VYSOKÁ ŠKOLA POLYTECHNICKÁ JIHLAVA

Katedra technických studií

Analýza optického rozpoznávání znaků

bakalářská práce

Autor práce: Ferenc Docsa

Vedoucí práce: Ing. Marek Musil

Jihlava 2020

Vysoká škola
polytechnická
Jihlava

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Autor práce:	**Ferenc Docsa**
Studijní program:	Elektrotechnika a informatika
Obor:	Aplikovaná informatika
Název práce:	**Analýza optického rozpoznávání znaků**
Cíl práce:	Cílem práce je provést a představit vyhodnocení úspěšnosti optického rozpoznávání znaků pomocí OCR systému Tesseract OCR a to v prostředí .NET/C#.

Ing. Marek Musil
vedoucí bakalářské práce

doc. Ing. Zdeněk Horák, Ph.D.
vedoucí katedry
Katedra technických studií

Abstrakt

Tato bakalářská práce se zabývá problematikou optického rozpoznávání znaků (OCR), zejména se zabývá analýzou nástrojů umožňující rozpoznávání textu. Jsou představeny možnosti těchto nástrojů a jejich efektivita – přesnost rozpoznávání. Práce se zaměřuje především na nástroj Tesseract OCR a knihovnu IronOCR jazyka C#. Jsou představeny výsledky rozpoznávání různých formátů vstupních dat a diskutována účinnost rozpoznání pomocí nástroje Tesseract OCR.

Klíčová slova

.NET , Analýza, C#, IronOCR, OCR, Tesseract, Tessnet

Abstract

This bachelor thesis deals with the problem of optical character recognition (OCR), especially with the analysis of tools for text recognition. The possibilities of these tools and their efficiency - accuracy of recognition are introduced. The thesis focuses mainly on the Tesseract OCR tool and the IronOCR library for the C # language. The results of recognition of various input data formats are presented and the efficiency of recognition using Tesseract OCR is discussed.

Key words

.NET , Analysis, C#, IronOCR, OCR, Tesseract, Tessnet

Prohlašuji, že předložená bakalářská práce je původní a zpracoval/a jsem ji samostatně. Prohlašuji, že citace použitých pramenů je úplná, že jsem v práci neporušil/a autorská práva (ve smyslu zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů, v platném znění, dále též „**AZ**“).

Souhlasím s umístěním bakalářské práce v knihovně VŠPJ a s jejím užitím k výuce nebo k vlastní vnitřní potřebě VŠPJ.

Byl/a jsem seznámen/a s tím, že na mou mé bakalářskou práci se plně vztahuje **AZ**, zejména § 60 (školní dílo).

Beru na vědomí, že VŠPJ má právo na uzavření licenční smlouvy o užití mé bakalářské práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé bakalářské práce (prodej, zapůjčení apod.).

Jsem si vědom/a toho, že užít své bakalářské práce či poskytnout licenci k jejímu využití mohu jen se souhlasem VŠPJ, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených vysokou školou na vytvoření díla (až do jejich skutečné výše), z výdělku dosaženého v souvislosti s užitím díla či poskytnutím licence.

V Jihlavě dne 17. prosince 2019

…….

Podpis studenta

Poděkování

Rád bych poděkoval svému vedoucímu bakalářské práce panu Ing. Marku Musilovi za cenné rady a všeobecnou pomoc při vytvoření této práce. A také bych chtěl poděkovat své rodině, mé přítelkyni a kamarádům za jejich dlouhodobou podporu a pomoc při vytvoření této bakalářské práce.

Obsah

Seznam obrázků

Seznam tabulek

Seznam použitých zkratek

AI	Artificial Intelligence
CBN	Clean Background Noise
DPI	Dots per inch
EC	Enhance Contrast
ER	Enhance Resolution
GUI	Graphical User Interface
HD	High Definition
MFP	Multi-function printer
OCR	Optical Character Recognition
PDF	Portable Document Format
PNG	Portable Network Graphics
RS	Rotate and Straighten
SDK	Software Development Kit
SPZ	Statní poznávací značka
TFA	Topological Feature Analysis
TIFF	Tagged Image File Format
UX	User Experience
WTBB	White Text on Black Background
XML	Extensible Markup Language

Úvod

Optické rozpoznávání znaků nebo **OCR** (z angl. *Optical Character Recognition*) je elektronická nebo mechanická konverze obrázků rukopisného nebo tišteného textu do strojově kódovaného textu, ať už ze skenovaného dokumentu, fotografie dokumentu, fotografie scény (například text na tabulích nebo billboardech) nebo z titulků překrývajících obraz (například z televizního vysílání). OCR se často používá jako forma zadávání informací načtením z tištěných dat – zde se jedná například o faktury, pasové doklady, smlouvy, potvrzení, bankovní výpisy nebo jakoukoliv vhodnou dokumentaci. Téměř všechno lze konvertovat do digitálního textu, který pak lze elektronicky editovat, prohledávat, ukládat kompaktněji nebo zobrazovat on-line. Mohu být použity v IT procesech, mezi které patří například kognitivní výpočty, text-to-speech, strojový překlad atd.

Počítačový program (OCR Software) převádí obraz automaticky, pokud není určeno jinak. Převedený text je vždy závislý na kvalitě skenovaného dokumentu a skeneru. Z tohoto důvodu **nemůže OCR program vždy rozpoznat všechna písmena správně**. (Arindam Chaudhuri, 2017) Rozpoznání klasického latinského písma stále není 100 % přesné, i když je k dispozici obrázek v perfektní kvalitě. Studie založena na rozpoznání novinových stránek z 19. a 20. století prokázala, že přesnost komerčních OCR softwarů je od **81 % do 99 %** (HOLLEY, 2009). Starší verze softwarů bylo třeba naučit pomocí obrázku každého písmena a rozpoznávání bylo pak prováděno po jednotlivých písmenech (symbolech).

Novější systémy jsou schopné produkovat vysoký stupeň přesnosti rozpoznávání a navíc rozlišovat víc než 150 jazyků a většinu písmen. Některé komerční produkty jsou schopné navíc reprodukovat nejen text, ale i obrázky, formátování, tabulky a ostatní netextové elementy. Ale i s novými funkcemi a možnostmi nejsou dnešní nástroje 100 % přesné, proto při rozpoznávání důležitých dokumentů (faktur, smluv atd.), kde se vyžaduje výjimečná přesnost, musí být výsledek dodatečně kontrolován člověkem.

Cíl práce

Cílem dané práce je provést analýzu a představit vyhodnocení úspešností optického rozpoznávání znaků pomocí OCR systému Tesseract v prostředí .NET/C#. Vzhledem k tomu, že systém Tesseract je napsán v programovacím jazyce C++ a tato práce se zaměřuje na .NET/C#, v průběhu jsem si vybral nástroj napsaný v programovacím jazyce C# – knihovnu IronOCR, která je založena na jádře Tesseract a přímo používá její metody.

V rámci práce bude řešeno a zjišťováno, do jaké míry tento nástroj pro optické rozpoznávání znaků může a nemůže rozpoznat vstupní data a jakou přesnost bude mít výstupní text vzhledem k vstupním datům. Hlavním cílem je ukázat jakou přesnost mají dnešní nástroje pro optické rozpoznávání znaků a potvrdit tvrzení o jejich spolehlivosti. Zároveň bych chtěl určit podmínky, při jakých budou mít OCR nástroje nejvyšší a nejnižší přesnost rozpoznávání.

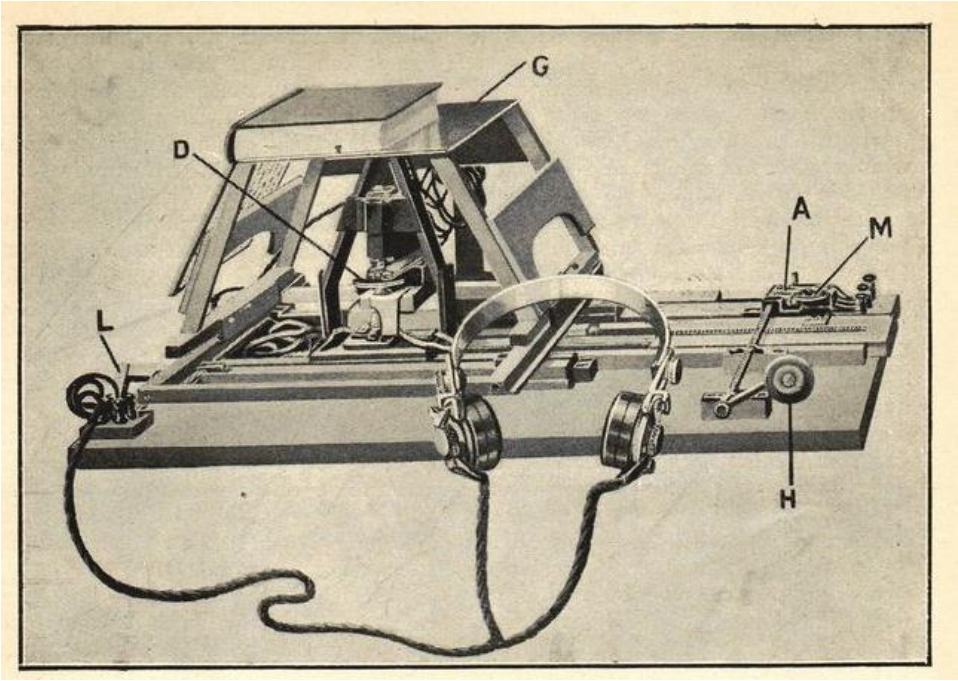
1 Vývoj OCR

V této kapitole je popsán historický přehled vývoje systémů pro optické rozpoznávání znaků, současný stav dnešních nástrojů a jejich použití.

1.1 Historie vývoje OCR

Problematikou optického rozpoznávání znaků se vědci začali zabývat již v 19. století. Prvním příkladem OCR se považuje skener sítnice – systém přenosu obrazu využívající mozaiku fotobuněk, který vynalezl americký vynálezce Charles R. Carey v roce 1870.

V roce 1912 irský fyzik, astrofyzik a chemik Dr. Edmund Fournier d'Albe vynalezl optofon – zařízení používané nevidomými, které skenuje text a generuje tóny pro identifikaci písmen. (Schantz, 1982)



Obrázek 1 – Pohled na optofon *(Vetenskapen och livet, 1922)*

Mezi roky 1931 a 1954 byly vynalezeny a aplikovány první OCR nástroje v průmyslu. Nástroje už byly schopné interpretovat Morseův kód a nahlas číst text. Intelligent Machines Research Corporation byla první společnost, která začala prodávat tyto nástroje (Norman, 2019). V roce 1962 americký profesor John G. Linvill vynalezl Optacon (Optical to Tactile Conveter) – přenosný elektromechanický přístroj, který pomáhal nevidomým číst tištěné texty. (Schantz, 1982)

Obrázek 2 – Optacon (Zdroj: Pietro Dipalma)

V roce 1974 americký vynálezce Ray Kurzweil založil společnost „Kurzweil Computer Inc.“, která vyvinula první OCR nástroj pro rozpoznání téměř jakéhokoliv tištěného písma. Později v roce 1978 společnost začala prodávat komerční verze svého OCR softwaru – čtecí stroj Kurzweila. Od této doby byly OCR nástroje často používány pro rozpoznání cenovek a cestovních pasů. (Hauger, 1995)

V roce 1989 David Yang založil “ABBYY“ – nejznámější společnost, která vyvíjí komerční software pro optické rozpoznávání znaků.

V roce 2000 byla technologie OCR adaptována jako webová služba – WebOCR. Toto umožnilo používat OCR na jakémkoliv vhodném zařízení s přístupem na internet. Rozpoznání se provádí na straně serveru a uživatel dostane výsledek v textovém formátu.

1.2 Současný stav

Nástroje pro optické rozpoznávání znaků jsou velmi náročné na používaný hardware, z tohoto důvodu na konci 20. století spolu s růstem výpočetních možností počítačů začal také vývoj komplikovanějších algoritmů pro provedení OCR. Současné nástroje používají nejen lineární algoritmy, ale například umělou inteligenci (angl. Artificial Intelligence), strojové učení a neuronové sítě. Tyto technologie umožňují nejen zlepšit celkovou přesnost výstupního textů, ale zároveň se v průběhu „učit“. Důležitým faktorem bylo

zlepšení zařízení pro skenovaní dokumentů, které umožňovalo získávat obrázky v mnohem lepší kvalitě a rozlišení. (Milan Sonka, 2014)

1.3 Použití OCR

Pravděpodobně nejznámějším případem použití technologie OCR je zpracovávání tištěných papírových dokumentů do elektronické podoby. Výsledek rozpoznání skenovaného papírového dokumentu lze upravovat v libovolném textovém editoru, například Microsoft Word, LibreOffice, Google Docs, Notepad++ atd. V minulosti, když technologie OCR ještě neexistovala, jedinou možností digitalizace tištěných dokumentů bylo ruční přepisování textu, což bylo nejen časově náročné, ale často docházelo k překlepům.

OCR je často používán jako "skrytá" technologie, která podporuje mnoho známých systémů a služeb v našem každodenním životě. Méně známá, ale v použití stejně důležitá technologie OCR zahrnuje automatizaci zadávání dat, indexování dokumentů pro vyhledávače, automatické rozpoznání poznávacích značek a také pomoc nevidomým a zrakově postiženým osobám.

OCR slouží jako nástroj pro širokou škálu poštovních třídících aplikací. Používá se nejen k dekódování celé škály domácích adres, ale také k vyřešení náročných problémů, jako je přesměrování pošty a zpracování neúplných adres. Všechny tyto aplikace přesahují pouhé rozpoznávání adres a provádějí širší škálu úkolů, včetně sofistikovanější analýzy dat.
Při provádění přesměrování pošty, tato technologie čte a porovnává jména osob / firemních adresátů obsažených v adresním poli se jmény obsaženými v databázi adres. Tím vyhledá a přečte zpáteční adresu. Tyto technologie se používají zejména na poštách v USA.

Technologie OCR se ukázala jako nesmírně užitečná při digitalizaci historických novin a textů. Ty byly převedeny na plně vyhledávatelé formáty a byl usnadněn přístup k těmto textům. Níže je uvedeno několik konkrétních příkladů, kde se OCR používá.

Digitalizace knih

Digitalizace knih je proces, při kterém se text z knihy převádí do digitálního (elektronického) textu. Digitální knihy lze snadno distribuovat, reprodukovat a číst na obrazovce.

Obrázek 3 – Příklad digitalizace knihy *(Dvortygirl, 2008)*

K převedení naskenované knihy do textového formátu se používá optické rozpoznávání znaků. Umožňuje převést stránky knihy do formátu digitálního textu: formát ASCII nebo jiný podobný formát. Tím je snížena velikost souboru, a především je tím umožněno text přeformátovat, vyhledávat v něm, nebo zpracovávat jinými aplikacemi. Použití této technologie je velmi přínosné, protože umožnuje rychle převádět tištěný text do elektronické podoby.

Čtení registračních značek

Už jste někdy dostali oznámení o překročení rychlosti? Pokud ne, tak jste vzorný řidič. Pokud ano, tak asi víte, jak vypadá toto oznámení:

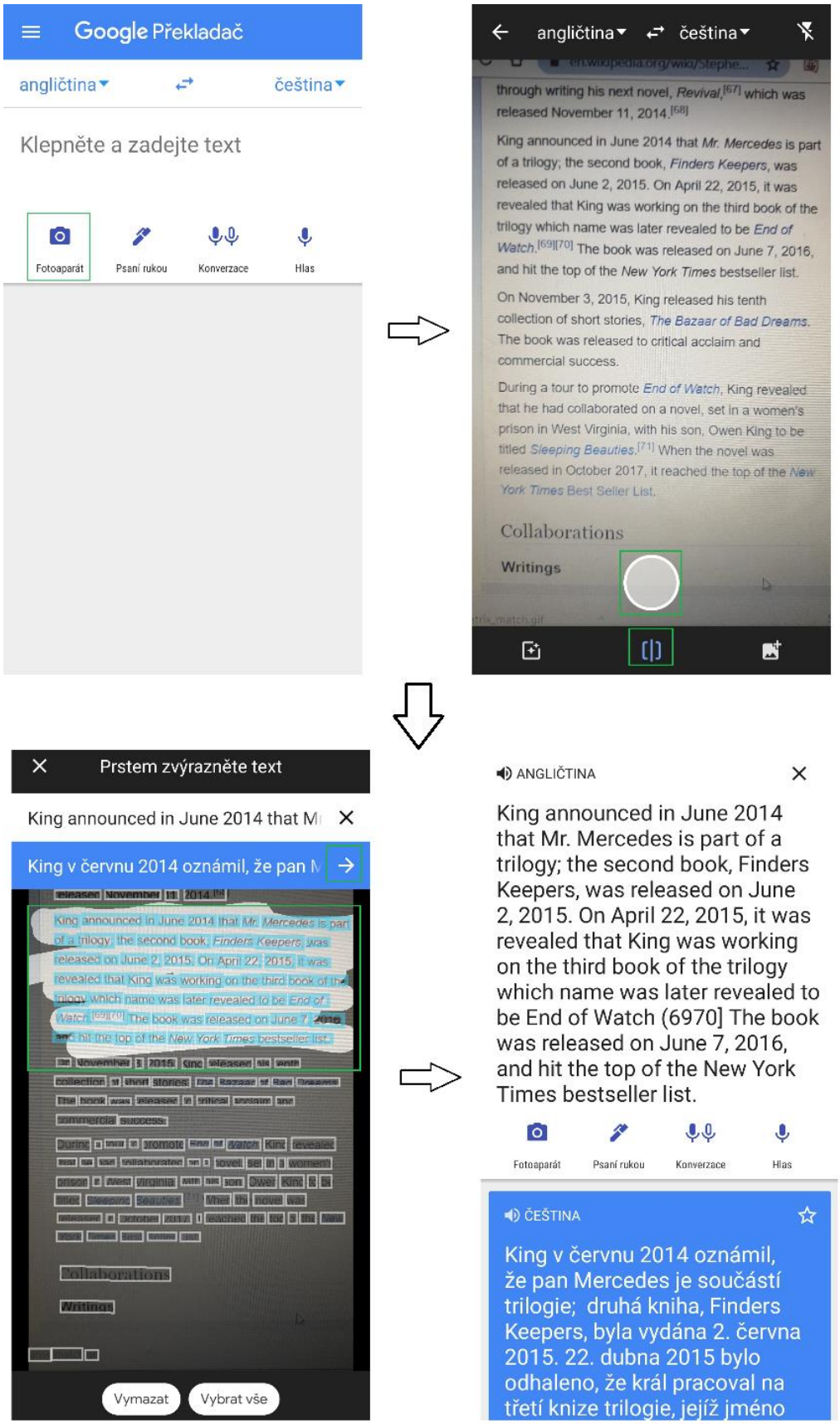
Obrázek 4 – Rozpoznávání registračních značek *(DOPRAVNÍ PŘESTUPKY, 2006)*

Jak systém pozná, komu připsat pokutu? Správná odpověď – pomocí OCR a rozpoznání Státní poznávací značky (SPZ). Radar zkontroluje rychlost auta a v případě překročení povolené rychlosti vyfotí kamera auto. Pořízený snímek je odeslán do centrálního systému. V centrálního systému je pak ze snímku rozpoznána SPZ.

Na levé straně obrázku 6 můžeme vidět příklady jednotlivých SPZ, které byly rozpoznány na autech (systém si je „vyříznul" z původní fotografie, která je uvedena na pravé straně obrázku). Metody, pomocí kterých si systém dokáže vybrat z obrázku jenom potřebná data, jsou popsané v kapitole 4.

Google Translate

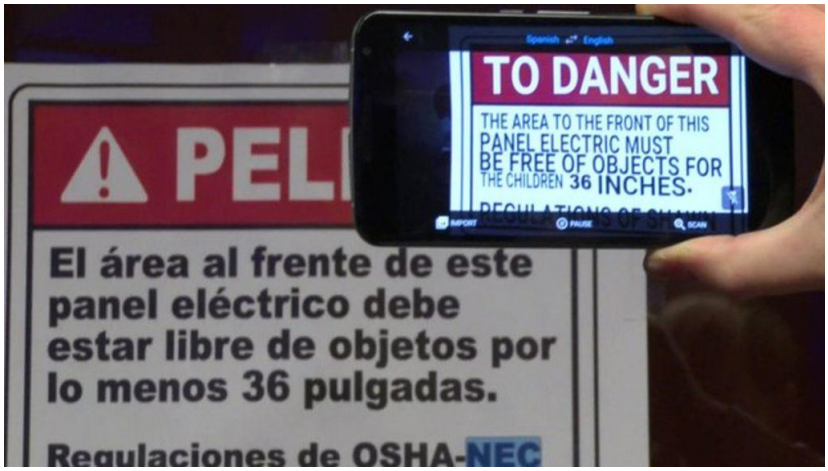
Google Translate je aplikace pro OS Android a je považována za velmi účinnou. Tato aplikace dokáže nejen přeložit jakýkoliv vstupní text do víc než 100 jazyků, ale i rozpoznat, přeložit a adaptovat obrázky zaznamenané kamerou mobilního zařízení v reálném čase (real-time).



Obrázek 5 - OCR pomocí aplikace Google Translate

Lze použít aplikaci jako OCR čtečku a zároveň přeložit načtený text. V aplikaci Google Translate se nejdříve vybere jazyk ručně nebo se nechá detekovat aplikací. Pro pořízený snímek textu, můžeme přeložit buď celý vyfocený text, nebo si vybrat potřebný kus textu.

Příkladem použiti této aplikace je například čtení a přeložení značek, jak je uvedeno v následujícím obrázku, nebo se může používat pro digitalizaci a čtení knih, časopisů v cizím jazyce.



Obrázek 6 – Přeložení textu pomocí Google Translate *(BBC, 2015)*

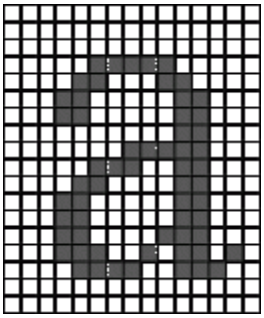
2 Princip OCR

V této kapitole bude popsán princip, na kterém funguje optické rozpoznávání znaků a také popsány metody, které se používají při zpracovaní informací.

2.1 Metody rozpoznávání znaků

Existují dvě základní metody optického rozpoznávání znaků: Matrix matching a Feature extraction. (Data ID, 2018)

Metoda Matrix Matching zahrnuje porovnávání vstupního obrázku s uloženým glyfem[1] „Pixel-by-pixel“ (to znamená, že software projde každý jednotlivý pixel vstupního obrázku a porovnává ho s uloženým glyfem). Tato metoda je také známá jako “pattern matching”, “pattern recognition” nebo “image correlation”. Metoda se opírá o to, že vstupní glyf je správně izolován od zbytku obrázku a uložený glyf je v podobném písmu a ve stejném měřítku.

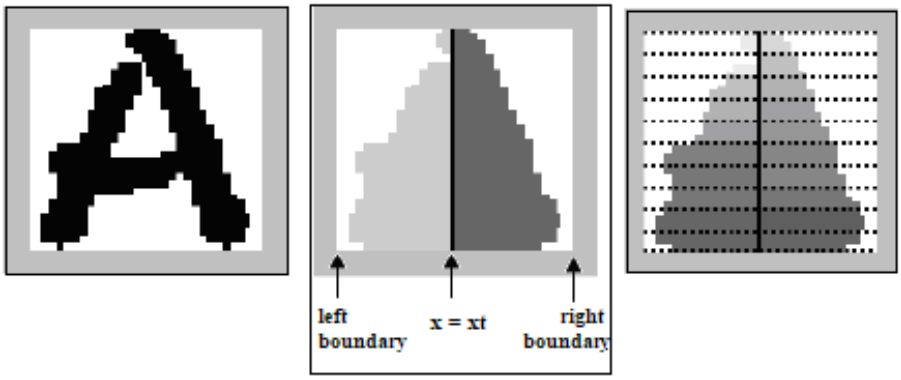


Obrázek 7 – Matrix Matching *(PCTechGuide, 2019)*

Metoda Feature Extraction je OCR bez nutnosti přesné shody s předepsanými šablonami. Také známý jako Intelligent Character Recognition (ICR) nebo Topological Feature Analysis (TFA). Tato metoda se liší podle toho, kolik "počítačové inteligence" výrobce používá. Počítač vyhledává obecné vlastnosti, jako jsou otevřené oblasti, uzavřené tvary, diagonální čáry, průsečíky atd. Tato metoda je mnohem univerzálnější než Matrix matching. Matrix matching funguje nejlépe, když se OCR software setká s omezeným repertoárem typových stylů, s malými nebo žádnými variacemi v rámci každého stylu. Pokud jsou znaky méně předvídatelné, funkční nebo topografická analýza je vhodnější metodou. Na následujícím obrázku je vidět jednotlivé kroky OCR softwaru. Na levé

[1] **Glyf** (ze starořeckého γλυφή – glyfein – tesat) je v lingvistice a typografii grafická realizace podoby grafému (písmena, číslice, znaku, piktogramu, interpunkčního a jiných znamének).

straně je normalizovaný obrázek. Uprostřed je rozdělení znaku na levou a pravou stranu. Na pravé straně jsou tzv. extracted features. Tmavší čtverce označují vyšší hustotu pixelů. Na základě takového rozložení software pozná, o jaký znak se jedná. (Muhammad'Arif Mohamad, 2015)



Obrázek 8 – Ukázka feature extraction

2.2 Metody zpracovávání

Ve většině případů není vstupní obrázek v dobré kvalitě, správně vyrovnán (má natočení textu vzhledem ke stránce) nebo není vůbec čitelný. Proto OCR software často předzpracuje vstupní obrázky, aby zvýšil šance úspěšného rozpoznání. (Wilhelm Burger, 2009). Některé z těchto metod jsou popsané níže.

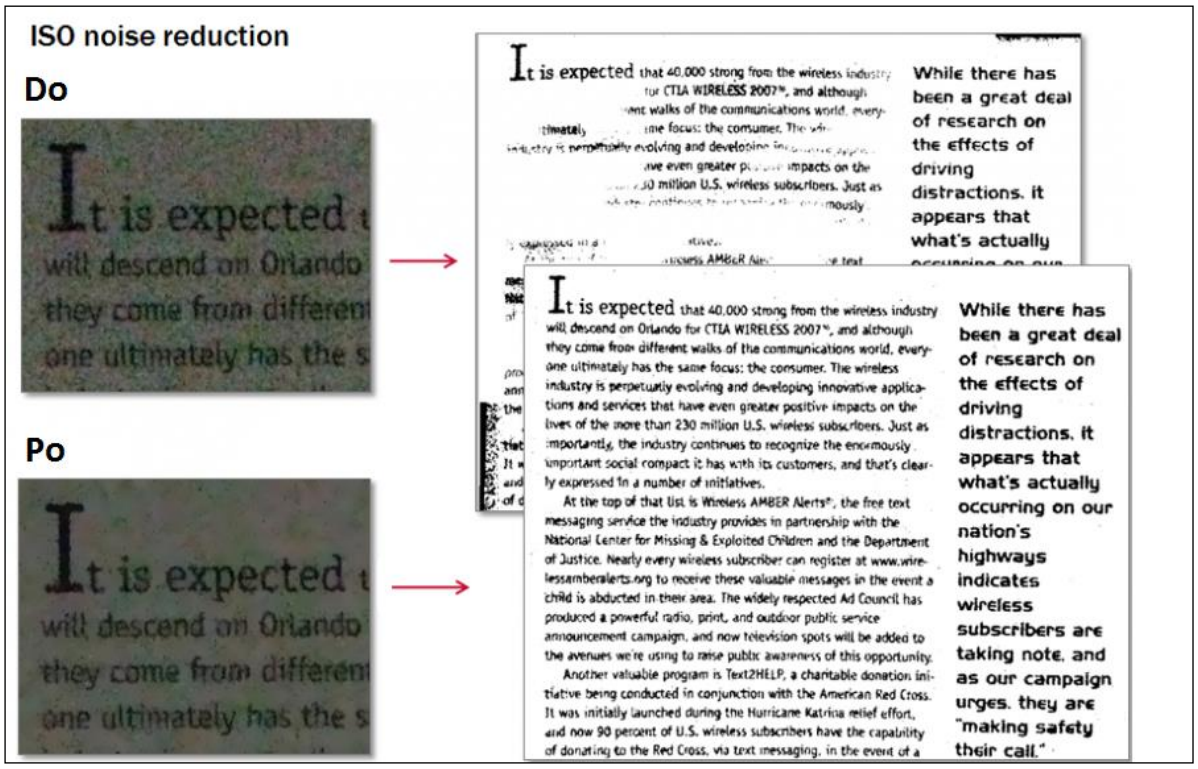
Pre-processing

Předzpracovaní (pre-processing) – proces, při kterém se vstupní data upravují za účelem snadnějšího rozpoznávání. Hlavním cílem je oddělit textovou část vstupního obrázku od nerelevantních informací. Nejčastěji používané metody jsou normalizace vstupních dat, redukce šumu a binarizace obrázku (Milan Sonka, 2014). Popis některých metod pro předzpracovaní je uveden dále.

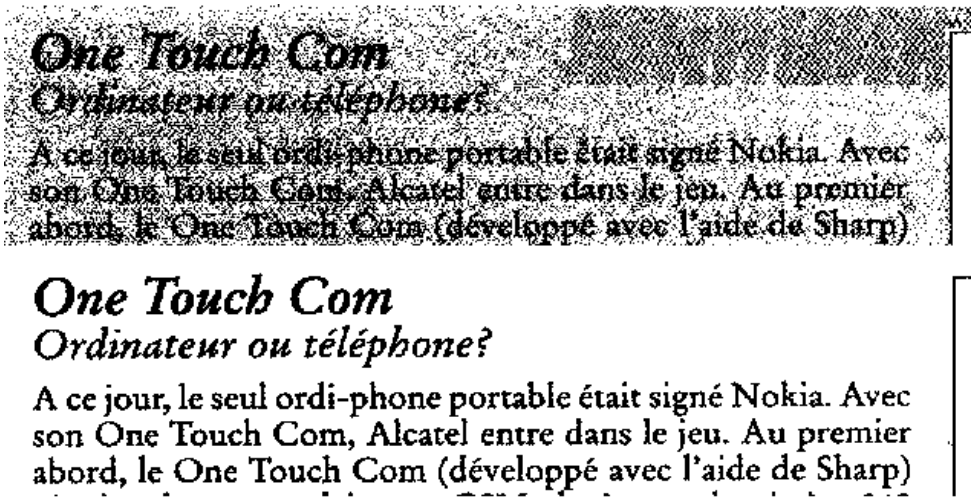
Metody předzpracování:

- **Otáčení** (De-skew) – v případě, že byl dokument špatně zarovnán při skenovaní nebo focení, může být otočen o pár stupňů.
- **Redukce šumu –** odstranění pozitivních a negativních skvrn, vyhlazení hran.
- **Odstranění řádků** – mazaní nepotřebných řádků. (Grooper, 2018)
- **Analýza rozložení (Layout analyse)** – označuje sloupce, odstavce, popisky atd. jako samostatné bloky. Tato metoda je důležitá při práci s tabulkami.

- **Normalizace** poměru stran a stupnice. (Damilies, 2018)
- **Binarizace** – konverze barevného nebo šedého obrázku do černo-bílého (tzv. "binární obrázek" - protože obsahuje pouze dvě barvy). Nejprve je obrázek převeden do stupňů šedi, poté se aplikuje práh (threshold). Cílem je oddělit text od pozadí. (TRIER, et al., 1995)



Obrázek 9 – Příklad redukcí šumu



Obrázek 10 – Příklad binarizace

Post-processing

I když dnes software pro optické rozpoznávání znaků dosahuje vysokého stupně přesností, je to stále daleko od dokonalosti. Proto přesnost OCR může být navýšena, pokud bude výstupní text omezen lexikonem – seznamem povolených slov. Mohou to být například všechna slova v českém/anglickém jazyce, nebo technický lexikon pro určitý obor, např. pro medicínu. Problém může nastat, jestliže se v dokumentu začnou vyskytovat slova, které nejsou v lexikonu, jako například vlastní jména. Tato metoda se nazývá lexikální korekce výstupu. Existuje další metoda – kontextová korekce. Ve chvíli, kdy lexikální korekce používá slovník pro porovnání slov, kontextová korekce využívá syntaktickou a sémantickou analýzu jazyka. Kontextová korekce určuje pravděpodobnost výskytu slov ve výstupním textu. (GISGeography, 2015)

Výstupem může být prostý text, textový nebo znakový soubor. Novější komerční OCR systémy mohou zároveň zachovávat originální rozložení (layout) dokumentů včetně pozic textů, obrázků, tabulek atd. a ukládat data například do souboru XML.

Software pro optické rozpoznávání znaků Tesseract**,** o kterém budeme mluvit v další kapitole**,** využívá svůj slovník k ovlivnění pořadí segmentací znaků za účelem zlepšení přesnosti rozpoznávání.

3 Software pro OCR

V této kapitole je uveden seznam aktuálních komerčních a nekomerčních řešení pro optické rozpoznávání znaků.

3.1 Komerční řešení

V dnešní době na trhu existuje mnoho komerčních řešení pro OCR. Obecně tyto řešení poskytují mnohem lepší uživatelskou zkušenost (user experience) než jejich nekomerční alternativy. Nabízejí intuitivní uživatelské rozhraní (GUI), podporu velké sady jazyků a také pokročilé metody zpracování výstupních informací.

ABBY FineReader

ABBYY FineReader je komerční aplikace pro optické rozpoznávání znaků vyvinutá společností ABBYY. První verze byla představena již v roce 1993.

Program umožňuje převod obrazových souborů (fotografií, dokumentů, PDF) do upravitelných elektronických formátů. Zejména se jedná o soubory s textovými formáty .DOC, .DOCX, .XLS, .XLSX, .PPTX, .RTF, .PDF, .HTML, .CSV, .TXT, .ODT, .DJVU, .EPUB, .FB2. Poslední verze aplikace (č. 15) podporuje rozpoznávání textů ve 192 jazycích a má vestavěnou kontrolu pravopisu pro 48 z nich. ABBYY dodává sadu SDK pro zabudované a mobilní zařízení, a to v programovacích jazycích C/C++, Visual Basic, .NET, Delphi, Java.

OmniPage

OmniPage je komerční aplikace pro optické rozpoznávání znaků vyvinutá společnosti Kofax Incorporated.

OmniPage byl jeden z prvních OCR programů spustitelných na osobních počítačích. Byl vyvinut na konci osmdesátých let a poté prodán společnosti Caere Corporation.

Program umožňuje převod obrazových dokumentů do formátů .DOC, .DOCX, .XLS, .XLSX, .PPTX, .RTF, .PDF, .PDF/A, .HTML, .XML, .EPUB, .MP3. Podporuje víc něž 120 jazyků a umí pracovat s tištěným a psaným písmem. Společnost dodává SDK pro programovací jazyky C/C++ a C#.

3.2 Nekomerční řešení

Naopak od komerčních řešení nekomerční mají mnohem méně možností, většinou tyto řešení nemají žádné uživatelské rozhraní, podporují jenom latinské písmo a mají nízkou přesnost. Zároveň neobsahují metody pro pre-processing a post-processing, díky tomuto musí uživatelé dodatečně upravovat vstup a výstup.

Google Drive OCR

Google Drive je služba ukládání a synchronizace souborů vyvinutá společností Google. Od roku 2015 mají uživatelé, kteří používají uložiště možnost konvertovat text v souborech ve formátu .JPEG, .PNG, .GIF nebo .PDF do formátu Google Docs. Není známo, kolik přesně jazyků umí rozpoznávat Google Drive OCR, ale předpokládá se, že OCR používá jádro Tesseract, to znamená, že by teoreticky měl umět rozpoznávat více než 100 jazyků.

Tesseract

Tesseract je nejznámější Optical Character Recognition engine[2] který je adaptován pro různé operační systémy. Jedná se o open-source software pod Apache Licence verze 2.0, vývoj který financuje společnost Google od roku 2006. Ve stejném roce byl Tesseract považován jedním z nejpřesnějších open-source OCR softwaru k dispozici.

Tesseract byl původně vyvinut jako proprietární software ve společnosti Hewlett Packard v Bristolu, Anglie a Greelej, Colorado mezi roky 1985 a 1994. V roce 1996 s některými dalšími změnami byl adaptován pro OS Windows. V roce 1998 byla většina kódu převedena z programovacího jazyka C do programovacího jazyka C++, část kódu je stále v C, ale je aktualizovaná pro kompilace C++ kompilátorem. V současnosti je v C++ napsáno 91,5 % kódu a 5.6 % v C. (Tesseract, 2018). Tesseract byl jedním z TOP 3 nejlepších OCR Engine z hlediska přesnosti rozpoznání znaků v roce 1995. (Ubuntu, 2016).

Zdrojový kód Tesseract je dostupný ke stažení na stránkách GitHub: https://github.com/tesseract-ocr/tesseract (10. 12. 2019). Nástroj je dostupný pro operační systémy Linux, Windows a MacOS.

[2] **Engine** – je v informatice označení pro jádro, „ústřední“ či „kritickou“ část softwarového produktu, nenabízí uživatelské rozhraní.

Obrázek 11 – Logo Tesseract OCR od Google *(Google, 2015)*

Tesseract až do verze 2.0 mohl zpracovávat pouze obrázky ve formátu TIFF a jednoduchý text v jednom sloupci. Tyto starší verze nepodporovaly analýzu rozložení a obrázky nebo rovnice pak způsobily zkomolený výstup. Původní verze mohla rozpoznat jenom anglický text. Tesseract verze 2.0 přidala podporu 6 dalších jazyků (francouzština, italština, němčina, španělština, portugalština, holandština). Verze 3 byla doplněna podporou mnoha dalších jazyků jako například čínština, japonština, arabština a hebrejština.

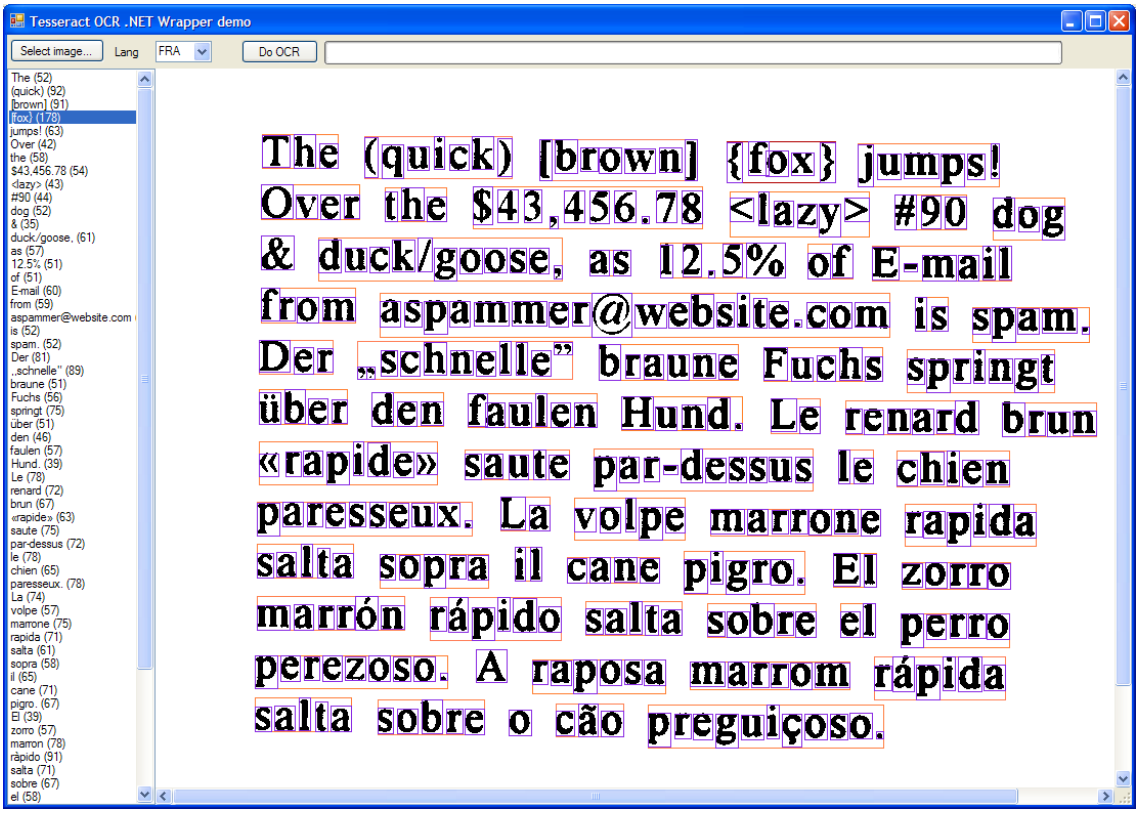
Již od verze 3.0 začal Tesseract podporovat formátování výstupního textu, hOCR (standart reprezentace dat pro formátovaný text obdržený pomocí OCR) a analýzu rozložení stránky. Podpora pro řadu nových formátů obrázků byla přidána pomocí knihovny Leptonica.

Systém Tesseract se považuje za nejlepší nekomerční řešení pro optické rozpoznávání znaků, a to díky tomu, že umí na rozdíl od jiných nekomerčních řešení předzpracovávat vstupní obrázky a ošetřovat výstup rozpoznávání. Aktuální verze podporuje více než 110 jazyků, a téměř každý ho může "naučit" rozpoznávat jiné jazyky (Ubuntu, 2016). Cílem dané bakalářské práce je provést analýzu systému OCR zejména pomocí tohoto systému, ale díky tomu, že tento systém je napsán v programovacím jazyce C++, je vhodné si vybrat systém napsaný v programovacím jazyce C# s jedním důležitým upozorněním – systém má byt založen na jádře Tesseract. Pro tyto účely jsem si zvolil jeho alternativu.

Tessnet2 C#

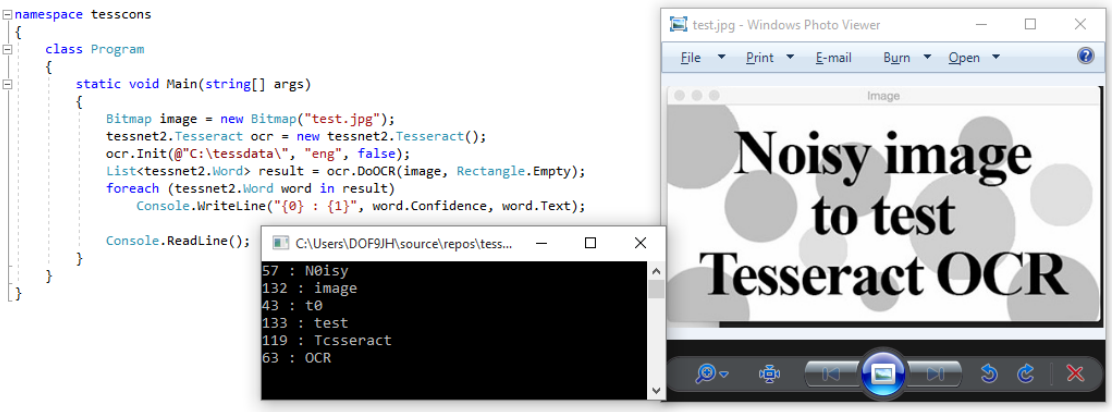
Tessnet2 je .NET knihovna, která používá velmi jednoduché metody pro provedení optického rozpoznávání znaků. Knihovna je založena na jádře Tesseract. (Rémi, 2018).

Tessnet2 je vícevláknový nástroj, ale je omezený ve své funkčnosti, umí pouze základní věci. Neobsahuje dodatečné metody pro předzpracování vstupních obrázků, a proto ho nelze používat pro přesné rozpoznávání. Formát rozpoznání je uveden na **Obrázek *12***.



Obrázek 12 – Ukázková aplikace tessnet2 *(Rémi, 2018)*

Jako příklad jsem připravil jednoduchou aplikaci s využitím knihovny tessnet2 a pokusil se rozpoznat testovací obrázek (**Obrázek *13***). Z výsledku lze vyvodit, že daný nástroj má problémy s rozpoznáváním jednoduchých znaků, jako například "O" a "0" nebo "a" a "e". Z tohoto důvodu nemá cenu tento nástroj nijak důkladněji testovat.



Obrázek 13 – OCR pomocí knihovny tessnet2

IronOCR C#

IronOCR je .NET C# knihovna, založena na Tesseract Engine, vyvinuta spol. IronSoftware v roce 2018. Má bezplatnou licenci pro vývojové účely. Komerční licence začíná od 399 USD. (IronSoftware, 2019)

IronOCR je mnohem účinnější než tessnet2 a většina nekomerčních řešení a navíc nabízí více možností pro předzpracování obrázku.

Možnosti předzpracování obrázku:

- Odstranění šumu z pozadí (Clean Background Noise)
- Zvětšení kontrastu (Contrast Enhance)
- Rotace a vyrovnaní obrázku (Rotate And Straighten)
- Rozlišení vice jazyků v jednom dokumentů (Multi-Language Recognition)
- Čtení čarových kódu (Barcode scanner)
- Specifikace plochy (Area Scan)

IronOCR má 2 režimy rozpoznávání:

- **Automaticky** (AutoOcr) - nástroj sám určí, jaké metody se budou používat pro zpracování obrázku a jaký je jazyk vstupního textu. Avšak nejedná se vždy o nejrychlejší metodu.
- **Manuální** (AdvancedOcr) – nástroj umožňuje vybrat metody, které se budou používat při zpracování obrázku a také vybrat jazyk vstupního textu. Při typizaci vstupu se nástroj může nastavit na rychlejší zpracování vstupních dat, díky tomu se zmenší celkový čas na rozpoznávání.

IronOCR přímo podporuje poslední verze .NET Frameworku, snadně se instaluje a ovládá. S ohledem na to, kolik možností a nastavení má tato knihovna, bezplatnou licenci pro nekomerční vývoj, a také to nejdůležitější, používá se jádro Tesseract – vybral jsem IronOCR jako hlavní nástroj pro analýzu systému optického rozpoznávání znaků. IronOCR je knihovna, proto se pro testovaní musí napojit na existující projekt a pro snadnější ovládaní navrhnout uživatelské rozhraní. Pro tyto účely jsem vyvinul vlastní aplikaci s využitím této knihovny. Postup je popsán v následující kapitole.

4 Vývoj aplikací s využitím knihovny IronOCR

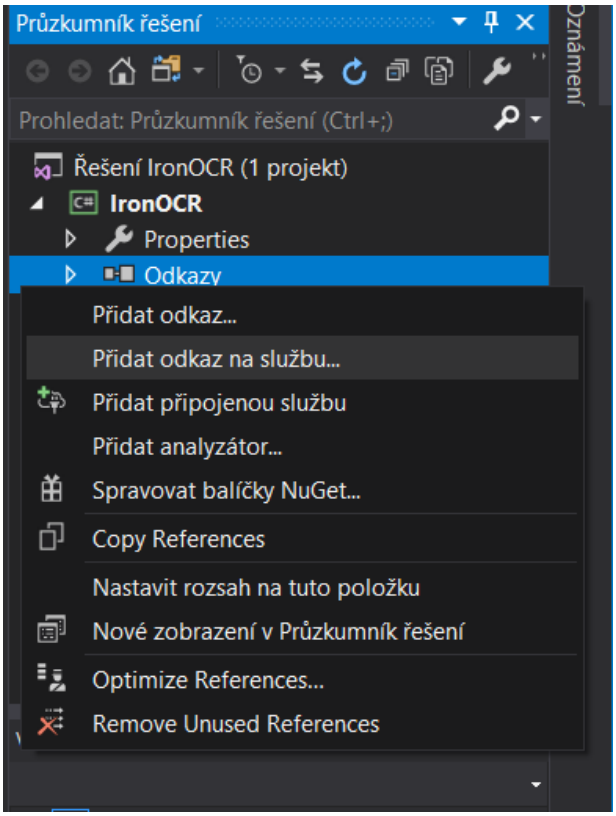
Pro ovládání a testovaní se musí knihovna IronOCR napojit na existující projekt. Pro tyto účely budu vyvíjet vlastní aplikaci. Vývoj se bude provádět v prostředí Microsoft Visual Studio Community 2017.

4.1 Založení projektu a připojení knihovny

Založení projektu v prostředí Microsoft Visual Studio Community je jednoduché **Soubor – Nový – Projekt – Visual C# – Aplikace Windows Forms (.NET Framework)**. Aplikace bude napsaná v jazyce C#, bude se používat .NET Framework verze 4.5.2.

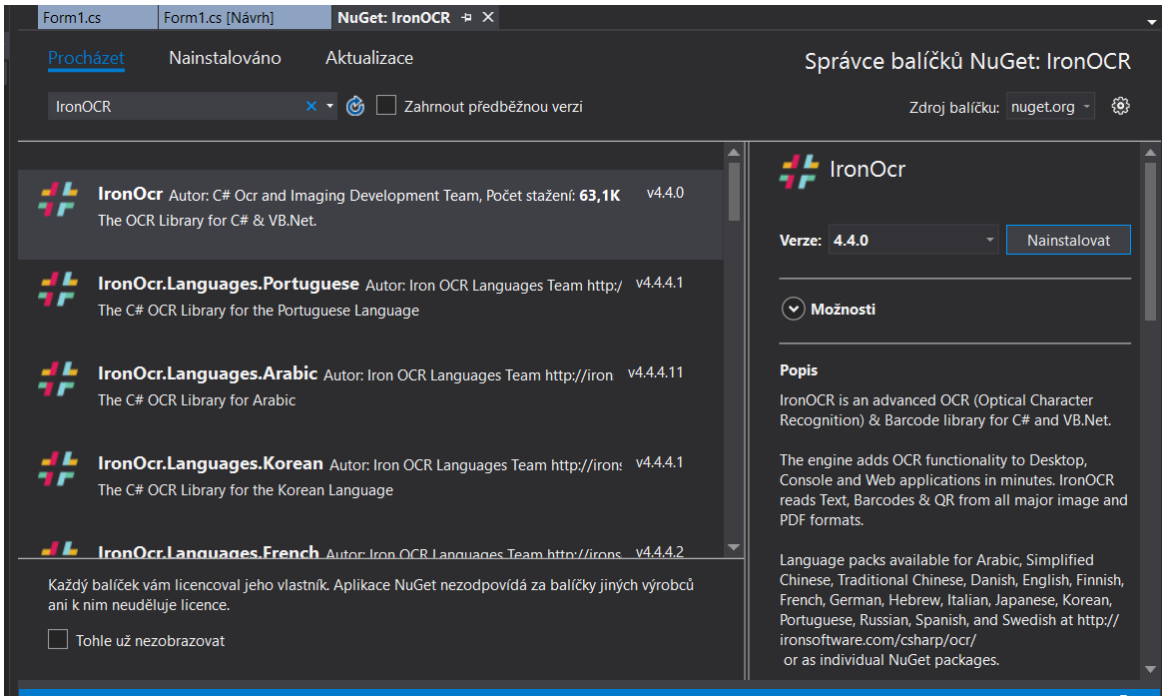
Existují dvě metody připojení knihovny k projektu. První metoda, knihovnu IronOCR si můžeme stáhnout přímo z oficiálních stránek webu společností IronSofware: https://ironsoftware.com/packages/IronOcr.zip (11. 12. 2019)

A poté v projektu přidat odkaz na staženu knihovnu pomocí **Projekt – Odkazy – Přidat odkaz…** .



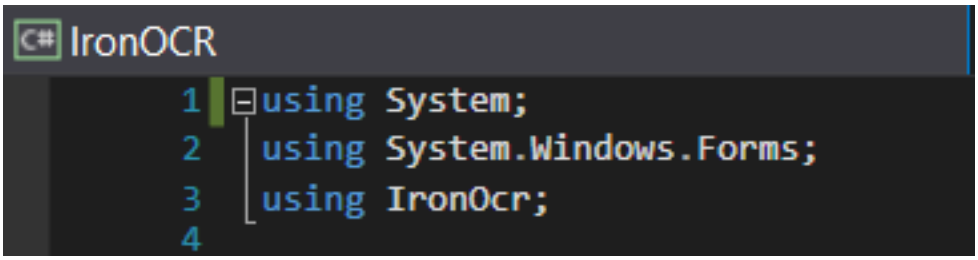
Obrázek 14 – Ukázka přidaní odkazu

Existuje jednodušší metoda pomocí správce balíčků NuGet. Ve správci zadáme do vyhledávacího políčka „IronOCR“ a stiskneme Nainstalovat. Tím si zajistíme vždy tu nejnovější verzi a také možnost rychlé aktualizace komponenty. Zároveň si můžeme stáhnout potřebné jazykové baličky.



Obrázek 15 – Ukázka připojení knihovny pomocí NuGet

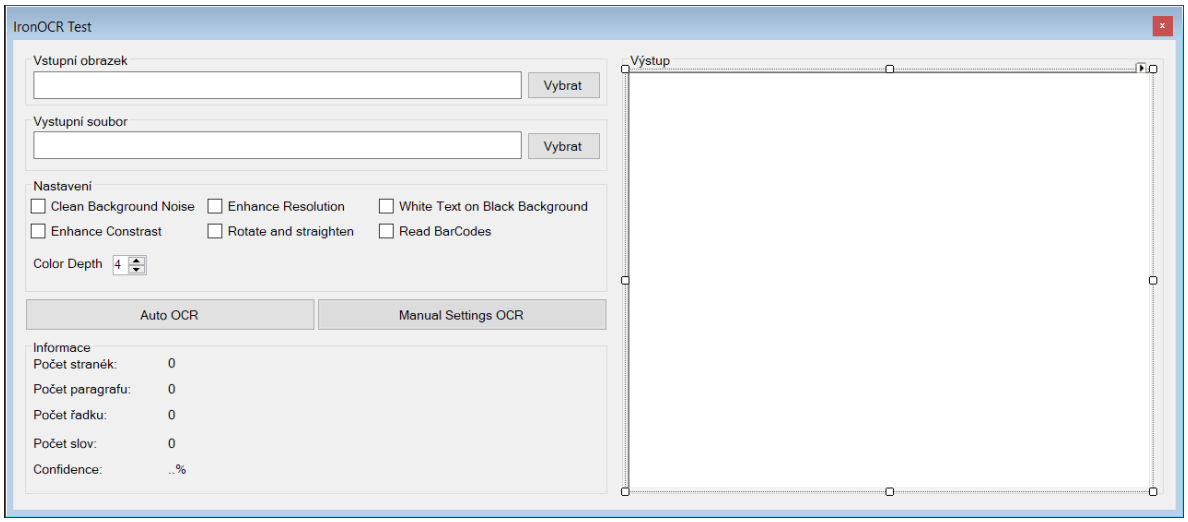
V samostatném projektu pak přidáme referenci na knihovnu:



Obrázek 16 – Reference IronOCR v projektu

4.2 Návrh rozhraní aplikace

Pomocí „Sady Nástrojů“ ve Visual Studiu jsem navrhnul design aplikace



Obrázek 17 – Design aplikací

Do políčka [Vstup] se bude pomocí tlačítka [Vybrat] zadávat vstupní obrázek ve formátech .PNG, .JPG, a .TIFF. Pro manuální režim pomocí zaškrtávacích políček v sekci Nastavení zadáme parametry zpracování vstupního obrázku.

Tlačítka [Auto OCR] a [Manual Settings OCR] spustí samostatné rozpoznávání a poté se výsledek zapíše do textového políčka [Výstup]. Zároveň jsem přidal možnost vypisovat výsledek do textového souboru, který se vybírá pomocí políčka [Výstupní soubor]. Do políčka [Informace] se budou vypisovat veškeré informace o rozpoznaném obrázku.

4.3 Logická vrstva

Nejdříve jsem vytvořil logiku pro tlačítka výběru a uložení souboru. Pomocí vlastnosti *.InitialDirectory* zadáme výchozí složku a pomocí vlastnosti *.Filter* zadáme filtraci souborů. Cestu k vybranému souboru poté zapíšeme do textového políčka Vstup.

Zároveň budeme potřebovat logiku pro uložení souboru s výsledkem rozpoznávání. Výstupný soubor bude ve formátu .txt, to je nastaveno pomocí vlastnosti *.Filter*. Po vybrání souboru se jeho cesta zapíše do políčka Výstup. Ukázky kódu jsou v přílohách A.1 a A.2.

Dalším krokem je logika pro tlačítka OCR. Tlačítko *„Auto OCR“* bude používat metodu *AutoOcr()*, tlačítko *„Manual Settings OCR“* bude používat metodu *AdvancedOcr(parametry)*, pomocí které lze určit parametry pro zpracování vstupního obrázku. Nejprve jsem pro obě metody přidal kontrolu existence souboru, v případě špatného vstupu uživatel dostane hlášku.

Poté jsem přidal samostatné konstruktory pro jednotlivé funkce. Bezparametrický konstruktor *AutoOcr()* provádí automatické rozpoznání vstupního obrázku a parametrický konstruktor *AdvancedOcr(parametry)* umožňuje určit parametry pro zpracování. Tyto parametry jsou získány prostřednictvím zaškrtávacích políček v aplikaci. Zaškrtávací políčka mají vlastnost *.Checked*, která nabývá hodnot *true/false* v závislosti na jejich stavu.

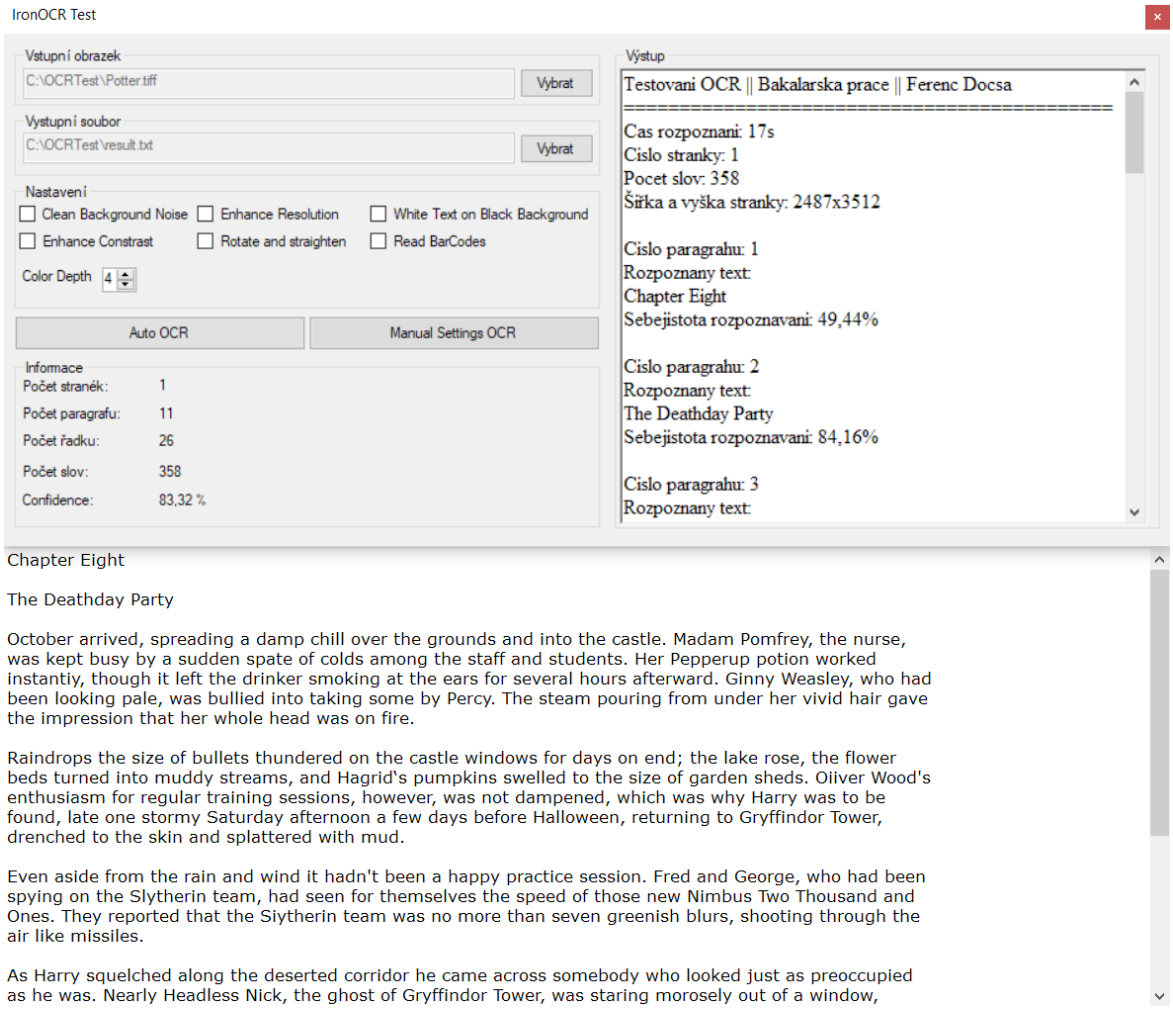
Výsledek rozpoznávání se bude zapisovat do textového souboru. Detailní výsledek rozpoznávání, spolu s dodatečnými informacemi se poté bude zapisovat do políčka *„Výstup“*.

Po rozpoznávání IronOCR eviduje veškeré údaje analyzovaného textu. Celý text se rozdělí do stránek, pak paragrafů, v nichž jsou řádky, ve kterých máme jednotlivá slova. Pro každý z těchto prvků může IronOCR určovat Confidence (sebejistotu) rozpoznávání, což bylo zmíněno v předchozích kapitolách. Navíc nastroj po předzpracování umí rozdělovat paragrafy do jednotlivých obrázků, to je také užitečné při testovaní. Přidáme do kódu výpis této informace. Jednotlivé častí kódu jsou v přílohách A.3 a A.4.

Tím pádem je aplikace připravena pro testovaní. Zkusím provést rozpoznávání většího kusu textu, například text z obrázku v příloze B. Použijeme automatický režim.

Na dalším obrázku je vidět výstup rozpoznávání v textovém souboru a zároveň doplňující informace o rozpoznávání v samostatném softwaru.

Po manuálním přečtení rozpoznaného textu, jsem nenašel ani jednu chybu, avšak aplikace nás upozorňuje, že předpokládaná sebejistota činí 83,32 %. V průběhu testovaní také určíme, jak moc tyto data odpovídají skutečnosti. Je také potřeba vzít v úvahu celkový čas, který aplikace potřebovala na rozpoznávání – 17 s, to je relativně hodně na 358 slov.



Obrázek 18 – Výsledek rozpoznání textu pomocí aplikace IronOCR

4.4 Jazyk rozpoznávání

Jako standardní jazyk rozpoznávání je nastavena angličtina. Pro změnu jazyka se nejprve musí stáhnout jazykový balíček, bud' přes oficiální stránky IronOCR, nebo přes správce balíčků NuGet. Poté u konstruktoru *AdvancedOcr()* stačí změnit parametr *.Language* na potřebný jazyk. Dostupné jazyky pro rozpoznávání jsou: arabština, čínština, čeština, dánština, finština, francouzština, němčina, hebrejština, maďarština, italština, japonština, korejština, norština, polština, portugalština, ruština, španělština, švédština, thajština, turečtina.

5 Postup testování a způsob vyhodnocení testů

V této kapitole se určí všechny podmínky pro testování, včetně metodiky vyhodnocení, vypočtu přesností, výběr přístrojů pro skenovaní a samostatný algoritmus testování.

5.1 Způsob vyhodnocení testů

5.1.1 Vyhodnocení parametrů OCR systému

V průběhu testovaní systému OCR provedeme jeho vyhodnocení z hlediska vlivů různých nastavení skenerů, které zahrnují různé obrazové formáty, rozlišení, typ obrazu atd. Přesnost rozpoznávání znaků (v knihách, časopisech, novinách) a různé nastavení testovacího softwaru IronOCR.

5.1.2 Výpočet přesností

Každý znak, slovo, nebo věta, bud' špatně rozpoznána nebo nerozpoznána vůbec se bude považovat za chybu. Všechny vstupní znaky budou ve formátu Unicode.

Přesnost výstupů budeme počítat pomocí vzorce:

$$Přesnost\ rozpoznání\ znaku\ \% = \left(\frac{N_{znak} - N_{nerozpoznáno}}{N_{znak}}\right) * 100 \quad (1)$$

Kde N_{znak} je počet znaků v textu a $N_{nerozpoznáno}$ je počet nerozpoznaných znaků.

Také budeme používat vzorec pro vypočet přesnosti rozpoznaných slov:

$$Přestnost\ rozpoznání\ slov\ \% = \left(\frac{N_{slov} - N_{nerozpoznáno}}{N_{slov}}\right) * 100 \quad (2)$$

Nástroj IronOCR umí vypočítat Confidence (Sebejistotu) svého rozpoznávání. Zároveň tak máme možnost porovnávat přesnost podle vzorců a porovnávat sebejistotu softwarů.

5.1.3 Výběr testovacích dat

Jako vstupní data pro testování jsem vybral několik článků z veřejných zdrojů, jako například Wikipedia, digitální skripta, časopisy a knihy. Texty jsou napsané v angličtině, češtině
a ruštině. Veškeré testované texty a obrázky jsou uvedeny v *Příloze*. V této bakalářské práci se na ně bude v průběhu odkazovat, jako například *Příloha 1*.

Při provedení analýzy budeme používat níže uvedené parametry:

- **Skenovací přístroj**: skener, kamera mobilního telefonu.
- **DPI**: 75 DPI; 150 DPI; 300 DPI; 600 DPI; 900 DPI.
- **Rozlišení vstupního obrázku**:

  1080p 1920 x 1080; 720p 1280 x 720; 480p 854 x 480; 360p 640 x 360;
  240p 426 x 240;
- **Kvalita a stav vstupního obrázku:**

  šikmost, obrazový šum, kvalita papíru, náklon kamery, vzdálenost od objektu a jiné defekty

5.1.4 Hodnocení výsledků

Hodnotit výsledky rozpoznávání budeme podle následujících parametrů, které se spolu s výsledkem budou zapisovat do tabulky:

Tabulka 1 – Pro zápis výsledků rozpoznávání (ukázková data)

Parametr	**Hodnota**
Stav a kvalita obrázku	Perfektní
DPI	300
Rozlišení obrázku	1280x720
Počet slov	250
Počet znaků	1750
Nastavení IronOCR	Auto
Čas rozpoznání	5 sekund
Sebejistota IronOCR	85 %
Přesnost rozpoznání znaků	90 %
Přesnost rozpoznání slov	96 %
Známka	**B**

Při zápisu použitých pro rozpoznávání nastavení IronOCR do tabulky, budou se používat zkratky metod **CBN** (Clean Background Noise), **RS** (Rotate and Straighten), **EC** (Enhance Contrast), **ER** (Enhance Resolution), **WTBB** (White Text on Black Background).

Podle výsledné přesností rozpoznání znaků se bude aplikovat následující hodnocení výsledků:

Tabulka 2 – Hodnocení výsledků rozpoznávání podle výsledné přesnosti

Výsledná přesnost rozpoznávání	Známka
>95 %	A
90-95 %	B
80-89 %	C
70-79 %	D
60-69 %	E
<60 %	F

5.1.5 Výstupní data

Výstupem rozpoznání bude samostatný rozpoznaný text ve formátu .TXT, takže informace o rozpoznání, jako například čas na rozpoznaní obrázku, počet paragrafů, řádků, slov a sebejistota rozpoznávání obrázku. Tato informace se bude také počítat při vyhodnocení výsledku.

5.1.6 Skenovací přístroje

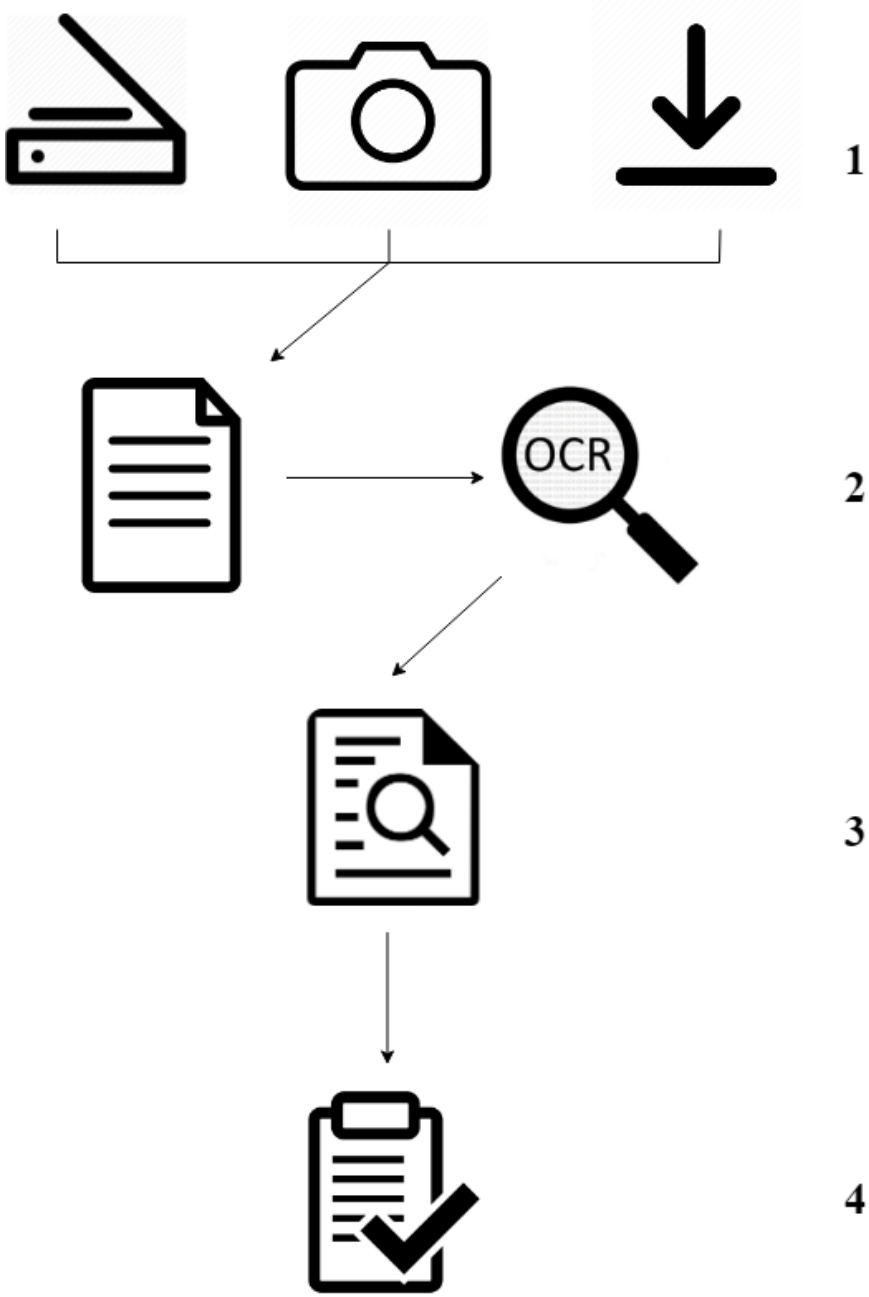
Pro skenovaní bude se používat MFP Canon Pixma E414. Skener umožňuje skenovat obrázek v rozlišení DPI 150 až 700 . Pro skenovaní se bude používat standardní software od výrobce daného zařízení.

Pro porovnání výsledků se také bude používat kamera mobilního telefonu Samsung Galaxy S9. U kamery budeme používat rozlišení 4032 x 2268 a poměr stran 16 : 9.

5.2 Algoritmus testovaní

Testování systému pro optické rozpoznávání znaků se bude provádět podle následujícího algoritmu:

1. Příprava vstupního obrázku (skenování, focení, stažení, ...).
2. Rozpoznání obrázku pomocí IronOCR v automatickém a manuálním režimu.
3. Manuální kontrola výsledku rozpoznávání se spočítáním chyb podle vzorců.
4. Zápis výsledku do ***Tabulka 1*** a evaluace výsledku podle ***Tabulka 2***.



Obrázek 19 – Diagram postupu testovaní

6 Testování

V této kapitole se bude provádět samostatné testování systému pro optické rozpoznávání znaků na klíčových případech. Testování se bude provádět pomocí algoritmu uvedeného v Kapitole 0. Dále budou popsány jednotlivé případy testování, a nakonec celkové zhodnocení dosažených výsledku.

6.1 Případ 1 – perfektně čitelný obrázek

Jako první případ jsem zkusil rozpoznat obrázek ve výborné kvalitě, jedná se o snímek článku z webu Biography.com v HD rozlišení (Příloha C.1). Výsledkem rozpoznání je soubor pripad_1.txt kam program zapsal rozpoznaný text. Vstupní a výstupní soubory jsou umístěny v příloze k této bakalářské práci.

Tabulka 3 – Hodnocení rozpoznávání (Případ 1)

Parametr	**Hodnota**		
Stav a kvalita obrázku	Perfektní kvalita		
DPI	120 DPI	120 DPI	70 DPI
Rozlišení obrázku	1280 x 720	409 x 360	409 x 360
Počet slov	280		
Počet znaků	1653		
Nastavení IronOCR	Auto		
Čas rozpoznání	7 sekund	8 sekund	18 sekund
Sebejistota IronOCR	95.73 %	89.63 %	76.08 %
Přesnost rozpoznání znaků	100 %	99.39 %	94.86 %
Přesnost rozpoznání slov	100 %	97.17 %	91.07 %
Hodnocení	**A**	**A**	**B**

Původní obrázek softwaru se podařilo rozpoznat se 100 % přesností a je to výborný příklad toho, že při perfektním vstupu software pro OCR dosahá téměř 100 % přesnost rozpoznávání. První výsledek bodují za **A**. Poté obrázek jsem upravil, zmenšil rozlišení z 720p do 360p, což už mělo vliv na přesnost rozpoznávání o pár desetin procenta, jsou to jenom drobné překlepy, takže bodují za **A**. Nakonec jsem zmenšil DPI do 70, výsledná přesnost rozpoznávání znaků zmenšila se až o ~5.2 %, v textu se vyskytuje o hodně víc překlepu, každopádně je to výsledek z >90 % přesností, což bodují za **B**.

6.2 Případ 2 – skenovaný dokument

Jako další ukázku jsem zkusil rozpoznat skenovaný dokument (Příloha C.2). Z webu Wikipedia.org jsem si vytisknul a následně naskenoval úryvek textu z biografie Stevena Kinga v anglickém jazyce. Nejprve jsem zkusil automatický režim rozpoznávání (Auto OCR) a už mě čekaly první chyby při rozpoznávání. Následně pomocí manuálních nastavení jsem zkusil rozpoznat obrázek ještě jednou. Výsledek byl znatelně lepší. Při porovnání těchto dvou výsledku jsem si všiml, že software neměl chyby při rozpoznání století, jako například „1900 s“, kde program rozpoznal číslo „5“ místo „s“. Avšak výsledek stále není perfektní, vyskytují se další drobné chyby. Taky bych chtěl upozornit na celkový čas rozpoznávání 34 sekundy na 1 stránku se 649 slovy, což znamená, že software zpracovává ~19 slov za sekundu, to není až tak mnoho, v dnešní době výkonných počítačů. Předpokládám, že delší doba rozpoznávání je způsobena algoritmy pro předzpracovaní obrázku.

Tabulka 4 – Hodnocení rozpoznávání (Případ 2)

Parametr	**Hodnota**		
Stav a kvalita obrázku	Dobrá kvalita		
DPI	400 DPI	400 DPI	200 DPI
Rozlišení obrázku	1920 x 1080	1080 x 720	1080 x 720
Počet slov	649		
Počet znaků	4004		
Nastavení IronOCR	CBN, EC, RS		
Čas rozpoznání	34 sekund	45 sekund	77 sekund
Sebejistota IronOCR	93.31 %	86.77 %	73.47 %
Přesnost rozpoznání znaků	99.07 %	82.63 %	23.06 %
Přesnost rozpoznání slov	98.84 %	73.22 %	12.29 %
Hodnocení	**A**	**C**	**F**

První výsledek bodují za **A**. Po zmenšení rozlišení obrázku z 1080p do 720p, přesnost klesla až o 17 %. Zároveň zvětšil se celkový čas rozpoznávání o 11 sekund, což je způsobeno horší kvalitou obrázku. Tento výsledek bodují za **C**. A naposledy jsem zmenšil DPI z 400 do 200. Téměř softwaru se nepodařilo rozpoznat vstupní obrázek skoro vůbec, přičemž celková doba rozpoznávání stanovila víc než minutu – 77 sekund. Poslední

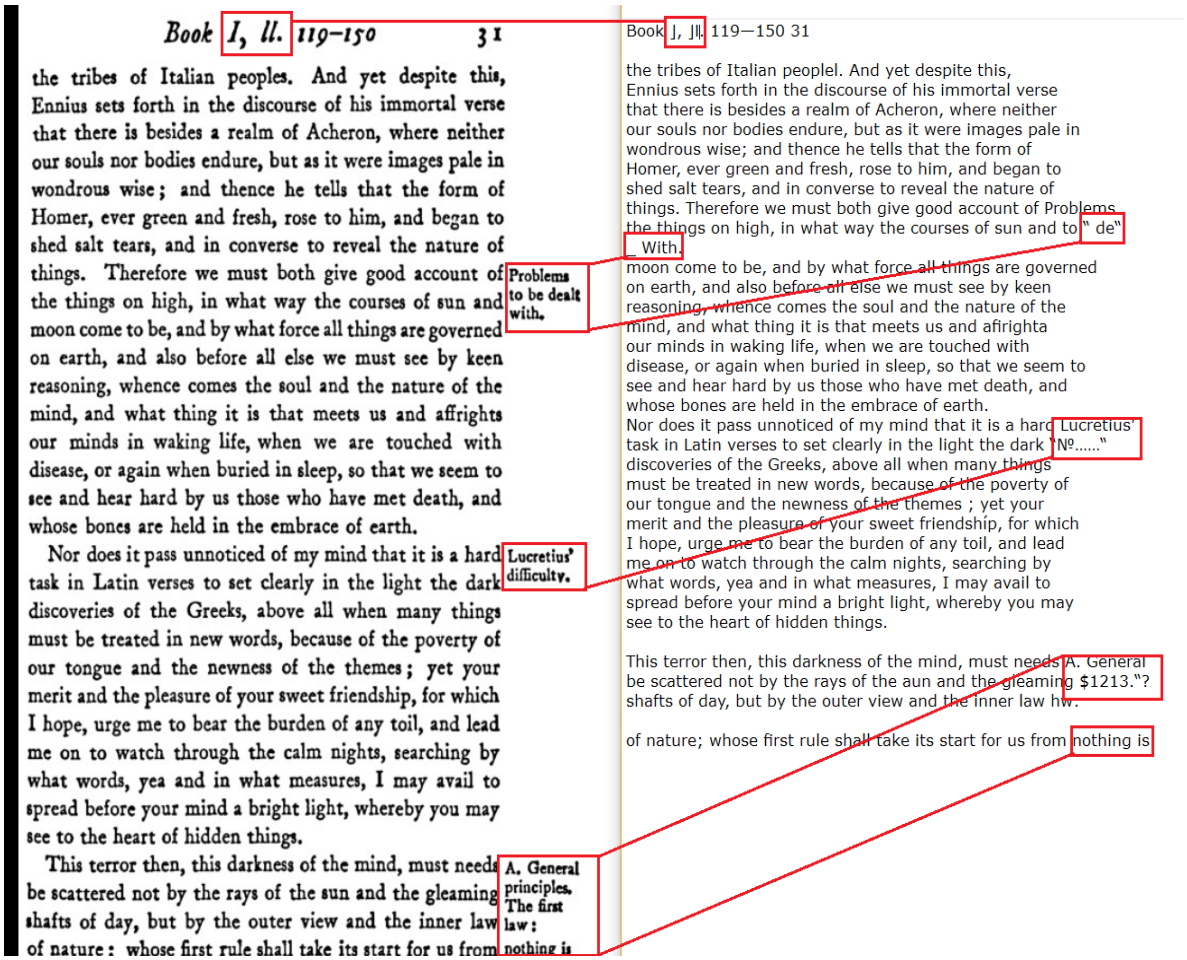
výsledek bodují za **F**. Tento případ je jasným příkladem toho, jako moc záleží výsledná přesnost na rozlišení a DPI vstupního obrázku.

6.3 Případ 3 – kniha

Jako třetí vzorek jsem se pokusil digitalizovat stránku z knihy „*De Rerum Natura*“ filosofa Lucretiusa (Příloha C.3). Má netypické rozložení a písmo, proto jsem se rozhodl, že je tato kniha vhodná pro testování. Po rozpoznání jsem si všiml pár chyb, první chyba je to, že OCR neumí rozpoznávat římská čísla, to je ovšem logické, jelikož jazyk rozpoznávání je nastaven na angličtinu. Druhá chyba spočívá v tom, že softwaru se někdy nepodařilo správně rozpoznat rozložení stránky, což způsobilo „naházení“ textu do jedné věty (***Obrázek 20***). Avšak souhlasím, že toto rozložení stránky se nepoužívá moc často. Jinak bych řekl, že s výstupem rozpoznávání jsem docela spokojen a konečný výsledek lze za par vteřin upravit tak, aby odpovídal původnímu textu na 100 %. Tento případ ukazuje přibližný postup jako při běžné digitalizací knih. Podle tabulky hodnocení přesnost rozpoznávání boduji za **B**.

Tabulka 5 – Hodnocení rozpoznávání (Případ 3)

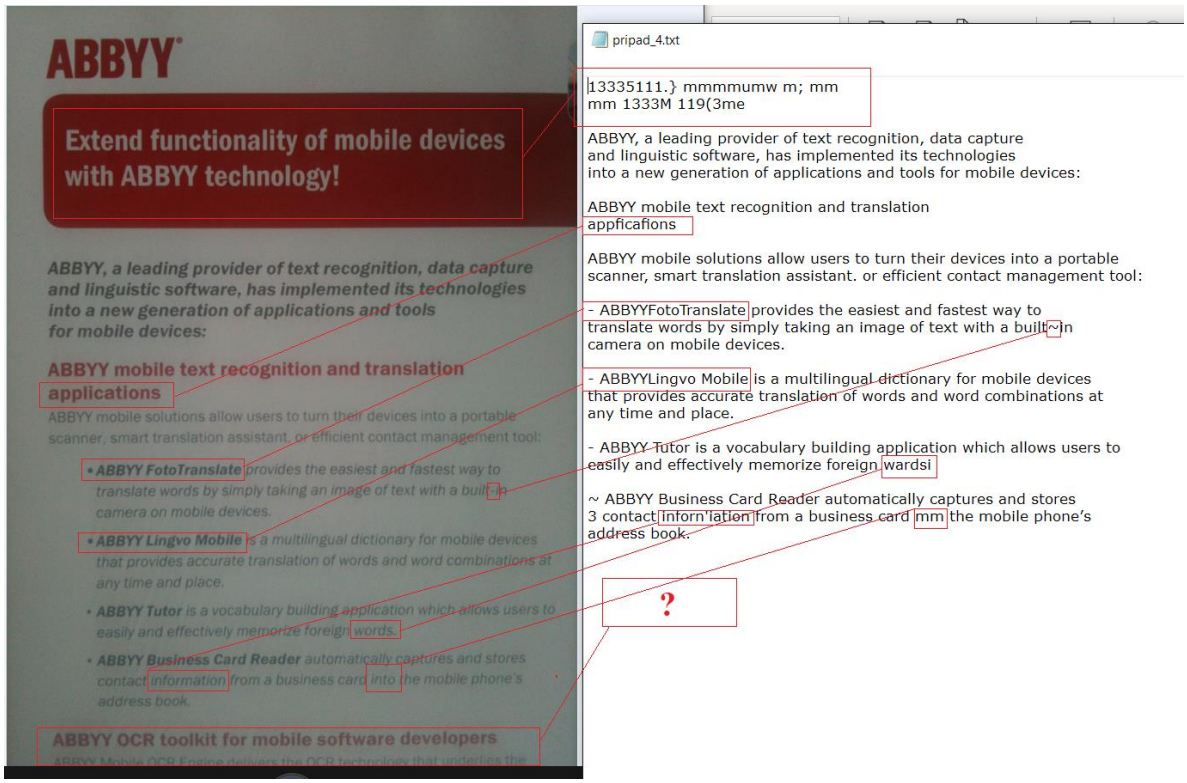
Kritérii	**Výsledek**
Stav a kvalita obrázku	Dobrá kvalita
DPI	400 DPI
Rozlišení obrázku	578 x 843
Počet slov	361
Počet znaků	2178
Nastavení IronOCR	CBN, EC, ER, RS
Čas rozpoznání	16 sekund
Sebejistota IronOCR	90.12 %
Přesnost rozpoznání znaků	91.68 %
Přesnost rozpoznání slov	95.56 %
Hodnocení	**B**



Obrázek 20 – Vstupní obrázek spolu s výstupem rozpoznávání (Případ 3)

6.4 Případ 4 – mobilní fotografie

Jako další případ jsem se pokusil rozpoznat mobilní fotografii ve špatné kvalitě. Na internetu jsem našel brožuru ABBYY FineReader, vytiskl jsem jí, a následně vyfotil pomocí svého mobilního telefonu. Důležité je zmínit, že při focení nebyl dostatek světla a schválně jsem nepoužil blesk. Při náhledu výsledku už bylo jasně vidět, že se zhoršením stavu a kvality vstupních obrázku neustále klesá přesnost rozpoznávání. Nejlepší výsledek jsem dostal při použití automatického režimu zpracování, manuální režim produkoval nižší přesnost. V automatickém režimu systému se podařilo rozpoznat 80 % textu. Na dalším obrázku červené obdélníky ukazují na místa chyb v původním textu a odpovídajícím chybám při rozpoznávání. Některé častí textu softwaru se nepodařilo rozpoznat vůbec.



Obrázek 21 – Vstupní obrázek spolu s výstupem rozpoznávání (Případ 4)

Tabulka 6 – Hodnocení rozpoznávání (Případ 4)

Parametr	**Hodnota**
Stav a kvalita obrázku	Mobilní fotografie, nedostatek světla
DPI	Nerelevantní
Rozlišení obrázku	1536 x 2048
Počet slov	176
Počet znaků	1017
Nastavení IronOCR	Auto OCR
Čas rozpoznání	13 sekund
Sebejistota IronOCR	71.67 %
Přesnost rozpoznání znaků	83.43 %
Přesnost rozpoznání slov	85.55 %
Hodnocení	**C**

Při analýze výsledku jsem si všiml, že systému se nepodařilo rozpoznat kus textu "Extend functionality of mobile…". Přišlo mi zajímavé, že po zpracování obrázku a jeho rozdělení do jednotlivých paragrafů, se systému stálé nepodařilo tento kus textu

rozpoznat. Na níže uvedeném obrázku je náhled tohoto paragrafu. Je vidět, že obrázek je relativně čitelný a teoreticky by mělo být snadné ho rozpoznat.

Obrázek 22 – Část obrázku po zpracování IronOCR

Poté jsem zkusil tento paragraf rozpoznat jako samostatný obrázek. Zkoušel jsem oba režimy, avšak při jakémkoliv nastavení program nemohl rozpoznat obrázek a pokaždé vypisoval prázdný nebo špatný výstup. Tento problém mě zaujal, a proto jsem ho zkusil rozpoznat pomocí několika jiných OCR nástrojů.

Tabulka 7 – Porovnaní výsledku rozpoznávání různých OCR nástrojů (Případ 4)

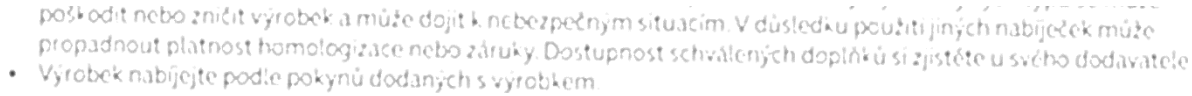
Nastroj	Výsledek rozpoznaní
IronOCR	No recognized text
Tesseract OCR C++	thjmmwd?ud?b?m
Tessnet2	mm) LQBBW Lt-Ieil??h?w
WebOCR (onlineocr.net)	No recognized text
WebOCR (ocr.space)	Etendtund ttoialigyof mobile deitces with ABBW tedhnoog

Je vidět, že skoro všechny OCR nástroje, které jsem použil pro rozpoznání, nemohly rozpoznat tento obrázek. Jediný online OCR systém (https://ocr.space), který jsem použil pro rozpoznávání, nejlépe (v porovnání s jinými) rozpoznal obrázek. Předpokládám, že neschopnost rozpoznat ho, je způsobena metodami, které IronOCR použil při předzpracování daného obrázku. Zejména černý šum, který se nepodařilo nástroji úplně odstřihnout. V kapitole 2.1, kde byl popsán princip, na kterém funguje OCR, metody Matrix Matching, a Feature Extraction potřebují „čistý“ vstup znaku pro správné rozpoznávání. Kvůli černému šumu, který otočí znaky, při rozpoznání OCR nemůže rozpoznat – kde končí písmeno a kde začíná pozadí. V tomto případě nástroje pro předzpracovaní obrázku nebyly úplně efektivní.

6.5 Případ 5 – nízké a vysoké DPI

U dalšího vzorku se pokusím rozpoznat text napsaný v českém jazyce – technickou brožuru pro moje sluchátka (Příloha C.4). Pro ztížení rozpoznávání a důkladnější testování jsem ji naskenoval v minimálním možném DPI pro můj skenovací přístroj (150 DPI). Zároveň při skenovaní jsem jí nevyrovnal správně. Důležitým faktorem je velikost písma, která odpovídá ~6 - 8pt.

Výsledky nebyly překvapující, softwaru se téměř nepodařilo rozpoznat vstupní obrázek. Místo výstupu jsem dostal pouze nesmyslnou sadu znaků. Zkoušel jsem ručně upravovat vstupní obrázek, správně jsem ho vyrovnal a odstřihnul jenom potřebné části, to ale nemělo žádný vliv na výstup. Když jsem se podíval do složky z odříznutými paragrafy, tak jsem zároveň zjistil příčinu tohoto výstupu.



Obrázek 23 – Část obrázku po zpracování IronOCR v 150 DPI

Je vidět, že kvůli příliš malému písmu a nízkému DPI jsem dostal téměř nečitelný text.

Naskenoval jsem brožuru znovu, ale tentokrát v 300 DPI. Jinak jsem zachoval veškeré podmínky předchozího skenovaní. Výsledky už byly mnohem lepší. Dostal jsem čitelný text, s vysokou přesností rozpoznání. Na uvedeném obrázku je představen kus textu po zpracování.

Obrázek 24 – Část obrázku po zpracování IronOCR v 300 DPI

Je vidět, že odstřihnuté paragrafy jsou v mnohem lepší kvalitě, než jsou v předchozím případě, a proto OCR rozpoznává text s vetší přesností. V tomto případě velikost písma už nebyla tak kritická pro OCR, protože při zpracování obrázku se používá metoda Enhance Resolution, která maximálně bezeztrátově zvětšuje původní obrázek.

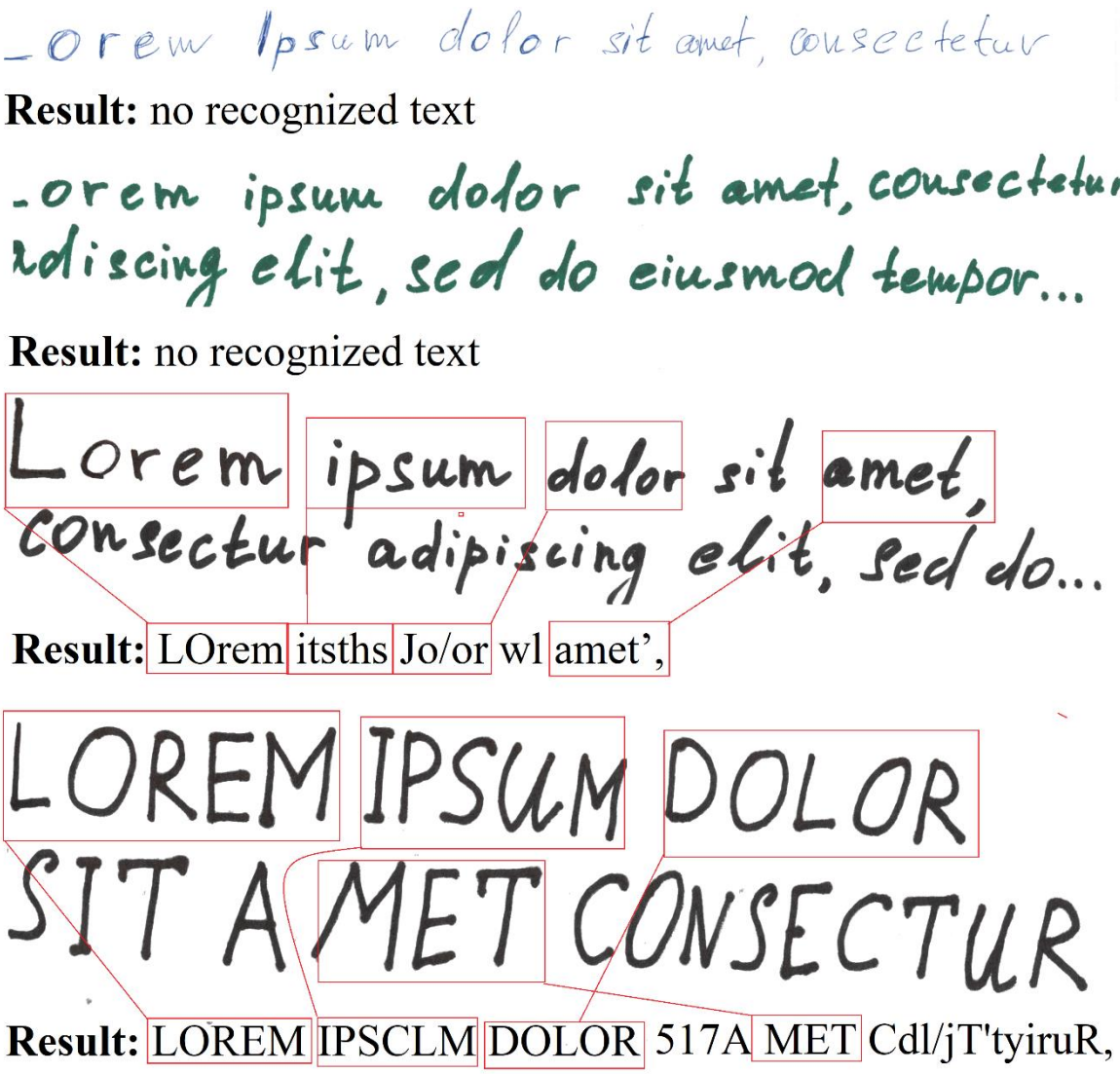
Tabulka 8 – Hodnocení rozpoznávání (Případ 5)

Parametr	**Hodnota**	
Stav a kvalita obrázku	Uspokojivá	Dobrá
DPI	150 DPI	300 DPI
Rozlišení obrázku	1650 x 780	1920 x 1080
Počet slov	740	
Počet znaků	5239	
Nastavení IronOCR	Auto OCR	ER, RS, CBN
Čas rozpoznání	35 sekund	23 sekund
Sebejistota IronOCR	53.33 % /	84.89 %
Přesnost rozpoznání znaků	5.02 %	94.59 %
Přesnost rozpoznání slov	5.08 %	96.01 %
Hodnocení	**F**	**B**

První výsledek boduji za **F**, druhý za **B**. Tím pádem považuji, že minimální DPI pro úspěšné rozpoznávání by mělo být minimálně 300. Samozřejmě čím vetší DPI bude mít skenovaný dokument, tím jasnější obrázek dostaneme, což přímo ovlivňuje přesnost rozpoznání.

6.6 Případ 6 – psaný text

Dále bych chtěl otestovat chovaní nástroje při rozpoznávání psaného textu. Na jeden list formátu A4 jsem napsal několik vět různým písmem a poté jsem ho naskenoval v 400 DPI. Na ***Obrázek 25*** jsou uvedeny jednotlivé řádky a výstupy rozpoznávání. Do výsledku jsem zapisoval nejlepší dosažené výsledky. Na obrázku je také vidět, že například psaný text IronOCR neumí rozpoznávat vůbec. Proto pro zvýšení šancí rozpoznávání jsem se snažil psát tiskací písmenka. Na 3. řádku je už vidět, že OCR začíná rozpoznávat nějaká písmena. V posledním řádku softwaru se už podařilo skoro bezchybně rozpoznat celá slova. Proto považuji, že tento OCR software není přizpůsobený na rozpoznávání psaného textu. Tento případ se bodovat nebude, protože bylo určeno že software není použitelný na ručně psané texty.



Obrázek 25 – Psaný text a výsledky jeho rozpoznávání

6.7 Případ 7 - PDF soubor s vice stránkami

V 7. případě testování jsem zkoušel rozpoznat soubor, který má víc stránek. Jedná se o otázky z programování, které jsem vypracoval pro přípravu na SZZ. Dokument jsem vytiskl
a následně naskenoval v 300 DPI, poté ho uložil jako soubor ve formátu PDF. Tento příklad obsahuje 4 stránky českého textu.

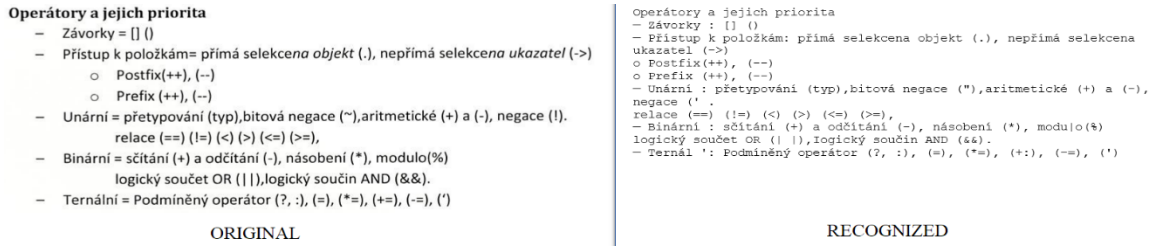
Po rozpoznávání jsem byl překvapen výsledkem. S automatickými nastavení softwaru se podařilo dosáhnout 87 % přesnosti, což je velmi dobrý výsledek. Taky bych chtěl upozornit na celkový čas rozpoznávání, který byl 47 sekund pro 2368 slov, což odpovídá

až 50 slovy za sekundu. Možná je to způsobeno dobrou kvalitou vstupního obrázku, tím pádem software neztratil moc času na jeho předzpracovaní.

Tabulka 9 – Hodnocení rozpoznávání (Případ 7)

Parametr	**Hodnota**
Stav a kvalita obrázku	Dobrá
DPI	300 DPI
Rozlišení obrázku	2314x3250
Počet slov	2368
Počet znaků	12420
Nastavení IronOCR	Auto OCR
Čas rozpoznání	47 sekund
Sebejistota IronOCR	86.03 %
Přesnost rozpoznání znaků	87.55 %
Přesnost rozpoznání slov	93.32 %
Hodnocení	**C**

Programu se taky úspěšně podařilo rozpoznat většinu matematických operátorů v textu a také dobře rozpoznává závorky jakéhokoliv typu.



Obrázek 26 – Porovnaní vstupu a výstupu rozpoznávání (Případ 7)

6.8 Případ 8 – vícejazyčný text

V dokumentací IronOCR je uvedeno, že knihovna umí rozpoznávat víc jazyku najednou, tuto možnost bych chtěl otestovat. Pro tyto účely jsem nejdříve upravil zdrojový kód programu. V konstruktoru *IronOcr()* u parametru *.Language* se musí upravit:

```
1. Language =
2. IronOcr.Languages.MultiLanguage.OcrLanguagePack(
3. IronOcr.Languages.JAZYK1.OcrLanguagePack,
4. IronOcr.Languages.JAZYK2.OcrLanguagePack,
5. …)
```

Místo **JAZYK1** a **JAZYK2**, se může vybrat jakýkoliv dostupný jazyk pro rozpoznávání. Počet jazyků je omezen jenom počtem dostupných jazyků. Předem se musí stáhnout odpovídající jazykový balíček.

V daném případě jsem si připravil obrázek úryvku textu z biografie Tomáše G. Masaryka. Čeština a angličtina používá latinské písmo, proto pro lepši přehlednost jsem jako třetí jazyk zvolil ruštinu, která používá azbuku.

Systému se úspěšně podařilo rozpoznat daný obrázek. Všechny 3 kusy textu v různých jazycích mají skoro 100 % úspešnost rozpoznávání. Dokonce v některých místech, kde se mícha ruština a čeština systém neměl problémy při rozpoznávání. Takže je to pravda, IronOCR umí rozpoznávat text ve více jazycích. Výsledek boduji za **A**. Tento případ slouží jako předmluva pro další.

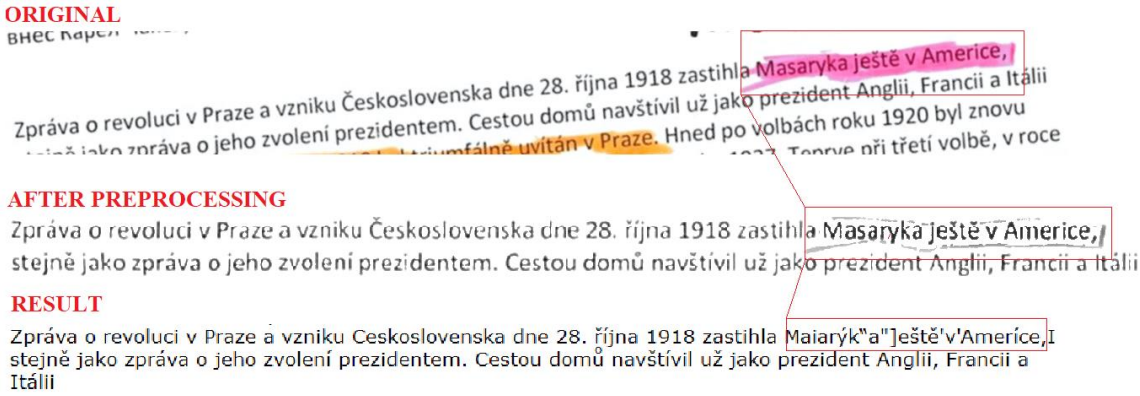
Tabulka 10 – Hodnocení rozpoznávání (Případ 8)

Parametr	**Hodnota**
Stav a kvalita obrázku	Výborná
DPI	Nerelevantní
Rozlišení obrázku	1634 x 756
Počet slov	503
Počet znaků	2985
Nastavení IronOCR	Auto OCR
Čas rozpoznání	17 sekund
Sebejistota IronOCR	89.93 %
Přesnost rozpoznání znaků	98.01 %
Přesnost rozpoznání slov	99.55 %
Hodnocení	**A**

6.9 Případ 9 – komplexní

V posledním testovacím případě bych chtěl otestovat komplexnější případ, který by shrnoval všechny předchozí případy. Použil jsem text z předchozího případu, předem jsem ho upravil tak, aby texty v různých jazycích měly různý font. Text jsem vytiskl a pustil se do jeho upravení. Papír jsem zmačkal, přidal jsem pár nadpisů a „náhodou“ jsem na něj vylil trošku kávy. Potě jsem ho naskenoval v nízkém 150 DPI, navíc jsem ho otočil o 180 stupňů a špatně zarovnal (Příloha C.5). Při takových úpravách jsem nepočítal, že přesnost rozpoznávání bude přesahovat alespoň 50-60 %. Avšak po dokončení

rozpoznávání jsem byl velice překvapen výsledkem. Systému se podařilo nejen rozpoznat původní text s ~90 % přesností, ale i zachytit pár slov, které jsem psal ručně na papír.



Obrázek 27 – Porovnaní vstupu a výstupů (Případ 9)

V některých místech měl software stále pár chyb, ale to jsou pouze drobnosti. Například na předchozím obrázku je vidět, že systém špatně rozpoznal text, který byl na papíře barevně zvýrazněn, bez ohledu na to, že při předzpracovaní, pomocí binarizace, softwaru se ho podařilo téměř úplně odstranit. A to je stejný problém, jako v případě s mobilní fotografii, nějaká část obarvení zasahuje do textu. Systému se podařilo rozpoznat 90 % od celkového textu, což pro vstup v takové kvalitě považují za velice dobrý výsledek.

Tabulka 11 – Hodnocení rozpoznávání (Případ 9)

Parametr	**Hodnota**
Stav a kvalita obrázku	Špatná
DPI	150 DPI
Rozlišení obrázku	1653 x 2339
Počet slov	518
Počet znaků	2732
Nastavení IronOCR	ER, EC, RS
Čas rozpoznání	32 sekund
Sebejistota IronOCR	77.23 %
Přesnost rozpoznání znaků	88.03 %
Přesnost rozpoznání slov	92.23 %
Hodnocení	**C**

6.10 Souhrn

Zde jsou uvedeny v souhrnu uvedeny výsledky všech provedených testů za účelem možnosti porovnání. Je vyhodnocena průměrná doba rozpoznání připadající na jeden znak (symbol). Průměrná doba rozpoznání jednoho slova není relevantní, protože slova mají různou délku (různý počet symbolů). Z toho důvodu tento údaj neuvádím. Tabulka je postačující k vizualizaci výsledků, žádný další graf není potřebný.

Pokud máme obrázek v dobré kvalitě, minimálně v 300 DPI a minimálně v 720p můžeme počítat s tím, že výsledná přesnost rozpoznávaní symbolů bude nejmíň 85%. Při takových podmínkách, software dosahá rychlostí rozpoznávání až ~200-250 symbolů za sekundu.

Je zřejmě že s poklesem kvality vstupního obrázku klesá přesnost výsledku a zároveň software tráví víc čas na jeho rozpoznávaní. Za nejhorších podmínek rychlost klesá až do ~70-100 symbolů za sekundu.

Tabulka 12 – Souhrn výsledku pro případy

	Případ							
Parametr	**1**	**2**	**3**	**4**	**5**	**7**	**8**	**9**
Stav	Perf.	Dob.	Dob.	Špa.	Dob.	Usp.	Výb.	Špa.
DPI	120	400	400	Nerel.	300	300	Nerel.	150
Čas rozpoznaní	7 s	34 s	16 s	13 s	23 s	47 s	17 s	32 s
Počet slov	280	649	361	176	740	2368	503	518
Počet symbolů	1653	4004	2178	1017	5239	12420	2985	2732
Rozp. znaků za sekundu	236	117	136	78	227	264	213	85
Přesnost OCR	100 %	99.07 %	91.68 %	83.43 %	94.59 %	87.55 %	98.01 %	88.03 %
Hodnocení	**A**	**A**	**B**	**C**	**B**	**C**	**A**	**C**

7 Diskuze

V dnešní době existuje celá řada komerčních a nekomerčních nástrojů pro optické rozpoznávání znaků. Tak to není divné, problematikou OCR se vědci začali zabývat ještě na konci 19. století. Za tuto dobu nástroje prošly cestu od jednoduchých přístrojů pro přečtení textu pro nevidomé, do pokročilejších systémů, využívajících strojové učení a umělou inteligencí pro rozpoznávání nejen samostatného textu, ale i různých objektů, jako například cestovní pasy, faktury, dopisy, plakátovací tabulky, obrázky, dopravní značky, SPZ atd.

Systémy pro optické rozpoznávání znaků napájejí spoustu větších systémů, o kterých běžný uživatel nemusí ani vědět. OCR se používá například při indexaci dokumentů, pro automatizaci zadávání dat, při rozpoznávání značek a také pomáhá nevidomým převádět text na zvuk. Tímhle OCR šetří stovky a stovky hodin pro lidi kteří ho používají.

Ale OCR má nejen kladné, ale i záporné strany. Hlavním problémem dnešních nástrojů (stejné jako u starších) je přesnost, se kterou rozpoznávají vstupní data. Vývojáři těchto systémů s tím bojují každý den, aby co nejvíce zvýšili přesnost výstupních dat. Vyvíjí se nové metody pro předzpracování vstupní informace, zároveň i novější metody pro postzpracování výstupu. Za nejperspektivnější řešení považuji strojové učení a umělou inteligenci, kdy si dokážeme „naučit“ systém do takového stupně, kdy bude produkovat výstup s maximální přesností a zároveň ošetřovat tento výstup.

V průběhu samostatného testovaní jsem zjistil, že dnešní nástroje mají spoustu možností a mají docela vysokou přesnost rozpoznávání. Když jde o obrázek v perfektně kvalitě, který chceme rozpoznat, tak už můžeme mít jistotu, že výsledný text bude odpovídat původnímu alespoň na 95 %, což bylo dokázáno v průběhu testovaní. Určitě, v zvláštních případech, kdy obrázek je nepřehledný, má defekty, nebo je v nízké kvalitě, může přesnost může klesat s velkou rychlostí. Navíc, když máme text s nestandardním písmem, nebo například psaný text, OCR ho může nerozpoznat vůbec. Avšak, dnešní OCR systémy mají v sobě spoustu nástrojů, které pomáhají ošetřovat vstupní data – metody pre-processingu a post-processingu. V průběhu testovaní jsem dokázal, že obrázek i v nejhorší kvalitě lze upravit alespoň do čitelného formátu, díky tomu o hodně zvětšit šance úspěšného rozpoznávání.

Závěr

V dané bakalářské práci jsem provedl analýzu úspešností optického rozpoznávání znaků v prostředí .NET Framework pomocí knihovny IronOCR, která je založena na nejznámějším open-source řešení Tesseract OCR.

Hlavním cílem této bakalářské práce bylo provést a představit vyhodnocení OCR systému. Tento cíl považují za splněný, vzhledem k tomu, že jsem provedl zkoumaní systému
na klíčových případech a prokázal, do jaké míry a s jakou přesností výstupu může OCR systém rozpoznávat vstupní data. Je pravda, že jsem otestoval zdaleka ne všechny možné scenerie použití, ale myslím, že hlavní možností IronOCR jsem úspěšně otestoval a ke každému případu jsem se dostatečně vyjádřil.

Je pravda, že při vypracování této bakalářské práce jsem měl omezení v tom, že Tesseract mám používat v prostředí .NET/C#, díky tomu výběr nástrojů pro testovaní byl omezen a nemohl jsem využit všechny možností původního systému Tesseract, jako například strojové učení. To ale nebyl velký problém, vzhledem k tomu, že IronOCR bez těchto možností, má spoustu vlastích řešení, které byly postačující při provedení analýzy a do určité míry více než původní Tesseract.

Na závěr bych rád zmínil, že po vypracovaní této bakalářské práce stále nepovažují OCR systémy za 100 % spolehlivé, ale s minimální lidskou supervisí je možné používat OCR jako velice přínosný nástroj pro široké spektrum úkolů. A to nejen pro zkušené vývojáře, ale i pro běžného uživatele. Na závěr hodnotím výstupy této bakalářské práce za velmi kladné a přínosné.

Seznam použité literatury

Arindam Chaudhuri, Krupa Mandaviya, Pratixa Badelia, Soumya K. 2017. *Optical Character Recognition Systems for Different Languages with Soft Computing.* Springer International Publishing, 2017. ISBN: 978-3-319-50252-6.

BBC. 2015. Google Translate 'turns interpreter' with voice function. *BBC.* [Online] 2015. [Citace: 10. 10 2019.] https://www.bbc.com/news/technology-30812277.

Damilies. 2018. Basic OCR in OpenCV. *Damilies Blog.* [Online] 2018. [Citace: 30. 10 2018.] http://blog.damiles.com/2008/11/20/basic-ocr-in-opencv.html.

Data ID. 2018. OCR Introduction. *What's OCR?* [Online] 2018. [Citace: 30. 06 2019.] http://www.dataid.com/aboutocr.htm.

DOPRAVNÍ PŘESTUPKY. 2006. Oznámení o Překročení Rychlosti. [Online] 2006. [Citace: 10. 05 2019.]

Dvortygirl. 2008. *Book digitalization using OCR.*

GISGeography. 2015. Image Classification Techniques in Remote Sensing. [Online] 2015. [Citace: 10. 10 2019.] https://gisgeography.com/image-classification-techniques-remote-sensing/.

Grooper. 2018. Grooper Document Capture. *Image Optimization.* [Online] 2018. [Citace: 10. 10 2019.] https://grooper.com/image-optimization.html.

Hauger, J. Scott. 1995. *Reading Machines for the Blind.* Blacksburg, Virginia : Faculty of the Virginia Polytechnic Institute and State University, 1995.

HOLLEY, Rose. 2009. How Good Can It Get? *Analysing and Improving OCR Accuracy in Large Scale Historic Newspaper Digitisation Programs.* [Online] Květen 2009. http://www.dlib.org/dlib/march09/holley/03holley.html. ISSN 1082-9873.

IronSoftware. 2019. IronOCR The C# Library. *IronSoftware.* [Online] 2019. [Citace: 15. 06 2019.] https://ironsoftware.com/csharp/ocr/.

Milan Sonka, Vaclav Hlavac, Roger Boyle. 2014. *Image Processing, Analysis, and Machine Vision.* místo neznámé : Springer US, 2014. ISBN: 978-1-4899-3216-7.

Muhammad'Arif Mohamad, Dewi Nasien, Haswadi Hassan, Habibollah Haron. 2015. *A Review on Feature Extraction and Feature Selection for Handwritten Character Recognition.* 2015. ISBN: 6(2):204–212.

Norman, Jeremy. 2019. The First OCR System: "GISMO". *History Of Information.* [Online] 2019. [Citace: 03. 10 2019.] http://www.historyofinformation.com/detail.php?entryid=885.

PCTechGuide. 2019. OCR Technology. *pctechguide.com.* [Online] 2019. [Citace: 03. 10 2019.] https://www.pctechguide.com/scanners/ocr-technology.

Rémi, T. 2018. Tessnet2 a .NET 2.0 Open Source OCR assembly using Tesseract engine. *pixel-technology.com.* [Online] 2018. http://www.pixel-technology.com/freeware/tessnet2/.

Schantz, H. F. 1982. *The history of OCR: optical character recognition.* místo neznámé : Recognition Technologies Users Association, 1982. ISBN: 0943072018.

Tesseract. 2018. *tesseract-ocr.* [Online] 2018. [Citace: 30. 10 2018.] https://github.com/tesseract-ocr/tesseract.

TRIER, Oeivind Due a Jain, Anil K. 1995. Goal-directed evaluation of binarisation methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* [Online] 1995.

Ubuntu. 2016. OCR - Optical Character Recognition. *Ubuntu.* [Online] 2016. [Citace: 10. 07 2019.] https://help.ubuntu.com/community/OCR.

Vetenskapen och livet. 1922. 1922.

Wilhelm Burger, Mark James Burge, Mark James Burge, Mark James. 2009. *Principles of digital image processing.* London : Springer, 2009. 978-1-84800-194-7.

Přílohy

Příloha A.1 – Výběr souboru

Příloha A.2 – Uložení souboru

Příloha A.3 – OCR pomocí manuálních nastavení

Příloha A.4 – Výpis detailní informace o rozpoznávání

Příloha B – Obrázek pro původní testovaní IronOCR

Příloha C.1 – Testovací obrázek pro případ 1

Příloha C.2 – Testovací obrázek pro případ 2

Příloha C.3 – Testovací obrázek pro případ 3

Příloha C.4 – Testovací obrázek pro případ 5

Příloha C.5 – Testovací obrázek pro případ 9

Příloha D – CD se zdrojovým kódem, spustitelným souborem, vstupními a výstupními obrázky

Příloha A.1 – Výběr souboru

```
1.  //Funkce pro vyběr souboru
2.  private void VybratBtn_Click(object sender, EventArgs e)
3.  {
4.      try
5.      {
6.          //Nastaveni openFileDialog
7.          openFileDialog.InitialDirectory = @"C:\IronOCR\"; //Vychozi složka
8.          openFileDialog.Title = "Vyberte obrazek"; //Titulek okna
9.          openFileDialog.Filter = "PNG File (.png)|*.png|TIFF File (.tiff)|*.tif
    f|JPG file (.jpg)|*.jpg|PDF File (.pdf)|*.pdf"; //Filter
10.         openFileDialog.FileName = "";
11.
12.
13.         if (openFileDialog.ShowDialog() == DialogResult.OK) //jestli soubor vy
    bran spravne
14.         {
15.             vstup_txt.Text = openFileDialog.FileName; //zapsat jeho cestu do t
    extoveho policka
16.             vstup_soubor = openFileDialog.FileName; //zapsat do promenny
17.         }
18.     }
19.     catch (Exception ex)
20.     {
21.         MessageBox.Show("Error: \n" + ex, "Exception", MessageBoxButtons.OK);
    //v pripade chyby, oznamit uzivatela
22.     }
23. }
```

Příloha A.2 – Uložení souboru

```
1.  //Uložení souboru
2.  void vystupBtn_Click(object sender, EventArgs e)
3.  {
4.      saveFileDialog.InitialDirectory = @"C:\IronOCR\";
5.      saveFileDialog.Filter = "Text files(*.txt)|*.txt"; //Filter
6.      saveFileDialog.FileName = "";
7.
8.      if (saveFileDialog.ShowDialog() == DialogResult.Cancel)
9.      {
10.         return;
11.     }
12.
13.     vsouborTxt.Text = saveFileDialog.FileName;//Zapiseme cestu do t. policka
14.     vystup_soubor = saveFileDialog.FileName;//Zapiseme cestu do prommeny
15. }
```

Příloha A.3 – OCR pomocí manuálních nastavení

```
1.          private void OCRbtn_Click(object sender, EventArgs e)
2.          {
3.              var watch = System.Diagnostics.Stopwatch.StartNew(); //timer pro spocitani casu rozpoznavani
4.              //Kontrola existence souboru
5.              if (!File.Exists(vstup_txt.Text)) //Kontrolujeme jestli vybrany soubor existuje
6.              {
7.                  MessageBox.Show("Políčko Vstup je prazdne, nebo soubor neexistuje. Prosim opakujte vyber", "Chyba",
8.                      MessageBoxButtons.OK, MessageBoxIcon.Warning);
9.                  return;
10.             }
11.             if (vystup_soubor == string.Empty) //Kontrolujeme jestli vystupni soubor je vybran
12.             {
13.                 MessageBox.Show("Políčko Vystup je prazdne, nebo cesta neni spravna. Prosim opakujte vyber", "Chyba",
14.                     MessageBoxButtons.OK, MessageBoxIcon.Warning);
15.                 return;
16.             }
17.             var Ocr = new AdvancedOcr() //Konstuktor IronOCR z parametry
18.             {
19.                 CleanBackgroundNoise = chk_CleanBackgroundNoise.Checked,
20.                 EnhanceContrast = chk_EnhanceContrast.Checked,
21.                 EnhanceResolution = chk_EnhanceResolution.Checked,
22.                 DetectWhiteTextOnDarkBackgrounds = chk_WTOBB.Checked,
23.                 RotateAndStraighten = chk_Rotate.Checked,
24.                 ReadBarCodes = chk_ReadBarcodes.Checked,
25.                 ColorDepth = (int)ud_colorDepth.Value, //Nastaveni parametru zpracovaní
26.                 Strategy = IronOcr.AdvancedOcr.OcrStrategy.Advanced, //Rezim zpracovani
27.
28.             };
29.             var result = Ocr.Read(vstup_txt.Text); //Samostatne rozpoznavaní souboru
30.             watch.Stop(); //zastavit timer
31.     //...
32. }
```

Příloha A.4 – Výpis detailní informace o rozpoznávání

```
1.  //...
2.      int pocet_stranek = 0;
3.      int pocet_paragrafu = 0;
4.      int pocet_radku = 0;
5.      int pocet_slov = 0;
6.      double confidence = 0; //prommeny pro zapis informace
7.
8.      string detailed_result = "Bakalarska prace || Ferenc Docsa \n";
9.      detailed_result  += "=========================================\n";
10.    //pomocne hlavicky
11.     detailed_result += "Cas rozpoznani: " + (watch.ElapsedMilliseconds / 1000)
    .ToString() + "s\n"; //vypis casu rozpoznani
12.
13.     //Zapis veskere potrebne informace do promenny detailed_result
14.     //Stranky
15.     foreach (var page in result.Pages)
16.     {
17.         detailed_result += "Cislo stranky: " + page.PageNumber + "\n";
18.         detailed_result += "Pocet slov: " + page.WordCount + "\n";
19.         detailed_result += "Šiřka a vyška stranky: " + page.Width + "x" + page
    .Height + "\n\n";
20.
21.         pocet_stranek++;
22.         pocet_slov += page.WordCount;
23.
24.         //Paragrafy
25.         foreach (var paragraph in page.Paragraphs)
26.         {
27.
28.             detailed_result += "Cislo paragrahu: " + paragraph.ParagraphNumber
     + "\n";
29.             detailed_result += "Rozpoznany text: " + "\n" + paragraph.Text + "
    \n";
30.             detailed_result += "Sebejistota rozpoznavani: " + Math.Round(parag
    raph.Confidence, 2) + "%" + "\n\n";
31.
32.             System.Drawing.Image paragraphImage = paragraph.Image;
33.             paragraphImage.Save( "C:\\OCRTest\\images\\Paragraph_" + paragraph
    .ParagraphNumber + ".png");
34.             //ulozeni obrazku kazdeho uriznuteho paragrafu
35.             pocet_paragrafu++;
36.             confidence += paragraph.Confidence;
37.             foreach (var line in paragraph.Lines)
38.             {
39.                 pocet_radku++;
40.             }
41.         }
42.
43.     }
44.
45.     vystupTxt.Text = detailed_result; //zapis vysledku do RichTextBox
46.     File.WriteAllText(vystup_soubor, result.Text); //zapis vysledku do souboru

47.
48.     pocetStranLbl.Text = pocet_stranek.ToString();
49.     pocetParagTxt.Text = pocet_paragrafu.ToString();
50.     pocetSlovTxt.Text = pocet_slov.ToString();
51.     pocetRadkuTxt.Text = pocet_radku.ToString();
52.     confidenceTxt.Text = Math.Round((confidence / pocet_paragrafu), 2).ToStrin
    g() + " %";
53.
54. }
```

Příloha B – Obrázek pro původní testovaní IronOCR

Chapter Eight

The Deathday Party

October arrived, spreading a damp chill over the grounds and into the castle. Madam Pomfrey, the nurse, was kept busy by a sudden spate of colds among the staff and students. Her Pepperup potion worked instantly, though it left the drinker smoking at the ears for several hours afterward. Ginny Weasley, who had been looking pale, was bullied into taking some by Percy. The steam pouring from under her vivid hair gave the impression that her whole head was on fire.

Raindrops the size of bullets thundered on the castle windows for days on end; the lake rose, the flower beds turned into muddy streams, and Hagrid's pumpkins swelled to the size of garden sheds. Oliver Wood's enthusiasm for regular training sessions, however, was not dampened, which was why Harry was to be found, late one stormy Saturday afternoon a few days before Halloween, returning to Gryffindor Tower, drenched to the skin and splattered with mud.

Even aside from the rain and wind it hadn't been a happy practice session. Fred and George, who had been spying on the Slytherin team, had seen for themselves the speed of those new Nimbus Two Thousand and Ones. They reported that the Slytherin team was no more than seven greenish blurs, shooting through the air like missiles.

As Harry squelched along the deserted corridor he came across somebody who looked just as preoccupied as he was. Nearly Headless Nick, the ghost of Gryffindor Tower, was staring morosely out of a window, muttering under his breath, ". . . don't fulfill their requirements . . . half an inch, if that . . ."

"Hello, Nick," said Harry.

"Hello, hello," said Nearly Headless Nick, starting and looking round. He wore a dashing, plumed hat on his long curly hair, and a tunic with a ruff, which concealed the fact that his neck was almost completely severed. He was pale as smoke, and Harry could see right through him to the dark sky and torrential rain outside.

"You look troubled, young Potter," said Nick, folding a transparent letter as he spoke and tucking it inside his doublet.

"So do you," said Harry.

Příloha C.1 – Testovací obrázek pro případ 1

Who Is Stephen King?

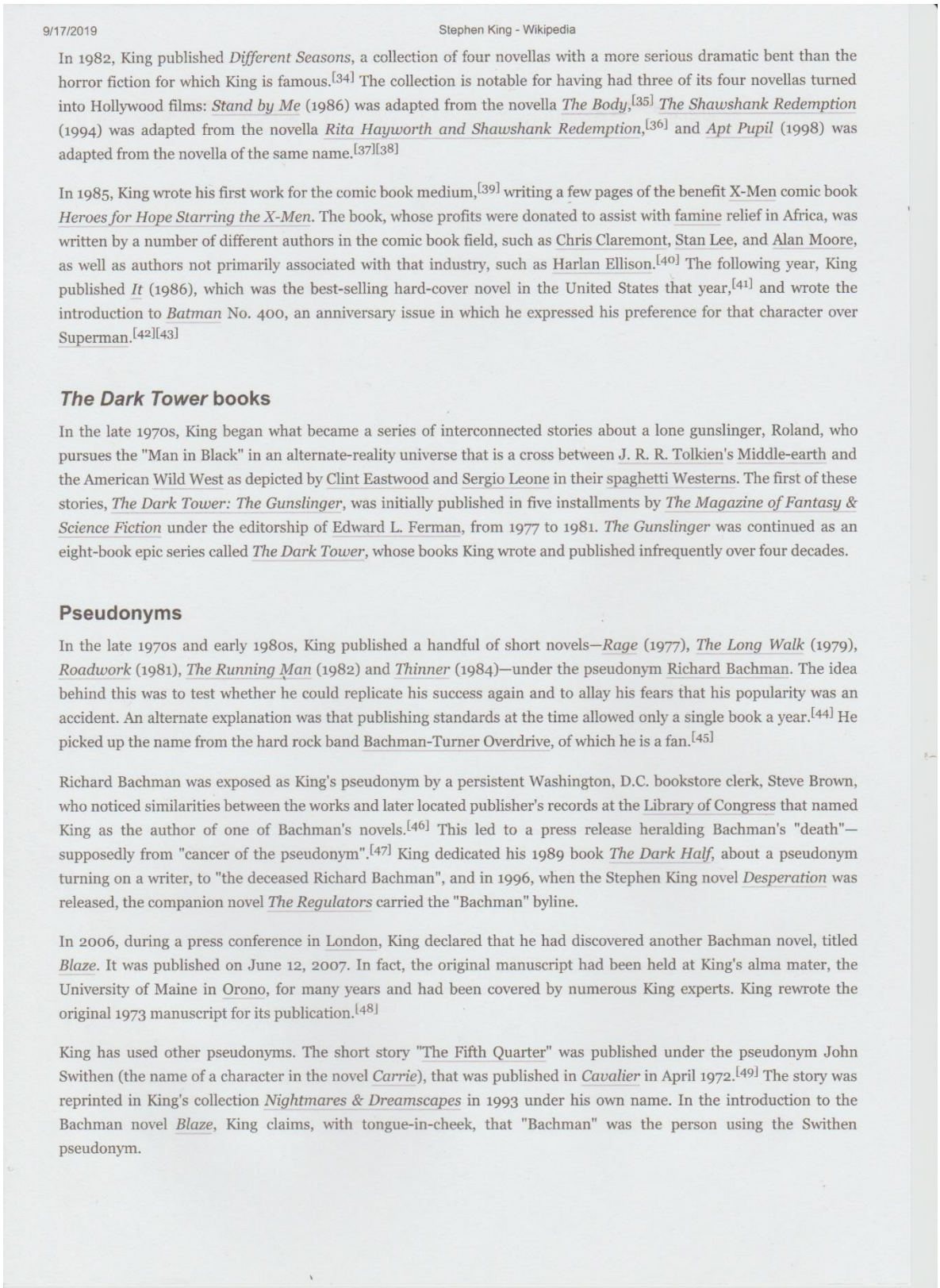
Stephen King was born on September 21, 1947, in Portland, Maine. He graduated from the University of Maine and later worked as a teacher while establishing himself as a writer. Having also published work under the pseudonym Richard Bachman, King's first horror novel, *Carrie*, was a huge success. Over the years, King has become known for titles that are both commercially successful and sometimes critically acclaimed. His books have sold more than 350 million copies worldwide and been adapted into numerous successful films.

Early Life and Education

Author Stephen Edwin King was born on September 21, 1947, in Portland, Maine. King is recognized as one of the most famous and successful horror writers of all time. His parents, Donald and Nellie Ruth Pillsbury King, split up when he was very young, and he and his brother David divided their time between Indiana and Connecticut for several years. King later moved back to Maine with his mother and brother. There he graduated from Lisbon Falls High School in 1966.

King stayed close to home for college, attending the University of Maine at Orono. There he wrote for the school's newspaper and served in its student government. While in school, King published his first short story, which appeared in *Startling Mystery Stories*. After graduating with a degree in English in 1970, he tried to find a position as a teacher but had no luck at first. King took a job in a laundry and continued to write stories in his spare time until late 1971, when he began working as an English educator at Hampden Academy. It was that year that he also married fellow writer Tabitha Spruce.

Příloha C.2 – Testovací obrázek pro případ 2



9/17/2019 Stephen King - Wikipedia

In 1982, King published *Different Seasons*, a collection of four novellas with a more serious dramatic bent than the horror fiction for which King is famous.[34] The collection is notable for having had three of its four novellas turned into Hollywood films: *Stand by Me* (1986) was adapted from the novella *The Body*,[35] *The Shawshank Redemption* (1994) was adapted from the novella *Rita Hayworth and Shawshank Redemption*,[36] and *Apt Pupil* (1998) was adapted from the novella of the same name.[37][38]

In 1985, King wrote his first work for the comic book medium,[39] writing a few pages of the benefit X-Men comic book *Heroes for Hope Starring the X-Men*. The book, whose profits were donated to assist with famine relief in Africa, was written by a number of different authors in the comic book field, such as Chris Claremont, Stan Lee, and Alan Moore, as well as authors not primarily associated with that industry, such as Harlan Ellison.[40] The following year, King published *It* (1986), which was the best-selling hard-cover novel in the United States that year,[41] and wrote the introduction to *Batman* No. 400, an anniversary issue in which he expressed his preference for that character over Superman.[42][43]

The Dark Tower books

In the late 1970s, King began what became a series of interconnected stories about a lone gunslinger, Roland, who pursues the "Man in Black" in an alternate-reality universe that is a cross between J. R. R. Tolkien's Middle-earth and the American Wild West as depicted by Clint Eastwood and Sergio Leone in their spaghetti Westerns. The first of these stories, *The Dark Tower: The Gunslinger*, was initially published in five installments by *The Magazine of Fantasy & Science Fiction* under the editorship of Edward L. Ferman, from 1977 to 1981. *The Gunslinger* was continued as an eight-book epic series called *The Dark Tower*, whose books King wrote and published infrequently over four decades.

Pseudonyms

In the late 1970s and early 1980s, King published a handful of short novels—*Rage* (1977), *The Long Walk* (1979), *Roadwork* (1981), *The Running Man* (1982) and *Thinner* (1984)—under the pseudonym Richard Bachman. The idea behind this was to test whether he could replicate his success again and to allay his fears that his popularity was an accident. An alternate explanation was that publishing standards at the time allowed only a single book a year.[44] He picked up the name from the hard rock band Bachman-Turner Overdrive, of which he is a fan.[45]

Richard Bachman was exposed as King's pseudonym by a persistent Washington, D.C. bookstore clerk, Steve Brown, who noticed similarities between the works and later located publisher's records at the Library of Congress that named King as the author of one of Bachman's novels.[46] This led to a press release heralding Bachman's "death"—supposedly from "cancer of the pseudonym".[47] King dedicated his 1989 book *The Dark Half*, about a pseudonym turning on a writer, to "the deceased Richard Bachman", and in 1996, when the Stephen King novel *Desperation* was released, the companion novel *The Regulators* carried the "Bachman" byline.

In 2006, during a press conference in London, King declared that he had discovered another Bachman novel, titled *Blaze*. It was published on June 12, 2007. In fact, the original manuscript had been held at King's alma mater, the University of Maine in Orono, for many years and had been covered by numerous King experts. King rewrote the original 1973 manuscript for its publication.[48]

King has used other pseudonyms. The short story "The Fifth Quarter" was published under the pseudonym John Swithen (the name of a character in the novel *Carrie*), that was published in *Cavalier* in April 1972.[49] The story was reprinted in King's collection *Nightmares & Dreamscapes* in 1993 under his own name. In the introduction to the Bachman novel *Blaze*, King claims, with tongue-in-cheek, that "Bachman" was the person using the Swithen pseudonym.

Příloha C.3 – Testovací obrázek pro případ 3

the tribes of Italian peoples. And yet despite this, Ennius sets forth in the discourse of his immortal verse that there is besides a realm of Acheron, where neither our souls nor bodies endure, but as it were images pale in wondrous wise; and thence he tells that the form of Homer, ever green and fresh, rose to him, and began to shed salt tears, and in converse to reveal the nature of things. Therefore we must both give good account of the things on high, in what way the courses of sun and moon come to be, and by what force all things are governed on earth, and also before all else we must see by keen reasoning, whence comes the soul and the nature of the mind, and what thing it is that meets us and affrights our minds in waking life, when we are touched with disease, or again when buried in sleep, so that we seem to see and hear hard by us those who have met death, and whose bones are held in the embrace of earth.

Problems to be dealt with.

Nor does it pass unnoticed of my mind that it is a hard task in Latin verses to set clearly in the light the dark discoveries of the Greeks, above all when many things must be treated in new words, because of the poverty of our tongue and the newness of the themes; yet your merit and the pleasure of your sweet friendship, for which I hope, urge me to bear the burden of any toil, and lead me on to watch through the calm nights, searching by what words, yea and in what measures, I may avail to spread before your mind a bright light, whereby you may see to the heart of hidden things.

Lucretius' difficulty.

This terror then, this darkness of the mind, must needs be scattered not by the rays of the sun and the gleaming shafts of day, but by the outer view and the inner law of nature; whose first rule shall take its start for us from

A. General principles. The first law: nothing is

Příloha C.4 – Testovací obrázek pro případ 5

Výstražné upozornění a prohlášení

Kabelové a bezdrátové - Evropa a Střední Východ

Prostudujte všechny příslušné odstavce.
Pro vaši ochranu a pohodlí společnost GN Audio A/S (dále jen „GN") u tohoto výrobku přijala ochranná opatření s cílem udržet bezpečnou úroveň hlasitosti a zajistit, aby výrobek fungoval v souladu s bezpečnostními normami.

UPOZORNĚNÍ NA ZVUKOVÝ VÝROBEK!
DLOUHODOBÉ VYSTAVENÍ ZVUKU O VYSOKÉ HLASITOSTI MŮŽE VÉST K TRVALÉ ZTRÁTĚ SLUCHU. VÝROBEK POUŽÍVEJTE PŘI NEJNIŽŠÍ MOŽNÉ ÚROVNI HLASITOSTI.
Vyvarujte se dlouhodobému používání náhlavních souprav při nadměrné úrovni akustického tlaku.
Před prvním použitím tohoto výrobku si prostudujte níže uvedené bezpečnostní pokyny.
Budete-li se řídit těmito bezpečnostními pokyny, můžete snížit riziko poškození sluchu:

1. Před použitím tohoto výrobku proveďte tyto kroky
- Před použitím výrobku nastavte ovladač hlasitosti na nejnižší úroveň.
- Nasaďte si náhlavní soupravu (dle potřeby)
- Ovladač hlasitosti pomalu nastavte na příjemnou úroveň.

2. Během používání tohoto výrobku
- Udržujte hlasitost na nejnižší možné úrovni;
- Pokud je vyšší hlasitost nezbytná, nastavujte ovladač hlasitosti pomalu;

Pokud máte nepříjemné pocity nebo zvonění v uších, okamžitě přestaňte výrobek používat.
Při trvalém používání vysokých hlasitostí si mohou vaše uši na vysokou hladinu zvuku zvyknout, a to může mít za následek trvalé poškození sluchu, aniž byste přitom měli nepříjemný pocit.

VŠEOBECNÉ BEZPEČNOSTNÍ INFORMACE
- Máte-li kardiostimulátor nebo jiná lékařská elektrická zařízení, před použitím tohoto výrobku byste se nejdříve měli poradit se svým lékařem.
- Toto balení obsahuje malé součástky, které mohou být nebezpečné pro děti. Výrobek vždy uchovávejte mimo dosah dětí. Sáčky samotné nebo mnoho malých částí, které jsou v nich obsaženy, mohou po požití způsobit udušení.
- Nikdy se nepokoušejte výrobek sami rozebírat ani do výrobku vtlačovat předměty jakéhokoli druhu, mohlo by tím dojít ke zkratu, který by mohl mít za následek požár nebo úraz elektrickým proudem.
- Uživatel nemůže vyměňovat ani opravovat žádnou ze součástí. Otevřít výrobek smějí pouze oprávnění zástupci a servisní střediska. Pokud je nutné některou část výrobku z jakéhokoli důvodu vyměnit, včetně běžného opotřebení,

roztržení nebo rozbití, obraťte se na obchodního zástupce.
- Výrobek nevystavujte dešti ani jiným kapalinám.
- Veškeré produkty, šňůry a kabely udržujte mimo zařízení, které je v provozu.
- Ve vyznačených prostorách, například v nemocnici či na letišti, věnujte pozornost všem vývěskám a pokynům, které vyžadují vypnutí elektrických zařízení nebo rádiových RF produktů.
- Pokud dojde k přehřátí, upuštění, poškození či namočení výrobku do tekutiny, přestaňte jej používat.
- Likvidaci výrobku provádějte v souladu s právními předpisy a normami příslušného státu (viz www.jabra.com/weee).

Pamatujte: Jezděte vždy bezpečně, nerozptylujte se a dodržujte místní zákony!
Používání reproduktorů během řízení motorového vozidla může být upraveno místními zákony. Při řízení motorového vozidla, motocyklu, plavidla nebo bicyklu může být používání náhlavní soupravy nebezpečné a v některých jurisdikcích je nezákonné, stejně tak jako není v některých těchto jurisdikcích povoleno užívání náhlavní soupravy se zakrytýma ušima. Vše si ověřte u místních úřadů.

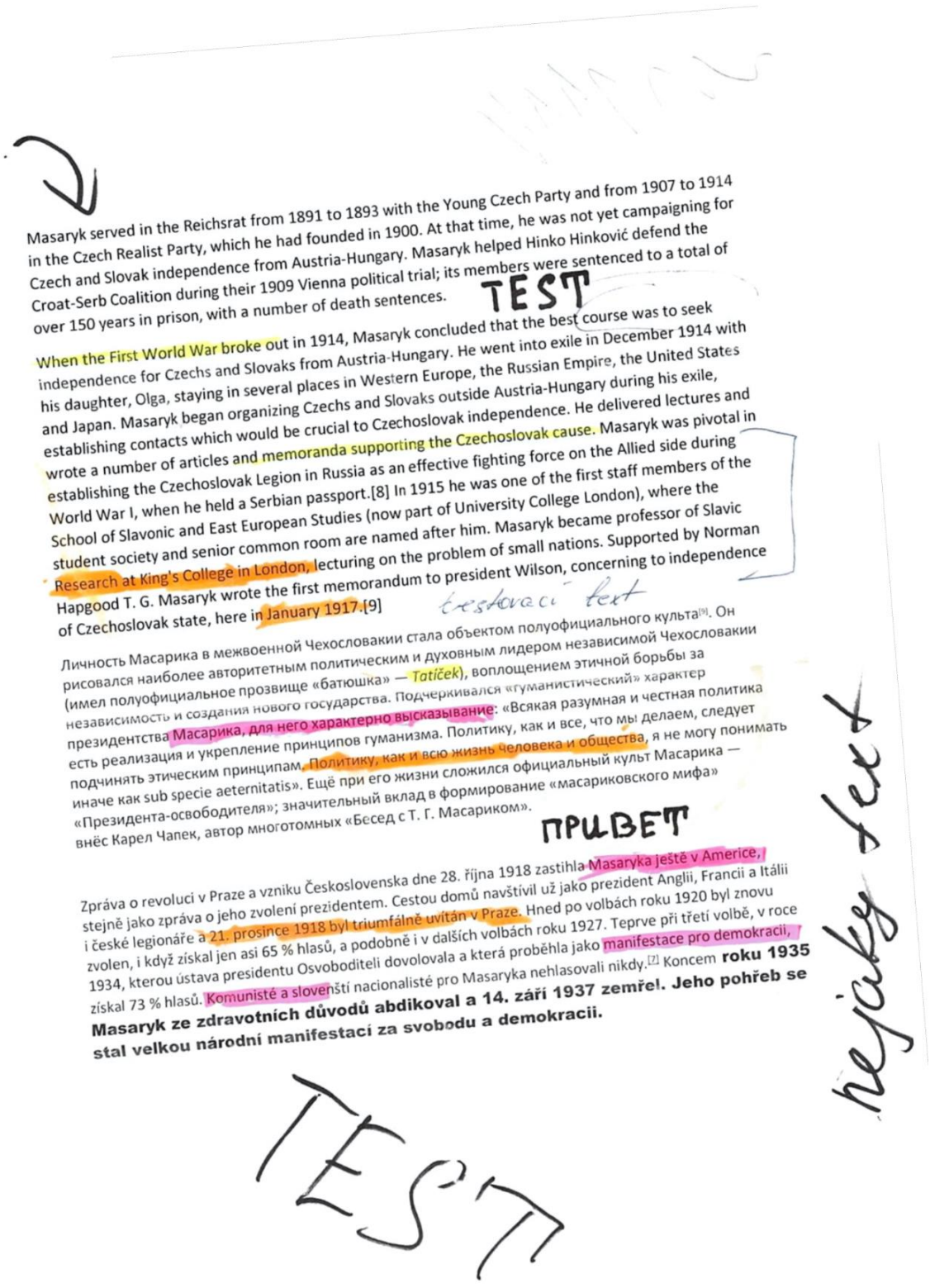
PÉČE O VESTAVĚNOU BATERII: Pokud výrobek obsahuje baterii, dodržujte prosím následující pokyny.
- Výrobek je napájen z dobíjecí baterie. Plného výkonu nové baterie je dosaženo až po dvou nebo třech cyklech kompletního dobití a vybití.
- Baterie snese stovky cyklů dobití a vybití, nakonec se však opotřebuje. Neponechávejte plně nabitou baterii připojenou k nabíječce, přebíjení může zkrátit její životnost.
- Pokud se plně nabitá baterie nepoužívá, časem se vybije.
- Ponecháním výrobku v teplém či chladném prostředí dojde ke snížení kapacity a zkrácení životnosti baterie. Baterii se vždy snažte udržovat při teplotách od 15 °C do 25 °C. Výrobek s horkou nebo chladnou baterií může dočasně přestat pracovat, i když je baterie plně nabitá.

Výstraha týkající se baterie!
- „Pozor" - Baterie použitá v tomto výrobku může při nesprávném použití představovat riziko požáru či chemických popálenin. Poškozená baterie může explodovat.
- Baterii dobíjejte pouze dodanými schválenými nabíječkami určenými pro tento výrobek.
- Baterie likvidujte v souladu s místními předpisy. Pokud možno, zajistěte jejich recyklaci. Baterie nevyhazujte do domácího odpadu ani je nelikvidujte v ohni, neboť mohou explodovat.
- Není-li v uživatelské příručce či v rychlém průvodci stanoveno jinak, uživatel nesmí ve svém výrobku baterii vyjmout ani vyměnit. Tyto pokusy jsou riskantní a mohou vést k poškození výrobku.

Více informací o bateriích naleznete na www.jabra.com/batteries.

PÉČE O NABÍJEČKU: Byl-li výrobek dodán s nabíječkou, dodržujte prosím následující pokyny.
- K dobíjení/napájení výrobku používejte výhradně dodanou nabíječku. Použitím jakýchkoli jiných typů se může poškodit nebo zničit výrobek a může dojít k nebezpečným situacím. V důsledku použití jiných nabíječek může propadnout platnost homologizace nebo záruky. Dostupnost schválených doplňků si zjistěte u svého dodavatele.
- Výrobek nabíjejte podle pokynů dodaných s výrobkem.

Příloha C.5 – Testovací obrázek pro případ 9



Masaryk served in the Reichsrat from 1891 to 1893 with the Young Czech Party and from 1907 to 1914 in the Czech Realist Party, which he had founded in 1900. At that time, he was not yet campaigning for Czech and Slovak independence from Austria-Hungary. Masaryk helped Hinko Hinković defend the Croat-Serb Coalition during their 1909 Vienna political trial; its members were sentenced to a total of over 150 years in prison, with a number of death sentences.

TEST

When the First World War broke out in 1914, Masaryk concluded that the best course was to seek independence for Czechs and Slovaks from Austria-Hungary. He went into exile in December 1914 with his daughter, Olga, staying in several places in Western Europe, the Russian Empire, the United States and Japan. Masaryk began organizing Czechs and Slovaks outside Austria-Hungary during his exile, establishing contacts which would be crucial to Czechoslovak independence. He delivered lectures and wrote a number of articles and memoranda supporting the Czechoslovak cause. Masaryk was pivotal in establishing the Czechoslovak Legion in Russia as an effective fighting force on the Allied side during World War I, when he held a Serbian passport.[8] In 1915 he was one of the first staff members of the School of Slavonic and East European Studies (now part of University College London), where the student society and senior common room are named after him. Masaryk became professor of Slavic Research at King's College in London, lecturing on the problem of small nations. Supported by Norman Hapgood T. G. Masaryk wrote the first memorandum to president Wilson, concerning to independence of Czechoslovak state, here in January 1917.[9]

testovací text

Личность Масарика в межвоенной Чехословакии стала объектом полуофициального культа[9]. Он рисовался наиболее авторитетным политическим и духовным лидером независимой Чехословакии (имел полуофициальное прозвище «батюшка» — *Tatíček*), воплощением этичной борьбы за независимость и создания нового государства. Подчеркивался «гуманистический» характер президентства Масарика, для него характерно высказывание: «Всякая разумная и честная политика есть реализация и укрепление принципов гуманизма. Политику, как и все, что мы делаем, следует подчинять этическим принципам. Политику, как и всю жизнь человека и общества, я не могу понимать иначе как sub specie aeternitatis». Ещё при его жизни сложился официальный культ Масарика — «Президента-освободителя»; значительный вклад в формирование «масариковского мифа» внёс Карел Чапек, автор многотомных «Бесед с Т. Г. Масариком».

ПРИВЕТ

Zpráva o revoluci v Praze a vzniku Československa dne 28. října 1918 zastihla Masaryka ještě v Americe, stejně jako zpráva o jeho zvolení prezidentem. Cestou domů navštívil už jako prezident Anglii, Francii a Itálii i české legionáře a 21. prosince 1918 byl triumfálně uvítán v Praze. Hned po volbách roku 1920 byl znovu zvolen, i když získal jen asi 65 % hlasů, a podobně i v dalších volbách roku 1927. Teprve při třetí volbě, v roce 1934, kterou ústava presidentu Osvoboditeli dovolovala a která proběhla jako manifestace pro demokracii, získal 73 % hlasů. Komunisté a slovenští nacionalisté pro Masaryka nehlasovali nikdy.[7] Koncem **roku 1935 Masaryk ze zdravotních důvodů abdikoval a 14. září 1937 zemřel. Jeho pohřeb se stal velkou národní manifestací za svobodu a demokracii.**

TEST

nejaky text