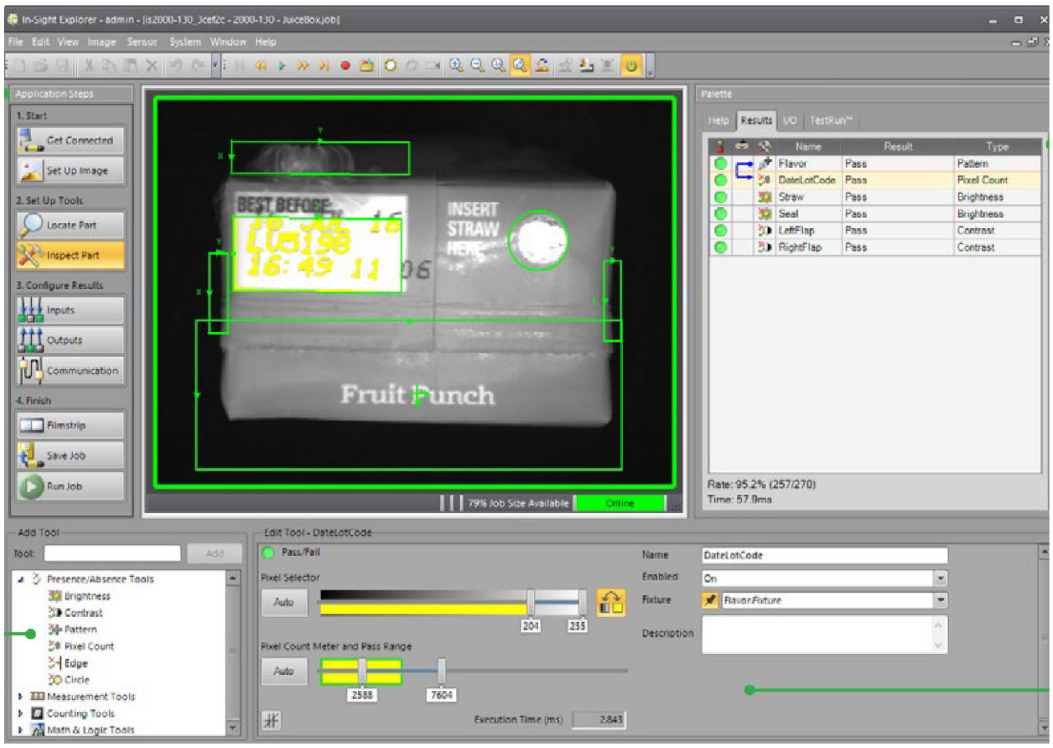
iNspekt disponuje dohromady více než dvaceti nástroji pro definici kontrolních metrik. Pro detekci kruhových objektů jsou obzvláště užitečné tyto funkce:

- Speciální funkce pro nalezení kruhů a jejich středů, nalezení kruhové výseče, a dvojice kruhů vložených do sebe.
- Nástroj pro počítání objektů v regionu. Ten je možný použít k ověření, zda počet nalezených kruhů v dané oblasti odpovídá specifikaci.
- Funkce pro měření vzdáleností. Tu je možno využít jak k posouzení distance mezi středy dvou nalezených kruhů, tak mezi jejich okraji. Také lze počítat vzdálenost středu kruhu od jeho okraje.
- Funkce pro rozpoznání barev. Ta může být velmi užitečná pro posouzení toho, kterou barvou LED svítí. Červená barva LED může signalizovat jiné chování výrobku než například žlutá.
- iNspekt také disponuje speciálními nástroji pro ověření charakteru kruhu, které využívají histogramu pixelů uvnitř kruhu a histogramu pixelů poblíž kružnice.

In-Sight 2000

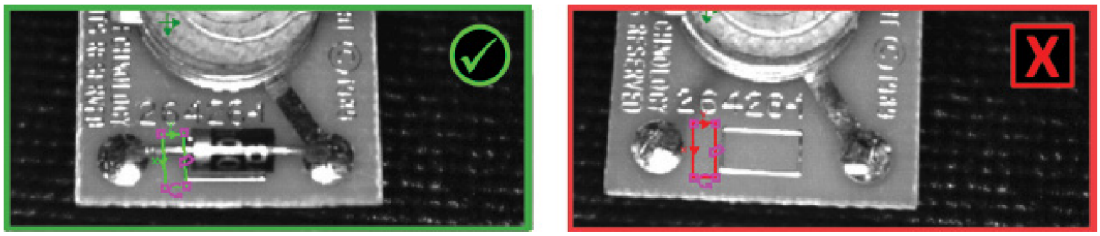
Kamerový systém In-Sight 2000 slouží k nastavení a monitorování inspekčních úloh pro strojového vidění. Způsob používání tohoto kamerového systému je podobný, jako u dvou předchozích kamerových systémů, o niž jsem pojednal výše. K nastavení konfigurace používá In-Sight uživatelské rozhraní EasyBuilder. Toto rozhraní se podle výrobce vyznačuje velkou intuitivností a provede uživatele procesem nastavování kontrolních úloh krok za krokem a umožní mu tak snadný vývoj aplikace. S EasyBuilderem mohou snadno pracovat i začínající uživatelé. Pomůcka TestRun pro In-Sight Explorer pomáhá zajistit, aby spolehlivost inspekce neovlivnila ani změna podmínek při snímání na montážní lince [3]. Na obrázku 2.6 je ukázán vzhled daného uživatelského rozhraní.

In-Sight 2000 může kontrolovat více cílů v rámci jednoho obrazu a provádět zároveň několik inspekcí na základě různých kontrolních metrik. Jsou to například:



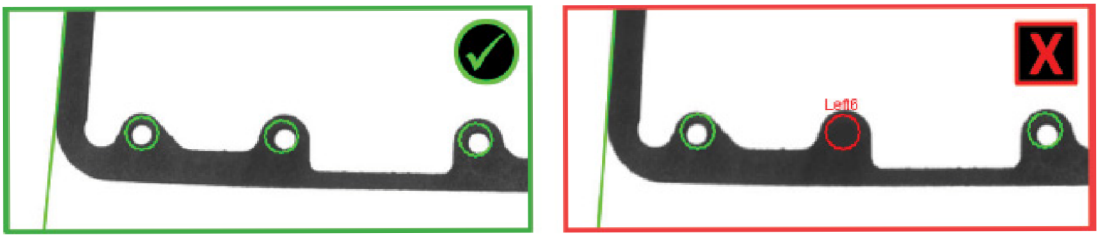
Obrázek 2.6: popisuje vzhled vývojového prostředí In-Sight 2000. Ikony pro definování metrik jsou v horní části, ty pro práci se souborem se nacházejí vlevo a v pravé části obrázku je seznam definovaných úloh. V dolní části se nachází podrobnější specifikace aktuální úlohy. Převzato z [3]

- Metriky pro lokalizaci a přítomnost nebo nepřítomnost hledané komponenty. Například nástroj Pattern při praktickém použití ověří, zda byl do kartonu přidán správný počet sešívacích proužků [3].
- Funkce pro vyhodnocení jasu a kontrastu umožňují validovat kvalitu některých komponent. Například pomocí nástroje Jas jde poznat přítomnost pilulek v blistru. Nástroj Kontrast umožňuje v cílové oblasti identifikovat přítomnost diody. Výsledek této operace je ukázán na obrázku 2.7.



Obrázek 2.7: ukazuje výsledek použití nástroje Kontrast. V levé části jde vidět detekovaná dioda, v pravé části pak chybí. Převzato z [3]

- Funkce pro stanovení hodnoty barvy pixelů potvrzuje přítomnost samostatných děr v cílovém objektu. Výsledek funkce je ukázán na obrázku 2.8. Tímto způsobem lze také například posoudit integritu balení.



Obrázek 2.8: ukazuje výsledek použití nástroje pro detekci barvy pixelů. V levé části jdou vidět tři detekovaná díry, v pravé části pak oproti specifikaci jedna chybí a proto je výrobek vyhodnocen jako vadný. Převzato z [3]

Kapitola 3

Metody zpracování obrazu pro vizuální inspekci

Tato kapitola je věnována popisu metod, které jsem použil při realizaci systému zaměřeného na detekci kruhových objektů v obraze. Zvláště se zaměřuje na popis principu fungování Houghovy kruhové transformace, kterou jsem se rozhodl v práci používat.

3.1 Detekce kruhových objektů v obraze pomocí Houghovy transformace

Houghova transformace se používá pro detekci objektů ve snímku. Původně byla používána pro detekci linií, později se začala často používat i pro nalezení kružnic a elips. Je ji možno využít i pro detekci dalších objektů.

Vstupem Houghovy transformace bývá binární obraz, který byl předtím předzpracovaný hranovým detektorem. Předzpracování přináší několik výhod. Jednak snižuje výpočetní náročnost celé operace díky tomu, že sníží množství hran, které bude potřeba zpracovávat. Také se díky tomu zvyšuje pravděpodobnost nalezení hledané struktury v obrazu. Výstupem Houghovy transformace je parametrický prostor, který obsahuje informace o pravděpodobnosti výskytu kandidátů hledané struktury. Tento prostor je nazýván jako Houghův prostor, nebo také jako akumulátor. To kolik rozměrů bude daný akumulátor mít, určuje počet neznámých parametrů v rovnici tvaru, který je v obraze vyhledáván. V případě hledání kruhových objektů se jedná o tři rozměry, je tedy nutné použit trojrozměrný akumulátor [14].

Parametrické vyjádření kružnice

V této podkapitole popisuji, jakými způsoby je kružnice vyjádřena. Tyto informace pomohou s pochopením fungování Houghovy kruhové transformace.

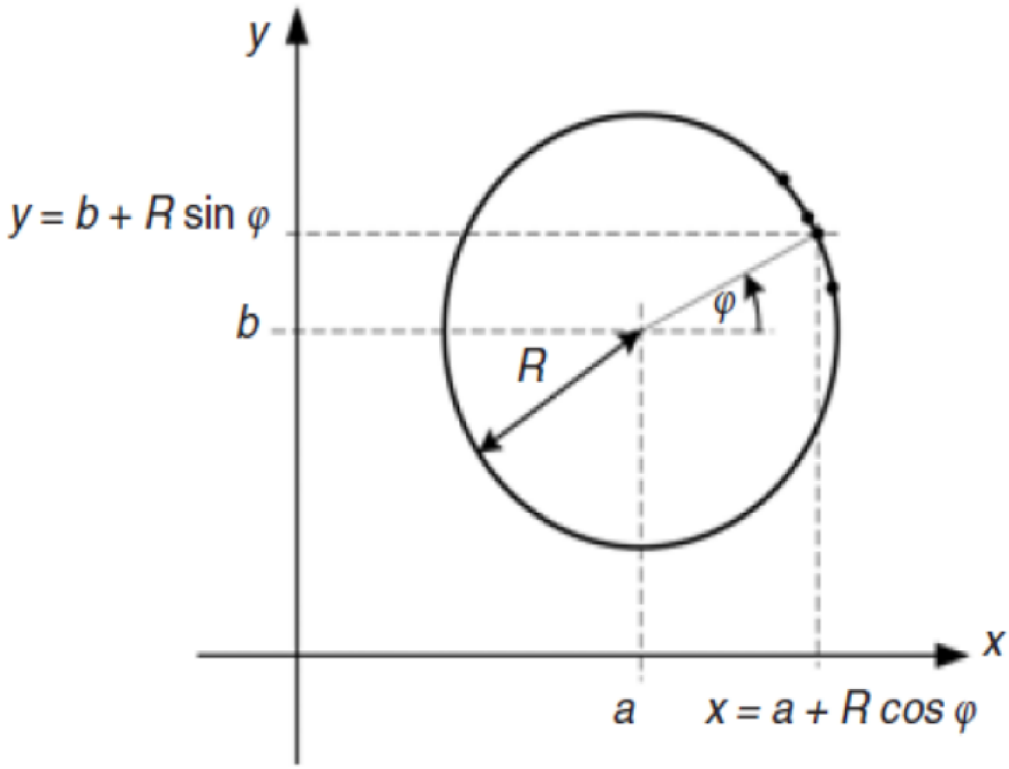
Analyticky je kružnice vyjádřena ve tvaru (3.1).

$$r^2 = (x - a)^2 + (y - b)^2 \tag{3.1}$$

kde r je poloměr kružnice. Parametry a a b představují souřadnice středu kružnice na osách x a y. Jak je zjevné, v rovnici jsou tři neznámé a tudíž parametrický prostor při provádění Houghovy kruhové transformace bude mít tři rozměry [18]. Dle parametrického vyjádření kružnice můžeme polohu bodů na kružnici popsat vztahem

$$\begin{aligned} x &= a + r\cos\varphi \\ y &= b + r\sin\varphi \end{aligned} \tag{3.2}$$

Vzhled této kružnice je vizualizována obrázkem 3.1.



Obrázek 3.1: zobrazuje vzhled kružnice popsané parametrickým vyjádřením. Převzato z [18].

Pokud hledáme body ležící na kružnici s poloměrem r, tak se nejprve vypočtou podle rovnice (3.2) jejich souřadnice a poté určí hodnoty parametrů a, b podle následujícího vztahu

$$\begin{aligned} a &= x - r\cos\varphi \\ b &= y - r\sin\varphi \end{aligned} \tag{3.3}$$

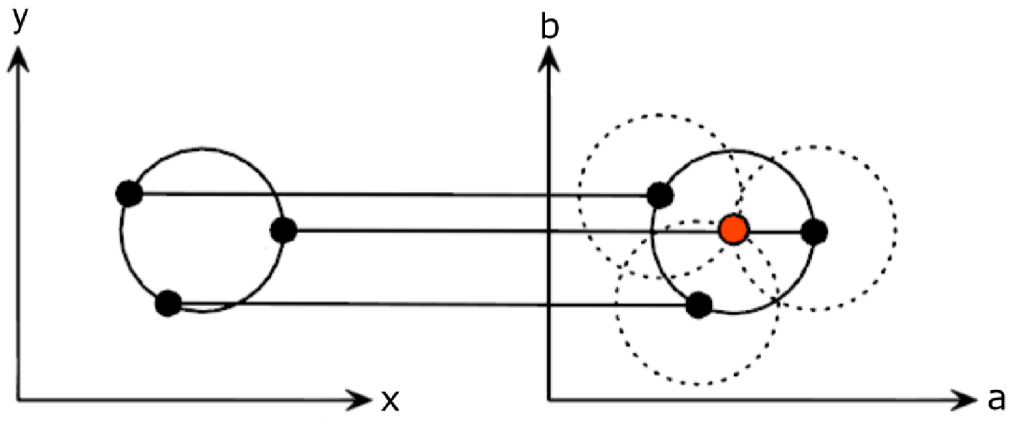
Houghova kruhová transformace

Vstupem Houghovy kruhové transformace bývá binární obraz, který byl předtím předzpracován hranovým detektorem. Hrany mají v binárním obrázku hodnotu jedna a jsou bílé, zatímco zbytek má hodnotu nula a je černý. Příklad toho jak se binární obraz liší od původního je ukázán na obrázku 3.2.

Nejprve je vysvětlen princip plnění Houghova prostoru pro pevně zadaný poloměr. V tomto případě je parametrický prostor dvourozměrný, neboť jsou zde jen dvě neznámé. Postup plnění akumulátoru je následující. Souřadnice každého bodu z binárního obrazu s hodnotou jedna se dosadí do rovnice 3.3. Následně se vykreslují kružnice se středem

Obrázek 3.2: demonstruje jak se liší předzpracovaný binární obraz vlevo od původního vpravo. Binární obraz je vhodným vstupem pro Houghovu transformaci. Převzato z [9]

v dané souřadnici. Pokud se tyto kružnice protnou v jednom bodě, tak jsme nalezli hledaný střed původní kružnice [8]. Celý tento proces je znázorněn na obrázku 3.3.



Obrázek 3.3: ukazuje proces hledání středu kružnice v binárním obraze. Body z něj se převedou do parametrického prostoru a vykreslují se kružnice. Pokud se jejich středy protnou znamená to, že byl nalezen střed původní kružnice [13].

V praxi běžně neznáme poloměr kružnice ve vstupním obrazu, tudíž se musí za proměnnou r dosazovat vhodné rozmezí hodnot. Z toho vyplývá, že parametrický prostor musí být třírozměrný. Postup plnění Houghova prostoru je pak podobný jako v předchozím příkladu. Dosazením souřadnic jednoho bodu do rovnice 3.3, vytvoří množina všech možných řešení (a, b, r) v parametrickém prostoru kužel. To je zapříčiněno třetím rozměrem prostoru, jímž je poloměr r. V případě, že se tímto způsobem promítnou do parametrického prostoru všechny pixely ležící ve vstupním binárním obraze na kružnici, bude získáno několik kuželů,

které se protnou v jediném bodě. Tento bod poté nese informace o hledaných parametrech (a, b, r) díky nimž je možné dle vztahu 3.2 stanovit přesný popis kružnice ve vstupním obrazu [18].

Algoritmus detekce je následující. Používá se trojrozměrný akumulátor. Pro každý pixel vstupního binárního obrazu, který má intenzitu jedna, se vypočtou dle 3.3 hodnoty parametrů a a b. Za parametr poloměru r se dosazuje předem určené rozmezí hodnot a za parametr φ rozmezí hodnot úhlů od 0 do $2\pi rad$, s pevně stanoveným krokem. Poté se na všech vypočtených pozicích (a, b, r) inkrementuje hodnota v akumulátoru. Při použití přesného poloměru vznikne postupnou inkrementací v Houghově prostoru maximum, které značí střed kružnice v binárním obrazu [19].

3.2 Kalibrace kamery

Kalibrace kamery slouží k odstranění zkreslení snímaného obrazu. Jedná se o techniku, která je pro systémy automatické vizuální kontroly velmi důležitá. Pokud by byl snímek zkreslený a neodpovídal by přesně skutečnosti, nebylo by možné provádět detekci s jistotou správného výsledku. Účinnost algoritmů počítačového vidění drasticky klesá, pokud kamera není správně kalibrována. Vývojář by pak nikdy nemohl s jistotou ověřit výkon svého algoritmu, pokud by hrozilo, že samotný vstup bude zkažený. Kalibrace kamery je obzvláště důležitá pro přesné měření vzdálenosti bodů v obraze.

Při kalibraci kamery rozlišujeme kalibraci vnitřních a vnějších parametrů. Kalibrace vnějších parametrů spočívá ve zjištění pozice kamery vůči výrobku. Je nezbytná pro správné měření vzdáleností v obraze. Při používání statické kamery, stačí kalibrace vnějších parametrů provést jen jednou. Pokud se poloha kamery může měnit, je třeba provádět kalibraci vícekrát.

Co se vnitřních parametrů kamery týká, zkreslení vzniká z následujících důvodů. Kamera se skládá z obrazové roviny a čočky, která zajišťuje transformaci mezi prostorem objektu a prostorem obrazu. Úhel vystupujícího paprsku není zcela stejný, jako u vstupujícího paprsku a poloha zobrazeného bodu se tedy mírně liší od správné polohy. Velikost odchylky se mění s radiální vzdáleností od středu objektivu. K radiálnímu zkreslení dochází, když se světelné paprsky ohýbají blíže okrajů čočky než v jejím optickém středu. Čím menší je objektiv, tím větší je zkreslení [16]. Příklad radiálního zkreslení můžeme vidět na obrázku 3.4.

Obrázek 3.4: popisuje vzhled kalibrační šachovnice. Obrázek vlevo ukazuje nezkreslený obraz. Na prostředním obrázku lze vidět příklad soudkovitého zkreslení a na tom vpravo poduškovité zkreslení. Převzato z [5].

Jako jeden z obrazů pro kalibraci se používá šachovnice jakou můžeme vidět na obrázku 3.4. Při kalibraci se nejprve změří vzdálenost mezi body, které se nacházejí v rozích čtverců v kalibračním obraze. Toto měření je třeba provést velmi přesně. Poté je zapotřebí kalibrační šachovnici nasnímat z různých úhlů. Počítač získá z obrazu vzdálenost mezi body na nasnímaných obrazech. Tu porovná se vzdáleností, která mezi nimi byla změřena. Rozdíl mezi těmito dvěma hodnotami odpovídá indexu zkreslení kamery. Takto získané parametry budou poté používány ke korekci zkreslení objektivu [6].

Pokud je kamera kalibrována s ohledem jak na její vnitřní, tak i vnější parametry, umožní to přesné měření velikosti objektů a jejich vzdálenosti od sebe.

Kapitola 4

Návrh řešení jednoduše konfigurovatelného kontrolního kamerového systému

V mé práci jsem se rozhodl zvláště se zaměřit na detekci LED (Light-Emitting Diode). Důvod zaměření se na detekci LED je následující. Běžné kontroly pomocí automatické vizuální inspekce se často soustřeďují pouze na odhalení povrchového poškození výrobku, jak ostatně vyplynulo z průzkumu současných řešení 2.2. Vadu, skrývající se pod povrchem, však odhalit nedokážou. Díky kontrole signalizačních LED je však možno takový defekt odhalit. Pokud LED nesvítí, nebo svítí jiná, než by měla, může to signalizovat poškození výrobku. Také jiná barva LED, než je očekávaná, může značit nestandardní chování produktu. Mnou navržený systém se tedy soustřeďuje na detekci samotného výskytu LED a i toho jestli právě svítí. Pokud je LED rozsvícená, tak systém vyhodnocuje, jakou má barvu.

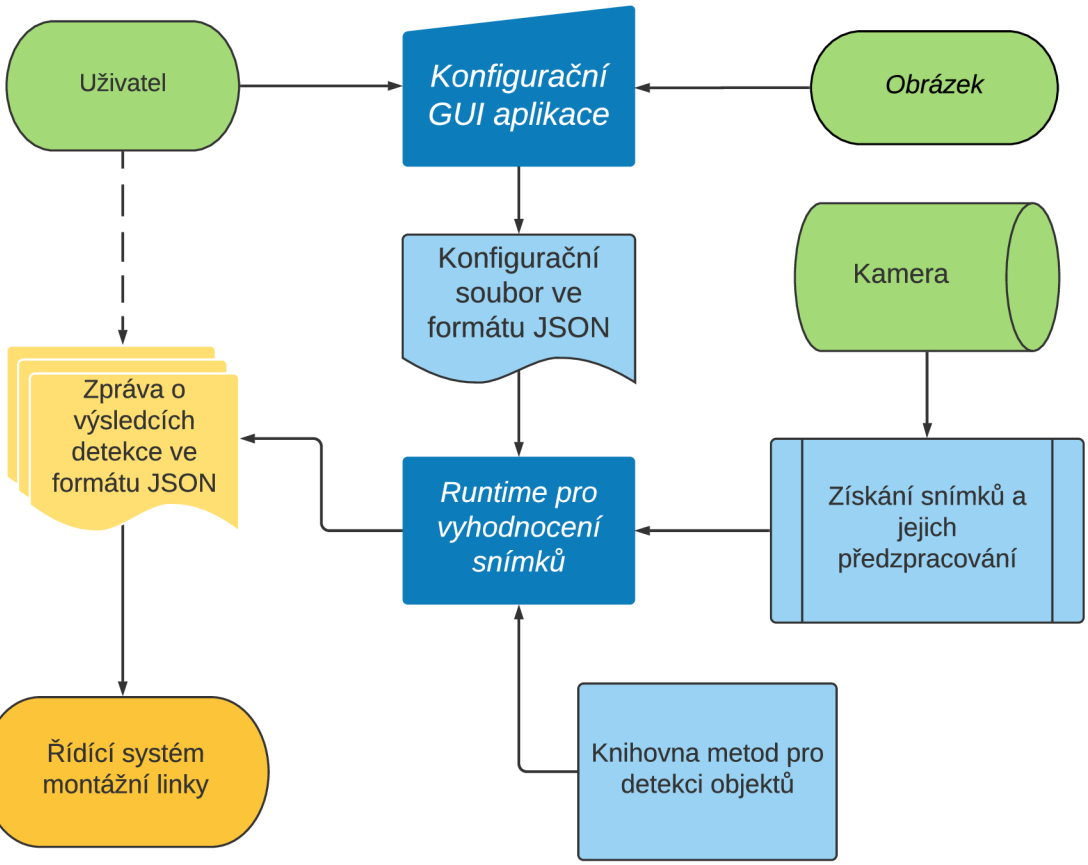
Cílem práce je navrhnout a implementovat jednoduše konfigurovatelný kontrolní kamerový systém. Z požadavku jednoduché konfigurovatelnosti plyne, že uživatel by měl mít možnost snadno definovat a měnit kontrolní metriky. Proto je kromě samotného nástroje pro analýzu snímků z montážní linky realizováno i grafické uživatelské rozhraní (GUI), v němž může uživatel snadno definovat konfiguraci úloh pro detekci objektů. Celková koncepce jednoduše konfigurovatelného kontrolního kamerového systému je vizualizována na obrázku 4.1.

Systém sestává ze dvou základních částí, které jsou vzájemně oddělené. Jsou to:

- Aplikace pro definici kontrolních úloh s grafickým uživatelským rozhraním určeného uživateli pro definování kontrolních metrik.
- Runtime, který pracuje samostatně a provádí samotné zpracování a vyhodnocení snímku.

Mezi těmito základními komponentami je přenášená konfigurace uložená v konfiguračním souboru. Ten jsem implementoval ve formátu JSON. JSON jsem se rozhodl použít pro to, že je nejen dobře použitelný jakožto vstup pro samostatně operující runtime, ale je i snadno čitelný pro uživatele. Osoba pracující s konfigurační GUI aplikací v ní totiž může sledovat stav JSON souboru a informace v něm obsažené. Ty se po každé změně konfigurace provedené uživatelem aktualizují.

Uživatel pracuje pouze s konfiguračním GUI. Vstupem konfigurační aplikace je tedy zpracovávaný obrázek a příkazy od uživatele. Runtime stačí pouze spustit s počátečními



Obrázek 4.1: popisuje schéma celého systému. Zelenou barvou jsou vyznačeny vstupy systému, kterými jsou obrázky a připojená kamera a také akce uživatele. Žlutá barva naopak značí výstup. Tím je zpráva o výsledcích detekce, která je předávána řídícímu systému montážní linky. Modře je zaznačen samotný kontrolní systém. Tmavě modrá vyznačuje jeho základní komponenty totiž GUI a runtime. Konfigurace je mezi GUI a runtime přenášena ve formátu JSON. Šipky ukazují vzájemnou závislost mezi jednotlivými komponentami systému.

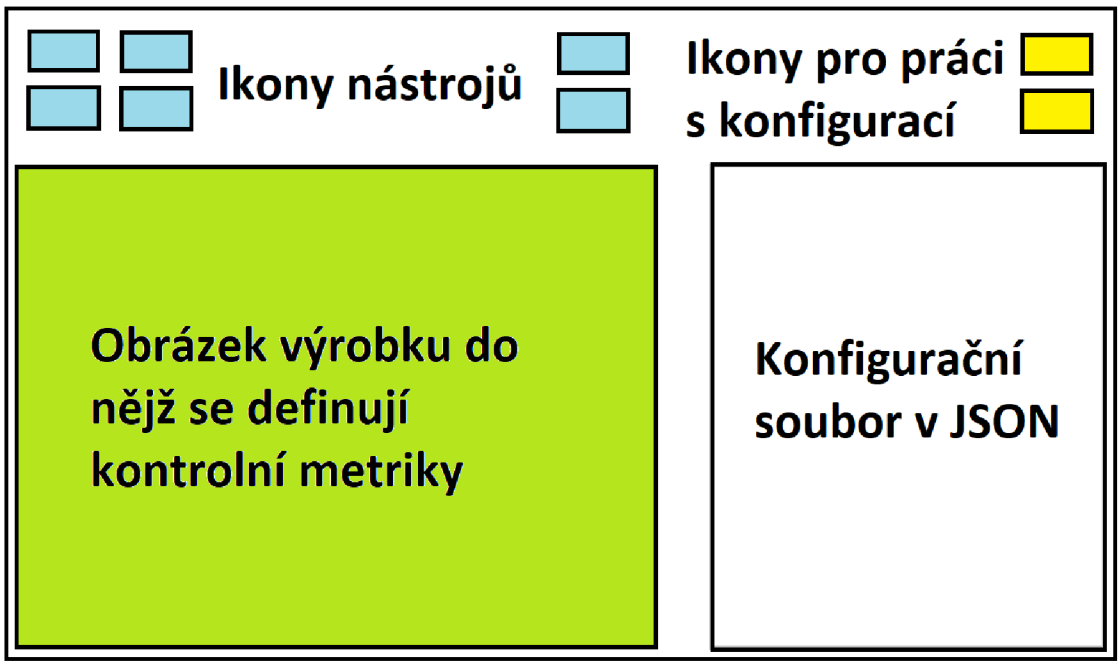
parametry. Runtime následně pracuje samostatně bez přítomnosti uživatele. Získává v určeném časovém intervalu z kamery jednotlivé snímky. Provede s nimi předzpracování, aby na ně mohli být efektivně aplikovány funkce pro detekci objektů. Ty jsou volány z dynamicky propojené knihovny.

Výsledky zpracování obrazu jsou stejně jako konfigurace ukládány ve formátu JSON. Uživatel má možnost si je prohlížet, primárně jsou však určeny pro Řídící systém montážní linky. Proto je v grafu mezi uživatelem a výsledky přerušovaná čára.

4.1 Konfigurační GUI aplikace

Díky desktopové konfigurační aplikaci s UI má uživatel možnost nastavovat, které metriky budou v obraze kontrolovány. Za vzor jejího vzhledu i funkcionality jsem si při návrhu vzal komerční systémy, o nichž pojednává kapitola 2.2. Rozložení komponent je situováno do čtyř skupin. jedná se jednak o samotný obrázek, poté ikony nástrojů a ikony pro práci

s konfiguračním souborem a také okno pro zobrazování konfigurace. Mockup navrhovaného GUI je vizualizován na obrázku 4.2.



Obrázek 4.2: ukazuje mockup konfiguračního GUI. V levé horní části se nacházejí ikony pro určené pro definici kontrolních metrik. Pod nimi je obrázek výrobku do nějž uživatel metriky zanáší. Seznam již definovaných úloh se nachází v pravé dolní části snímku a nad ním jsou ikony pro práci s konfigurací.

Při průzkumu současných řešení 2.2 jsem zjistil, že jako jednu z hlavní devíz svých produktů označují firmy to, že je uživatelské rozhraní jejich produktu přehledné a intuitivní a že jej dokáže efektivně používat i nepříliš zkušený zaměstnanec. Při návrhu a implementaci jsem se proto snažil o to, aby mé GUI bylo intuitivní a bylo navrženo takovým způsobem, aby s ním mohl snadno pracovat i nepříliš zkušený uživatel. O tom, jak se tento cíl podařil splnit, pojednávám v kapitole 6.1, kde hodnotím výsledky uživatelského testování konfigurační aplikace, které bylo speciálně zaměřeno na její intuitivnost.

Práce s aplikací probíhá následujícím způsobem. Uživatel si načte jeden vzorový snímek ze souboru, nebo ho získá z kamery. V něm následně může definovat jednotlivé kontrolní úlohy, v nichž nastaví co a jak se má v obraze detekovat a měřit. Aplikace uživateli umožňuje:

- Uživatel může definovat počet hledaných objektů.
- Uživatel má možnost nad obrázkem zadefinovat obdélníkovou oblast, ve které se mají objekty detekovat. Ty mimo oblast budou při kontrole ignorovány.
- Dále může vybrat barvu, kterou chce v obraze detekovat a nastavit k ní toleranci.
- Může vytvářet více konfiguračních úloh a při návrhu má možnost mezi nimi přepínat a dodatečně je upravovat.
- Má možnost nastavit parametry runtime, jako je rychlost získávání jednotlivých snímků z kamery.

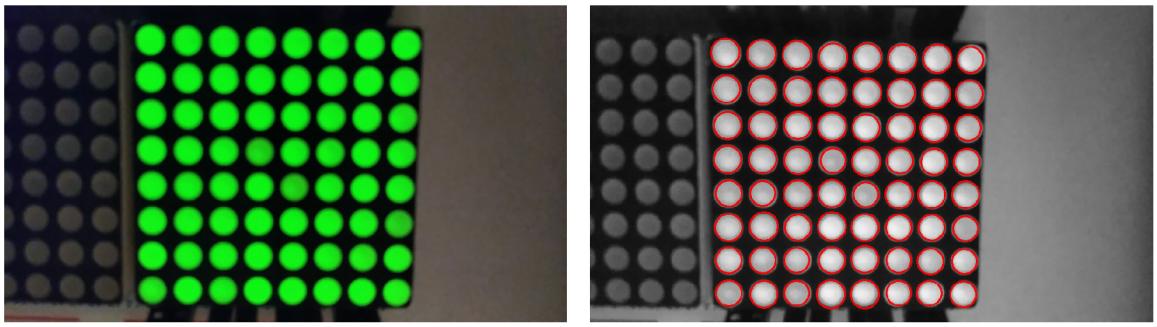
- Může také sledovat stav konfiguračního souboru v rámci GUI a manuálně ho upravovat.

4.2 Runtime pro vyhodnocení snímků

Runtime pracuje samostatně, bez přítomnosti uživatele. Informace ke svému provozu získává z konfiguračního souboru ve formátu JSON, který předtím uživatel zadefinoval. Kromě něj je runtime při jeho spuštění předána identifikace kamery a také adresa složky do níž budou ukládány výsledky. K výsledky jsou poté předávány řídícímu systému montážní linky.

Runtime při své činnosti začíná načtením video souboru. Z něj v pevně stanoveném intervalu získává snímky. Tento interval je definován uživatelem v rámci konfigurace. Pokud tomu tak není, tak runtime použije pevně stanovenou hodnotu. Ve snímku zkontroluje, zda se v něm nachází výrobek. Pokud ne, tak další detekci neprovádí. Pokud ano, tak následuje výběr metod pro zpracování videa. Jaké metody budou použity, vyplývá z konfiguračním souboru. Následně je provedena detekce objektů v obraze. Algoritmy pro zpracování videa jsou implementovány v dynamicky propojené knihovně. Jak vypadá vstupní obrázek a jak se jeho vzhled změní po provedení detekce kruhových objektů je ilustrováno na obrázku 4.3.

Výsledky se ukládají do logu, jež je stejně jako konfigurační soubor implementován ve formátu JSON. Výsledný soubor poté bude předán dalším částem systému, které na jeho základě mohou rozhodnout, zda je kontrolovaný výrobek poškozen a je jej tedy nutné vyřadit z výrobního procesu.



Obrázek 4.3: první obrázek ukazuje snímek načtený z kamery. Druhý ukazuje, jak vypadá výsledný obraz po provedení předzpracování, prahování a detekce svítících LED za pomocí Houghovy kruhové transformace.

Kapitola 5

Implementace jednoduše konfigurovatelného kontrolního kamerového systému

V této kapitole popisuji, jakým konkrétním způsobem byl návrh jednoduše konfigurovatelného kontrolního kamerového systému, popsaný v kapitole 4 implementován. Aplikace byla napsána v programovacím jazyce Python. GUI je vytvořeno za pomoci knihovny Tkinter (Tk interface). Pro detekci objektů využívá runtime knihovnu OpenCV (Open Computer Vision).

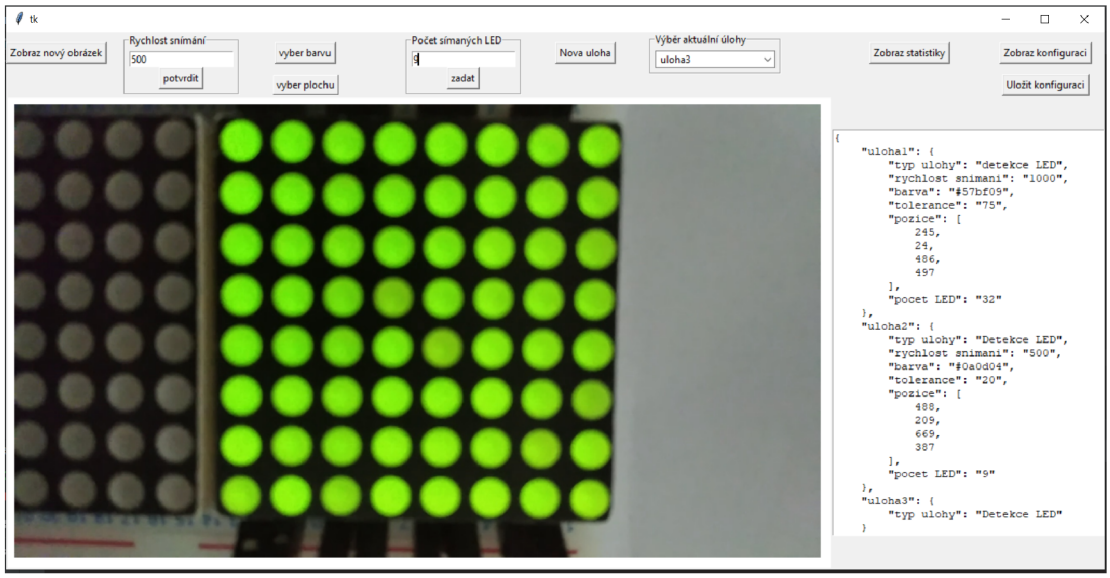
5.1 Konfigurační GUI aplikace

Celá konfigurační GUI aplikace je v kódu koncipována jako jeden objekt, který obsahuje jednotlivé komponenty v podobě tlačítek, popisků, vstupních polí, plátna pro zobrazení snímku a textového pole pro výpis konfigurace, nebo výsledků. K definování vzájemné polohy jednotlivcích komponent konfiguračního GUI jsem využil správce rozmístění Grid. Objekt také obsahuje definice funkcí, které jsou volány při aktivizaci konkrétních tlačítek. Obrázek 5.1 ilustruje vzhled výsledného konfiguračního GUI.

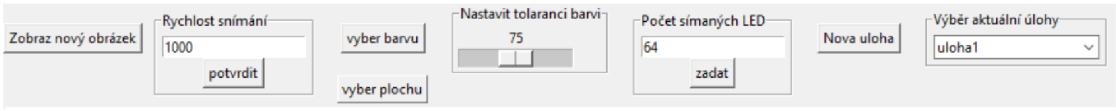
Tlačítka jsou vytvářena jako instatnce třídy Button. Kromě nich obsahuje aplikace i vstupní pole, která umožňují uživateli zadávat textový vstup. K popisu toho, k čemu vstupní pole slouží používám instanci třídy LabelFrame. Do ní je vloženo jak textové pole, tak i tlačítko, po jehož stisknutí se hodnota nacházející se v textovém poli uloží do konfigurace. Všem tlačítkům a textovým polím je při jejich inicializaci přiřazen parametr Command. Ten zajišťuje to, že při aktivaci tlačítka bude volána konkrétní funkce, která je takto s nimi asociována.

Plocha pro zobrazování obrázku je realizována pomocí třídy Canvas (plátno). Pro provádění operací s obrázky využívá konfigurační aplikace knihovnu PIL (Python Imaging Library). Zobrazení konfiguračního souboru je realizováno pomocí textového pole. To umožňuje i zobrazování výsledků detekce prováděných runtime. Pro práci s konfiguračním souborem a výsledky je využívána knihovna json.

Na obrázku 5.2 se nachází lišta s ikonami sloužícími pro volání konfiguračních funkcí. Jedná se o zvětšený výřez z obrázku celé aplikace. Tlačítka umožňují uživateli provádět následující akce:



Obrázek 5.1: vizualizuje vzhled výsledného konfiguračního GUI. V horní části se nacházejí ikony určené pro definici kontrolních metrik a pro práci s konfiguračním souborem. Většinu plochy zabírá obrázek výrobku, do nějž uživatel metriky zanáší. Seznam již definovaných úloh se nachází v pravé části snímku.



Obrázek 5.2: zobrazuje lištu s ikonami, které slouží pro volání konfiguračních nástrojů.

- Po kliknutí na tlačítko Zobraz nový obrázek se uživateli objeví vyskakovací okno v němž zadá adresu obrázku. Poté, co uživatel svou volbu potvrdí, se obrázek vloží do plátna. Tato operace využívá funkce knihovny Tkinter zvané filedialog.askopenfilename.
- Do pole Rychlost snímání se zadává hodnota v milisekundách určující rychlost získávání jednotlivých snímků z kamery. Stisknutím tlačítka Potvrdit se napsaná hodnota uloží.
- Poté, co uživatel klikne na nástroj Vyber barvu musí znovu kliknout do obrázku. Funkce takto získá barevnou hodnotu vybraného pixelu. Tu může pracovník manuálně upravovat za pomoci modulu colorchooser. Po provedení této operace se na hlavním ovládacím panelu objeví nástroj pro vybrání tolerance barvy. Tímto je zaručeno, aby nešlo zadat toleranci barvy, bez jejího předchozího definování. Hodnota tolerance barvy určuje, jak přísně bude runtime při detekci postupovat.
- Tlačítko Vyber plochu uživateli umožňuje nad obrázkem zadefinovat obdélníkovou oblast, ve které se mají objekty detekovat. Uživatel klikne do obrázku a tažením myší obdélník roztáhne. Objekty mimo definovanou oblast budou při kontrole ignorovány.
- Funkčnost textového pole Počet hledaných LED je z popisku jednoduše odvoditelná.

- Po klepnutí na tlačítko Nová úloha se objeví vyskakovací okno, v němž se zadává typ nové úlohy. Pracovník může vytvářet více konfiguračních úloh a při návrhu má možnost mezi nimi přepínat a dodatečně je upravovat. Přepínání se realizuje pomocí seznamu Výběr aktuální úlohy.

Uživatel může v rámci GUI také sledovat stav konfiguračního souboru a manuálně ho upravovat. Také mu aplikace umožňuje zobrazování výsledků. O jejich struktuře pojednává následující podkapitola.

5.2 Přenos konfigurace a zobrazování výsledků

Konfigurace vytvořená uživatelem je přenášena ve formátu JSON. Na obrázku 5.3 je ilustrována typická struktura konfiguračního souboru.


```
{
    "uloha1": {
        "typ ulohy": "detekce LED",
        "rychlost snimani": "1000",
        "barva": "#57bf09",
        "tolerance": "75",
        "pozice": [
            245,
            24,
            486,
            497
        ],
        "pocet LED": "32"
    },
    "uloha2": {
        "typ ulohy": "Detekce LED",
        "rychlost snimani": "500",
        "barva": "#0a0d04",
        "tolerance": "20",
        "pozice": [
            488,
            209,
            669,
            387
        ],
        "pocet LED": "9"
    },
```


Obrázek 5.3: popisuje typickou strukturu konfiguračního souboru.

Každý soubor obsahuje jednu až libovolné množství konfiguračních úloh. Jako identifikátor úlohy slouží její jméno, které je vždy ve formátu "*ulohax*" kde x značí pořadové číslo úlohy. Úloha obsahuje různé podmínky, jako je její typ, rychlost snímání, počet hledaných LED, hledaná barva a hodnota tolerance pro detekci objektů. Typ úlohy může nabývat hodnoty Detekce LED, nebo Měření vzdálenosti. Hodnota barvy je v RGB kde

každý z barevných kanálů je reprezentován číslem od 0 do 255 vyjádřeným v šestnáctkové soustavě. Nejkomplexnějším z údajů je pozice cílové oblasti. Pozice obsahuje seznam čtyř hodnot. První dvě značí koordinanty x a y levého horního rohu okna, další dva značí totéž u pravého dolního rohu. Další parametry jsou popsány v tabulce 5.1. V tomto formátu je konfigurace předávána runtime.

	Rychlost snímání	Tolerance barvy	Pozice
Hodnota v	ms	hodnoty RGB	pixely
Povolený rozsah	500–2000	0–150	rozsah obrázku

Tabulka 5.1: zobrazuje typy parametrů a hodnoty, jichž mohou nabývat.

Výsledky jsou stejně jako konfigurace uloženy ve formátu JSON a mají podobnou strukturu. Obsahují vždy identifikátor každého kontrolovaného výrobku. U každého je vypsán seznam metrik, které na něj byly aplikovány a to jestli byly úspěšně splněny nebo jestli byla odhalena chyba. Ta je poté blíže specifikována, například je popsáno o kolik LED svítí v obrázku méně oproti očekávanému stavu. Výsledky si může uživatel zobrazit v rámci GUI.

5.3 Runtime pro vyhodnocení snímků

Runtime provádějící detekci kruhových objektů bývá spouštěn s následujícími parametry:

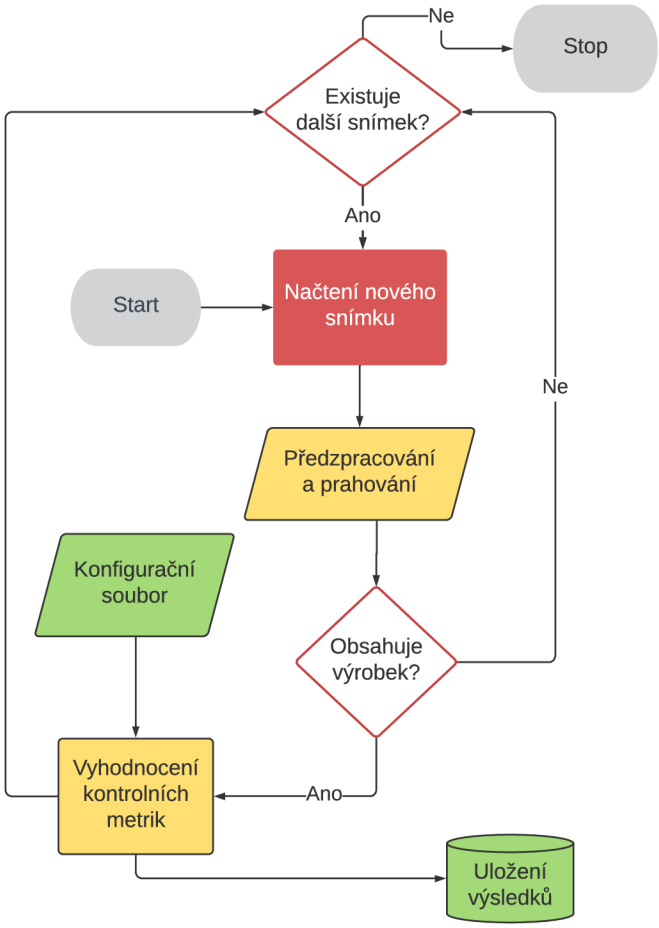
- Odkaz na video, z něhož budou načítány snímky.
- Odkaz na konfigurační soubor.
- Odkaz na soubor pro zapisování výsledků.

Pro zpracování parametrů využívá skript knihovnu getopt. Poté, co parametry zpracuje, začne načítat snímky z videa. V tomto pokračuje, dokud video neskončí, nebo mu není zaslán signál, aby přerušil svou činnost. Schéma činnosti runtime je ukázáno na obrázku 5.4.

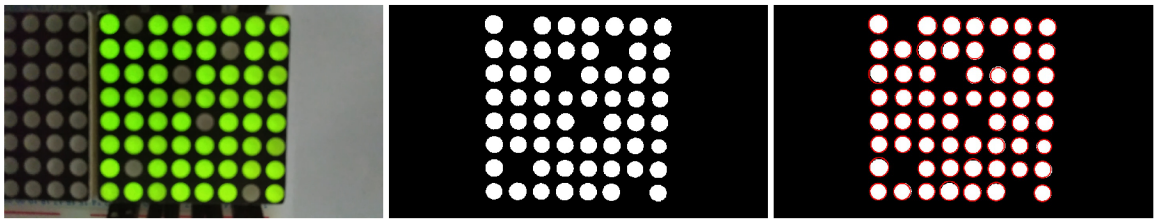
Načtený snímek je předzpracován a je na něj aplikováno prahování. Systém detekuje přítomnost kruhových objektů. Pokud se v obrázku nachází produkt, začne se s vyhodnocováním kontrolních metrik. Hledá se v obraze tolik objektů, kolik zadal uživatel. Pokud tuto informaci uživatel nezadal, je již v rámci GUI do konfiguračního souboru vybrána předdefinovaná hodnota. Ta v případě detekce LED činí 64.

Pokud je v konfiguraci uvedena hodnota pro pozici pole, v němž se má kontrola provádět, runtime ořízne obrázek pomocí funkce pro ořezávání a pracuje jen s takto získaným výřezem. Funkce pro ořezávání přijímá čtyři parametry. Dva z nich představují koordinanty horního levého rohu výřezu na osách x a y, další dva značí totéž u pravého dolního rohu. Pokud parametr pozice chybí, tak je zpracováván celý snímek.

Pro nalezení kontrolovaných objektů se využívá detekce uživatelem definované barvy. K prahování na základě barevné hodnoty systém využívá funkci inRange. Ta přijímá jako své parametry dolní a horní hranici. Dolní hranice je vypočítána tak, že je od uživatelem zadané hodnoty barvy odečtena velikost tolerance. Horní hranice se získává stejným způsobem, pouze s tím rozdílem, že se tolerance k hodnotě barvy přičítá. Poté je provedeno prahování. Pixely, jejichž barevná hodnota spadá mezi horní a spodní hranici mají ve výsledném binárním obraze hodnotu 1 a při vykreslení jsou bílé. Ostatní pixely dostávají



Obrázek 5.4: ilustruje činnost runtime. Proces začíná získáním snímku z kamery. Ten je poté předzpracován a je na něj aplikováno prahování. Poté systém vyhodnocuje, zda se ve snímku nachází výrobek. Pokud ne, načítá se další snímek. Pokud ano, tak je provedeno vyhodnocení kontrolních metrik. Výsledky tohoto procesu se pokaždé ukládají. Ve chvíli, kdy není žádný další snímek na vstupu, činnost programu končí.



Obrázek 5.5: první obrázek ukazuje snímek načtený z kamery. Druhý zobrazuje jak vypadá snímek po provedení předzpracování, prahování a detekce zelené barvi. třetí pak popisuje výsledek aplikace Houghovy kruhové transformace na předchozí obrázek.

hodnotu 0 a jsou černé. Na takovýto binární obrázek je aplikována Houghova kruhová transformace, která v něm detekuje LED. Obrázek 5.5 vizualizuje, jak vypadá snímek v průběhu zpracování kontrolním systémem.

Pokud není v obrázku detekována žádná kružnice, systém vyhodnotí, že se ve snímku žádný výrobek nenachází. Zpracovávaný snímek je dále ignorován a načítá se nový.

Výsledky procesu detekce se po provedení každé z jeho součástí ukládají do výstupního souboru. Ve chvíli, kdy není žádný další snímek na vstupu, činnost programu končí.

Kapitola 6

Testování jednoduše konfigurovatelného kontrolního kamerového systému

Systém je rozdělen na dvě základní části, a to na konfigurační GUI aplikaci a runtime pro detekci objektů v obraze. Z tohoto důvodu jsem se rozhodl i testování rozdělit na dvě části a funkčnost každé ze složek výsledného systému otestovat zvlášť. V první části se věnuji uživatelskému testování konfigurační aplikace, které má ověřit, zda je její GUI dostatečně intuitivní. Také jsem při tomto testování chtěl získat další podněty na rozšíření funkcionality aplikace do budoucna. V druhé části se věnuji testování runtime. Popisuji, jakým způsobem jsem vytvořil testovací dataset a poté na něm zkoušel efektivnost detekčních metod.

6.1 Uživatelské testování GUI konfigurační aplikace

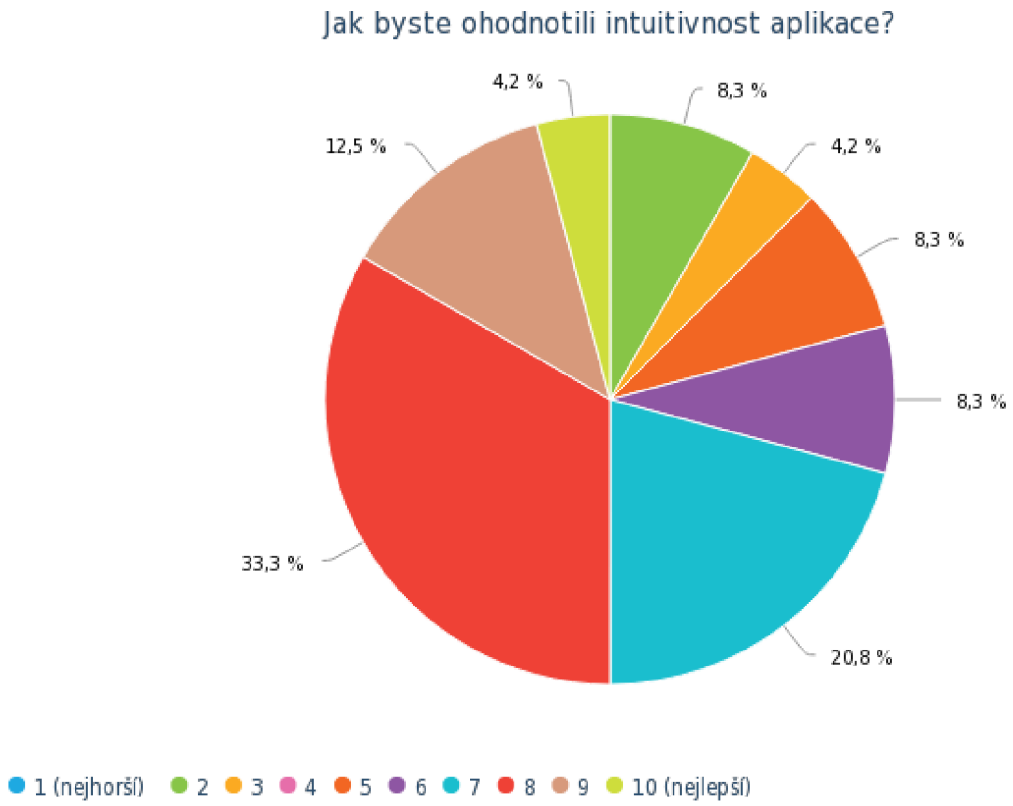
Funkčnost a intuitivnost uživatelského rozhraní jsem vyhodnocoval za pomoci dotazníku. Dotazník sestával z devíti otázek, které byly jednak zaměřeny na věk a pohlaví uživatelů a také na jejich zkušenosti s programováním. Dále se dotazník soustředil na to, jaký dojem na ně uživatelské rozhraní udělalo. Znění dotazníku je uvedeno v příloze B.

Odpovědi na ty nejzajímavější z otázek, jsou vizualizovány v textu pomocí grafů. Na dotazník odpovědělo 24 respondentů, z nichž 13 uvedlo, že mají zaměstnaní v IT, nebo jej studují. Nejvíce respondentů bylo ve věkové kategorii 18 až 30 let.

Z odpovědí vyplynulo, že uživatelé při celkovém hodnocení shledávají aplikaci poměrně intuitivní. Průměrné hodnocení intuitivnosti představovalo 6,875 bodu z deseti. Souhrnné výsledky této otázky jsou ukázány v grafu 6.1.

Při hodnocení vzhledu aplikace udávali uživatelé průměrnou hodnotu číslo 3.375 z pěti. Dle konkrétnějších recenzí není aplikace příliš hezká, ale u průmyslových aplikací není vzhled tím hlavním kritériem.

Většina uživatelů, tedy přes 60%, neměla problém s pochopením způsobu používání žádného z nástrojů jak ukazuje graf 6.2. Přesto však alespoň třetina měla s pochopením přesné funkcionality některého z nástrojů potíže. Z tohoto důvodu jsem naznal, že by bylo dobré aplikaci doplnit návodem na její používání. Také by mohlo být vhodné GUI upravit tak, aby tlačítka pro volání konfiguračních funkcí umožňovala tooltip (pro zobrazení popisku její funkcionality).



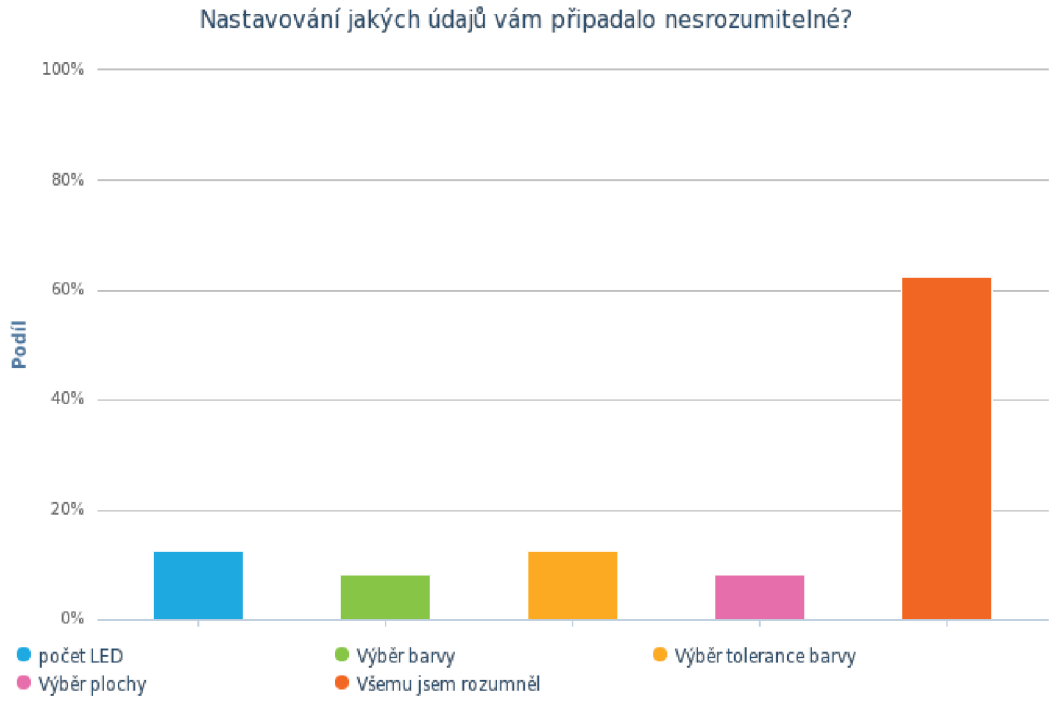
Obrázek 6.1: popisuje výsledky odpovědí na otázku, zda se uživatelům jevila aplikace dostatečně intuitivní. Uživatelé mohli své hodnocení uvést v rozmezí od jedné do desíti, přičemž hodnota deset značila nejlepší hodnocení. Nejčastějším hodnocením byla hodnota osm následovaná hodnotou sedm.

Použití žádného z nástrojů nebylo výrazně obtížnější než používání ostatních. Největší problémy měli uživatelé bez zkušenosti s programováním. Testovacím subjektům také připadal zmatený systém přepínání mezi jednotlivými úlohami.

Důležitou součástí uživatelského testování byla snaha zjistit, o jakou funkcionalitu by lidé chtěli, aby byl prototyp jednoduše konfigurovatelného kontrolního kamerového systému rozšířen. Nejčastější odpověď představovalo měření vzdálenosti mezi objekty. Dále následovalo hledání středů kružnic a vyhledávání jiných objektů, než jen kruhového tvaru. Odpovědi jsou vizualizovány na grafu 6.3.

Nakonec měli účastníci průzkumu možnost se v otevřené otázce vyjádřit podrobněji k aplikaci. V otevřené otázce na hodnocení aplikace uživatelé uvedli že:

- Je potřeba rozdělit tlačítka pro tvorbu konfiguračních úloh do souvisejících bloků.
- Přepínání mezi úlohami je zmatené. Chtělo by ho přepracovat, aby bylo celé řešeno pomocí jednoho tlačítka.
- Hodilo by se, kdyby okno pro výběr plochy šlo v obrázku posouvat a zvětšovat/zmenšovat.

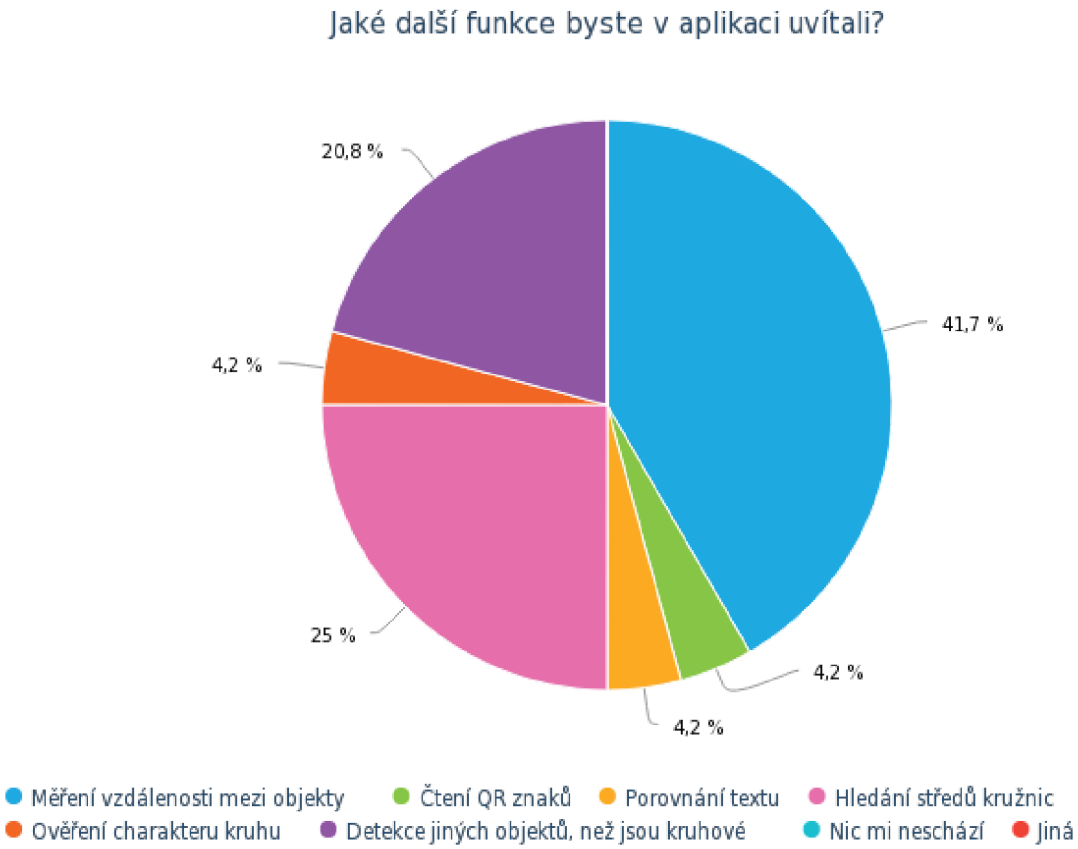


Obrázek 6.2: ukazuje zda měli uživatelé problém s pochopením používání některého z nástrojů. Většina uvedla, že ne.

- Změnil bych to, jakým způsobem vyskakuje výběr tolerance barvy po kliknutí na výběr barvy.
- Vzhled aplikace sice není příliš dobrý, ale u průmyslových aplikací to asi není příliš důležité...

Na základě reakcí uživatelů jsem změnil některé části uživatelského rozhraní. Předně jsem zjednodušil systém přepínání mezi úlohami. Ten byl předtím řešen pomocí dvou tlačítek, které vybíraly úlohu a výběr potvrzovaly. Kromě nich se ještě na v GUI nacházel textový popisek, který vypisoval, jaká z úloh je aktuálně vybraná. Toto jsem zjednodušil tak, aby celý výběr probíhal pomocí jednoho seznamu, v němž jde zároveň jasně vidět aktuálně vybraná úloha. Také jsem všechna tlačítka přeskupil do logicky souvisejících bloků a opatřil je popisky. Díky průzkumu jsem zjistil, že bude dobré, aby při nepodařeném zadefinování výběru plochy, nebylo nutné okno umisťovat do obrázku znova, ale dalo se myší přetahovat a zvětšovat, nebo zmenšovat.

Do budoucna plánuji aplikaci rozšířit o možnost hledání středů kružnic. Také je v plánu umožnit uživatelům měření vzdálenosti mezi jednotlivými objekty, neboť tato funkce byla podle výsledků uživatelského testování tou nejžádanější. Užitečné také bude do budoucna se zaměřit na možnost detekování i jiných objektů, než jsou jen kruhové. Potenciálně je také možné zaměřit se na detekci a porovnávání textů a čárových a QR kódů, což je oblast, které se komerční aplikace rozsáhle věnují.



Obrázek 6.3: ukazuje jakou další funkcionality by uživatelé v aplikaci uvítali. Nejčastější odpovědí je měření vzdálenosti mezi objekty, nasledovalo hledání středů kružnic a vyhledávání jiných objektů, než jen kruhového tvaru.

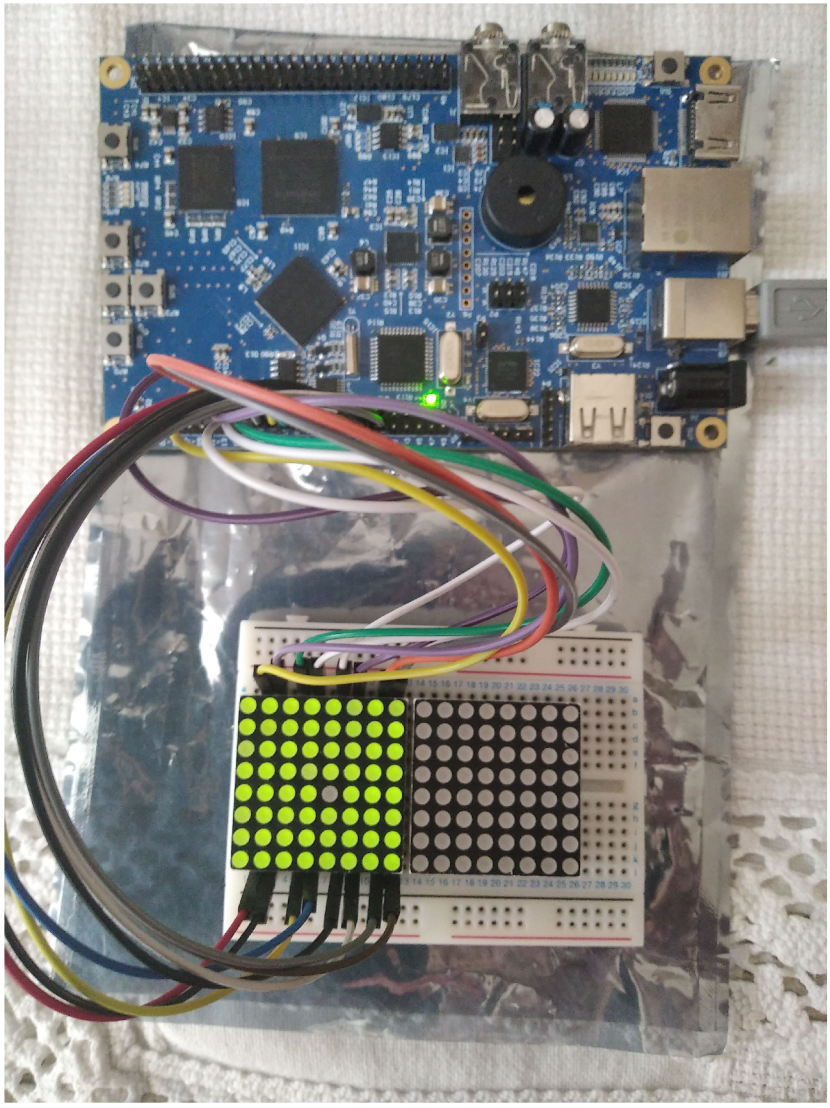
6.2 Testování runtime pro vyhodnocení snímků

V rámci testování runtime jsem si nejprve vytvořil testovací dataset, na němž jsem poté zkoušel detekci LED. Při testování jsem chtěl porovnat efektivnost různých postupů pro detekci kruhových objektů ve snímku.

Testovací dataset a jeho příprava

Najít volně dostupný dataset, sloužící k trénování průmyslových aplikaci je velmi obtížné, neboť si je firmy chrání. V práci jsem se proto rozhodl využívat dataset, který jsem vlastnoručně vytvořil za pomocí samostatného hardware FITkit3 – Minerva. K němu jsem připojil externí LED displej o velikosti 8x8. Spojení bylo provedeno pomocí kabelů, které jsou připojeny na piny na Fitkitu. Dohromady bylo využito 16 pinů, které se všechny nacházejí v poli P1. Zapojení je demonstrováno na obrázku 6.4.

Poté jsem FitKit naprogramoval tak, aby maticový displej vykresloval správné kombinace rozsvícených LED. Nejprve bylo třeba na FitKitu provést inicializaci nezbytných komponent. Pomocí několika funkcí bylo provedeno základní nastavení hodin a vypnut



Obrázek 6.4: ukazuje samostatný hardware FITkit3 – Minerva. K němu je pomocí šestnácti kabelů připojena matice LED 8x8. Na matici právě probíhá zobrazování vizuálních efektů.

watchdog. Dále jsem inicializoval porty PTA, PTE a PTD, které se používají pro komunikaci s LED maticí. Poté program volá funkce pro vytváření světelných efektů.

Matice nedokáže zároveň rozsvítit všech 64 LEDek, ale pouze jeden řádek o osmi LED. Tento problém jsem řešil následujícím způsobem. Vykreslení každého snímku se děje ve smyčce for. V ní jsou postupně v krátkých intervalech po sobě rozsvíceny jednotlivé řádky, které se mají vykreslovat. Pro nastavení délky rozsvícení každého řádku používám funkce shortWait, normalWait a longWait. Tyto funkce provádějí zpoždění, které slouží k tomu, aby dioda svítila dostatečně dlouhý čas a člověk ji tak mohl vidět. Nejčastěji se v programu používá shortWait. V některých případech, byl jas diod při použití shortWait nízký, a proto jsem vytvořil další dvě funkce, které umožňují delší čekání. Tímto se zajistí, že všechny diody svítí jasně. Kód ve smyčce for se provádí s takovou frekvencí, aby došlo k rozsvícení všech řádků padesátkrát za sekundu. Takto vysoká hodnota stačí k tomu, že je dosaženo efektu, při kterém se pozorovateli jeví, jako by všechny řádky svítily současně. Zdání, že

svítí současně je vyvoláno právě opakovaným spouštění řádků rychle po sobě. Aby bylo dosaženo efektu, že některé LED v matici nesvítí a ostatní ano, tak se řádek nerozsvěcuje celý, ale rozsvěcují se jen ty LED, které se nacházejí na průsečíku aktivovaného řádku a sloupce.

Svícení LED se spouští a vypíná v předem stanovených intervalech. To simuluje pohyb výrobků na montážní lince. Pokud LED svítí, značí to, že výrobek je na scéně. Pokud jsou všechny vypnuty, simuluje to, že v daném okamžiku žádný produkt na výrobní lince není. Dataset je snadno rozšiřitelný. Stačí pouze za pomoci FitKitu nakonfigurovat další vizuální efekty na LED matici. Tyto efekty se poté natočí kamerou. Tak vzniká video, které může kontrolní runtime přijímat na svém vstupu. Současný dataset sestává ze série jednadvaceti videí na niž jsem natočil různé konfigurace rozsvěcování LED. Celkem obsahuje 226 konfigurací LED matice, které představují výrobky pohybující se na montážní lince. Z toho je 128 výrobku bez vady a 98 obsahuje nějakou chybu. Jako bezvadné jsou počítány ty, na nichž svítí všechny LED, jako poškozené ty, na nichž některá nesvítí. LED matice umožňuje, aby LED svítili červeně a zeleně. Při natáčení se objevil problém s tím, že červeně svítící LED má dvakrát vyšší světelnou intenzitu než zeleně svítící. Následkem toho při současném rozsvícení obou barev docházelo k tomu, že kamera špatně zachycovala červené LED v nichž se pak jevily být dva kruhy.

Výsledky testování

Při testování jsem pracoval s úlohou řešící detekci svítících LED a pak porovnával, zda stačí použít pouze jasové prahování, nebo je lepší uvažovat i barvu a operovat s prahováním na základě detekce barvy.

Testováním jsem chtěl ověřit, zda pro efektivní detekci rozsvícených LED, bez ohledu na to, jakou svítí barvou, stačí použít pouze prahování na základě intenzity jasu v obrazu, nebo zda je potřeba používat i detekci barev. Pokud by se využívalo jenom prahování jasu, mělo by to výhodu, že detekce bude nezávislá na barvě konkrétní LED. Ta uživatele v tomto případě nezajímá, protože testuje pouze to, zda LED aktuálně svítí, takže barvu není nutné zjišťovat. Pokud by detekce barvy byla nutná, pak bude třeba vyzkoušet vyhledávání všech barev, jichž může LED potenciálně nabývat. Díky tomu musí kontrolní runtime provést větší množství operací. Pouhé prahování jasu tedy šetří výpočetní čas systému.

Testovací snímky nemají omezenou oblast detekce. LED se tedy vyhledávají v celém snímku. Na videích některé výrobky obsahují 64 rozsvícených LED. Tyto výrobky jsou klasifikovány jako bezvadné. Na jiných svítí menší počet LED. Tyto jsou klasifikovány jako vadné.

K vyhodnocení výsledků jsem využíval následující kritéria:

- Míru zásahu, neboli procento správných produktů, na niž nebyla detekována žádná závada. To znamená, že počet detekovaných LED odpovídá hodnotě 64.
- Míru detekce chyb, která značí procento poškozených produktů, na niž byla detekována závada. To znamená, že bylo správně detekováno, že výrobky obsahují méně LED, než 64.
- Míru minutí, což je procento nezjištěných vadných výrobků. Došlo tedy k tomu, že bylo detekováno více LED než svítilo.
- Míru falešného poplachu, neboli procento funkčních výrobku nesprávně detekovaných jako poškozené. Došlo tedy k tomu, že bylo detekováno méně LED než svítilo.

Na vstup kontrolního skriptu jsem dával různá videa a sledoval výsledky detekce. Tímto způsobem jsem vyzkoušel jak detekci pomocí pouhého prahování, tak i tu za pomoci vyhledávání konkrétních barev. Výsledky testů pro detekci barev jsou v následující tabulce 6.1 a pro prahování v tabulce 6.2.

	Správné	Vadné	Nezjištěné chyby	Falešně chyby
Počet výrobků	126	95	2	3
Procentuální podíl	55,75%	42,04%	0,88%	1,33%

Tabulka 6.1: zobrazuje úspěšnost testování při použití detekce barev.

Při použití detekce barev bylo správně detekováno 97,79% výrobků, bud jako vadné, nebo správně fungující. Chybovost byla 0,88% a podíl falešného poplachu tvořil 1,33%.

	Správné	Vadné	Nezjištěné chyby	Falešně chyby
Počet výrobků	93	73	28	32
Procentuální podíl	41,15%	32,3%	12,39%	14,16%

Tabulka 6.2: zobrazuje úspěšnost testování při použití prahování.

Při použití prahování jasu bylo správně detekováno 73,45% výrobků, bud jako vadné, nebo správně fungující. Chybovost byla 12,39% a podíl falešného poplachu tvořil 14,16%.

Při testování, při kterém bylo používáno pouze prahování jasu, docházelo k velkému množství falešných poplachů a nezjištěných chyb. Ty vznikaly hlavně následkem nerovnoměrného osvětlení kamerových snímků. V některých případech se nedala nalézt správná hodnota prahu, při které by byly odhaleny všechny svítící LED a zároveň by nebyla falešně detekována ta, která nesvítí. Z tohoto důvodu docházelo k tomu, že skript vyhodnotil snímek, na němž žádná nesvítila, jako snímek obsahující výrobek a prováděl pro něj kontrolu. Testování tedy prokázalo nutnost použití funkcí pro detekci barvy. Pouhé prahování jasu bylo natolik nepřesné, že se nedá v praxi použít.

Další experimenty prováděné na základě konfiguračního souboru definovaného uživatelem v GUI dosahovaly rozdílné úspěšnosti v závislosti na tom, jak uživatel kontrolní metriky definoval.

Kapitola 7

Závěr

Cílem této práce bylo vytvořit prototyp jednoduše konfigurovatelného kamerového systému sloužícího ke kontrole výrobků na automatizované montážní lince. Práce byla zvláště zaměřena na problematiku detekce LED. Detekována je jejich samotná přítomnost ve snímku a dále systém vyhodnocuje, zda LED svítí, a pokud ano, tak jakou barvou.

Vytvořený systém sestává ze dvou částí, totiž z Grafického uživatelského rozhraní (GUI) a runtime pro detekci objektů v obraze. Díky GUI má uživatel možnost samostatně definovat kontrolní metriky, pro detekci objektů. Uživatel si při práci načte vzorový snímek a v něm poté nastavuje, co bude kontrolováno. Výsledná konfigurace se uloží do konfiguračního souboru ve formátu JSON. Ten je následně předán runtime, který pracuje samostatně, bez přítomnosti uživatele. Kromě konfiguračního souboru si v pevně stanoveném časovém intervalu načítá i snímky z přednastavené kamery. V nich poté provádí detekci objektů na základě kontrolních metrik. Celková správnost detekce počtu svítících LED při testování dosáhla 97,79%. Runtime zapisuje výsledky do výstupního souboru ve formátu JSON, který může být předáván dalším částem systému, které by na jeho základě rozhodly, zda je kontrolovaný výrobek poškozen a je jej tedy nutné vyřadit z výrobního procesu.

Výsledný prototyp plánuji do budoucna rozšiřovat a vylepšovat. Inspiraci k dalšímu vylepšování a rozšiřování práce jsem získal na základě uživatelského testování a také díky rešerši již existujících komerčních systémů. Do budoucna plánuji aplikaci rozšířit o možnost hledání středů kružnic. Také je v plánu umožnit uživatelům měření vzdálenosti mezi jednotlivými objekty, neboť tato funkce byla podle výsledků uživatelského testování tou nejžádanější. Užitečné také bude do budoucna se zaměřit na možnost detekování i jiných objektů, než jsou jen kruhové. Potenciálně je také možné zaměřit se na detekci a porovnávání textů a čárových a QR kódů, což je oblast, které se komerční aplikace rozsáhle věnují. Dalším z vylepšení je, aby se okno definující plochu, v níž se vyhledává, mohlo po obraze přesouvat myší a zvětšovat nebo zmenšovat.

Literatura

[1] *Artificial Intelligence Machine Visio from Innomiles Solutions* [online]. Innomiles, srpen 2017 [cit. 2021-04-08]. Dostupné z: https://www.innomiles.com/artificial-intelligence-by-machine-vision/.

[2] *BOA Teledyne DALSA* [online]. Teledyne DALSA, 2021 [cit. 2021-04-03]. Dostupné z: https://www.teledynedalsa.com/en/products/imaging/smart-cameras/boa/.

[3] *In-Sight 2000 Vision Sensors Cognex* [online]. Cognex Corporation, 2021 [cit. 2021-04-03]. Dostupné z: https://www.cognex.com/products/machine-vision/vision-sensors/in-sight-2000-vision-sensors.

[4] *Intuitivní kamerový systém - Řada CV-X KEYENCE International Belgium* [online]. Omron Microscan, 2021 [cit. 2021-04-03]. Dostupné z: https://www.keyence.eu/cscz/products/vision/vision-sys/cv-x100/.

[5] Alqahtani, F., Banks, J., Chandran, V. a Zhang, J. Detection and Tracking of Faces in 3D Using a Stereo Camera Arrangements. *International Journal of Machine Learning and Computing*. Únor 2019, sv. 9, s. 35–43. DOI: 10.18178/ijmlc.2019.9.1.762.

[6] Clarke, T. A. a Fryer, J. G. The Development of Camera Calibration Methods and Models. *The Photogrammetric Record*. 1998, sv. 16, č. 91, s. 51–66. DOI: https://doi.org/10.1111/0031-868X.00113. Dostupné z: https://onlinelibrary.wiley.com/doi/abs/10.1111/0031-868X.00113.

[7] Huang, S.-H. a Pan, Y.-C. Automated visual inspection in the semiconductor industry: A survey. *Computers in Industry*. 2015, sv. 66, s. 1–10. DOI: https://doi.org/10.1016/j.compind.2014.10.006. ISSN 0166-3615. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0166361514001845.

[8] Jan, J. *Medical image processing, reconstruction, and restoration: concepts and methods*. Boca Raton, FL: Taylor & Francis, 2006. Signal processing and communications. ISBN 9780824758493. OCLC: ocm57192949.

[9] Karuppanagounder, S., Kalaividya, P., Thiruvenkadam, K., Krishnamoorthy, R. a Praveenkumar, S. Edge detection using Chebyshev's orthogonal polynomial and brain extraction from magnetic resonance images of human head. *International Journal of Imaging Systems and Technology*. Říjen 2018, sv. 29. DOI: 10.1002/ima.22297.

[10] Kuo, C.-F. J., Fang, T.-y., Lee, C.-L. a Wu, H.-C. Automated optical inspection system for surface mount device light emitting diodes. *Journal of Intelligent*

Manufacturing. únor 2019, sv. 30, č. 2, s. 641–655. DOI: 10.1007/s10845-016-1270-6. ISSN 0956-5515, 1572-8145. Dostupné z: http://link.springer.com/10.1007/s10845-016-1270-6.

[11] Malamas, E., Petrakis, E., Zervakis, M., Petit, L. a Legat, J.-D. A survey on industrial vision systems, applications and tools. *Image and Vision Computing*. Únor 2003, sv. 21, s. 171–188. DOI: 10.1016/S0262-8856(02)00152-X.

[12] Mar, N., Yarlagadda, P. a Fookes, C. Design and development of automatic visual inspection system for PCB manufacturing. *Robotics and Computer-Integrated Manufacturing*. 2011, sv. 27, č. 5, s. 949–962. DOI: https://doi.org/10.1016/j.rcim.2011.03.007. ISSN 0736-5845. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0736584511000457.

[13] Prabakaran, K., Khan, F., Abrar, N. M., Abbas, F. M. a Khrshid, S. A smart sensing and quantification of platelets, red blood cells (RBC), white blood cells (WBC) and classification of WBC's using microscopic blood image. *Int. J. Appl. Med. Sci. Res.* 2015, sv. 1, č. 1, s. 1–9.

[14] Russ, J. C. *The image processing handbook*. Boca Raton, FL: CRC Press, 2011. ISBN 9781439840634. OCLC: 714644349. Dostupné z: http://www.crcnetbase.com/isbn/9781439840634.

[15] Shirvaikar, M. Trends in automated visual inspection. *Journal of Real-Time Image Processing*. březen 2006, sv. 1, č. 1, s. 41–43. DOI: 10.1007/s11554-006-0009-6. ISSN 1861-8200, 1861-8219. Dostupné z: http://link.springer.com/10.1007/s11554-006-0009-6.

[16] Sturm, P. a Maybank, S. On plane-based camera calibration: A general algorithm, singularities, applications. In: *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*. 1999, sv. 1, s. 432–437 Vol. 1. DOI: 10.1109/CVPR.1999.786974.

[17] Thomas, A., Rodd, M., Holt, J. a Neill, C. Real-time Industrial Visual Inspection: A Review. *Real-Time Imaging*. 1995, sv. 1, č. 2, s. 139–158. DOI: https://doi.org/10.1006/rtim.1995.1014. ISSN 1077-2014. Dostupné z: https://www.sciencedirect.com/science/article/pii/S1077201485710145.

[18] Vlach, J. Hledání úseček a kružnic s využitím Houghovy transformace při zpracování obrazu v LabView. *Automa*. 2011, s. 42–44. ISSN 1210-9592. Dostupné z: http://www.odbornecasopisy.cz/res/pdf/42983.pdf.

[19] Yuen, H., Princen, J., Illingworth, J. a Kittler, J. Comparative study of Hough Transform methods for circle finding. *Image and Vision Computing*. 1990, sv. 8, č. 1, s. 71–77. DOI: https://doi.org/10.1016/0262-8856(90)90059-E. ISSN 0262-8856. Dostupné z: https://www.sciencedirect.com/science/article/pii/026288569090059E.

Příloha A

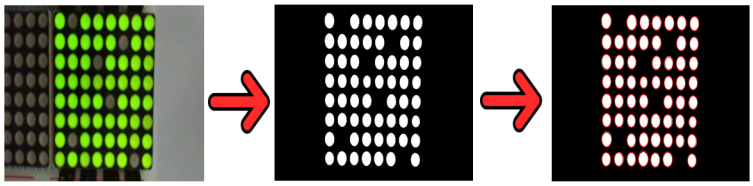
Plakát

Jednoduše konfigurovatelný kontrolní kamerový systém pro průmyslové aplikace

Autor: Ondřej Andrla
Email: xandrl09@stud.fit.vutbr.cz
Vedoucí práce: Ing. Michal Španěl, Ph. D

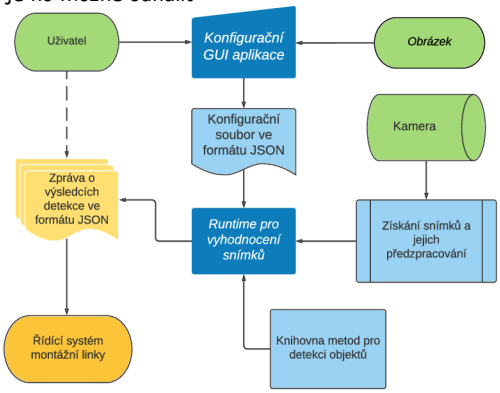
Akademický rok: 2020/21

Cíl práce
- Prototyp aplikace pro kontrolu výrobků na montážní lince
- Zaměření na detekci LED

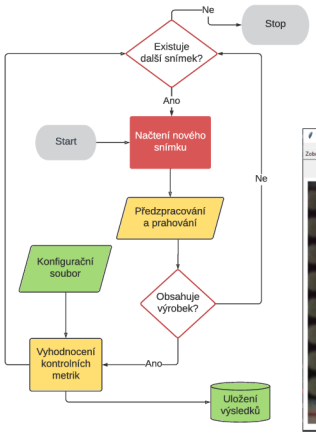


Motivace
- Běžná vizuální inspekce neodhalí vnitřní poškození výrobku
- Díky získání informací z kontrolních LED je ho možné odhalit

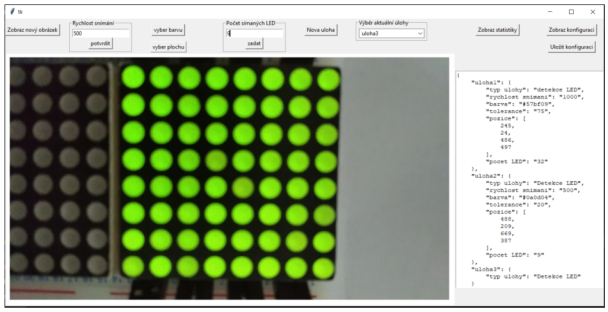
Koncepce řešení
- Oddělená GUI uživatelská aplikace a samostatně pracující runtime pro detekci kruhových objektů
- Přenos konfigurace ve formátu JSON



Runtime pro detekci
- K detekci používána Houghova kruhová transformace



Aplikace pro definování kontrolních úloh s GUI
- Uživatel definuje metriky, jako je počet LED, jejich barva, oblast detekce, ...



Příloha B

Dotazník

1. Jaký je váš věk?
2. Jaké je vaše pohlaví?
3. Věnujete se studijně/profesně oblasti informatiky?
4. Jak byste ohodnotili intuitivnost aplikace?
5. Jak srozumitelný vám připadá systém přepínání mezi jednotlivými úlohami?
6. Nastavování kterých údajů vám připadalo nesrozumitelné?
7. Jaké další funkce byste v aplikaci uvítali?
8. Jak se vám líbí grafický vzhled aplikace?
9. Sem můžete napsat další postřehy k aplikaci...

Příloha C

Obsah paměťového média

- Složka obsahující konfigurační soubory, která se jmenuje *konfigurace*.
- Složka obsahující testovací obrázky, která se jmenuje *prezentace*.
- Složka jménem *videa* obsahující zpracovávaná videa.
- Složka jménem *vysledky* obsahující JSON soubory s výsledky.
- Skript *configuration_GUI.py* spouštějící konfigurační aplikaci.
- Skript *detection_runtime.py* spouštějící runtime pro detekci objektů.
- Skript *run_runtime.py* spouštějící sktipt *detection_runtime.py* s nastavenými parametry.
- Soubor *plakat.pdf*, na němž je plakát.
- PDF s technickou zprávou.
- Zdrojový kód technické zprávy.
- Soubor README s návodem na použití.
- Soubor *requirements.txt* pro stažení potřebných knihoven.