FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání diplomové práce

Název:	Detekce a extrakce informací z fotografií dokladů
Student:	Bc. Tomáš Kuchař
Vedoucí:	Ing. Jan Zelenka
Studijní program:	Informatika
Obor / specializace:	Webové a softwarové inženýrství, zaměření Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	do konce letního semestru 2022/2023

Pokyny pro vypracování

1) Proveďte rešerši metod automatického rozpoznávání objektů a textů z fotografií.
2) Vytvořte službu, která na vstupu přijme obrázek osobního dokladu (občanský průkaz, řidičský průkaz, cestovní pas) a na výstupu vrátí strukturované textové údaje vyčtené z fotografie dokladu.
3) Služba bude napojena na veřejnou databázi neplatných dokladů poskytovanou ministerstvem vnitra České republiky. V odpovědi s textovými údaji z dokladu se navíc vrátí i informace, zda se přečtený doklad nachází na seznamu neplatných dokladů.
4) Služba bude napojena na veřejnou databázi RÚIAN. Po přečtení adresy z dokladu služba zkontroluje platnost vyčtené adresy a vrátí v odpovědi informaci o výsledku.
5) Důkladně otestujte vytvořenou službu pro všechny zmíněné typy dokladů.

Elektronicky schválil/a Ing. Michal Valenta, Ph.D. dne 3. května 2021 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Detekce a extrakce informací z fotografií dokladů

Bc. Tomáš Kuchař

Katedra softwarového inženýrství
Vedoucí práce: Ing. Jan Zelenka

5. prosince 2021

Poděkování

V první řadě bych chtěl poděkovat svému vedoucímu za to, že jsem měl možnost dostat se k tomuto zajímavému tématu. Dále za volnost a možnost vlastního zvolení postupů a technologií při vývoji řešení. Také bych chtěl poděkovat své rodině za podporu a čas, který mi dali, protože díky nim jsem měl vůbec možnost studovat vysokou školu. Děkuji.

České vysoké učení technické v Praze
Fakulta informačních technologií

Odkaz na tuto práci

Kuchař, Tomáš. *Detekce a extrakce informací z fotografií dokladů.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2021.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. Dále prohlašuji, že jsem s Českým vysokým učením technickým v Praze uzavřel dohodu, na základě níž se ČVUT vzdalo práva na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona. Tato skutečnost nemá vliv na ust. § 47b zákona č. 111/1998 Sb., o vysokých školách, ve znění pozdějších předpisů.

V Praze dne 5. prosince 2021 ……

Abstrakt

Cílem této práce je prozkoumat možnosti automatického rozpoznávání objektů a textů z obrázků a následně vytvořit službu, která na vstupu přijme fotografii osobního dokladu, nalezne doklad v obrázku a vyčte z něj strukturovaná data, která vrátí jako výsledek. Dále bude služba napojena na databázi neplatných dokladů poskytovanou Ministerstvem vnitra České republiky, díky které dokáže navíc k datům z fotografie přidat informaci o platnosti dokladu. Díky databázi RÚIAN se pro doklady obsahující adresu provede navíc validace této adresy. Vytvořená služba bude otestována pro český občanský a řidičský průkaz a cestovní pas.

Klíčová slova: osobní doklady, vyčtení dat z obrázku, občanský průkaz, řidičský průkaz, cestovní pas, OCR, opencv, tesseract

Abstract

The goal of this thesis is to explore methods of automated recognition of object and texts in images and to create a service, which consumes a photo of personal identity document at the input, finds document in image and reads structured data as the output. Service will take into account database of invalid documents provided by Ministry of the Interior of the Czech Republic, which allowes to include status about validity of found document in text response. For documents containing address will be this address validated according to RÚIAN database of valid addresses. Created service will be tested for czech ID card, driving licence and passport.

Keywords: identity document, text recognition from images, national identity card, driving licence, passport, OCR, opencv, tesseract

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Doba se vyvíjí a časy, kdy doklady četli pouze lidé, jsou pryč. V některých sektorech, kde je potřeba ověřit identitu osoby (například v bankách, pojišťovnách, půjčovnách vozidel, atd.), s přibývající digitalizací společnosti přicházejí požadavky na digitální uložení fotografie dokladu. Nebo se požadavky a registrace nových klientů vyřizují online. A tak přirozeně přichází potřeba kontrolovat fotografie dokladů, ověřovat a přepisovat data z nich.

Pro člověka toto není žádný problém, ovšem při větším množství požadavků může práce zabírat mnoho času a hodilo by se proces zautomatizovat. Pro počítač tento úkol ovšem není triviální.

Samozřejmě se toto téma už nějakou dobu řeší a existuje již řada komerčních řešení, která v rámci možností zvládnou úlohu splnit, ovšem většina z nich obsahuje různá omezení. Například nepodporují potřebné typy českých dokladů, fotografie musí být pořízena jen z určitého úhlu a vzdálenosti, musí být propojeny se skenerem nebo používat konkrétní aplikaci a nedají se tak napojit přes rozhraní do dalšího systému nebo jsou příliš pomalé (občas i více než 10 sekund). Tato omezení nevyhovují zadavateli práce, budu se tudíž snažit vytvořit vlastní službu na míru, ve které budou tyto problémy všechny vyřešeny.

Kapitola 1

Cíl práce

Hned ze začátku bych rád objasnil, proč tato práce vůbec vznikla. Zadavatelem práce je banka, která chtěla pomocí této služby zjednodušit proces vyplňování osobních údajů a kontroly fotografií osobních dokladů nejen zaměstnancům banky, ale i uživatelům internetového bankovnictví.

Hlavní příklad použití je při online zakládání nového zákaznického účtu u banky. Uživatel prochází postupně několika kroky a v každém kroku zadává nějaké údaje. Součástí procesu je i nahrání fotografií dvou různých typů platných osobních dokladů. Tento krok je jedním z prvních v pořadí. Po jejich nahrání se fotografie odešlou vytvořené službě, která z nich vyextrahuje požadovaná data a v dalším kroku vytváření zákaznického účtu automaticky zákazníkovi předvyplní pole strojově vyčtenými údaji z dokladu. Zákazník tak pouze zkontroluje předvyplněná data a pokud jsou v pořádku, tak pokračuje na další krok.

Další příklad použití je při zřizování nového účtu na pobočce banky. Zákazník přijde na přepážku, kde obsluha naskenuje fotografie dokladů a uloží je do systému CRM. Jedním tlačítkem pak může spustit čtení dokladu a služba opět předvyplní všechna potřebná textová pole ve formuláři a zároveň ověří platnost dokladu.

Výsledkem práce by tedy měla být REST služba, která na vstupu přijímá jeden nebo dva obrázky osobních dokladů. Možnost nahrát dva obrázky je z důvodu, že některé doklady obsahují data z obou stran a tyto dvě strany mohou (ale nemusí) být vyfotografovány zvlášť na dvou fotografiích. Při nahrání dvou souborů tak dopředu není známo, na které fotografii se nachází která strana dokladu. Podporované formáty obrázků jsou JPEG, PNG a BMP.

Dále služba přijímá nepovinný parametr s typem dokladu. Pokud typ dokladu není uveden, služba automaticky zjistí, jaký doklad se nachází na fotografiích. Podporované typy dokladů jsou český občanský průkaz (vzor 2005 a 2012), český řidičský průkaz a český cestovní pas. V odpovědi pak služba vždy vrátí typ nalezeného dokladu a vyčtené textové údaje, které zadavatele zajímají.

Ministerstvo vnitra České republiky zveřejňuje databázi neplatných dokladů, která je každý den aktualizována a je volně dostupná na webových stránkách [1]. Pro typy dokladů, pro které jsou tato data dostupná, se při přečtení údajů z průkazu navíc ověří, zda se doklad nachází na zmíněném seznamu neplatných dokladů. Tato informace se zahrne v odpovědi služby.

Český úřad zeměměřický a katastrální poskytuje veřejný Registr územní identifikace, adres a nemovitostí (RÚIAN), kde lze nalézt seznam všech existujících adres v České republice s dalšími užitečnými údaji [2]. Pro vybrané doklady, které obsahují adresu trvalého pobytu držitele, provede služba ověření automaticky přečtené adresy podle tohoto seznamu a v odpovědi vrátí informaci, zda byla přečtená adresa nalezena v databázi. Pokud ano, zjistí a zahrne v odpovědi navíc i poštovní směrovací číslo.

Cílem práce tedy je navrhnout službu, která bude splňovat všechny tyto požadavky zadavatele a bude použitelná v produkčních systémech pro každodenní používání. Služba bude pracovat s předem pořízenými fotografiemi a nebude běžet v reálném čase. Rychlost vyčtení dat tedy není nejdůležitější faktor a je potřeba zaměřit se spíše na kvalitu extrakce informací.

Je potřeba brát v potaz, že vstupní fotografie pochází od samotných uživatelů, kteří nejsou limitováni ničím jiným, než aby byl na fotografiích čitelný doklad. Snímky tak pořizují různými způsoby – různými typy fotoaparátů v různém prostředí nebo pomocí skenerů, kde bývá navíc naskenovaný celý zbytek listu A4 atd. Důležité tedy bude, aby nebylo řešení závislé na zdroji fotografie, úhlu vyfocení, perspektivnímu zakřivení, orientaci, formátu a velikosti fotografie, velikosti dokladu vůči velikosti fotografie, množství světla a stínů, vzorku pozadí, částečnému překrytí dokladu, odlesků světla, zkrátka aby služba dokázala co nejlépe zpracovat jakýkoliv vstup od uživatele.

Součástí práce nebude integrace služby do ostatních systémů.

Kapitola 2

Rešerše existujících řešení

Prvním logickým krokem před samotným vývojem je prozkoumat současný trh a najít, zda již neexistuje nějaké řešení a nešlo by použít, případně se jím inspirovat a zdokonalit ho.

2.1 Zdarma dostupná řešení

Už dopředu je jasné, že pro toto téma nebude existovat příliš kvalitních nástrojů, které jsou dostupné zadarmo. Po důkladném hledání jsem nenašel žádné volně použitelné řešení, které by přímo dokázalo vyčíst textové údaje z fotografií českých dokladů.

Jediné zdarma dostupné produkty, které se blíží problematice zadání, jsou knihovny do různých programovacích jazyků, které dokáží přečíst strojově čitelné oblasti dokladů (anglicky machine-readable zone, MRZ). Toto však není pro projekt dostatečné, jelikož v MRZ nejsou dostupné všechny potřebné údaje, jméno a příjmení je tam uvedené bez diakritiky a dokonce na českém řidičském průkazu tato oblast vůbec není zahrnuta.

Příkladem takovéto knihovny je PassportEye [3] psaná v Pythonu. Dokáže pro libovolný doklad obsahující MRZ vyextrahovat data s 80% úspěšností. Bohužel je poměrně pomalá. Autor uvádí, že pro různé dokumenty může proces trvat i více než 10 sekund. Na pozadí se používá knihovna Tesseract od společnosti Google, pomocí které se provádí samotné OCR.

2.2 Komerční produkty

Z komerčních produktů lze zmínit například aplikaci Accurascan [4], která nabízí patnáctidenní zkušební verzi zdarma. Lze použít buď mobilní aplikaci na platformy Android a iOS nebo webové rozhraní. Software umožňuje číst různé doklady skoro z 200 zemí, ovšem je nutné dopředu zvolit zemi a typ dokladu. Pro Českou republiku podporují cestovní pas, nový typ občanského

průkazu a víza. Na webových stránkách se výrobce pyšní svými zákazníky, jako je například T-Mobile, a také ambiciózní 100% spolehlivostí přečtení dat, ovšem při snaze vyčíst data z občanského průkazu mi aplikace doklad v obrázku několikrát vůbec nenašla. Ani rychlost není ideální, některé požadavky přes webové rozhraní zabraly i více než 10 sekund. Navíc textový výstup nepodporuje českou diakritiku.

Aplikace BlinkID [5], která je dostupná na různá mobilní zařízení i přes webové rozhraní, dokáže také rozeznávat české doklady. Dokonce umí automaticky rozpoznat typ dokladu a není tak nutné dopředu zvolit, jaký průkaz se bude číst. Při testování přes webové rozhraní vracela výsledky překvapivě rychle, většinou do jedné sekundy. V polovině testování občanských průkazů však doklad ve fotografii nenalezla. Zároveň měla problém se zadní stranou občanky, kde dokázala přečíst pouze strojově čitelnou oblast a například adresu nebo rodné číslo při testování nikdy nerozpoznala. Příjemným překvapením však je podpora i starších typů občanských průkazů. Podobně úspěšně si BlinkID vedla i při testování řidičských průkazů. Občas se vrátil výsledek, že se jedná o švédský řidičák, avšak vyčtené údaje byly správné. Cestovní pas aplikace rozpoznala také pouze v polovině testování, největší úspěšnost měla v případech, kdy byla fotografie oříznuta přímo na doklad. Nevýhodou u pasů je absence diakritiky ve jméně a příjmení, zřejmě se tyto údaje získávají ze strojově čitelné zóny, kde diakritika chybí.

Dalším komerčním produktem podporujícím všechny tři požadované české doklady je Smart Engines [6]. Nabízí mobilní aplikaci, webové rozhraní i instalaci na desktop či server. Otestování služby je možné pouze přes mobilní aplikaci. Už ve vzorovém videu [7] však aplikace pro český řidičský průkaz nedokázala správně přečíst některé údaje. Toto se ověřilo i při mých několika pokusech o přečtení různých dokladů. Nicméně z doposud zmíněných produktů si Smart Engines vedl při mém testování výrazně lépe než ostatní. Jediný problém občas nastával při pokusu naskenovat obě strany dokladu najednou z jedné fotografie. V tomto případě aplikace často nalezla pouze jednu stranu. Nevýhodou také je nutnost zvolit předem typ dokladu. Řešení od Smart Engines používá řada světových bank a dalších klientů. Z jmen známých u nás lze vybrat například Raiffeisen Bank, Home Credit Bank nebo platformu pro nabízení spolujízdy BlaBlaCar.

Dále lze na internetu dohledat spoustu dalších služeb, které nabízí vytěžování dat z dokladů, ovšem nepodporují české typy dokladů nebo jsou zaměřené pouze na konkrétní zemi. Některé služby fungující na principu neuronových sítí také nabízí přidání podpory dalších dokladů po poskytnutí trénovacího balíku vzorků. K nalezení je také spousta komerčního softwaru pro univerzální čtení strojově čitelných zón různých průkazů, který funguje kvalitněji než bezplatné alternativy.

2.3 České výtvory

Obecná nevýhoda všech univerzálních zahraničních produktů je, že nejsou důkladně přizpůsobené na všechny typy českých dokladů. Kromě toho samozřejmě neumožňují propojení se zmíněnými databázemi neplatných dokladů a existujících adres, ale to by šlo k výstupu doplnit dodatečně. Hojně mají také problémy s diakritikou. Naproti tomu tuzemští výrobci o svých řešeních nezveřejňují mnoho informací a nelezl jsem tak pouze dva výrobce, kteří nabízí produkt podobný zadání této práce a uvedli o něm nějaké podrobnější informace. Ostatní společnosti pouze tvrdí, že toho ohledně tématu OCR spoustu umí, ale konkrétní možnosti nezmiňují.

První ze zmíněných produktů pochází od firmy Codeware s.r.o. [8] a nabízí software propojený s hardwarovou čtečkou, která podle dostupných informací načítá pouze strojově čitelnou část obsahu dokladů. Následně je software napojený na další systémy, ze kterých si stáhne doplňující informace. V ukázkovém videu s příkladem použití se ale například adresa z dokladu nevyplnila správně a video nejen díky tomuto nedostatku působí amatérsky. Navíc jsou podporovány pouze doklady se strojově čitelnými údaji, tudíž nelze zpracovat řidičský průkaz.

Druhým produktem je služba ZenID [9] od firmy Trask solutions a.s., která nejlépe ze všech uvedených případů splňuje zadání této práce. Podporuje občanský a řidičský průkaz i cestovní pas. Společnost nabízí integraci do jakéhokoliv systému. Současně je služba také napojena na databázi neplatných dokladů, existujících adres a dokonce i do insolvenčního rejstříku. Kromě vytěžování informací z dokladů nabízí i další služby, jako je detekce padělaného dokladu nebo ověřování obličeje. Uživateli ZenID jsou například Komerční banka, Moneta, Sazka nebo Cofidis. Autoři slibují 95% úspěšnost při vytěžování informací z dokladů, bohužel však nenabízí možnost službu vyzkoušet. Hlavní nevýhodou služby je její licencování, které je založené na ročním předplatném omezeném počtem zpracovaných požadavků. Dále také podmínka, že veškerá volání služby, vstupy a výstupy jsou pro účely auditu zaznamenávány do interní databáze poskytovatele.

Mé řešení se funkcionalitou nebude příliš lišit od veřejného popisu softwaru ZenID, pokusím se mu tedy konkurovat a dodat vlastní produkt bez drobných nevýhod ZenID a nutnosti platit každoroční předplatné.

Kapitola 3

Popis podporovaných dokladů

Pro lepší představu o vytvářené službě se hodí zanalyzovat si doklady, které bude služba podporovat. Mezi ně patří český občanský a řidičský průkaz a český cestovní pas.

3.1 Občanský průkaz

Od zavedení povinnosti občanských průkazů v tehdejším Československu se podoba tohoto dokladu poměrně často měnila. První typ občanského průkazu byl zaveden 17. března 1939 nařízením říšského protektora Konstantina von Neuratha. Průkaz byl navržen podle podobného dokumentu s fotografií z Třetí říše. [10]

V roce 1948 se průkaz změnil na knížku o rozměrech 89×130 mm. Desky byly plátěné v odstínu červené barvy a uvnitř bylo 30 číslovaných stran. Na stránkách knížky byly uvedené osobní informace o držiteli průkazu, které zahrnovaly i údaje o dětech, rodičích a manželech, rodné číslo držitele, nejvyšší dosažené vzdělání, informace o branném poměru, zaměstnavatele, ale i zdravotní údaje jako prodělaná očkování, krevní skupinu či závažné choroby. Tento styl knížky se zachoval až do roku 1993, přičemž mezi tím došlo celkem pětkrát k různým změnám ve vzhledu, ochranných znacích, formátování, uspořádání a povinnosti uváděných údajů . Průkaz byl vydáván všem československým občanům při dosažení věku 15 let. [10] [11]

V květnu roku 1993 se po osamostatnění České republiky začal českým občanům vydávat nový typ průkazu, který už neměl formu knížky, ale oboustranné zalaminované karty o rozměrech 82×111 mm. I na tomto průkazu se postupně v letech 1994 a 1996 měnily různé grafické detaily a uspořádání údajů. [11]

První typ dokladu obsahující strojově čitelnou oblast přišel v roce 2000 a vydával se do konce roku 2004. Byl v mezinárodním formátu ID2. Oproti předchozím verzím přibyly anglické překlady názvů jednotlivých polí. Na před-

ní straně dokladu byl kromě osobních údajů o držiteli navíc zahrnut jeho podpis. [10]

Od začátku roku 2005 se předchozí model drobně upravil. Byl přidán překlad názvů položek do francouzštiny a přesunut údaj o místě narození. V září téhož roku byl opraven překlad některých francouzských názvů a od dalšího roku byl na zadní stranu přidán údaj o uzavření registrovaného partnerství a nepovinně údaje o partnerovi. [10] Tento typ zalaminované karty velikosti 74×105 mm byl vydáván do konce roku 2011 standardně s desetiletou dobou platnosti pro osoby nad 15 let [12]. Pro osoby starší 70 let se vydával s platností 35 let, je tedy možné, že tento typ průkazu v době psaní práce ještě někdo vlastní a bude proto muset být podporován vyvinutým softwarem.

Obrázek 3.1: Vzor českého občanského průkazu z roku 2005. Převzato z [10].

Další výrazná změna přišla 1. ledna roku 2012, kdy starý model z roku 2005 nahradila plastová karta formátu ID1 s rozměry platební karty (54×85.6 mm). Přední strana obsahuje kromě osobních údajů i gravírovanou černobílou fotografii a gravírovaný podpis držitele. Jméno, příjmení a číslo dokladu jsou provedeny taktilním gravírováním, lze je tak poznat hmatem. Oproti starší verzi opět zmizel francouzský překlad názvů polí a možnost zapsat si na zadní stranu rodinné příslušníky. Občanský průkaz může být vydán s elektronickým čipem (od srpna 2018 povinně), liší se pak oproti průkazu bez čipu pouze jeho viditelností na zadní straně. V květnu 2014 došlo k drobné grafické změně ve způsobu tisku státního znaku a v použití transparentního kinegramu namísto hologramu na přední straně přes fotografii. [12]

Tento typ občanského průkazu je poprvé možné vydat i osobám mladším

15 let na dobu platnosti 5 let a také osobám s trvalým pobytem mimo území České republiky. Pro osoby mezi 15 a 70 lety se vydává na dobu platnosti 10 let, pro starší osoby na dobu platnosti 35 let. Uvádění rodinného stavu se stalo volitelným údajem stejně jako zápis rodného čísla od začátku roku 2020. [10] Vzor občanského průkazu z roku 2012 je v současnosti nejrozšířenější, proto bude služba pro vytěžení textových údajů zaměřena hlavně na tento typ.

Obrázek 3.2: Aktuální vzor českého občanského průkazu z roku 2012. Převzato z [10].

V srpnu 2021 nabylo účinnosti nařízení Evropského parlamentu a Rady EU 2019/1157, které standardizuje vzhled občanského průkazu po celé Evropské unii [10]. Oproti předchozí podobě můžeme v horním levém rohu přední strany najít modrý obdélník se symbolem Evropské unie a uvnitř mezinárodní zkratku České republiky "CZ". Na zadní straně přibylo logo biometrického dokladu. [13]

3.2 Řidičský průkaz

Historie řidičských průkazů v České republice byla podobně pestrá jako historie průkazů občanských. V průběhu dvacátého století se průkaz vyvíjel od papírových ručně psaných jízdních licencí, různých knížek a skládaček přes podobu zalaminovaných kartiček podobných starším občanským průkazům až po současnou podobu plastové karty poprvé použité v roce 2004. [14]

Od začátku května roku 2004 se v České republice začal vydávat řidičský průkaz Evropské unie, který má stejně jako nový typ občanského průkazu formát platební karty s rozměry (54×85.6 mm). Na přední straně můžeme najít osobní informace o držiteli jako příjmení, jméno, datum a místo narození, rodné číslo, bydliště, dále číslo a rozsah platnosti dokladu, vystavující autoritu a kategorie řidičských oprávnění. Nechybí ani černobílá fotografie a podpis majitele. V levém horním rohu se nachází symbol Evropské unie.

Na zadní straně jsou uvedeny detaily jednotlivých kategorií řidičských oprávnění a další podrobnější informace. Zadní část dokladu prošla drobnou

změnou od 19. ledna 2013, kdy byly lehce přeuspořádány jednotlivé řádky v tabulce kategorií a byla přesunuta legenda k nenadepsaným políčkům. [15]

Pro zadavatele práce jsou dostačující údaje na přední straně řidičského průkazu. Z tohoto důvodu ani nebude vytvořený program hledat zadní stranu dokladu a vystačí si pro vytěžení požadovaných dat se stranou přední.

Obrázek 3.3: Aktuální vzor českého řidičského průkazu z roku 2013. Převzato z [15].

3.3 Cestovní pas

Cestovní pas má už od založení samostatné České republiky podobu sešitu s pevnými deskami o rozměrech 88×125 mm. Od roku 1993 vzniklo celkem 7 verzí tohoto dokladu. Současná verze s červenými deskami a strojově čitelnými údaji se vydává od 1. září 2006. Platnost cestovního pasu je 5 let pro osoby mladší 15 let a 10 let pro osoby starší. [15]

Výjimečně lze také v současné době vidět dočasný cestovní pas se zelenými deskami a platností 6 měsíců. Tato verze neobsahuje strojově čitelné údaje. V této práci se dočasný cestovní pas nebude brát v potaz, jelikož se vydává pouze ve výjimečných případech. Vytvořená služba bude podporovat pouze klasický cestovní pas s červeným obalem, který můžeme v dnešní době vidět pouze v nejnovější verzi z roku 2006. [15]

Oproti občanskému a řidičskému průkazu má cestovní pas všechny hlavní údaje uvedeny pouze na jedné straně dokladu. Pole jsou nadepsána česky, anglicky a francouzsky. Tato strana obsahuje příjmení a jméno držitele, datum a místo narození, rodné číslo, pohlaví, číslo dokladu, rozsah platnosti pasu, vydávající autoritu, podpis a černobílou fotografii majitele pasu.

Na spodní části stránky jsou dva řádky strojově čitelných údajů. Tyto řádky obsahují stejně jako u občanského průkazu jméno a příjmení majitele bez diakritiky, číslo dokladu, zkrácené datum narození, pohlaví, datum expirace dokladu a rodné číslo majitele. Díky kontrolním součtům je možné při

strojovém čtení těchto údajů detekovat chyby ve čtení a pokus případně zopakovat.

Obrázek 3.4: Aktuální vzor českého cestovního pasu z roku 2006. Převzato z [15].

Kapitola 4

Návrh řešení

Postup od nahrání obrázků uživateli až po vrácení textových údajů nazpátek se bude skládat z několika samostatných kroků. Zjednodušený popis postupu bude vypadat zhruba takto:

1. Nalezení požadovaného dokladu ve fotografii. Je potřeba určit všechny jeho čtyři rohy.
2. Otočení a perspektivní transformace nalezeného dokladu podle jeho čtyř rohů.
3. Oříznutí a změna velikosti obrázku, aby zůstal pouze narovnaný doklad s požadovanými rozměry.
4. Předzpracování oříznuté fotografie a příprava na čtení.
5. Vyčtení všech požadovaných údajů.
6. Korekce a validace vytěžených textových údajů.
7. Případné doplnění dat z napojených externích služeb.

Konkrétní detailnější postup pro všechny zmíněné kroky rozeberu v následujících podkapitolách.

4.1 Nalezení dokladu v obrázku

Nalezení samotného dokladu uvnitř poskytnuté fotografie je z celého procesu ten nejsložitější a nejdůležitější úkol, jelikož se od něj následně odvíjí úspěšnost všech dalších kroků. Situaci komplikuje naprostá volnost uživatelů ve způsobu pořízení fotografie. Jedinou podmínkou je, aby byl doklad na fotografii čitelný pro lidské oko, a počítač se tak musí pro lepší pohodlnost uživatelů dokázat přizpůsobit.

Hledaný doklad může být na fotografii libovolně umístěný, otočený o libovolný úhel, vyfocený z perspektivy, s různým množstvím světla včetně ostrých stínů a světelných odlesků, částečně překrytý prsty či jinými předměty a může být dokonce oříznutý tak, že nepodstatná část dokladu úplně chybí. Všechny tyto požadavky vychází ze zkušenosti s nahrávanými fotografiemi reálných uživatelů, nelze z nich proto slevit.

Metod nalezení objektů ve fotografiích je v současné době dostupných několik. Zaměřím se na hlavní způsoby, které by bylo možné použít pro tuto práci. Některé názvy ponechám v angličtině, jelikož často nemají český překlad a při pokusu přeložit je vlastním způsobem by nemuselo být zřejmé, o které metody a pojmy se jedná.

4.1.1 Metody detekce objektů pomocí neuronových sítí

Jedním z hlavních možných způsobů nalezení dokladu ve fotografii je použití nějakého z algoritmů pracujících na bázi neuronových sítí. Hledání objektů ve fotografiích a videu je dokonce jedno z nejrozšířenějších současných použití strojového učení a neuronových sítí. Tyto sítě se v dnešní době nechají za pomoci různých dostupných nástrojů a dostatečného množství vzorků hledaných objektů vytrénovat k nalezení těchto objektů s použitelnou úspěšností. Příkladem z reálného světa může být odemykání mobilního telefonu pomocí obličeje, automatické rozpoznání státní poznávací značky automobilů při příjezdu na parkoviště, sledování různých objektů v reálném čase (například sledování míče při fotbalovém zápase pro automatický pohyb kamery), samořídící automobily nebo různé využití v robotice. [16]

Jedním z úplně prvních způsobů detekce a lokalizace objektů v obrázku používající strojové učení pomocí neuronových sítí je metoda Region-based Convolutional Neural Network (R-CNN). V roce 2014 ji popsal čerstvě vystudovaný Američan Ross Girshick. Celý postup se skládá ze tří oddělitelných kroků. V první fázi se navrhne několik tisíc obdélníkových regionů v obrázku, které jsou podezřelé, že by se v nich mohly nacházet nějaké objekty. Tyto oblasti se hledají pomocí selektivního vyhledávání, které bere v úvahu podobnou barvu sousedních pixelů, podobný jas, texturu a další faktory, pomocí kterých pak navrhne hranice regionů. V dalším kroku se s pomocí předem natrénovaných neuronových sítí z každého navrženého regionu vyextrahují tzv. features, čímž se v oblasti strojového vidění rozumí podezřelá místa, která splňují určité (např. geometrické) vlastnosti a pomáhají tak identifikovat obsah obrázku. V další fázi se potom porovnávají tyto features se známými modely a pomocí toho se určuje, zda se v regionu nachází hledaný objekt a případně se lehce posunují hranice regionu pro přesnější zaměření objektu.

Největší nevýhodou tohoto řešení je dlouhá doba zpracování obrázku, která brání použití v reálném čase. To pro tento projekt není takový problém, ovšem čím rychlejší zpracování, tím lépe. V roce 2015 se tento problém pokusili vyřešit pomocí vylepšení předchozí metody s názvem Fast R-CNN. Na rozdíl od

předchůdce tato metoda neextrahuje features pro každý navržený region, ale provede tuto operaci pouze jednou pro celý obrázek a zjištěné informace pak použije v jednotlivých regionech.

Ani tato metoda nebyla dokonalá, proto o rok později přišla varianta Faster R-CNN, která navíc vylepšila vyhledávání podezřelých regionů. Pro stanovení hranic regionů se používá také neuronová síť, která zohledňuje nalezené features a dokáže díky nim detekovat přesnější regiony. Těch oproti předchozí variantě stačí zhruba desetina, což ještě více urychlí tuto metodu, která v době vzniku při testování na GPU trvala zhruba 200 milisekund. [17]

Další rodinou metod pro detekci a lokalizaci objektů v obrázku je YOLO (z anglického You Only Look Once). Byla popsána v roce 2015 Josephem Redmonem, ovšem spoluautorem byl i Ross Girshick, který má na svědomí předchozí popsané metody. Stejně jako R-CNN používá předem natrénovanou neuronovou síť, rozdílem však je, že přijme na vstupu celou originální fotografii a v jednom průchodu obrázkem provede všechny potřebné operace k nalezení regionů s detekovanými objekty. Díky tomu je metoda rychlá a vhodná k použití v reálném čase. Zpracování jedné fotografie metodou YOLO v základní neoptimalizované verzi trvá zhruba 20 milisekund, což je více než tisíckrát rychlejší než metoda R-CNN a více než stokrát rychlejší než Fast R-CNN.

Postup YOLO spočívá v tom, že na začátku rovnoměrně rozdělí celou fotografii na síť $n \times n$ shodných obdélníků. Pro každou část pak samostatně probíhá detekce pomocí natrénované neuronové sítě, zda se v něm nachází střed nějakého zkoumaného objektu. Pokud ano, vrátí se odhadnuté rozměry hledaného objektu, které mohou sahat i mimo zkoumaný obdélník a pravděpodobnost, s jakou se tam objekt skutečně vyskytuje. Po průchodu všech obdélníků vznikne seznam všech regionů ve fotografii, kde se nachází detekované objekty, a jejich pravděpodobnosti. Může se stát, že se objekt nalezl ve dvou sousedních obdélnících, ale jedná se o stejný objekt. V takovém případě se spočítá poměr společné oblasti v obou regionech vůči součtu oblastí, kde se regiony nepřekrývají a pomocí tohoto poměru se rozhodne, zda se jedná o jeden dvakrát nalezený objekt nebo o dva různé objekty vedle sebe. Pokud se jedná pouze o jeden, pak se jako výsledný region bere ten s vyšší pravděpodobností. [17]

V dalších letech pak vznikaly nové verze YOLO metody, které postupně vylepšovaly výkon, úspěšnost a způsoby řešení některých částí algoritmu. Vznikly tak například verze YOLO v2, YOLO v3, YOLO v4, YOLO v5, PP-YOLO nebo YOLOR.

Kromě uvedených dvou hlavních metod stojí ještě za zmínku algoritmus Single Shot Detector (SSD), kterému stejně jako YOLO stačí pouze jeden průchod obrázkem a stíhá tak fungovat i ve videu v reálném čase. Nevýhodou je nižší kvalita pro malé objekty.

Další důležitou metodou je RetinaNet uvedená v roce 2017, která v tu dobu byla nejlepším dostupným způsobem detekce objektů ve fotografii pomocí neuronových sítí. Rychlostí se vyrovnala již existujícím metodám YOLO

v2 a SSD a navíc měla lepší úspěšnost srovnatelnou s pomalejší metodou R-CNN. Nejen díky tomu našla RetinaNet časté využití například při zpracování leteckých a satelitních snímků. [16]

Na internetu je k nalezení řada knihoven, které mají implementované uvedené metody pro detekci a lokaci různých objektů ve fotografiích. Nejčastěji hledané objekty pro běžné použití už bývají v knihovnách zabudovány a často nabízí i možnost vytrénovat neuronovou síť pro vlastní typy objektů.

Příkladem je knihovna ImageAI do jazyka Python, která používá známý framework pro deep learning TensorFlow. Knihovna nabízí spoustu různých operací spojených s detekcí a lokalizací známých objektů, trénování vlastních objektů a podporuje kromě obrázků i zpracování videa. Nabízí předtrénovanou neuronovou síť pro 80 nejběžnějších předmětů a v jednom obrázku dokáže nalézt až 1000 objektů.

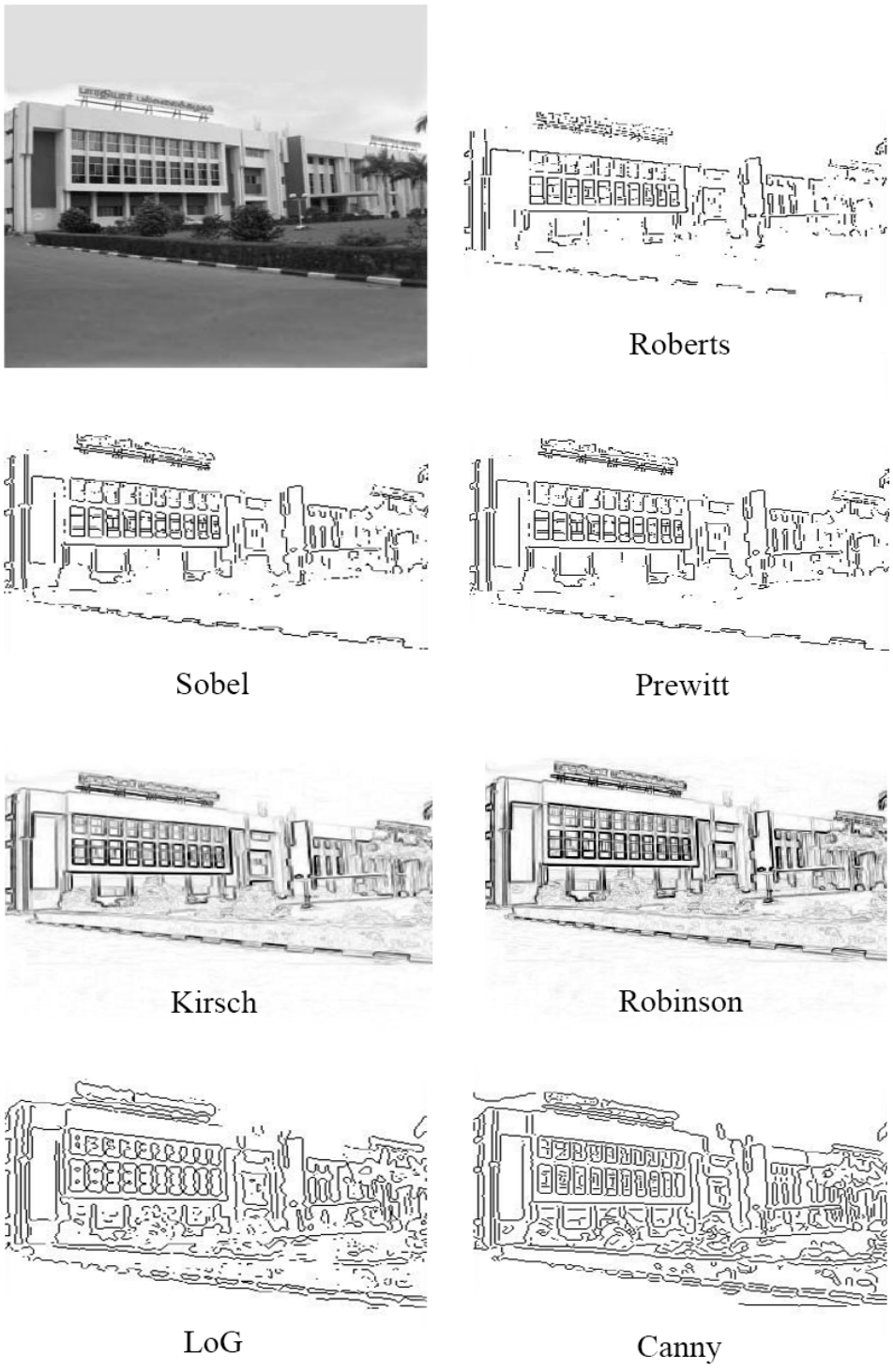
Další z nejlepších dostupných knihoven je GluonCV, která kromě všech důležitých funkcí včetně možnosti vycvičení na vlastní typy objektů podporuje i rozpoznání polohy těla člověka. Díky tomu dokáže v reálném čase pojmenovat aktivity, které lidé vykonávají.

Mezi další zajímavé knihovny podporující trénování vlastních objektů patří Detectron2 od společnosti Facebook, YOLOv3_TensorFlow lépe podporující malé objekty nebo knihovna Darkflow. [16]

Obecnou nevýhodou metod používajících neuronové sítě je, že pro to, aby se dokázaly naučit vyhledávat ve snímcích vlastní typy objektů, potřebují co největší množství trénovacích dat. Pro dobrou úspěšnost jsou potřeba alespoň tisíce vzorků hledaných předmětů. Existují programy, které dokážou uměle vygenerovat z několika základních vzorků spoustu obrázků, kde vzorky umístí na odlišná pozadí a různě je upraví, ovšem vždy je lepší a přesnější, když jsou data skutečná.

4.1.2 Metoda hledání hran čtyřúhelníků

Myšlenkou metody hledání hran čtyřúhelníků je projít celou fotografii a pokusit se detekovat rovné linky v obraze. Následně se znalostí poměru stran hledaného dokladu vybrat čtyři vhodné nalezené hrany, které s nejvyšší pravděpodobností vymezují čtyřúhelník obsahující doklad. V rozhodování, o které hrany půjde, může pomoci i známá struktura konkrétního dokladu. Například u občanských průkazů vydávaných od roku 2012 víme, že na přední straně obsahují vlevo obdélníkovou fotografii. Budeme tedy hledat čtyřúhelník, který v sobě obsahuje další menší ve správném poměru. Obecně také pomáhá hledat spíše vnější čtyřúhelníky, protože uvnitř hledaných dokladů se nachází spousta dalších hran. Problémem pak může být velká plocha pozadí fotografie, která obsahuje rovné hrany. Jakmile se však naleznou správné hrany dokladu, není už problém vypočítat jejich průsečíky, dostat tak souřadnice rohů a doklad dále zpracovat.



Obrázek 4.1: Příklady rozdílných metod pro detekci hran ve fotografii. Převzato z [18], upraveno.

V praxi se hledají hrany ve spoustě různých aplikací. Tato metoda se uplatňuje například v softwarových skenerech, které používají fotoaparát mobilního telefonu, aby nalezly hrany a ořízly skenovaný dokument. Dále je možné se s hledáním hran setkat i v některých variantách již zmíněného čtení státních poznávacích značek automobilů. Dokonce i některé aplikace pro čtení dokladů z provedené rešerše (např. BlinkID [5]) tuto metodu obsahují.

V různých grafických knihovnách je dostupná spousta algoritmů pro detekci hran ve fotografiích. Jedním z nejčastěji zmiňovaných je Cannyho metoda detekce hran vytvořená v roce 1983 jako diplomová práce, která však i v současnosti stále překonává spoustu novějších metod a je proto používaná v řadě odvětví. První krok algoritmu je Gaussovské rozostření pro potlačení šumu v obrázku. Následně se vypočítá gradient pomocí zvoleného operátoru. Pro každý pixel se potom vyhodnotí, zda se jedná o hranu a v jakém směru. V posledním kroku se pak pro pixely s hranami kontroluje okolí a jednobodové hrany se stejným směrem se spojují do velkých výsledných hran. [18]

Dalšími zástupci, kteří se hodí spíše pro hledání vodorovných a svislých hran, jsou například Sobelův algoritmus nebo Prewittův algoritmus. Dalších dostupných metod je spousta, nejnázorněji ilustruje rozdíly v detekci hran obrázek 4.1.

Metoda hledání hran čtyřúhelníků má však zásadní nevýhody, kvůli kterým se nehodí pro použití v této práci. Největším problémem je, že není zajištěna viditelnost všech hran dokladu na zkoumané fotografii. Doklad může být z části překrytý nějakým předmětem, fotografie může být oříznutá až do plochy dokladu nebo může nějaká strana dokladu splývat s pozadím takovým způsobem, že přesná hrana nebude viditelná.

Další komplikací může být fotka z perspektivy, na které protilehlé hrany dokladu nebudou rovnoběžné a může nastat problém s výběrem správných hran. Problémem může být i pozadí fotografie. V některých reálných vzorcích totiž doklad leží na klávesnici počítače nebo čtverečkovaném papíru a pozadí tak obsahuje velké množství rovnoběžných matoucích hran, které mohou vést ke špatné detekci dokladu. Dokonce i ostré stíny tvoří na fotografii hrany, které znesnadňují nalezení.

Při nalezení správných hran ohraničujících doklad pak nastává další problém a to zjištění orientace dokladu ve čtyřúhelníku. Doklad totiž může být otočený o 180° a metoda toto nezvládne detekovat.

4.1.3 Template matching

Dalším způsobem nalezení konkrétního objektu ve fotografii je hledání pomocí šablony. Pokud znám dopředu přibližnou podobu hledaného objektu, mohu si vytvořit šablonu, pomocí které následně hledám v originálním obrázku podobnou část. Tento způsob se hodí právě na problém zadaný v této práci, protože dopředu vím, jak mají vypadat všechny typy hledaných dokladů

v obrázcích a tyto doklady se vždy ve většině obsahu shodují. Liší se pouze některé texty, případně fotografie hlavy majitele dokladu a podpis.

Algoritmy pro template matching se nechají rozdělit do dvou skupin. První skupina je v angličtině pojmenovaná "template-based approach"(někdy také "area-based approach"), druhá skupina "feature-based approach". Algoritmy z první skupiny jsou jednodušší na implementaci. Jejich princip spočívá v tom, že do rohu zkoumaného obrázku vloží obrázek šablony, který následně pixel po pixelu posunují po původním obrázku a přitom porovnávají, v jaké míře se shoduje šablona s překrytou plochou původního obrázku. Pro přesnější výsledek se obrázky převádí do černobílé podoby a využívají se další techniky zpracování obrázků a měření shody. Jedná se tedy o řešení hrubou silou.

Samozřejmě nastává problém, pokud jsou rozměry objektu v hledaném obrázku jiné než rozměry šablony. Tento problém algoritmy řeší opět hrubou silou, vyzkouší šablonu přeškálovat na několik různých velikostí a opět porovnávají, kdy se bude nejlépe shodovat s nějakým úsekem zkoumaného obrázku. Další problém je v případě, že objekt je v hledaném obrázku otočený o nějaký úhel oproti šabloně. Tento případ řeší Ciratefi algoritmus z roku 2007, který má stejnou úspěšnost jako zkoušení všech úhlů hrubou silou, ovšem je 400× rychlejší.

Tento algoritmus si však stále neporadí s perspektivní transformací nebo větším překryvem nepodstatné části dokladu a není tak úplně vhodný pro řešení této práce. Navíc pro velké obrázky je hodně pomalý.

Do druhé skupiny patří algoritmy využívající již popsané features. Jedná se o hrany, rohy, něčím zajímavé body, ale i místa v obrázku, která člověku nepřipadají ničím výjimečná, ale pro počítač splňují určité důležité vlastnosti. Features jsou unikátní pro každý obrázek a díky tomu pomáhají rozhodnout o tom, jak moc se různé obrázky shodují nebo liší. Jejich hlavní výhodou je, že zůstanou stejné i v případě, že se obrázku změní velikost nebo se orotuje (existují i algoritmy, kde toto neplatí stoprocentně). [19]

Pro detekci a extrakci features z obrázku existuje několik různých metod. Jednou z nejznámějších je SIFT (Scale-Invariant Feature Transform). Metoda byla vytvořena v roce 1999 a do loňského jara byla patentovaná v USA. Jak název napovídá, metoda je nezávislá na změnách velikosti obrázku. Kromě toho je navíc nezávislá na otočení, zrcadlení, částečně i na jakékoliv afinní transformaci, 3D projekci a změně osvětlení. Tím se liší oproti dříve vyvinutým metodám, které povětšinou tyto všechny vlastnosti neměly. Díky její složitosti je však poměrně pomalá a kvůli tomu nepoužitelná pro zpracování snímků v reálném čase.

Další důležitou metodou je SURF (Speeded Up Robust Features), která částečně vychází ze SIFT, je však několikanásobně rychlejší. Vznikla v roce 2006 a mimo akademické účely je také doposud patentovaná v USA, na Českou republiku by se však patent neměl podle dostupných informací z databáze patentů a užitných vzorů [20] Úřadu průmyslového vlastnictví vztahovat. Tato metoda je oproti předchozí robustnější vzhledem k transformacím obrázku,

změně osvětlení a rozmazání. Pokud však tyto změny nejsou v obrázku zásadní a nezáleží na rychlosti, úspěšnost SIFT stále vítězí.

Dále dostupnou metodou detekce features je FAST (Features from Accelerated Segment Test, 2006), která je velmi výpočetně efektivní a zaměřuje se na detekci rohů. Zvládá fungovat v reálném čase při zpracování videí a používá se tak ke sledování objektů ve videích. Metoda bohužel neobsahuje druhý potřebný krok extrakce features a je nutné ji zkombinovat s nějakým deskriptorem features jako je například BRIEF. Metody BRISK (Binary Robust Invariant Scalable Keypoints) a ORB (Oriented FAST and Rotated BRIEF) vznikly v roce 2011 jako bezpatentové a rychlejší alternativy k SIFT a SURF, v některých případech i s lepší úspěšností. [21]

Celý algoritmus pro nalezení objektu ve fotografii pomocí feature-based template matchingu se skládá ze čtyř kroků. První dvě operace jsou detekce a extrakce features pomocí již popsaných metod. Musí proběhnout ve zkoumaném obrázku i v hledané šabloně. Dalším krokem je spárování shodných features ze zkoumaného obrázku a šablony. Kolekce optimalizovaných algoritmů FLANN (Fast Library for Approximate Nearest Neighbors) se postará o vytvoření nejpodobnějších spojení nalezených features z obou obrázků. Pomocí Lowe's ratio testu se vyfiltrují pouze nejpravděpodobnější z nich. Ve čtvrtém posledním kroku se pomocí iterativního algoritmu RANSAC (Random Sample Consensus) nalezne homografie mezi obrázky. To znamená, že se ve zkoumaném obrázku pomocí vyfiltrovaných shodných features odhadne pozice šablony. Získají se souřadnice čtyř rohů, které jsou umístěné libovolně v obrázku nebo i mimo něj a definují čtyřúhelník obsahující hledanou šablonu. [19]

Detekce objektu ve fotografii tímto způsobem má i nevýhody. Pro obrázky obsahující málo výrazných features nemusí být tato metoda úspěšná a může vracet horší výsledky než jiné zmíněné metody. Problémem také může být předpoklad, že se v obrázku nachází pouze jeden hledaný objekt. Pokud by jich tam bylo více, budou se podobné features opakovat, což může vést k rozložení napárování se šablonou pro více nalezených objektů a následně ke špatnému nalezení homografie.

4.1.4 Použití a vlastní vylepšení základní metody

Pro tento projekt jsem se po důkladném zvážení všech výhod a nevýhod jednotlivých způsobů detekce dokladu v obrázku rozhodl použít metodu template-matching pomocí párování features v šabloně a ve zkoumané fotografii. Z reálných fotografií dokladů jsem si vytvořil několik anonymizovaných šablon, aby bylo možné používat software komerčně a neobsahoval ničí osobní údaje. Pomocí těchto šablon pak hledám ve fotografii jednotlivé doklady.

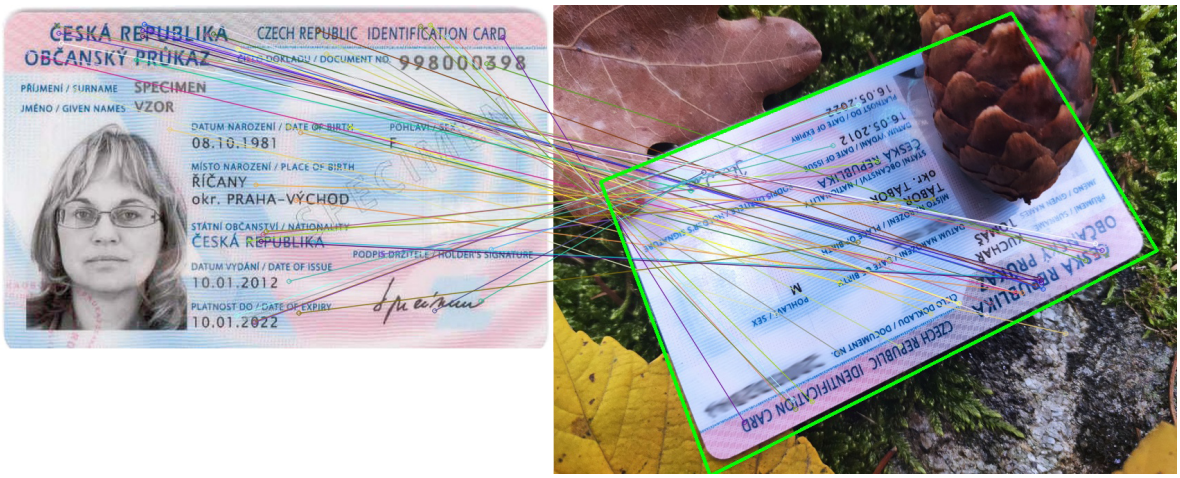
Vyzkoušel jsem pro podporované typy dokladů všechny zmíněné detektory features a dokonce ještě některé další méně používané. Největší úspěšnost pro tento problém měla metoda SIFT, kterou jsem nakonec použil ve výsledném

řešení. Sice je lehce pomalejší než ostatní metody, nicméně dává nejlepší výsledky a v porovnání s ostatními prováděnými operacemi v celém procesu je rozdíl doby výpočtu SIFT oproti jiným detektorům zanedbatelný.

Pro zvýšení přesnosti jsem ze šablony zkoušel oříznout více proměnlivé části, aby se díky nim špatně nepárovaly features a nevedlo to kvůli tomu k nepřesnějším výsledkům. Tato myšlenka ovšem nepomohla a nejlepší výsledky dávalo i přes rozdílné texty v dokladech použití šablony s celým dokladem, pouze s ořízntutým pozadím.

K dalšímu drobnému zlepšení výsledků pomohlo i opakované hledání dokladu podle různých šablon. Použití jedné univerzální šablony pro všechny fotografie nedává vždy nejpřesnější ohraničení hledaného dokladu, na každou fotografii se hodí trochu jiná. Proto jsem vytvořil více vzorových obrázků a při špatném nalezení pomocí jedné šablony se postup opakuje s jinou šablonou.

Řešení také komplikuje možnost uživatelů nahrát celkem až dvě fotografie, protože u oboustranných dokladů nevíme, v jaké fotografii se nachází jaká strana. Obě strany ale mohou být nahrány i v jedné fotografii. Z toho vyplývá, že řešení musí být vždy schopné nalézt obě strany v jedné fotografii. V případě, kdy jsou dodány dva obrázky, se tyto obrázky nejprve spojí do jednoho a následně se pokračuje v obou případech stejně a hledá se přední strana dokladu ve spojeném obrázku. Nalézt zadní stranu je trochu složitější, protože neobsahuje tolik markantů jako strana přední. Pokud byly nahrány dvě fotografie a známe polohu přední strany ve spojeném obrázku, lze předpokládat, že zadní strana dokladu bude spíše na druhé fotografii, a mohu se při hledání zadní části dokladu zaměřit spíše na tuto oblast. Tento postup po otestování také lehce pomohl zvýšit celkovou úspěšnost.



Obrázek 4.2: Ukázka detekce dokladu ve fotografii pomocí features.

4.2 Zpracování nalezeného dokladu

Předchozí popsané kroky vedou k získání souřadnic čtyř rohů nalezeného dokladu ve fotografii. Důležité je zdůraznit, že díky zvolené metodě víme, který nalezený roh je který. V tomto kroku tedy je potřeba narovnat a oříznout nalezený doklad do nového obrázku o pevně daných rozměrech a co nejlépe ho připravit pro následné čtení textových údajů.

4.2.1 Perspektivní transformace

V původní fotografii může být doklad vyfocený z úhlu nebo různě orotovaný. Toto se stává v nějaké míře téměř pokaždé. Při znalosti souřadnic 4 rohů dokladu ve fotografii a výsledných rozměrů, které chceme dostat, je perspektivní transformace pouze matematický přepočet pozic jednotlivých obrazových bodů. Tuto základní funkci nabízí spousta grafických knihoven. Prvním krokem je vypočítat transformační matici, s pomocí které se následně obrázek transformuje.

Důležité je také brát v potaz, že nemusí jít pouze o otočení obrázku, ale i o změnu celkové velikosti, protože do dalšího kroku potřebujeme obrázek s přesně danou šířkou a výškou. Díky tomu mohou vznikat nějaké ztráty v obrazových bodech (při zmenšování obrázku) nebo naopak může nastat potřeba vyplnit chybějící pixely vhodnými barvami (při zvětšování obrázku). Grafické knihovny pro geometrické transformace nabízí několik možností interpolačních algoritmů, které ovlivňují výslednou kvalitu obrázku po provedené transformaci a zároveň se liší v délce trvání výpočtu.

Obrázek 4.3: Ukázka použití perspektivní transformace pro srovnání nalezeného dokladu.

4.2.2 Obrazové filtry

V této fázi je obrázek dokladu narovnaný do obdélníku o předem daných rozměrech a v dalším kroku bude následovat čtení textových dat z obrázku. Mezi tím se hodí pomoci čtecí knihovně použitím obrazových filtrů, které zvýrazní černý text a odstraní ostatní šum, který by mohly čtecí funkce mylně interpretovat jako textové znaky, snažit se je přečíst a vracet kvůli tomu nepřesné výsledky.

Samozřejmě dává smysl tyto filtry vyzkoušet až ve spolupráci s knihovnou pro čtení textu z obrázku, jelikož okem příliš nepoznáme, co této počítačové knihovně pomůže dělat méně chyb ve čtení. Takto jsem ve skutečnosti postupoval při vytváření celkového řešení, ovšem rozepíšu se o tom zde pro zachování hierarchie jednotlivých kroků, které na sebe v celém procesu navazují. Po důkladném testování různých intenzit různých obrazových filtrů jsem dostal nejlepší výsledky při kombinaci třech filtrů.

Prvním z nich je Gaussovské rozostření. Jde o jednu z nejčastějších forem rozmazání obrázku. Algoritmus pro každý jednotlivý pixel vypočítá vážený průměr barev z okolních pixelů v zadaném poloměru. Přitom pixelům nejblíže k centru dává nejvyšší váhu a s postupným vzdalováním od středu se váha snižuje. Díky tomu při použití filtru nepřijdeme kompletně o informaci v jednotlivých pixelech a rozostření je přirozené. Hodnoty vah se řídí Gaussovým (normálním) rozdělením.

Příklad Gaussovského rozostření o průměru pěti obrazových bodů lze reprezentovat maticí o rozměrech 5×5 s hodnotami:

$$\frac{1}{256} \cdot \begin{pmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{pmatrix}$$

Důležitý je zlomek před maticí, který zajistí, že se všechny hodnoty matice nasčítají na jedničku a filtr tak nezpůsobí zvýšení intenzity obrázku, pouze zprůměruje barvy pixelů. V jednotlivých složkách matice je také názorně vidět poměr vah jednotlivých obrazových bodů v okolí.

Pro barevné obrázky se filtr používá zvlášť pro každou barevnou složku, v případě černobílých obrázků se použije pouze jednou, princip filtru se tedy nemění. V různých implementacích se však můžou lišit drobnosti jako zpracování hraničních pixelů obrázku, kde se chybějící pixely buď dopočítají podle nejbližších sousedů a nebo se použije hodnota středově souměrného pixelu. Filtr může mít i směrové varianty, které dávají větší váhu obrazovým bodům v určitém směru. [22]

Rozostření obrázku však čtení textu nepomůže. Fotografii je naopak potřeba zostřit. Pro docílení zostření obrázku a vytažení hran se používá neintuitivní postup, kdy pro každý pixel spočítáme $(1 + n)$ násobek jeho hodnoty

v původním obrázku a od výsledku odečteme n násobek hodnoty pixelu na stejné pozici v rozostřeném obrázku. Tímto postupem lze docílit zaostření textu ve fotografii a lepší výsledky strojového čtení.

Tento postup kromě vytažení požadovaných hran zvýrazní i případný šum v obrázku. Pro jeho odstranění lze použít bilaterální filtrování. Výhoda tohoto neiterativního algoritmu je, že kromě efektivního odstranění šumu zachovává ostrost hran. Základním principem je porovnávání barev pixelů v okolí a průměrování hodnot podobných barev. Vznikají tak v obrázku stejnobarevné zóny, které si zachovávají ostré hrany. [23]

Je potřeba správně odhadnout potřebnou intenzitu filtru a nastavit vhodné parametry. Při malé intenzitě nebude mít filtr dostatečný efekt, při vysoké intenzitě začne kromě drobného šumu vyhlazovat i čtený text. Se znalostí struktury a vždy stejných rozměrů obrázku srovnaného dokladu lze tyto parametry nastavit jednotně pro libovolný uživatelský vstup a filtr tak vždy bude dávat požadovaný efekt.

4.3 Vytěžení textu

Dalším logicky navazujícím krokem v celém procesu výsledného softwaru je vyčtení informací z dokladu do textové podoby. Po předchozích operacích je k dispozici oříznutá srovnaná fotografie průkazu, která má fixní rozměry a je co nejlépe připravená pro čtení textu.

Úlohou strojového čtení textu z digitálních obrázků se lidé zabývají již od osmdesátých let minulého století. Od té doby se metody postupně vyvíjely a zdokonalovaly a vzniklo několik programů od různých výrobců, které v současné době tento problém řeší s velmi použitelnými výsledky.

Zpočátku se řešení zaměřovala zejména na čtení textů z naskenovaných snímků dokumentů. Používaly se různé techniky pro odhalení řádku textu v obrázku. Nalezený řádek se následně podle mezer rozdělil na jednotlivá slova a tyto slova se potom rozdělila na samostatná písmena. Jednotlivé znaky se pak porovnávaly s naučenou databází znaků. Některá řešení navíc používají slovníky s databází existujících slov v různých jazycích, pomocí které se snaží vylepšit výsledky čtení.

Postupem času se OCR (Optical Character Recognition) technologie začaly využívat pro odhalování textu v reálné scéně a tomu se začal přizpůsobovat i dostupný software, který už není omezený primárně na dokumenty, ale zvládá hledat text i ve složitějších obrázcích. V praxi se tak už běžně setkáváme například se strojovým čtením poznávacích značek automobilů nebo s autonomními vozy, které hledají text na dopravních značkách. [24]

4.3.1 Dostupné knihovny

Pro možnost používat OCR funkce pro čtení textu z fotografií v této práci je potřeba vybrat správnou knihovnu. Dostupných je spousta možností, které

se liší nejen úspěšností. Některé knihovny jsou jednoduché na použití a nabízí přímo funkce pro čtení, u některých je potřeba pro zprovoznění trocha programování, některé vyžadují hlubší znalosti a hodně programování navíc. Jsou dostupná drahá komerční řešení, ale i bezplatné open-source knihovny.

Prvním příkladem zdarma dostupné knihovny je Calamari. Je založená na bezplatné platformě s otevřeným zdrojovým kódem TensorFlow, která nabízí svou kapacitu neuronových sítí a strojového učení. Knihovna je dostupná pro jazyk Python. Calamari je přímočará na použití, bohužel ale nenabízí kompletní řešení a pro různé úpravy obrázku jako zvýšení kontrastu, odstranění kurzívy či segmentaci obrazu je potřeba knihovnu zkombinovat s dalším nástrojem. Výrobci pro tyto operace doporučují OCRopus.

OCRopus je kolekce několika nástrojů pro analýzu dokumentů, která zahrnuje i funkce strojového čtení textu. Byla napsána taktéž pro jazyk Python a nabízí výstup čtení ve formátu prostého textu, PDF nebo ve formátu hOCR, což je otevřený standard reprezentující výsledky OCR, který kromě samotného textu obsahuje informace o umístění jednotlivých slov a řádků v obrázku a důvěryhodnosti jejich přečtení. OCRopus není příliš univerzální řešení, má problémy s obrázky s nízkým rozlišením a s textem, který není úplně v rovině. Ani jedno z těchto omezení by však v tomto projektu díky předzpracování fotografií nemělo být problém. [24]

Zástupcem komerčního softwaru je například ABBYY FineReader, který nabízí jak lokální použití na Linux, Windows i Mac, tak i online rozhraní. V nejnovější verzi podporuje až 210 světových jazyků, zhruba u třetiny z nich má k dispozici i slovník s databází existujících slov. Knihovna je dostupná pro jazyky C a C++, dále pak výrobce nabízí wrapper pro jazyk Java. Podporuje desítky vstupních obrázkových a kancelářských formátů. Výstup pak kromě prostého textu dokáže uložit v širokém množství textových formátů nebo například i ve formátech Microsoft Office. [25] Již v roce 2015 měl software celosvětově více než 20 milionů uživatelů a na začátku loňského roku byl dokonce vyhlášen nejkvalitnějším OCR nástrojem na trhu. [26]

Dalším komerčním programem je OmniPage – jeden z nejstarších nástrojů pro extrakci textu z obrázků, který se začal používat již v osmdesátých letech na osobních počítačích. V současné době se dodává jako kompletní software v několika verzích pro lokální instalaci na všechny nejpoužívanější operační systémy, nabízí i poměrně drahé (stovky tisíc českých korun) SDK pro napojení na ostatní aplikace nebo serverové rozhraní. Stejně jako u předchozího FineReaderu se výrobce chlubí nejvyšší úspěšností na trhu. Podporuje více než 125 světových jazyků včetně různých typů písem, zvládne například i arabštinu, hebrejštinu, čínštinu, vietnamštinu, japonštinu nebo korejštinu. Samozřejmostí je i podpora spousty vstupních grafických formátů a výstupních textových formátů. Zajímavým způsobem výstupu většiny zkoumaných komerčních softwarů je převedení původního obrázku do formátu obrazového PDF, kde původní text v obrázku je překryt přečteným textem a lze ho tak v PDF označovat. [27]

Knihovna	Vznik	Jazyk	Online	Linux	Win	Mac
Tesseract	1985	C, C++, wrappery pro dalších více než 10 jazyků	Ne	Ano	Ano	Ano
OmniPage	1988	C, C++, C#	Ano	Ano	Ano	Ano
ABBYY FineReader	1993	C, C++, Java	Ano	Ano	Ano	Ano
CuneiForm	1993	C, C++	Ne	Ano	Ano	Ano
Asprise OCR	1998	C, C++, C#, Java, Python, VB.NET, Delphi	Ano	Ano	Ano	Ano
GOCR	2000	C, C++	Ano	Ano	Ano	Ano
Dynamsoft OCR	2003	C, C++	Ano	Ne	Ano	Ne
Ocrad	2003	C++	Ano	Ano	Ne	Ano
OCRopus	2007	Python	Ne	Ano	Ne	Ano
Google Cloud Vision	2015	C++, Java, Python, Go, Ruby	Ano	Ne	Ne	Ne
Calamari	2018	Python	Ne	Ano	Ano	Ano

Tabulka 4.1: Porovnání různých OCR knihoven poskytujících SDK [24][28].

Asprise OCR je na trhu také již více než dvacet let. Na rozdíl od ostatních svých komerčních kolegů nabízí daleko širší paletu programovacích jazyků, pro které poskytuje své SDK. Podporované jsou například C, C++, C#, Java, Python, VB.NET nebo Delphi. Hlavní výhodou oproti ostatním produktům je také rychlost čtení, kterou lze pomocí parametrů přizpůsobit a například pro projekty v reálném čase ještě zvýšit na úkor kvality. Nevýhodou je nižší množství podporovaných jazyků textu a vstupních i výstupních formátů. [29]

Z bezplatných projektů poskytujících knihovny pro napojení do vlastní aplikace si ještě zaslouží zmínku GOCR a Ocrad pro jazyk C/C++, nebo CuneiForm, který se stal open-source projektem až v roce 2008, po patnácti letech s uzavřenou placenou licencí. [28]

Dále existují také online služby poskytující OCR funkce přes webové rozhraní. Hlavním zástupcem je Google Cloud Vision, který nabízí nejen OCR služby, ale pomocí strojového učení a neuronových sítí i vyhledávání růz-

ných objektů v obrázcích. Nabízí bezplatné vyzkoušení dobře zdokumentované služby až na tisíci obrázků. Za každou další tisícovku požadavků si poskytovatel účtuje 1,5 dolaru. Za tuto cenu však v porovnání s konkurencí nabízí kvalitní výsledky. [24]

4.3.2 Tesseract

Vlastní podkapitolu si zaslouží knihovna Tesseract, kterou jsem záměrně nechal na konec a popíši ji podrobněji, jelikož jsem si pro tuto práci z několika důvodů zvolil právě ji.

Tesseract je bezplatný open-source software nabízející OCR engine s názvem `libtesseract` zahrnující knihovnu s API pro jazyky C a C++ a dále program `tesseract`, který umožňuje snadné použití enginu z příkazové řádky. Pro použití knihovny v jiných programovacích jazycích vznikla spousta wrapperů od různých autorů. Nejnovější verzi Tesseractu lze díky nim navíc použít v jazycích Java, Python, Objective-C, Swift, Flutter, R, Ruby, Elixir a Crystal. Starší verze podporují i .Net, Nodej.js, PHP, Go nebo Clojure. Zmíněné wrappery však podle vlastních zkušeností většinou nepodporují všechny funkce a možnosti knihovny, mají problémy se správou paměti nebo jsou výrazně pomalejší oproti použití v C++.

Tesseract vyrobila v roce 1985 firma Hewlett-Packard a stal se tehdy historicky jedním z prvních nástrojů pro čtení textu z obrázků. V roce 1996 došlo k různým změnám umožňujícím použití na systémech Windows a o dva roky později se kód modernizoval do novější verze C++. Následující dekádu nedocházelo k mnoha změnám a v roce 2005 autoři vydali software jako open-source s Apache 2.0 licencí. Od roku 2006 je vývoj Tesseractu sponzorován firmou Google. Knihovna je velmi dobře zdokumentovaná.

Do jeho třetí verze Tesseract používal pro čtení textu techniky vyhledávání šablon znaků v obrázku. K zásadní změně došlo v říjnu 2018, kdy s uvedením čtvrté verze přišla možnost využití nového způsobu čtení pomocí neuronových sítí. S použitím LSTM (Long Short-Term Memory) neuronových sítí nabízí nová verze vyšší přesnost čtení, integrovanou podporu více než 100 světových jazyků včetně češtiny a možnost vytrénovat síť pro libovolný další jazyk. V jednom obrázku může být zahrnuto i více jazyků. Na rozdíl od předchozí metody je tato více zaměřená na rozpoznávání celých řádků textu. Obě metody detekce textu je dokonce možné zkombinovat a nechat volbu lepší metody pro konkrétní obrázek na Tesseractu.

Pro podporu čtení jednotlivých jazyků je potřeba knihovně stáhnout předpřipravené balíky s natrénovanými daty pro konkrétní jazyk. Tyto soubory se nazývají "traineddata"a Tesseract oficiálně nabízí tyto možnosti:

- `tessdata` – základní balík natrénovaných dat
- `tessdata_best` – balík natrénovaných dat s vyšší přesností čtení na úkor rychlosti

- `tessdata_fast` – balík natrénovaných dat pro rychlé čtení textu s nižší přesností

- `tessdata_contrib` – vlastní balíky natrénovaných dat od ostatních uživatelů, nejsou spravovány vývojáři knihovny

Pro konkrétní jazyk a použití nemusí vždy platit uvedený popis, je tedy potřeba vyzkoušet, který balík bude více vyhovovat. Při mém testování pro doklady v českém jazyce se nejvíce osvědčil základní balík.

Tesseract podporuje jako vstup základní grafické formáty zahrnující BMP, PNG, JPEG, TIFF, JFIF a PNM. Ostatní formáty musí být překonvertovány na jeden z těchto. Výstup nabízí ve formě prostého textu, hOCR, v různých použitích formátu PDF a několika dalších méně známých formátech.

Nevýhodou knihovny je horší kvalita čtení pro originální obrázky, které nejsou dopředu zpracované jiným nástrojem. Výrobci v dokumentaci uvádí několik tipů pro zlepšení kvality čtení. Tesseract sice vnitřně používá grafickou knihovnu Leptonica pro různé operace s obrázkem před samotným čtením, ovšem tyto operace v některých případech nejsou dostatečné pro dobré výsledky.

Knihovna si nedokáže poradit se světlým textem na tmavém pozadí, takovéto obrázky je potřeba nejprve invertovat. Špatně si také vede s obrázky, kde text není vodorovně a je otočený o nějaký úhel. Horší úspěšnost může také způsobit rozměr obrázku – pro nízké rozlišení je text hůře čitelný, pro vysoké rozlišení naopak v obrázku může existovat šum a být složitější správně nalézt text. Pro odhadnutí ideální velikosti písma v obrázku byly provedeny různé testy. Výsledkem bylo, že nejpřesněji se podařilo vyčíst údaje pro text vysoký okolo 32 pixelů. I tento rozměr beru ve výsledném řešení v potaz.

Před čtením textu nejprve Tesseract binarizuje obrázek pomocí Otsuova algoritmu. To znamená, že převede jednotlivé pixely buď na bílé nebo černé. Při podobném odstínu textu a pozadí může nastat problém, že algoritmus nerozezná rozdíl v odstínech a text nebude viditelný. Pomoci může vlastní binarizace obrázku před zpracováním Tesseractem pomocí jiných parametrů nebo jinou metodou.

Pomoci lepší úspěšnosti může i správná tloušťka písma. Pokud je text příliš tučný nebo naopak úzký, nemusí knihovna dávat tak dobré výsledky. Při předem známé tloušťce textu lze použít grafické funkce, které tento problém dokážou zlepšit. Dále samozřejmě pomáhá odstranění šumu nebo odstranění různých rámečků a čar, které nepatří k samotnému textu.

Před čtením textu musí Tesseract nejprve správně detekovat, zda se v obrázku nachází celý odstavec řádků textu, jeden řádek, jedno slovo nebo například pouze jeden znak. Následně obrázek rozdělí na menší části, které potom čte. Tomuto procesu se říká segmentace obrazu. Pro zjednodušení a lepší výsledky je možné napovědět knihovně, jakou strukturu má v obrázku očekávat. Zkoumané doklady mají vždy stejnou strukturu, je tedy vhodné tuto vlast-

nost využít a získat tak rychleji přesnější data. Knihovna nabízí zvolit tyto možnosti segmentace a zpracování obrázku:

- `PSM_OSD_ONLY` – pouze detekce orientace a typu skriptu (OSD)
- `PSM_AUTO_OSD` – automatická segmentace stránky s použitím OSD
- `PSM_AUTO_ONLY` – automatická segmentace stránky bez OSD a OCR
- `PSM_AUTO` – plně automatická segmentace stránky bez OSD
- `PSM_SINGLE_COLUMN` – předpokládá jeden sloupec textu s rozdílnými šířkami
- `PSM_SINGLE_BLOCK_VERT_TEXT` – předpokládá jeden uniformní blok vertikálně zarovnaného textu
- `PSM_SINGLE_BLOCK` – předpokládá jeden uniformní blok textu
- `PSM_SINGLE_LINE` – bere obrázek jako jeden řádek textu
- `PSM_SINGLE_WORD` – bere obrázek jako jedno slovo
- `PSM_CIRCLE_WORD` – bere obrázek jako jedno slovo v kroužku
- `PSM_SINGLE_CHAR` – bere obrázek jako jeden znak
- `PSM_SPARSE_TEXT` – snaží se najít v obrázku co nejvíc možného textu
- `PSM_SPARSE_TEXT_OSD` – snaží se najít v obrázku co nejvíc možného textu s použitím OSD
- `PSM_RAW_LINE` – bere obrázek jako jeden řádek textu, vynechává úpravy specifické pro knihovnu

Pro tuto práci jsem zvolil knihovnu Tesseract z důvodu, že je zdarma k použití, z ostatních bezplatných řešení je nejrozšířenější a poskytuje jedny z nejlepších výsledků. Nabízí několik možností konfigurace čtení, jednoduchou integraci a spoustu návodů a příkladů dostupných na internetu. [30]

4.3.3 Použití v projektu

Výhodou této práce je, že každý typ zkoumaného dokladu má vždy pevně danou strukturu přední i zadní strany. Díky tomu mohu určit přesnou pozici jednotlivých údajů na ploše dokladu a díky fixním rozměrům zpracovaného obrázku mohu jednoduše vyříznout jednotlivé obdélníkové oblasti pro každý zkoumaný údaj. Tyto podobrázky následně po jednom předám knihovně Tesseract, která z nich podle zvolených parametrů vytěží textová data.

Pokoušet se přečíst najednou celou stranu dokladu by nevedlo k dobrým výsledkům, protože v textovém výstupu by údaje mohly být špatně seřazené, zbytek fotografie obsahuje různý šum a navíc je velká část obsaženého textu pouze popis jednotlivých polí a nadpisy dokladu.

Některé údaje na dokladu mají vždy stejnou délku, například datumy, číslo dokladu nebo rodné číslo. To umožňuje poměrně přesně oříznout oblast, kde se údaj nachází. Některé pole ale takovou výhodu nemají a není dopředu známé, kolik budou obsahovat znaků. Příkladem je jméno, příjmení, místo narození, adresa trvalého bydliště nebo vydávající autorita. Údaj může být složen z jednotek znaků, ale i z více než dvaceti. Proto musí vyříznutá oblast být přes celou plochu, kde by se mohl objevit text. Pokud ale v takovéto vyříznuté ploše je obsažen jen krátký text, Tesseract má sklony hledat další text, který neexistuje, a občas vrátí za správně vyčtenými údaji další neexistující znaky navíc.

Našel jsem způsob, jak tomuhle problému částečně předejít. Místo oříznutí celé velké plochy vyříznu v prvním kroku pouze menší část a přečtu obsažený text. Pokud je text dostatečně dlouhý, mohlo se stát, že část textu je ještě za hranicemi ořezu, proto obrázek oříznu znovu na celou dostupnou plochu a čtení opakuji. Pokud při prvním čtení byl text krátký, čtení neopakuji a použiji vyčtený text, který není tak náchylný na chyby jako při čtení většího regionu. Tento postup se při testování skutečně osvědčil jako méně náchylný na chyby.

Kromě čtení samotných údajů, které budou výstupem služby, používám Tesseract ještě k jedné věci. Po perspektivní transformaci nalezeného dokladu ve fotografii se pokusím přečíst některé statické texty, které jsou stejné na každém dokladu, například nadpisy dokladu nebo názvy jednotlivých polí. Výsledky čtení následně porovnám se skutečnými údaji, které by měly být na

Obrázek 4.4: Vyznačení hranic čtených oblastí ve zpracované fotografii občanského průkazu. Vlevo jsou vyznačeny oblasti obsahující požadovaná data, vpravo jsou vyznačeny kontrolní oblasti.

daných místech dokladu uvedeny. Tato informace umožňuje částečně zkontrolovat, zda se doklad správně nalezl v obrázku v prvním kroku. Pokud se přečtený text z velké části neshoduje, opakuji hledání dokladu ve fotografii pomocí jiné šablony a jiných parametrů.

4.4 Korekce textů

Po vyčtení textů ze všech požadovaných polí z obou stran dokladu (případně pouze z přední strany, pokud zadní není) následuje korekce těchto údajů. Tento krok je potřebný, protože výstup OCR knihovny pro méně kvalitní fotografie často není stoprocentní a občas obsahuje chyby, které počítač dokáže odhalit a automaticky opravit. Některé údaje lze i validovat, zda odpovídají hodnotám a formátu, který mohou mít. V případě odhalení neopravitelných chyb používám přístup, že raději údaj nepoužiji vůbec, než abych vědomě dopustil, aby služba vrátila nesprávné výsledky.

4.4.1 Úpravy jednotlivých polí

Prvním krokem při korekci je projít každé pole zvlášť a samostatně ho zkontrolovat. U všech polí na všech podporovaných dokladech platí nějaká více či méně striktní pravidla pro text, který mohou obsahovat.

Základním omezením každého pole je množina přípustných znaků. Tesseract podporuje znakovou sadu UTF-8 a často se stává, že některý znak špatně vyhodnotí jako jiný jemu podobný a nebo kvůli šumu a nečistotám ve fotografii vrátí ve výstupu navíc znaky, které v textu ve skutečnosti nejsou. Tyto znaky často bývají pomlčky, tečky nebo jiné speciální znaky, které se běžně nevyskytují v údajích jako je jméno a příjmení a mohou se díky tomu z přečteného textu odstranit. Po přečtení každého údaje projdu celý text a vyfiltruji pouze ty znaky, které vím, že se v údaji mohou vyskytovat.

Jméno a příjmení mohou být složeny pouze z alfabetických znaků (písmen), případně mezery při více jménech. Pro všechny aktuálně vydávané doklady jsou tyto hodnoty navíc psány velkými písmeny.

Číslo občanského průkazu se skládá z devíti číslic, číslo cestovního pasu z osmi číslic. Číslo řidičského průkazu se až na výjimky skládá ze dvou písmen, mezery a šesti číslic.

Všechna pole obsahující datumy mají na podporovaných typech dokladů stejný formát. Den, měsíc a rok jsou v tomto pořadí odděleny tečkami a v případě jednočíslicových hodnot zleva doplněny nulou. V dokladech se používají pro datum narození a počátek a konec platnosti průkazu.

Místo narození a vydávající autorita jsou složeny převážně z alfanumerických znaků. Číslice se používají například pro definování městské části v Praze. Dále údaje mohou obsahovat tečku ve zkratkách dlouhých názvů a pomlčku v některých názvech měst (například "Frýdek-Místek").

Pohlaví se skládá pouze ze znaků "M" nebo "F", rodné číslo pouze z numerických znaků a lomítka, titul pouze z písmen a teček.

Ještě před vymazáním špatně přečtených znaků, které se v konkrétních údajích nevyskytují, je potřeba dát pozor, zda si OCR knihovna pouze nespletla nějaký znak s jiným podobným. Proto nejdřív opravím znaky, které by při pouhém odstranění následně chyběly ve výsledném textu. Často se pletou číslice s podobnými písmeny, například písmeno 'O' a nula, velké písmeno 'I' a jednička, písmeno 'Z' a dvojka, písmeno 'S' a pětka nebo velké písmeno 'B' a osmička. Některá písmena mají i podobnou podobu v malé a velké verzi a liší se pouze poměrem velikosti k ostatnímu textu. Tohle všechno je nejdřív potřeba opravit.

Některé OCR knihovny nabízí před čtením textu z obrázku možnost omezit množinu čtených znaků nastavením jejich seznamu ve whitelistu nebo zakázat čtení nějakých znaků v blacklistu. Tuto funkci podporuje i Tesseract a skvěle by se hodila právě pro čtení dat z dokladů. Funkci jsem zkoušel různě použít pro většinu čtených polí, ovšem při omezení znaků začala knihovna dávat horší výsledky než při výchozích parametrech. Největší úspěšnost měl postup přečtení libovolných znaků a následná vlastní korekce. Snad se tato funkce v novějších verzích knihovny zlepší.

Popsaný postup odstranění špatně přečtených znaků není kompletní řešení. Občas nastane případ, kdy OCR knihovna přečte chybně navíc znaky z množiny povolených znaků. Tyto přebytečné nesprávné znaky lze někdy odhalit a také promazat. Názorným příkladem může být reálný výstup čtení jména v nekvalitní fotografii: "L KAREL ooooo". Osoba může mít více jmen, ovšem lze předpokládat, že jméno nemůže být tvořeno jedním znakem, a tyto slova odstranit. Pokud je slovo tvořeno pouze dlouhou kombinací určitých samohlásek nebo souhlásek, případně pouze několika stejnými znaky, lze také předpokládat, že se jedná o chybu čtení a slovo vymazat.

Tuto korekci je potřeba provádět jen s velkou jistotou pouze na některých údajích. Je potřeba dát pozor na jednopísmenné předložky v adresách a názvech měst, cizí jména, římské číslice v některých adresách a další okrajové případy. V případě nejistoty je lepší nechat originálně přečtený text, případně zopakovat čtení trochu jiným způsobem.

4.4.2 Oprava známých slov

Technologie OCR si často pomáhají k přesnějším výsledkům předpokladem, že pro daný jazyk čtený text obsahuje existující slova, která mají uložená ve svém slovníku. I Tesseract podle dostupných informací používá slovníky, ovšem nepodařilo se mi zjistit, do jaké míry podle nich opravuje výstup a jak moc obsáhlé jsou pro jednotlivé podporované jazyky. Při čtení se tak stávalo, že v některých slovech byl občas špatně nějaký znak. Pro dlouhá unikátní slova lze tuto chybu automaticky opravit podle vlastního slovníku.

Při hledání chyb ve slovech je potřeba brát v potaz všechny tři typy chyb, které mohou nastat. Buď může v přečteném textu chybět znak, přebývat znak a nebo může být znak špatně přečtený. Při kontrolování znak po znaku by vynechání nebo přidání písmene znamenalo, že zbytek slova bude špatně. Proto je vhodné měřit odchylku od slovníkových slov pomocí Levenshteinova algoritmu pro měření editační vzdálenosti, jehož výstupem je minimální počet operací odstranění, přidání nebo nahrazení znaku potřebných k úpravě přečteného řetězce na slovo ze slovníku.

Pro podporované doklady používám automatické opravy slov pro dlouhé názvy měst. Variantu pro více slov pak používám pro víceslovná města, kde podle délky textového řetězce a podoby s jiným existujícím městem připouštím různě velkou chybu.

Některá pole obsahují často používaná slova, která se tímto způsobem také nechají opravit. Například v místě narození uvedeném v občanském průkazu často druhý řádek začíná slovem "okr." následovaným názvem okresního města. Při drobné chybě lze tento údaj zpřesnit. Políčka s vydávajícím úřadem často obsahují podřetězce "MěÚ", "ÚMČ" nebo "Magistrát města", jejichž občasná oprava dokáže celkovou statistiku úspěšnosti také o nějaké procento zlepšit.

Dále se v dokladech vyskytují pole, která mohou obsahovat jen několik výčtových hodnot. Například pro pohlaví se na průkazy naštěstí zatím uvádí pouze dvě oficiální hodnoty, takže stačí hledat do té doby, než knihovna přečte písmeno 'M' nebo 'F'. Rodinný stav muže v době vzniku této práce dokáže nabývat pouze hodnot "ženatý", "svobodný", "rozvedený", "vdovec" nebo "partnerství", rodinný stav ženy nabývá hodnot "vdaná", "svobodná", "rozvedená", "vdova" nebo "partnerství". Výčet různých akademických titulů nebo skupin vozidel v řidičském průkazu je také limitován. Se znalostí seznamu těchto hodnot lze opět opravit drobné chyby ve strojovém čtení.

4.4.3 Rodné číslo

Rodné číslo obsahují všechny typy podporovaných osobních dokladů. Pravidla pro jeho podobu jsou celkem komplikovaná a díky tomu lze ve většině případů odhalit chybu při čtení. Údaj se skládá pouze z číslic a jednoho lomítka. Prvním krokem tedy je z výstupu OCR knihovny převést všechna písmena podobná číslicím na tyto číslice a odstranit všechny znaky kromě čísel a lomítka. Následuje kontrola a případná oprava jednotlivých znaků ve zbytku textu.

Prvních šest znaků v českém rodném čísle jsou číslice reprezentující datum narození osoby. První a druhá číslice jsou jasně určeny posledními dvěma číslicemi roku narození. Další dvě číslice zahrnují informaci o měsíci narození. Pro zahrnutí odlišení pohlaví do rodného čísla se navíc ženám k této hodnotě přičítá 50. Od roku 2004 (zákonem č. 53/2004 Sb.) je navíc možnost k této hodnotě přičíst ještě 20 v případě, že se daný den narodí hodně dětí stejného pohlaví a počet dostupných rodných čísel nebude dostačovat. Třetí a čtvrtá

číslice tedy mohou nabývat hodnot 01–12, 21–32, 51–62 a 71–82. Pátá a šestá číslice obsahují pořadový den v měsíci narození. Povolené hodnoty jsou tedy 01–31, kde nejvyšší hodnoty musí souhlasit s měsícem, který může mít tolik dní. Sedmý znak rodného čísla je lomítko. Za lomítkem jsou tři znaky, které udávají pořadí narození dítěte v konkrétní den. Od začátku roku 1954 přibyla za lomítkem čtvrtá číslice, sloužící jako kontrolní. Vznikne jako zbytek po dělení devítimístného rodného čísla číslem jedenáct. Problém nastává, pokud je zbytek po dělení roven deseti. V takovém případě se jako kontrolní číslice použije nula. Tato výjimka byla použita zhruba u tisícovky rodných čísel a přidělování takových rodných čísel bylo roku 1985 podle interního předpisu Federálního statistického úřadu ukončeno. [31]

Při chybě ve čtení nějakého znaku lze díky všem uvedeným pravidlům většinou detekovat, že rodné číslo není platné, a případně opakovat pokus o vytěžení textu z obrázku.

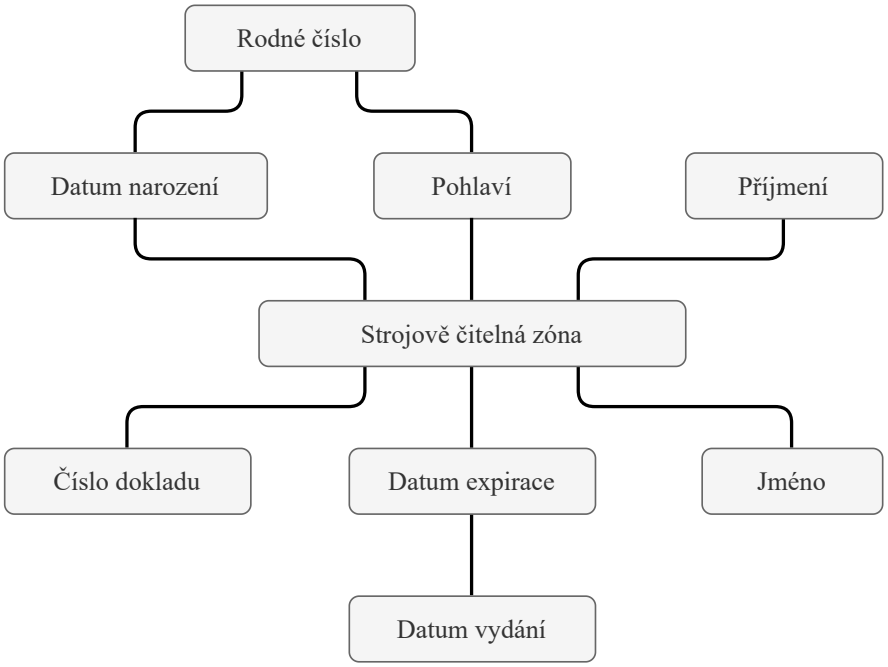
4.4.4 Kontrola vzájemně souvisejících údajů

Některé údaje na průkazech spolu částečně souvisí nebo jsou uvedeny duplicitně na více místech a díky tomu je možné zkontrolovat správné strojové přečtení duplicitních údajů porovnáním přečtených hodnot. V případě neshody údajů je někdy lze opravit a nebo se pokusit opakovaně zavolat OCR funkce s jinými parametry nebo upraveným obrázkem.

Již uvedeným příkladem může být rodné číslo, pomocí kterého jde ověřit datum narození nebo pohlaví. Pokud tyto informace nebyly správně přečteny a u rodného čísla sedí formát a kontrolní číslice, je možné doplnit hodnotu data narození a pohlaví. Pokud je naopak zjištěna chyba v rodném čísle a první šestice číslic nesedí s datem narození a pohlavím, je možné rodné číslo z těchto údajů opravit a znovu ověřit kontrolní číslici.

Datumy vydání průkazu a expirace jsou spolu také provázána. Den a měsíc bývají ve velké většině stejné hodnoty a roky jsou od sebe u všech zkoumaných dokladů v násobcích pěti. Konkrétní rozdíl lze zjistit i přesně podle typu dokladu a věku držitele.

Občanský průkaz a cestovní pas obsahují oproti řidičáku navíc strojově čitelnou zónu, kde jsou opět zduplikované některé hodnoty. Tyto řádky mají pevně danou strukturu a pravidla a obsahují několik kontrolních znaků, podle kterých lze detekovat úspěšnost čtení. Kontrolní znaky bývají pro jednotlivé údaje a navíc i pro celé řádky. Pomocí těchto dat lze zkontrolovat a opravit datum narození, datum expirace průkazu, číslo průkazu a pohlaví. Navíc je obsaženo jméno a příjmení držitele dokladu, ale bez diakritiky. Kvůli tomu jsem ho pro opravu údajů nezahrnul.



Obrázek 4.5: Grafická ukázka souvislostí jednotlivých údajů na občanském průkazu.

4.5 Napojení na externí služby

V této fázi celého procesu už jsou kompletně k dispozici všechny požadované textové údaje, které lze vytěžit z fotografie dokladu. Zbývá k těmto datům zjistit a doplnit dodatečné informace získané z napojených externích systémů.

4.5.1 Kontrola platnosti dokladů

Ministerstvo vnitra České republiky vydává a denně aktualizuje seznam neplatných dokladů. Tento seznam je veřejně dostupný pro české občanské průkazy, červené cestovní pasy, zbrojní průkazy a zbrojní licence. Seznamy obsahují doklady, které jsou evidované jako ztracené, odcizené, ze zákona neplatné a neplatné na základě rozhodnutí. Vždy je uvedeno i datum zneplatnění dokladu. V seznamu nejsou uvedena čísla dokladů s prošlou lhůtou platnosti a doklady skartované. [1]

Po úspěšném strojovém přečtení čísla občanského průkazu nebo cestovního pasu se vytvořená služba podívá do této databáze neplatných dokladů a vyhledá podle čísla, zda se v ní nachází zkoumaný doklad z fotografie. Ve své odpovědi kromě textových údajů ze samotné fotografie vrací navíc příznak, zda je doklad podle seznamu platný. Vrácená hodnota nabývá jedné ze tří

možností:

- `VALID` – číslo dokladu bylo úspěšně přečteno a nenachází se v databázi neplatných dokladů
- `INVALID` – číslo dokladu bylo úspěšně přečteno a nachází se v databázi neplatných dokladů
- `NO_DATA` – číslo dokladu nebylo úspěšně přečteno a nejde proto zkontrolovat platnost dokladu nebo není dostupná databáze neplatných dokladů

4.5.2 Validace adresy

Dalším požadavkem zadavatele práce je automatická validace adresy trvalého bydliště uvedené na zadní straně občanského průkazu. Ostatní podporované typy dokladů tento údaj neobsahují.

Český úřad zeměměřický a katastrální poskytuje veřejný Registr územní identifikace, adres a nemovitostí (RÚIAN), kde lze nalézt databázi všech existujících adres v České republice. Jednotlivé dílčí atributy adresy jsou rozděleny do samostatných údajů a lze v nich proto lehce vyhledávat. Nejdůležitější dostupné atributy pro každé adresní místo jsou uvedeny v tabulce 4.2. Kromě těchto údajů jsou k dispozici navíc i vnitřní kódy každého názvu části adresy a další podrobnosti, které nejsou pro tento projekt podstatné.

Pro možnost ověření existence adresy je nejdřív potřeba rozdělit strojově přečtenou adresu z průkazu na jednotlivé části. Pro tuto úlohu není triviální sestavit algoritmus, který bude stoprocentně úspěšný, jelikož nebývají údaje na dokladu uvedeny vždy stejným způsobem.

Někdy je celá adresa uvedena na dvou řádcích, někdy na třech. Na posledním řádku vždy bývá název okresu, který je kromě některých částí Prahy uveden zkratkou "okr.". Na prvním (případně druhém) řádku bývá v různém pořadí uvedena obec, část obce, ulice a číslo popisné. V případě absence pojmenování části obce se uvádí název obce pouze jednou, nebo dvakrát za sebou oddělený čárkou. Některé obce nemají zavedenou uliční síť a není uvedený název ulice. V takovém případě bývá číslo popisné u názvu části obce. Někdy je číslo popisné uvedeno zkratkou "č.p.", ale ne vždy. Poznat rozdíl mezi údaji "HRUBÁ SKÁLA, HRUBÁ SKÁLA 19", kde číslo označuje číslo popisné, a "ČESKÉ BUDĚJOVICE, ČESKÉ BUDĚJOVICE 3", kde číslo je součástí názvu části obce, může být pro počítač oříšek. Oba tyto údaje jsou z reálných občanských průkazů a oba se nacházejí na prvním řádku adresy. Číslice jsou občas obsaženy i v označení části větších měst nebo v názvech ulic jako "28. ŘÍJNA" či "NÁBŘEŽÍ 17. LISTOPADU". V delších názvech se někdy objevují zkratky jako "NÁM. ČSA", "TŘ. 28. ŘÍJNA" nebo "DR. E. BENEŠE".

Tato volnost ve formátu uvedení adresy v kombinaci s pestrostí českých adres velice komplikuje strojové parsování údaje. I přesto jsem se pokusil sestrojit co nejlepší algoritmus, který funguje pro všechny adresy (včetně uve-

dených příkladů), které jsem měl při testování k dispozici. Ve sporných případech, kde si algoritmus není přesně jistý nějakým údajem, provedu kontrolu vůči seznamu existujících adres vícekrát s různými variantami rozdělení na konkrétní adresní atributy. Pokud se alespoň jednou najde v databázi shodná adresa, potvrdí se tím platnost adresy a zároveň se určí, které údaje rozdělené hodnoty reprezentují. Výsledný příznak platnosti vyčtené adresy nabývá

Název atributu	**Popis atributu**
Kód ADM	Kód adresního místa vedeného v Informačním systému územní identifikace (ISÚI).
Název obce	Název obce vedené v ISÚI
Název MOMC	Název městského obvodu/městské části, je vyplněn pouze v případě členěných statutárních měst
Název obvodu Prahy	Název obvodu Prahy, je vyplněn pouze v případě Hlavního města Prahy
Název části obce	Název části obce v rámci nadřazené obce, ve které je číslován stavební objekt
Název ulice	Název ulice, která je navázána na adresní místo, může být vyplněn pouze u obcí, které mají zavedenu uliční síť
Typ SO	Typ stavebního objektu, může nabývat hodnot č.p. (číslo popisné stavebního objektu) nebo č.ev. (číslo evidenční stavebního objektu)
Číslo domovní	Číslo popisné nebo číslo evidenční, podle rozlišeného typu SO
Číslo orientační	Číslo orientační, slouží k orientaci v rámci nadřazené ulice
Znak čísla orientačního	Znak čísla orientačního, uveden v případě, že je znak k orientačnímu číslu přidělen
PSČ	Poštovní směrovací číslo
Souřadnice Y	Souřadnice Y definičního bodu adresního místa v systému S-JTSK (systém jednotné trigonometrické sítě katastrální), uvedené v [m]
Souřadnice X	Souřadnice X definičního bodu adresního místa v systému S-JTSK, uvedené v [m]

Tabulka 4.2: Nejdůležitější atributy adresních míst z databáze RÚIAN [32].

v odpovědi vytvořené služby stejných hodnot jako údaj o platnosti dokladu popsaný v předchozí kapitole.

V požadavcích na tuto práci je i zjištění poštovního směrovacího čísla vyčtené adresy. Jelikož RÚIAN databáze pro každé adresní místo obsahuje i jeho PSČ, není jeho zjištění problém. Dílčí údaje jsou již rozdělené z procesu validace adresy a pokud byla adresa vyhodnocena jako validní a nalezena v databázi, není problém k ní dohledat PSČ a uvést ho v odpovědi vytvořené služby.

4.6 Shrnutí celého procesu

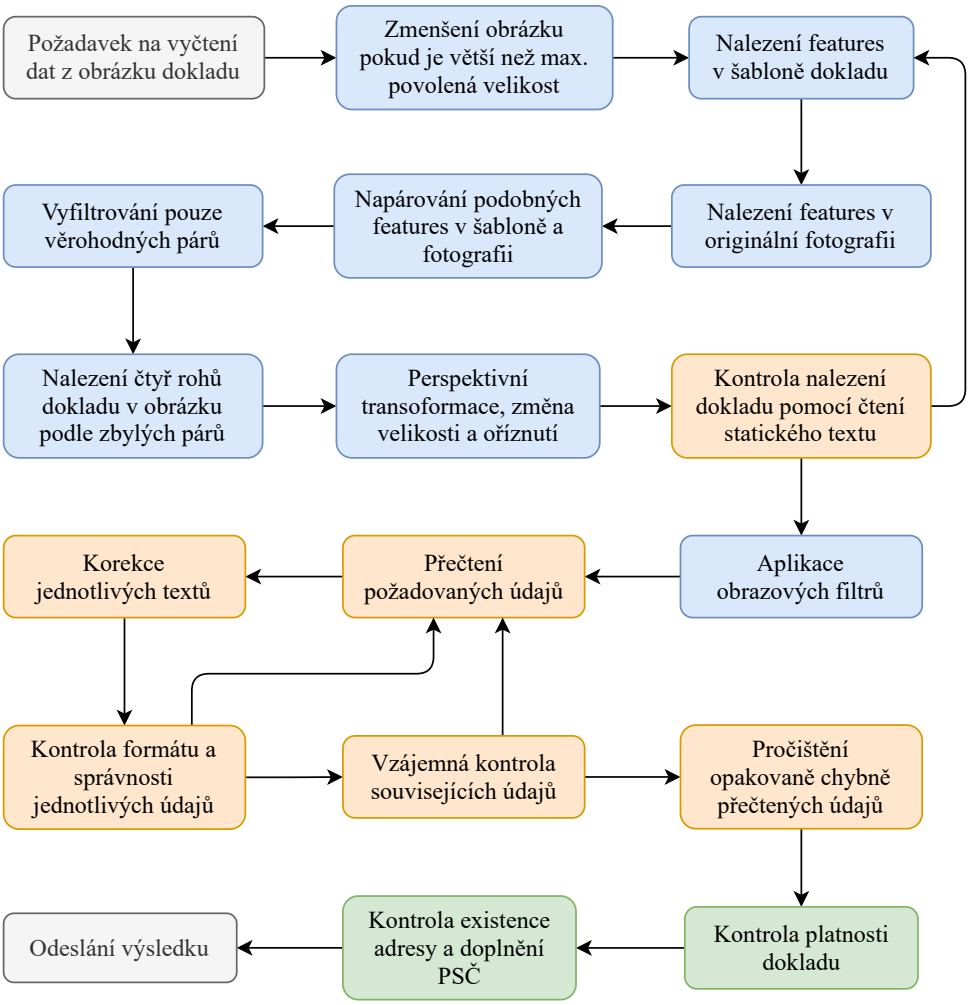
V této kapitole byl popsán celý proces detekce dokladu ve fotografii a extrakce textových údajů z něj. Proces se skládá z jednotlivých samostatně oddělitelných kroků, které byly postupně vysvětleny, byly uvedeny dostupné možnosti řešení těchto kroků a byl popsán výsledný způsob řešení zvolený v této práci. V následující podkapitole bych rád shrnul celý postup a upozornil na některé věci, které se nevešly do předchozích podkapitol.

Diagram v obrázku 4.6 názorně zobrazuje postup celého procesu od původní fotografie až po finální textový výstup. Modrou barvou jsou zvýrazněny operace související s úpravou fotografie, oranžově jsou obarveny úkony čtení a úpravy textu a zeleně jsou označeny kroky, kde se zjišťují doplňující informace z externích systémů.

Za povšimnutí stojí některé smyčky v diagramu. Jak již bylo párkrát zmíněno při popisu konkrétních kroků, v průběhu procesu se provádí několik kontrol, které přibližně ověří, zda se předchozí kroky podařily. Pokud se zjistí, že předchozí operace nebyly úspěšné, algoritmus se vrátí o několik kroků zpět a pokusí se trochu jiným způsobem zopakovat nezdařené operace.

Proces se drobně liší pro jednotlivé typy podporovaných dokladů. Pro občanský průkaz je postup komplexnější z důvodu rozdělení jednotlivých údajů na dvě strany dokladu. Nejprve se detekuje přední strana ve fotografiích, až poté se detekuje zadní strana. Po zpracování obrázků obou stran dokladu se z nich vytěží textová data a probíhají korekce všech údajů najednou.

Na řidičském průkazu se vyskytují všechny potřebné informace na přední straně. Při testování se ve většině případů stávalo, že některé údaje (zejména seznam získaných skupin řidičských oprávnění) nebyly vyčteny stoprocentně správně a bylo nutné opakovat proces s druhou šablonou a rozdílným způsobem čtení. Proto jsem pro tento doklad upravil postup a hned od začátku spouštím celý proces dvakrát paralelně s různými šablonami a některými dalšími rozdílnými parametry. Při vytěžování textu navíc využívám další šikovné vlastnosti Tesseractu, který dokáže pro provedenou operaci čtení textu vrátit pro každé přečtené slovo přibližnou hodnotu jistoty, že byl text přečten správně. Celý proces tedy spustím dvakrát nezávisle na sobě a při následných korekcích textů kontroluji všechny údaje a pro každý z nich porovnám pře-



Obrázek 4.6: Zjednodušený diagram celého procesu vytvářené služby.

čtené hodnoty z obou obrázků včetně jejich jistot a použiji variantu, kterou vyhodnotím jako správnější.

Tento postup jsem použil zatím pouze u řidičských průkazů, protože u ostatních dokladů nedocházelo tak často k chybám a opakování jednotlivých kroků. U řidičského průkazu se špatně četla některá složitější pole, například zmíněné skupiny řidičských oprávnění nebo číslo dokladu, protože tyto hodnoty obsahují kombinaci písmen, číslic a špatně čitelných mezer, které občas nebyly správně detekovány. U jiných typů průkazů by tento postup nepřinesl o mnoho lepší výsledky na úkor pomalejšího zpracování a vyplatí se tak postup opakovat až v případě detekce chyb.

Při testování služby pro fotografie s velmi vysokým rozlišením docházelo

častěji k nepřesnostem při hledání dokladu ve fotografii z důvodu existence více features v obrázcích a následného špatné napárování. Detekce dokladu navíc trvala několikanásobně déle než u menších fotografií. Z tohoto důvodu se ještě před popsaným zpracováním obrázku všechny příliš velké fotografie zmenší na maximální povolené rozměry. Toto platí obecně pro všechny podporované doklady.

Jedním z požadavků práce je automatická detekce typu dokladu ve fotografii. Veškerý předchozí postup předpokládal znalost typu dokladu, který se ve fotografii vyhledává. Tento problém řeším tak, že v případě požadavku na vyčtení textu z neznámého dokladu vyzkouším postupně nalézt všechny podporované typy dokladů ve fotografii a po ověřovacím čtení statických textů v transformovaném oříznutém obrázku dokážu říct, zda se konkrétní doklad ve fotografii skutečně nalezl nebo ne. Pokud se přečtené texty z nějakého minimálního procenta shodují se skutečnými texty, které měly být přečteny, tak dokončím postup pro konkrétní doklad. Pokud není shoda dostatečná, zkouším nalézt další typ dokladu. Tento postup funguje výborně a ve všech testovacích případech se doklad správně určil.

Pořadí hledání jednotlivých typů dokladů je dané statistikami využití služby. V praxi se služba nejčastěji používá pro nový typ občanských průkazů, dále pro řidičský průkaz, cestovní pas a nejméně často pro starý typ občanských průkazů, protože už jich v oběhu není mnoho. V tomto pořadí se tudíž zkouší doklady detekovat ve fotografiích, aby se minimalizovala doba trvání požadavku, která je při neznalosti typu dokladu vyšší než s počáteční znalostí této informace.

Kapitola 5

Technická stránka realizace

Volba použitých technologií, programovacího jazyka a knihoven zůstala kompletně na mně. Požadavkem zadavatele práce je podpora napojení do dalších systémů pomocí REST rozhraní. Výsledná mikroslužba (microservice) musí být schopna běžet v Docker kontejneru založeném na operačním systému Centos 7. Přípravu Docker obrazů, nastavení automatického sestavení služby, všechny části procesu průběžné integrace (continuous integration) a integraci do ostatních systémů banky jsem také zajistil, nicméně všechny tyto operace jsou specifické pro dané prostředí, kde služba běží, a jsou mimo rozsah této práce, proto se jimi nebudu dále zabývat.

Podporované vstupní formáty obrázků jsou JPEG, PNG a BMP. V textovém požadavku na REST službu jsou obrázky zakódovány pomocí Base64 kódování. V případě potřeby zpracovat fotografii v jiném formátu (například naskenovaná stránka ve formátu PDF) je potřeba před zavoláním služby překonvertovat soubor do jednoho z podporovaných formátů.

5.1 Použité technologie

Při rozhodování, které technologie pro projekt použiji, jsem v první řadě hledal nejlepší dostupné bezplatné knihovny pro strojové čtení textu a pro práci s obrázky. Programovací jazyk jsem volil až podle vybraných knihoven.

Klíčovým rozhodnutím byl výběr OCR knihovny Tesseract. Důvody volby této knihovny jsem již uvedl v kapitole 4.3.2. Tesseract nabízí knihovnu do jazyka C++ a existuje i několik wrapperů pro další programovací jazyky. Vyzkoušel jsem dva různé wrappery pro Javu, protože s Javou mám největší zkušenosti nejen já, ale i většina kolegů z firmy, která je zadavatelem této práce, a v budoucnu by to usnadnilo správu služby ostatními. Ovšem nebyl jsem s dostupnými wrappery spokojený, protože oproti stejnému použití přímo v C++ byl kód pomalejší, wrappery nepokrývaly všechny potřebné funkce a možnosti konfigurace a prakticky se jednalo pouze o volání funkcí z C++

přes Java Native Interface (JNI). Z tohoto důvodu jsem už neuvažoval nad dalšími dostupnými jazyky a rozhodl jsem se psát tuto část práce v jazyce C++.

Službu jsem vyvíjel na operačním systému Ubuntu 20.04, výsledný produkt bude běžet na systému Centos 7. Instalace knihovny Tesseract na oba systémy je velice jednoduchá, všechno potřebné zajistí výchozí správce balíčků. Po instalaci je potřeba pouze zkontrolovat dostupnost požadovaných typů balíků natrénovaných dat pro požadované jazyky a případně je dodatečně stáhnout z internetu. Ostatní potřebné závislosti (například již zmíněná knihovna Leptonica) se nainstalují automaticky.

Pro potřebné grafické operace s fotografiemi jsem použil bezplatnou otevřenou multiplatformní knihovnu OpenCV, které nabízí implementaci všech dříve popsaných operací a algoritmů. Nabízí funkce pro načtení obrázku, různé metody detekce, extrakce a párování features, nástroje pro dvojrozměrné transformace, změnu velikosti a ořez obrázku i rozmanité obrazové filtry. [33]

OpenCV je rozdělené na hlavní část a dále "contrib" část obsahující extra moduly a funkce, které nejsou ještě dostatečně otestované, jsou méně stabilní, méně používané nebo nejsou určeny pro bezplatné použití, přestože jsou známy jejich algoritmy. Příkladem může být zmíněný patentovaný SURF detektor features. Postupem času se některé oblíbené funkce přesouvají do hlavní části knihovny. Takto se to stalo nedávno s detektorem SIFT, kterému v loňském roce vypršel americký patent.

Při vývoji služby jsem pro lokální testování nainstaloval obě části OpenCV, abych měl možnost vyzkoušet funkce z "contrib" části. Instalace zahrnující tento modul je pracnější, protože se musí stáhnout zdrojové kódy a provést ruční instalace se správným nastavením potřebných parametrů. Ve výsledné službě však nejsou použity žádné funkce z doplňujícího modulu a instalaci hlavní části OpenCV lze provést jednoduše přes správce balíčků v operačním systému.

Veškerou funkcionalitu od načtení obrázku až po finální korekce textových údajů z dokladů jsem napsal v C++ za pomoci těchto dvou knihoven a vlastního kódu. Obě knihovny při správné konfiguraci podporují paralelní zpracování různých funkcí, díky tomu vytvořená služba běží paralelně v několika vláknech a celkový čas zpracování požadavků je nižší.

Pro zbývající funkcionality (konkrétně zajištění serveru s REST rozhraním, práci s JSON objekty a napojení na zmíněné externí služby) jsem se rozhodl pro usnadnění práce použít jazyk Java, ve kterém je snadnější zprovoznit tyto funkce pracující na vyšší úrovni.

Propojení Java části projektu a C++ části projektu je realizováno pomocí JNI. Z vytvořeného C++ kódu vnikne sdílená knihovna obsahující jedinou nativní funkci. Tato funkce přijímá jako argumenty dva obrázky včetně typu jejich formátu a případně informaci o typu hledaného dokladu ve fotografiích, pokud je tento údaj dostupný. Výstupem pak jsou vyextrahovaná textová data z dokladu.

Java část projektu je tvořená Spring Boot projektem. Tento framework nabízí snadnou konfiguraci webového serveru Apache Tomcat, který může být i přímo zabudovaný do výsledného spustitelného souboru. Vytvoření požadovaného API endpointu přijímajícího HTTP POST požadavky je otázka několika řádků kódu. Knihovna Jackson nabízí automatickou serializaci Java objektů do formátu JSON a naopak. Celá režie okolo samotného vytěžení dat z fotografií je díky těmto nástrojům jednoduchá záležitost.

Kostra výsledného Java programu od začátku až do konce vypadá zhruba takto:

1. Příjem REST požadavku s obrazovými daty a typem hledaného dokladu
2. Zavolání nativní C++ funkce, které se předají vstupní argumenty a výstupem jsou vytěžená textová data z fotografií dokladů
3. Případné zjištění doplňujících informací z externích systémů
4. Synchronní vrácení kompletní odpovědi na původní požadavek

5.2 Napojení na externí služby

Pro ověření platnosti dokladu a validaci adresy je potřeba napojit vytvořenou službu na externí zdroje. Tato úloha je zajištěna také v Java části projektu.

5.2.1 Databáze neplatných dokladů

Pro ověření neplatnosti dokladů nabízí Ministerstvo vnitra několik různých technických způsobů. Dostupné jsou ověřovací aplikace na různá mobilní zařízení, online webový formulář, webové rozhraní přijímající HTTP GET požadavky, které vrací strukturovanou odpověď ve formátu XML pro konkrétní jeden doklad, nebo nabízí možnost stažení kompletního seznamu všech neplatných dokladů ve formátu CSV. Ve stejném formátu jsou také denně zveřejňovány rozdílové seznamy, které obsahují čísla nově přidaných a odebraných dokladů za poslední den. [1]

Webové rozhraní s XML výstupem jsem zamítl z několika důvodů. Může se stát, že poskytnuté rozhraní nebude vždy dostupné (přeruší se internetové připojení na jedné ze stran, změní se webová adresa, atd.), dotazování se v průběhu požadavku dalšího webového serveru je zbytečně zdlouhavé a navíc by vznikala potřeba zpracovávat XML odpověď, u které není zaručeno, že se časem nebude měnit.

Z těchto důvodů jsem se rozhodl stáhnout si celý seznam neplatných dokladů ve formátu CSV a lokálně uložit hodnoty do databáze. Každou noc se spustí automatická úloha, která stáhne aktuální seznam a aktualizuje lokálně uložené hodnoty. Při potřebě ověřit konkrétní číslo dokladu se pouze pošle

dotaz do tabulky v databázi. V případě výpadku připojení nebo neočekávané změny struktury poskytovaných dat alespoň zbývá v databázi starší verze dat a je větší časový prostor pro opravu problému.

5.2.2 Databáze adres

Ověření adresy podle databáze RÚIAN lze také provést několika způsoby. V nabídce je webový formulář pro ověření konkrétní adresy online nebo možnost stažení kompletní databáze. Bohužel způsob stažení dat není vůbec přívětivý pro počítač. Po pročtení poskytnuté uživatelské příručky, technických požadavků a důkladném prozkoumání webu ČÚZK jsem našel dvě dostupné možnosti. [2]

První a zároveň doporučovaný způsob stažení dat je vyplnění formuláře se zhruba dvaceti formulářovými políčky na webu ČÚZK. Po odeslání formuláře se zobrazí stránkovaný seznam několika desítek odkazů na ZIP archivy které obsahují XML soubory s databází adres. Jednotlivé části adresy na sebe odkazují svými kódy. Pro získání celé adresy je nutné propojit a prohledat několik provázaných entit. Formát XML je nepřehledný a je potřeba pečlivě pročíst dokumentaci ke správnému pochopení dat. Databáze RÚIAN je často používaná v různých projektech, proto vznikla řada komerčních programů, které dokážou zdrojová data zpracovat do přehlednějšího formátu nebo poskytují zjednodušené REST rozhraní.

Druhý způsob stažení dat je daleko příjemnější, ovšem je složitější dohledat, že je k dispozici. Je taktéž oficiálně dostupný z webu ČÚZK, ovšem úřad na všech stránkách odkazuje a doporučuje spíše předchozí popsaný formát. Druhá metoda stažení databáze je pomocí ZIP archivu s CSV soubory, ve kterých je kompletní seznam všech existujících adresních míst se všemi důležitými parametry oddělenými středníky. Jedinou nevýhodou tohoto formátu oproti předchozímu je frekvence aktualizace, data jsou obnovována pouze jednou měsíčně na rozdíl od předchozí varianty, kde je možné získat navíc denní rozdíly v datech. [32]

Méně častá aktualizace dat zadavateli práce nevadí, tudíž jsem použil druhý popsaný formát. Automatická úloha opět každou noc zkontroluje, zda nejsou k dispozici novější data a případně je stáhne do lokální databáze. Validace adresy přečtené z dokladu a dohledání poštovního směrovacího čísla tedy funguje také offline.

Kapitola 6

Testování

Proces testování mě provázel už od samotného začátku vývoje služby. Jelikož jsem neměl žádné předchozí zkušenosti s tématy jako zpracování obrazu, strojové učení nebo strojové čtení textu, byla pro mne většina věcí nová a testoval jsem postupně všechny použité funkce a ověřoval výsledky pro různé volby parametrů. Výsledný finální produkt této práce je důkazem, že bez průběžného poctivého testování by řešení zdaleka nefungovalo tak, jak funguje.

Aby byly výsledky testů co nejvěrohodnější, veškeré testování probíhalo na reálných fotografiích reálných dokladů. Zadavatel práce mi poskytl reálné vzorky fotografií, které byly pořízené přímo běžnými uživateli, a díky tomu jsou velice pestré a přesně odpovídají datům, která bude služba v budoucnu zpracovávat. Nebylo tudíž potřeba vytvářet žádné umělé vzorky nebo pořizovat vlastní fotografie.

Poskytnuté rozmanité vzorky skvěle otestují službu pro různé velikosti a formáty obrázků, různé světlo, úhel a vzdálenost vyfocení, různé typy pozadí, otočení obrázku i všelijaké nečitelnosti, světelné odlesky a překryvy dokladů.

Průběžné testování jsem prováděl už v průběhu vývoje programu, abych si přibližně otestoval funkčnost jednotlivých kroků a službu částečně vyladil do nějakého počátečního použitelného stavu. Ze začátku jsem kladl důraz na C++ část, která provádí samotnou detekci dokladu, extrakci a korekce textových údajů. Až po důkladném vyladění této části jsem se pustil do Java obalu a napojení na externí databáze. Zpočátku vývoje jsem testoval ručně, spouštěl jsem službu pro jednotlivé obrázky a kontroloval výstup. Když se výsledky začaly podobat skutečným datům, potřeboval jsem proces testování zautomatizovat, abych mohl hromadně testovat na více vzorcích a měl kvalitnější statistiky úspěšnosti než pouhou ruční kontrolu každého spuštění programu.

Pomocí vytvářené služby jsem si nechal pro testovací vzorky fotografií jednotlivých dokladů vygenerovat vzorové výsledky, které jsem následně ručně procházel, kontroloval vytěžená data a případně je ručně opravil nebo doplnil, aby všechny údaje přesně souhlasily s údaji na dokladech. Tento proces byl velice zdlouhavý a nudný, ovšem dost mi pomohl získat představu, jaké jsou

nejčastější chyby a nedostatky vytvořené služby a tyto chyby opravit.

Následně jsem si vytvořil testovací program, který automaticky vygeneruje data pro všechny zvolené testovací fotografie, poté porovná své výsledky se vzorovými ručně opravenými skutečnými daty a vypočítá celkovou statistiku úspěšnosti pro jednotlivé údaje i souhrnně pro celé doklady. Spouštění tohoto automatického testování na stovkách vzorových fotografií pro každý podporovaný typ dokladu mi umožnilo ještě lépe vyladit výslednou službu a pomocí jednoduchého skriptu si vypsat všechny chybně přečtené údaje a pokusit se zdokonalit službu tak, aby se chyby v dalším testování neopakovaly.

Mým hlavním cílem bylo testovat službu nejen na kvalitně pořízených fotografiích, ale z dostupných vzorků vybírat převážně i složité špatně čitelné fotografie. Jedině tak je totiž při testování názorně vidět, jak se služba zpřesňuje, a stále existují rezervy, které je možné ještě vylepšit. Do automatických testů jsem zahrnul i obrázky, kde jsem měl problém s přečtením některých údajů i já. Použil jsem i fotografie, ve kterých byly některé údaje částečně zakryté nebo ořízнuté, ale daly se doplnit z ostatních údajů v dokladu. Příkladem tak může být občanský průkaz, který měl částečně oříznutou fotografii přední strany a chyběla polovina čísla dokladu. Z druhé fotografie zadní strany však vytvořená služba dokázala tento údaj správně doplnit přečtením strojově čitelné zóny dokladu, která také obsahuje informaci o čísle dokladu.

Celková doba zpracování jednoho požadavku se liší podle typu dokladu, velikosti a obsahu vstupních fotografií. Pro neznámý typ dokladu bývá proces lehce pomalejší než při předem známém typu. Svoji roli v celkovém čase hraje i kontrola platnosti dokladů a validace adresy pomocí dotazování v databázi. Průměrná doba kompletního paralelního zpracování požadavku se na mém notebooku pohybuje okolo 1,8 sekundy, na výkonnějším produkčním serveru je čas o něco kratší, vždy však záleží na konkrétních datech.

6.1 Statistiky úspěšnosti služby

V následujících tabulkách 6.1, 6.2 a 6.3 přikládám statistiky úspěšnosti služby pro jednotlivé podporované typy dokladů. V prvním sloupci je seznam názvů čtených polí z dokladu, druhý sloupec obsahuje poměr kompletně správně přečtených údajů (bez jediného chybného znaku) a ve třetím sloupci je uveden poměr správně přečtených znaků oproti skutečným datům. Tento ukazatel je měřen pomocí již zmíněného Levenshteinova algoritmu a jeho hodnota umožňuje porovnat, zda se v případě špatně přečteného pole jedná spíše o drobný překlep nebo se údaje nenalezly vůbec.

V posledním řádku každé tabulky je statistika pro celý doklad, kde se uvedená pravidla aplikují pro všechna pole dohromady. Poměr kompletně správně přečtených dokladů musí být logicky stejný nebo horší než nejhůře přečtené pole. Tato hodnota v praxi bývá řádově nižší než statistiky pro jednotlivá pole,

protože se zde hromadí chyby ze všech přečtených údajů. Čím více polí doklad obsahuje, tím je větší pravděpodobnost, že se v nějakém poli stane chyba.

Vysoce rozdílné hodnoty poměru kompletně správně přečtených dokladů a poměru správně přečtených znaků v celém dokladu (hodnoty z posledního řádku tabulek) poukazují na fakt, že ve většině dokladů, kde se stala nějaká chyba ve čtení, byla tato chyba pouze drobná a neznamenala, že se údaje nevyčetly vůbec. Toto jsem ověřil i kontrolou výstupních dat pro několik testovaných fotografií a v naprosté většině případů, které se započítaly jako chybně přečtené doklady, byla chyba pouze v nějakém diakritickém znaménku v jednom údaji.

Při hodnocení statistik měření úspěšnosti je také potřeba brát v potaz již zmíněný fakt, že součástí testovacích fotografií byly i extrémně nepřehledné nebo rozmazané snímky, s jejichž přečtením jsem občas měl problém i já. Při testování vlastních dobře osvětlených ostrých fotografií v dostatečném rozlišení se mi ve všech případech podařilo z dokladu vytěžit stoprocentně správná data.

Název pole	**Poměr správně přečtených údajů**	**Poměr správně přečtených znaků**
Číslo dokladu	0,996	0,996
Příjmení	0,978	0,9926
Jméno	0,99	0,9945
Datum narození	0,996	0,9978
Místo narození	0,948	0,9875
Datum vydání	0,986	0,9878
Platnost do	0,99	0,9936
Pohlaví	0,998	0,998
Rodné číslo	0,974	0,9759
Vydávající autorita	0,95	0,9575
Titul	0,994	0,9751
Rodinný stav	0,982	0,9763
Adresa 1. řádek	0,904	0,9488
Adresa 2. řádek	0,904	0,9478
Adresa 3. řádek	0,956	0,9566
Celý doklad	**0,824**	**0,9737**

Tabulka 6.1: Výsledky testování služby pro nový typ českého občanského průkazu (testováno na 500 vzorcích)

Název pole	Poměr správně přečtených údajů	Poměr správně přečtených znaků
Příjmení	0,932	0,9804
Jméno	0,942	0,9715
Titul	0,994	0,969
Datum narození	0,978	0,9906
Místo narození	0,946	0,9792
Datum vydání	0,964	0,9798
Platnost do	0,97	0,9826
Vydávající autorita	0,972	0,9856
Rodné číslo	0,97	0,971
Číslo dokladu	0,96	0,9678
Bydliště	0,89	0,9432
Skupiny oprávnění	0,88	0,9586
Celý doklad	**0,746**	**0,9727**

Tabulka 6.2: Výsledky testování služby pro český řidičský průkaz (testováno na 500 vzorcích)

Název pole	Poměr správně přečtených údajů	Poměr správně přečtených znaků
Číslo dokladu	0,97	0,9775
Příjmení	0,975	0,9772
Jméno	0,95	0,9761
Datum narození	0,985	0,985
Místo narození	0,97	0,9868
Rodné číslo	0,97	0,9699
Pohlaví	0,995	0,995
Datum vydání	0,98	0,989
Platnost do	0,98	0,9885
Vydávající autorita	0,945	0,9769
Celý doklad	**0,885**	**0,9809**

Tabulka 6.3: Výsledky testování služby pro český cestovní pas (testováno na 200 vzorcích)

6.2 Dodatečná měření

Kromě konečných statistik úspěšnosti celého procesu pro všechny podporované typy dokladů jsem samozřejmě zkoušel průběžně upravovat a testovat jednotlivé dílčí kroky. Pro důkaz, že všechny popsané kroky služby jsou podstatné a zlepšují konečné výsledky, jsem provedl dodatečná měření, kde jsem vynechal některé operace, bez kterých se služba obejde. V tabulce 6.4 jsou uvedeny poměry kompletně správně přečtených hodnot pro jednotlivá pole občanského průkazu. Prvním ořezáním algoritmu bylo vynechání předzpracování fotografie pomocí obrazových filtrů před strojovým čtením textu. Účinnost filtrů je podle výsledků měření znát hlavně v delších textových údajích jako místo narození, vydávající autorita nebo trvalá adresa. Nepostradatelnou roli hrají ve výsledné úspěšnosti i finální korekce vyčteného textu, což dokazuje poslední sloupec tabulky. Tyto korekce zachrání před špatným přečtením některého údaje zhruba každý druhý doklad.

Název pole	**Originální úspěšnost**	**Vynechání obrazových filtrů**	**Vynechání textových korekcí**
Číslo dokladu	0,996	0,994	0,766
Příjmení	0,978	0,958	0,966
Jméno	0,99	0,972	0,966
Datum narození	0,996	0,988	0,984
Místo narození	0,948	0,868	0,894
Datum vydání	0,986	0,988	0,944
Platnost do	0,99	0,99	0,972
Pohlaví	0,998	0,996	0,914
Rodné číslo	0,974	0,964	0,766
Vydal	0,95	0,876	0,878
Titul	0,994	0,986	0,966
Rodinný stav	0,982	0,964	0,898
Adresa 1. řádek	0,904	0,748	0,862
Adresa 2. řádek	0,904	0,738	0,87
Adresa 3. řádek	0,956	0,846	0,9
Celý doklad	**0,824**	**0,5**	**0,334**

Tabulka 6.4: Statistiky úspěšnosti služby pro český občanský průkaz s vynecháním některých kroků (testováno na 500 vzorcích)

V tabulce 6.5 jsem otestoval výsledky služby pro jednotlivá pole občanského průkazu s použitím různých metod detekce a extrakce features. Nejlepší výsledky poskytovaly metody SIFT a BRISK. V případě druhé metody byly statistiky lepší pro většinu jednotlivých údajů, ovšem celková úspěšnost celých dokladů byla paradoxně vyšší u metody SIFT, kterou jsem nakonec použil ve výsledném softwaru. Metoda FAST obsahuje pouze detekci features (nalezení konkrétních bodů), proto jsem ji musel zkombinovat s metodou BRIEF, která naopak poskytuje pouze extrakci features (vypočítání deskriptorů jednotlivých detekovaných bodů). Kombinace těchto dvou metod však neposkytuje příliš dobré výsledky pro toto konkrétní použití a vhodné jsou spíše ostatní uvedené metody, které najednou umožňují detekci i extrakci features.

Název pole	**SIFT**	**SURF**	**BRISK**	**FAST, BRIEF**
Číslo dokladu	0,996	0,996	0,998	0,364
Příjmení	0,978	0,982	0,98	0,33
Jméno	0,99	0,984	0,99	0,34
Datum narození	0,996	0,996	0,998	0,368
Místo narození	0,948	0,94	0,948	0,306
Datum vydání	0,986	0,984	0,982	0,31
Platnost do	0,99	0,994	0,992	0,362
Pohlaví	0,998	0,998	1	0,458
Rodné číslo	0,974	0,952	0,976	0,302
Vydávající autorita	0,95	0,938	0,944	0,308
Titul	0,994	0,994	1	0,832
Rodinný stav	0,982	0,974	0,986	0,48
Adresa 1. řádek	0,904	0,866	0,908	0,276
Adresa 2. řádek	0,904	0,884	0,904	0,29
Adresa 3. řádek	0,956	0,946	0,966	0,444
Celý doklad	**0,824**	**0,752**	**0,798**	**0,188**

Tabulka 6.5: Statistiky úspěšnosti služby pro český občanský průkaz s použitím různých detektorů features (testováno na 500 vzorcích)

Závěr

Cílem této práce bylo vytvořit REST službu, která na vstupu přijme jednu či dvě fotografie osobního dokladu, detekuje doklad v obrázku a vyextrahuje z něj strukturované textové údaje. Kromě informací dostupných přímo ve fotografii služba navíc ověří platnost dokladu, validuje trvalou adresu držitele a doplní poštovní směrovací číslo pomocí veřejně dostupných databází Ministerstva vnitra a Českého úřadu zeměměřického a katastrálního.

Dílo vzniklo pro externího zadavatele a bude mít reálné využití v bankovním systému. Z tohoto důvodu byl kladen důraz na požadavky zadavatele a na přizpůsobení služby tak, aby dokázala nepřetržitě a spolehlivě běžet v produkčním systému banky.

Při vytváření služby jsem se zaměřil zejména na kvalitu a přesnost vyčtených údajů na úkor rychlosti zpracování požadavků. Služba byla navržena tak, aby podporovala snadné rozšíření o případný další typ osobního dokladu. Testování výsledného produktu probíhalo řádově na tisících různých fotografií reálných osobních dokladů.

Bibliografie

1. MINISTERSTVO VNITRA ČESKÉ REPUBLIKY. *Neplatné doklady* [online] [cit. 2021-11-04]. Dostupné z: `https://aplikace.mvcr.cz/neplatne-doklady/`.
2. ČESKÝ ÚŘAD ZEMĚMĚŘICKÝ A KATASTRÁLNÍ. *Registr územní identifikace, adres a nemovitostí* [online] [cit. 2021-11-04]. Dostupné z: `https://www.cuzk.cz/ruian/`.
3. TRETYAKOV, Konstantin. *PassportEye* [online]. 2020-11-29 [cit. 2021-11-05]. Dostupné z: `https://pypi.org/project/PassportEye/`.
4. ACCURATECHNOLABS. *Protect Your Business with an Outstanding OCR Scanner App* [online] [cit. 2021-11-04]. Dostupné z: `https://accurascan.com/accura-sdk`.
5. MICROBLINK LTD. *BlinkID - Digital onboarding made easy* [online] [cit. 2021-11-05]. Dostupné z: `https://microblink.com/products/blinkid`.
6. SMART ENGINES. *Automatic and eco-friendly document scanning and OCR* [online]. 2019-02-06 [cit. 2021-11-05]. Dostupné z: `https://smartengines.com/`.
7. SMART ENGINES. *Smart ID Engine - EU Driver's License Automatic Scan and Recognition* [online] [cit. 2021-11-05]. Dostupné z: `https://www.youtube.com/watch?v=8A6r-d_ziaM`.
8. CODEWARE S.R.O. *Strojové čtení občanských průkazů nejen pro notáře* [online] [cit. 2021-11-04]. Dostupné z: `https://www.codeware.cz/blog/strojove-cteni-obcanskych-prukazu-nejen-pro-notare`.
9. TRASK SOLUTIONS A.S. *ZenID - Automatizované vytěžování informací z osobních dokladů a detekce podvodů* [online] [cit. 2021-11-04]. Dostupné z: `https://www.trask.cz/files/produktovy-list-trask-zenid-cz.pdf`.

10. WIKIPEDIE, OTEVŘENÁ ENCYKLOPEDIE. *Občanský průkaz* [online] [cit. 2021-11-06]. Dostupné z: https://cs.wikipedia.org/wiki/Ob%C4%8Dansk%C3%BD_pr%C5%AFkaz.

11. MINISTERSTVO VNITRA ČESKÉ REPUBLIKY. *Občanské průkazy bez strojově čitelných údajů* [online] [cit. 2021-11-06]. Dostupné z: https://www.mvcr.cz/soubor/obcanske-prukazy-bez-strojove-citelnych-udaju.aspx.

12. MINISTERSTVO VNITRA ČESKÉ REPUBLIKY. *Občanské průkazy se strojově čitelnými údaji* [online] [cit. 2021-11-06]. Dostupné z: https://www.mvcr.cz/soubor/obcanske-prukazy-se-strojove-citelnymi-udaji.aspx.

13. MINISTERSTVO VNITRA ČESKÉ REPUBLIKY. *Občanské průkazy se strojově čitelnými údaji vydávané od srpna 2021* [online] [cit. 2021-11-06]. Dostupné z: https://www.mvcr.cz/soubor/obcanske-prukazy-se-strojove-citelnymi-udaji-od-viii-2021.aspx.

14. WIKIPEDIE, OTEVŘENÁ ENCYKLOPEDIE. *Řidičský průkaz* [online] [cit. 2021-11-06]. Dostupné z: https://cs.wikipedia.org/wiki/%C5%98idi%C4%8Dsk%C3%BD_pr%C5%AFkaz.

15. EVROPSKÁ RADA A RADA EU. *PRADO - Veřejný rejstřík pravých dokladů totožnosti a cestovních dokladů online* [online] [cit. 2021-11-06]. Dostupné z: https://www.consilium.europa.eu/prado/cs/prado-documents/CZE/all/docs-all.html.

16. K, Bharath. *Object Detection Algorithms and Libraries* [online]. 2021-08-18 [cit. 2021-11-13]. Dostupné z: https://neptune.ai/blog/object-detection-algorithms-and-libraries.

17. BROWNLEE, Jason. *A Gentle Introduction to Object Recognition With Deep Learning* [online]. 2019-05-22 [cit. 2021-11-13]. Dostupné z: https://machinelearningmastery.com/object-recognition-with-deep-learning/.

18. MUTHUKRISHNAN.R, M.RADHA. *Edge detection techniques for image segmentation* [online]. 2011-12 [cit. 2021-11-14]. Dostupné z: https://airccse.org/journal/jcsit/1211csit20.pdf.

19. RAX AUTOMATION SUITE. *Template-based versus Feature-based Template Matching* [online]. 2019-11-16 [cit. 2021-11-14]. Dostupné z: https://medium.datadriveninvestor.com/template-based-versus-feature-based-template-matching-e6e77b2a3b3a.

20. ÚŘAD PRŮMYSLOVÉHO VLASTNICTVÍ. *Databáze patentů a užitných vzorů* [online] [cit. 2021-11-18]. Dostupné z: https://upv.gov.cz/informacni-zdroje/narodni-databaze/databaze-patentu-a-uzitnych-vzoru.

21. MAI THANH NHAT TRUONG, SANGHOON KIM. *A Review on Image Feature Detection and Description* [online]. 2016-11 [cit. 2021-11-14]. Dostupné z: https://www.koreascience.or.kr/article/CFKO201629368424723.pdf.

22. DAY, Lewin. *What exactly is a Gaussian blur?* [Online]. 2021-07-21 [cit. 2021-11-21]. Dostupné z: https://hackaday.com/2021/07/21/what-exactly-is-a-gaussian-blur/.

23. FISHER, Robert. *Bilateral Filtering for Gray and Color Images* [online] [cit. 2021-11-21]. Dostupné z: https://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/MANDUCHI1/Bilateral_Filtering.html.

24. TED HAN, AMANDA HICKMAN. *Our Search for the Best OCR Tool and What We Found* [online]. 2019-02-19 [cit. 2021-11-21]. Dostupné z: https://source.opennews.org/articles/so-many-ocr-options/.

25. ABBYY. *ABBYY FineReader Engine* [online] [cit. 2021-11-21]. Dostupné z: https://www.abbyy.com/ocr-sdk/specifications/.

26. MENDELSON, Edward. *ABBYY FineReader Review* [online]. 2020-02-06 [cit. 2021-11-21]. Dostupné z: https://www.pcmag.com/reviews/abbyy-finereader.

27. KOFAX INC. *The Most Robust OCR and Imaging SDK for Linux* [online] [cit. 2021-11-24]. Dostupné z: https://www.kofax.com/products/omnipage/omnipage-for-developers/omnipage-capture-sdk-for-linux.

28. WIKIPEDIA, THE FREE ENCYCLOPEDIA. *Comparison of optical character recognition software* [online] [cit. 2021-11-21]. Dostupné z: https://en.wikipedia.org/wiki/Comparison_of_optical_character_recognition_software.

29. ASPRISE. *C/C++ OCR and Barcode Recognition* [online] [cit. 2021-11-24]. Dostupné z: https://asprise.com/royalty-free-library/c-c++-ocr-api-overview.html.

30. TESSERACT-OCR. *Tesseract User Manual* [online] [cit. 2021-11-25]. Dostupné z: https://tesseract-ocr.github.io/tessdoc/.

31. LORENC, Miroslav. *Ověření správnosti rodného čísla* [online] [cit. 2021-11-27]. Dostupné z: https://lorenc.info/3MA381/overeni-spravnosti-rodneho-cisla.htm.

32. ČESKÝ ÚŘAD ZEMĚMĚŘICKÝ A KATASTRÁLNÍ. *Adresní místa RÚIAN ve formátu CSV* [online] [cit. 2021-11-04]. Dostupné z: https://vdp.cuzk.cz/vymenny_format/csv/ad-csv-struktura.pdf.

33. OPENCV. *About OpenCV* [online] [cit. 2021-11-30]. Dostupné z: https://opencv.org/about/.

Příloha A

Popis rozhraní vytvořené služby

Formát požadavku

- `documentData` – Binární data prvního obrázku v Base64 formátu
- `documentData2` – Binární data volitelného druhého obrázku (pokud jsou strany dokladu rozděleny do dvou obrázků) v Base64 formátu
- `contentType` – Content type prvního obrázku, podporované typy jsou `image/jpeg`, `image/png` a `image/bmp`
- `contentType2` – Content type volitelného druhého obrázku, podporované typy jsou `image/jpeg`, `image/png` a `image/bmp`
- `documentType` – Typ odeslaného dokumenetu k přečtení. Obsahuje hodnoty `ID_CARD_CZ` pro přečtení všech podporovaných typů českého občanského průkazu, `DRIVING_LICENCE_CZ` pro přečtení českého řidičského průkazu, `PASSPORT_CZ` pro přečtení českého cestovního pasu nebo hodnotu `UNKNOWN` pro automatické určení typu dokumentu (může být nepatrně pomalejší)

Formát odpovědi

- `documentType` – Typ přečteného dokumentu. V případě zadání konkrétního dokumentu v požadavku vrátí stejnou hodnotu, v případě neznámého typu dokumentu vrátí konkrétní hodnotu pro rozpoznaný dokument nebo `UNKNOWN` v případě, že nebyl nalezen žádný z podporovaných dokumentů
- `idCardCz` – Obsahuje data z přečteného českého občanského průkazu
- `drivingLicenceCz` – Obsahuje data z přečteného českého řidičského průkazu

- `passportCz` – Obsahuje data z přečteného českého cestovního pasu

Formát pole pro občanský průkaz

- `number` – Číslo dokladu
- `lastName` – Příjmení
- `firstName` – Křestní jméno
- `dateOfBirth` – Datum narození
- `placeOfBirth` – Místo narození
- `validFrom` – Datum vydání
- `validTo` – Platnost do
- `gender` – Pohlaví. Obsahuje hodnoty `M` nebo `F`
- `personalCode` – Rodné číslo bez lomítka
- `issuedBy` – Autorita, která vydala doklad
- `title` – Titul (pokud je uveden)
- `maritalStatus` – Rodinný stav (pokud je uveden). Obsahuje hodnoty `ženatý`, `svobodný`, `rozvedený`, `vdovec`, `partnerství` pro mužské pohlaví a hodnoty `vdaná`, `svobodná`, `rozvedená`, `vdova`, `partnerství` pro ženské pohlaví
- `address` – Adresa trvalého pobytu
  - `city` – Název obce
  - `cityPart` – Název části obce
  - `street` – Název ulice
  - `number` – Číslo popisné
  - `postalCode` – Dohledané poštovní směrovací číslo
  - `addressLine1` – První řádek adresy přečtené na občanském průkazu (originální data)
  - `addressLine2` – Druhý řádek adresy přečtené na občanském průkazu (originální data)
  - `addressLine3` – Třetí řádek adresy přečtené na občanském průkazu (originální data)
- `citizenship` – Národnost - ISO Alpha-2 standard (CZ)

- `expirationStatus` – Údaj, zda je průkaz expirovaný (mimo rozmezí datumů platnosti). Obsahuje hodnoty `VALID` (platný), `INVALID` (expirovaný) nebo `NO_DATA` (data nejsou k dispozici)
- `numberStatus` – Údaj, zda je průkaz vedený jako platný doklad v době přečtení. Obsahuje hodnoty `VALID` (platný), `INVALID` (neplatný) nebo `NO_DATA` (data nejsou k dispozici)
- `addressStatus` – Údaj, zda je rozpoznaná adresa platná. Obsahuje hodnoty `VALID` (platný), `INVALID` (neexistující) nebo `NO_DATA` (data nejsou k dispozici)
- `idCardType` – Typ rozpoznaného občanského průkazu. Jsou podporovány typy `ID_CARD_2012` (aktuální vzor OP vydávaný od roku 2012) a `ID_CARD_2005` (starší vzor OP vydávaný od roku 2005 do roku 2011)

Formát pole pro řidičský průkaz

- `number` – Číslo dokladu
- `lastName` – Příjmení
- `firstName` – Křestní jméno
- `title` – Titul (pokud je uveden)
- `dateOfBirth` – Datum narození
- `placeOfBirth` – Místo narození
- `validFrom` – Datum vydání
- `validTo` – Platnost do
- `personalCode` – Rodné číslo bez lomítka
- `issuedBy` – Autorita, která vydala doklad
- `residence` – Místo pobytu (pokud je uvedeno)
- `groupsOfVehicles` – Skupiny řidičského oprávnění oddělené mezerou

Formát pole pro cestovní pas

- `number` – Číslo dokladu
- `lastName` – Příjmení
- `firstName` – Křestní jméno

- `dateOfBirth` – Datum narození
- `placeOfBirth` – Místo narození
- `validFrom` – Datum vydání
- `validTo` – Platnost do
- `gender` – Pohlaví. Obsahuje hodnoty `M` nebo `F`
- `personalCode` – Rodné číslo bez lomítka
- `issuedBy` – Autorita, která vydala doklad
- `expirationStatus` – Údaj, zda je průkaz expirovaný (mimo rozmezí datumů platnosti). Obsahuje hodnoty `VALID` (platný), `INVALID` (expirovaný) nebo `NO_DATA` (data nejsou k dispozici)
- `numberStatus` – Údaj, zda je průkaz vedený jako platný doklad v době přečtení. Obsahuje hodnoty `VALID` (platný), `INVALID` (neplatný) nebo `NO_DATA` (data nejsou k dispozici)

Příloha B

Příklad volání služby

Občanský průkaz

Požadavek

```
POST /ocr-document HTTP/1.1
Content-Type: application/json
Content-Length: 1000
Host: localhost:8901

{
    "documentData" : "base64 data",
    "contentType" : "image/jpeg",
    "documentData2" : "base64 data",
    "contentType2" : "image/jpeg",
    "documentType" : "ID_CARD_CZ"
}
```

Odpověď

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 1000

{
    "documentType" : "ID_CARD_CZ",
    "idCardCz" : {
        "number" : "123456789",
        "lastName" : "NOVÁK",
        "firstName" : "JAN",
        "dateOfBirth" : "1950-01-01T00:00:00.000+0000",
        "placeOfBirth" : "BRNO okr. BRNO-MĚSTO",
        "validFrom" : "2015-01-01T00:00:00.000+0000",
        "validTo" : "2025-01-01T00:00:00.000+0000",
        "gender" : "M",
        "personalCode" : "5001010300",
        "issuedBy" : "Magistrát města BRNA",
        "title" : "Ing.",
        "maritalStatus" : "ženatý",
        "address" : {
            "city" : "BRNO",
            "cityPart" : "ČERNOVICE",
            "street" : "FAMĚROVO NÁMĚSTÍ",
            "number" : "111/22",
            "postalCode" : "61800",
            "addressLine1" : "BRNO, ČERNOVICE",
            "addressLine2" : "FAMĚROVO NÁMĚSTÍ č.p. 111/22",
            "addressLine3" : "okr. BRNO-MĚSTO"
        },
        "citizenship" : "CZ",
        "expirationStatus" : "VALID",
        "numberStatus" : "VALID",
        "addressStatus" : "VALID",
        "idCardType" : "ID_CARD_2012"
    }
}
```

Řidičský průkaz

Požadavek

```
POST /ocr-document HTTP/1.1
Content-Type: application/json
Content-Length: 1000
Host: localhost:8901

{
    "documentData" : "base64 data",
    "contentType" : "image/jpeg",
    "documentData2" : "base64 data",
    "contentType2" : "image/jpeg",
    "documentType" : "DRIVING_LICENCE_CZ"
}
```

Odpověď

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 1000

{
    "documentType" : "DRIVING_LICENCE_CZ",
    "drivingLicenceCz" : {
        "number" : "AB 123456",
        "lastName" : "NOVÁK",
        "firstName" : "JAN",
        "title" : "Ing.",
        "dateOfBirth" : "1950-01-01T00:00:00.000+0000",
        "placeOfBirth" : "PRAHA 8",
        "validFrom" : "2015-01-01T00:00:00.000+0000",
        "validTo" : "2025-01-01T00:00:00.000+0000",
        "personalCode" : "5001010300",
        "issuedBy" : "Mag. hl.m. PRAHA",
        "residence" : "PRAHA",
        "groupsOfVehicles" : "AM A1 B1 B"
    }
}
```

Cestovní pas

Požadavek

```
POST /ocr-document HTTP/1.1
Content-Type: application/json
Content-Length: 1000
Host: localhost:8901

{
    "documentData" : "base64 data",
    "contentType" : "image/jpeg",
    "documentData2" : "base64 data",
    "contentType2" : "image/jpeg",
    "documentType" : "PASSPORT_CZ"
}
```

Odpověď

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 1000

{
    "documentType" : "PASSPORT_CZ",
    "passportCz" : {
        "number" : "12345678",
        "lastName" : "NOVÁK",
        "firstName" : "JAN",
        "dateOfBirth" : "1950-01-01T00:00:00.000+0000",
        "placeOfBirth" : "PRAHA 8",
        "validFrom" : "2015-01-01T00:00:00.000+0000",
        "validTo" : "2025-01-01T00:00:00.000+0000",
        "gender" : "M",
        "personalCode" : "5001010300",
        "issuedBy" : "Magistrát města BRNA",
        "expirationStatus" : "VALID",
        "numberStatus" : "VALID"
    }
}
```

Příloha C

Seznam použitých zkratek

API Application Programming Interface

BRIEF Binary Robust Independent Elementary Features

BRISK Binary Robust Invariant Scalable Keypoints

CRM Customer Relationship Management

CSV Comma Separated Values

ČÚZK Český úřad zeměměřický a katastrální

EU Evropská unie

FAST Features from Accelerated Segment Test

FLANN Fast Library for Approximate Nearest Neighbors

GPU Graphics Processing Unit

HTTP Hypertext Transfer Protocol

ISÚI Informační systém územní identifikace

JNI Java Native Interface

JSON JavaScript Object Notation

LSTM Long Short-Term Memory

MRZ Machine-Readable Zone

OCR Optical Character Recognition

ORB Oriented FAST and Rotated BRIEF

OSD Orientation and Script Detection

PDF Portable Document Format

PSČ Poštovní směrovací číslo

RANSAC Random Sample Consensus

R-CNN Region-Based Convolutional Neural Network

REST Representational State Transfer

RÚIAN Registr územní identifikace, adres a nemovitostí

SDK Software Development Kit

SIFT Scale-Invariant Feature Transform

SO Stavební objekt

SSD Single Shot Detector

SURF Speeded Up Robust Features

USA Spojené státy americké

YOLO You Only Look Once

Příloha D

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `thesis.pdf` ..... text práce ve formátu PDF
- `src`
  - `impl` ..... zdrojové kódy implementace
    - `converter` ..... zdrojové kódy C++ knihovny
    - `service` ..... zdrojové kódy Java Spring Boot projektu
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX