## PROJECT REPORT

## On

OCR TECHNOLOGY

USING PYTHON

*Submitted*

*In partial fulfilment*

*For the award of the Degree of*

***Bachelor of Technology***

***in Department of Computer Science Engineering***

|  |
| --- |
|  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| Submitted By: | Submitted To: |
| TUMPA BERA  SOUMYADIPTI CHATTOPADHYAY | Ms. Keya Bhattacharjee |
| DEBANJAN GANGULY  DEPANKAR METLA  RAJAT MALO | H.O.D  Computer Science Department |

**CONTENTS**

* Introduction
* Key Technologies
  + Python
  + Flask
  + OpenCV
  + NumPy
  + Tensorflow
  + Keras
* Project Description
* Application Architecture
* Process Overview
* Implementation
* Limitation & Scope of Improvements
* Practical Use Case

**ACKNOWLEDGEMENT**

Before getting into the thick of the things, we would like to express our deep gratitude to the people who helped us during the course of this project. We are grateful to our project guide **Mr. Anunay Ghosh** for his guidance throughout this project research and work

We also wish to thank all the faculty members of Information Technology and our respectable Head of Department **Ms. Keya Bhattacharjee** for their constant help and efficient teaching procedures.

**Names of students :-**

**SOUMYADIPTI CHATTOPADHYAY (26900118004)**

**TUMPA BERA(26900118001)**

**DEBANJAN GANGULY (26900118042)**

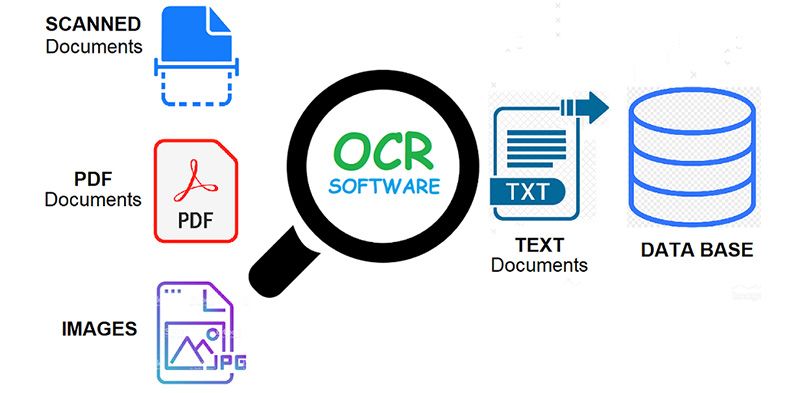
**DIPANKAR METLA (269001180018)**

**RAJAT MALO (26900118009)**

INTRODUCTION

One of the fundamental challenges of business process automation, in general, is how to efficiently extract data from semi-structured or unstructured sources, such as PDF files, and use that data to automatically perform data entry on other systems, with little or no need for human intervention.

Digitization can be implemented by converting the existing printed/typed data/documents into a digital format that is readable by the computer. Digital transformation creates greater processing efficiency and fewer risks of errors. It presents an opportunity to digitize business processes.

OCR or Optical Character Recognition-It is used to read text from images and converting them into text data for digital content management across many industries. It is mainly used as a substitute for data entry and also for information gathering, analysis purposes,andvariousotherpurposes. 

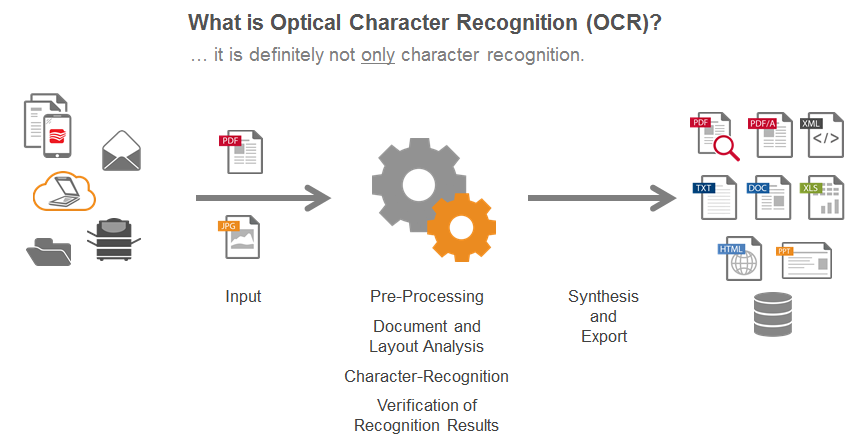
**How OCR Works**

OCR technology is used to convert images containing written text (typed, handwritten, or printed) into machine-readable text data. It is widely used as a form of data reading/entry/update from printed records.

Data reading through OCR requires three stages. Initially, some pre-processing is applied to the image which includes line removal, segmentation, de-skewing, decoloring, etc. These methods ensure the accuracy of the data that is read. Once the pre-processing is done, either Pattern Matching or Feature Extraction, are used to start the conversion process. Pattern matching makes use of stored glyphs that are equated to the data of the image being converted. This method is largely dependent on the font type and size and works best for images containing traditional fonts.

**The OCR sub-processes are as follows:**

* Pre-processing of the image
* Text Localization
* Character Segmentation
* Character recognition
* Post-Processing



**OCR Tools**

Tesseract (Freeware)

Tesseract is an open-source OCR Engine. It is a freeware available under the Apache License. It is one of the top few free OCR Engines available today. The latest version(v4) of OCR (available in GitHub) uses artificial intelligence for text recognition. It internally uses the LSTM (Long Short Term Memory) algorithm, which is based on Neural Networks logic. It currently supports the recognition of the scripts of more than 100 languages.

Tesseract has API interfaces for C++, Python, etc. and can be executed from the command line interface as well.

****

**Limitations of Tesseract**

* Since this is a freeware, it is not as accurate as some commercial solutions (Amazon Rekognition, Google Vision API) that are currently available in the markets.
* The accuracy of handwriting recognition is very low, almost unacceptable for most handwriting types.
* If the size of the noise is high, it may recognize the noise content as some ASCII characters.
* It may fail to read images with a lot of artifacts.
* It fails to read words/characters with lines crossed across them like in many bills or reports.

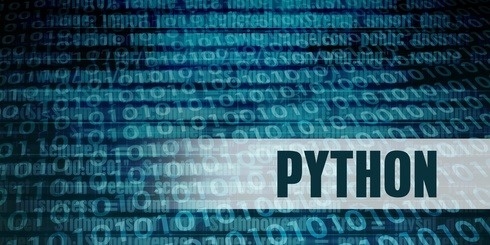
**Why OCR is the key element for all businesses**?

* OCR enables businesses to achieve higher productivity, efficiency, and reducing human costs through the automation of repetitive tasks.
* OCR increases the accessibility of data from digital documents ensuring organizations make better future decisions.
* OCR immensely improves the customer service experience by easily accessing information from text messages or voice calls.
* OCR helps to convert scanned documents and images into editable formats to update information constantly and regularly.

Key Technologies

Python

Python is a widely-used, interpreted, object-oriented, and high-level programming language with dynamic semantics, used for general-purpose programming. It was created by Guido van Rossum, and first released on February 20, 1991.



Python goals-

easy and intuitive language just as powerful as those of the major competitors;

open source, so anyone can contribute to its development;

code that is as understandable as plain English;

suitable for everyday tasks, allowing for short development times.

Python isn’t a young language. It is mature and trustworthy. It’s not a one-hit wonder. It’s a bright star in the programming firmament, and time spent learning Python is a very good investment.

Python Examples-

* Python is a great choice for: Web and Internet development (e.g., Django and Pyramid frameworks, Flask and Bottle micro-frameworks) Scientific and numeric computing (e.g., SciPy – a collection of packages for the purposes of mathematics, science, and engineering; Ipython – an interactive shell that features editing and recording of work sessions)
* Education (it’s a brilliant language for teaching programming!) Software Development (build control, management, and testing – Scons, Buildbot, Apache Gump, Roundup, Trac)

Flask

Flask is a web development framework developed in Python. It is easy to learn and use. Flask is “beginner-friendly” because it does not have boilerplate code or dependencies, which can distract from the primary function of an application.

Features of Flask

Some features which make Flask an ideal framework for web application development are:

* Flask provides a development server and a debugger.
* It uses Jinja2 templates.
* It is compliant with WSGI 1.0.
* It provides integrated support for unit testing.
* Many extensions are available for Flask, which can be used to enhance its functionalities.
* Flask is known as a micro-framework because it is lightweight and only provides components that are essential. It only provides the necessary components for web development, such as routing, request handling, sessions, and so on.

Open\_CV

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as Numpy which is a highly optimized library for numerical. which is a highly optimized library for numerical operations, then the number of weapons increases in your Arsenal i.e whatever operations one can do in Numpy can be combined with OpenCV.

Applications of OpenCV: There are lots of applications which are solved using OpenCV, some of them are listed below

* face recognition
* Automated inspection and surveillance
* number of people – count (foot traffic in a mall, etc)
* Vehicle counting on highways along with their speeds
* Interactive art installations
* Anamoly (defect) detection in the manufacturing process (the odd defective products)
* Street view image stitching
* Video/image search and retrieval
* Robot and driver-less car.

NumPY

NumPy is a short form for Numerical Python, which is applied for scientific programming in Python, especially for numbers. It comprises multidimensional objects in arrays and a package of integrating tools for Python implementation. It is basically a mix of C and Python used as an alternative for traditionally used MATLAB programming, where data in the form of numerals are treated as arrays for multidimensional functions and rearrangement operations.

Features :

* It is a combination of C and python.
* Multidimensional homogeneous arrays. Ndarray which are a ndimensional array.
* Various functions for arrays.
* Reshaping of arrays , Python can be used as an alternative to MATLAB.

Advantages

1. Numpy arrays take less space.

NumPy’s arrays are smaller in size than Python lists. A python list could take upto 20MB size while an array could take 4MB. Arrays are also easy to access for reading and writing.

2. The speed performance is also great. It performs faster computations than python lists.

TensorFlow

TensorFlow is an open-source library developed by Google primarily for deep learning applications. It also supports traditional machine learning. TensorFlow was originally developed for large numerical computations without keeping deep learning in mind. However, it proved to be very useful for deep learning development as well, and therefore Google open-sourced it.

TensorFlow accepts data in the form of multi-dimensional arrays of higher dimensions called tensors. Multi-dimensional arrays are very handy in handling large amounts of data.

**In TensorFlow, however, data can be stored and manipulated using three different programming elements:**

* **Constants**
* **Variables**
* **Placeholders**

Keras

Keras is a high-level, deep learning API developed by Google for implementing neural networks. It is written in Python and is used to make the implementation of neural networks easy. It also supports multiple backend neural network computation.

Keras is relatively easy to learn and work with because it provides a python frontend with a high level of abstraction while having the option of multiple back-ends for computation purposes. This makes Keras slower than other deep learning frameworks, but extremely beginner-friendly.

Keras allows you to switch between different back ends. The frameworks supported by Keras are:

* [Tensorflow](https://www.simplilearn.com/tutorials/deep-learning-tutorial/tensorflow)
* Theano
* PlaidML
* MXNet
* CNTK (Microsoft Cognitive Toolkit )

## 

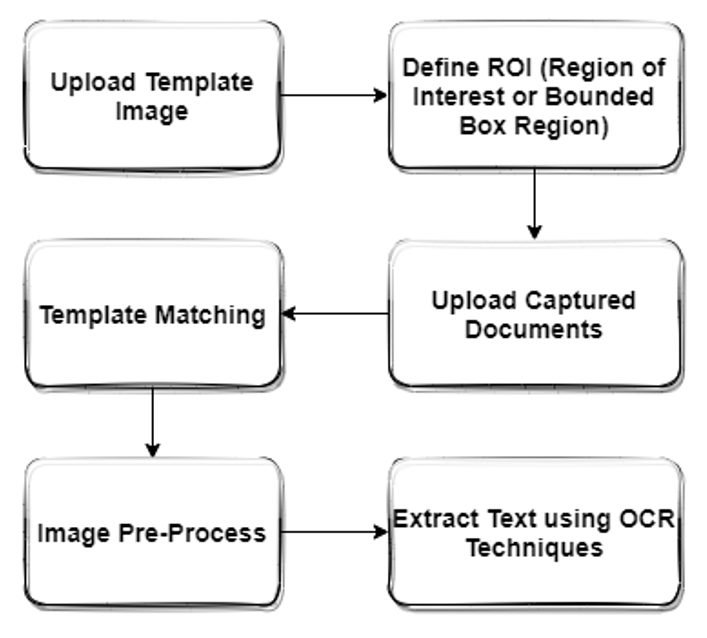
## Applications of Keras

* Keras is used for creating deep models which can be productized on smartphones.
* Keras is also used for distributed training of deep learning models.
* Keras is used by companies such as Netflix, Yelp, Uber, etc.
* Keras is also extensively used in deep learning competitions to create and deploy working models, which are fast in a short amount of time.

Project Desciption

OCR also called Optical Character Reader is a system that provides a full alphanumeric recognition of printed or handwritten characters at electronic speed by simply scanning the form. Forms containing characters images can be scanned through scanner and then recognition engine of the OCR system interpret the images and turn images of handwritten or printed characters into ASCII data (machine-readable characters). Therefore, OCR allows users to quickly automate data capture from forms, eliminate keystrokes to reduce data entry costs and still maintain the high level of accuracy required in forms processing applications.

This project has similar motivation of automating data-entry based process from any physical form, invoice, license plate, id card etc. First, User is required to upload a standard scanned document which can be referred as template. Then User will identify the ROI or Region of Interest and attach an unique label within it. On the next step, user can upload any number of captured documents, the uploaded documents then be matched with our pre-defined template using Template Matching Techniques. Once matched, we can extract the pre-defined ROI on uploaded documentation and extract the text written on it using our OCR powered engine. Then the extracted data/text will be returned to user in JSON format with their labels attached.



Application Structure

Project is in form of a custom structure using Python Flask module. Project can be served via a local for development (or a wsgi server in production) in form of a rest api response (JSON format)

## Heading 2

1. **App :**app is the heart of the project, it is separated by modules, middleware’s and app-specific services.
   1. **\_\_init\_\_.py :** all the routes registered here, also contains logic to handle http errors.
   2. **helper.py**: contains global functions
   3. **middleware.py :** contains app-specific validation rule
   4. **modules :** application logic is implemented here separating by function based folder

1.4.1: **controllers.py:** contains main business logic and routing registered to application

1.4.2 : **schema.py:** contains mapping between routes and validation rule

1.5. **services:** complex functions are separated within services folder to call from it anywhere within app when needed.

1.5.1. **open\_cv.py :** contains functions related to image pre-processing

1.5.2. **Text\_extractor.py :** contains functions related to extracting text using keras\_ocr

**2. Storage :** all user uploaded documents and app generated temp files will be stored over here and will be served as static from here.

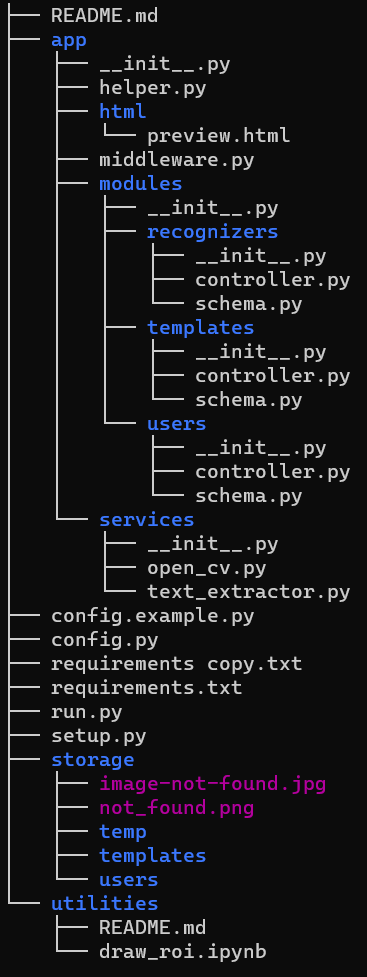
**3. Utilities :**a separated folder not linked to main project. Contains jupyter notebooks of complex function implementation throughout the project to help with various use cases like debugging, separated custom implementation etc.

**4.Config.py :** Contains app runtime configurations

**5.Requirements.txt :** Contains modules require to satisfy and bootstrap application. Can be installed with *pip3 install -r requirements.txt*

**6. setup.py :**contains scripts to install application requirements at one go.

**7.run.py :** app is served from this file



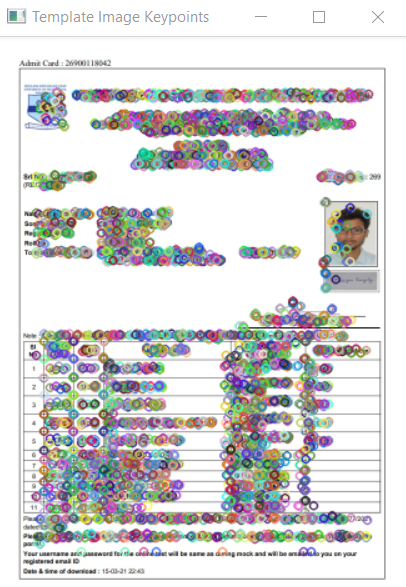
Process Description

* **Feature Extraction and Template Matching**:
* **1.1** : ORB or Oriented FAST and Rotated BRIEF is basically a fusion of FAST keypoint detector and BRIEF descriptor with many modifications to enhance the performance. First it use FAST to find keypoints, then apply Harris corner measure to find top N points among them. It also use pyramid to produce multiscale-features. But one problem is that, FAST doesn't compute the orientation.It computes the intensity weighted centroid of the patch with located corner at center. The direction of the vector from this corner point to centroid gives the orientation. To improve the rotation invariance, moments are computed with x and y which should be in a circular region of radius r, where r is the size of the patch.

Now for descriptors, ORB use BRIEF descriptors. But we have already seen that BRIEF performs poorly with rotation. So what ORB does is to "steer" BRIEF according to the orientation of keypoints. For any feature set of n binary tests at location (xi,yi), define a 2×n matrix, S which contains the coordinates of these pixels. Then using the orientation of patch, θ, its rotation matrix is found and rotates the S to get steered(rotated) version Sθ.

ORB discretize the angle to increments of 2π/30 (12 degrees), and construct a lookup table of precomputed BRIEF patterns. As long as the keypoint orientation θ is consistent across views, the correct set of points Sθ will be used to compute its descriptor.

**Our Template Image**

We’ve used ORB to detect Image Keypoints and its correspondence descriptors on our template Image, which looked like similarly below. The base keypoints to be considered is 6000.  Most useful ones are *nFeatures* which denotes maximum number of features to be retained (by default 500), *scoreType* which denotes whether Harris score or FAST score to rank the features (by default, Harris score) etc. Another parameter, WTA\_K decides number of points that produce each element of the oriented BRIEF descriptor. By default it is two, it selects two points at a time. In that case, for matching, NORM\_HAMMING distance is used. If WTA\_K is 3 or 4, which takes 3 or 4 points to produce BRIEF descriptor, then matching distance is defined by NORM\_HAMMING2.

1.2 : Brute Force Matching : Brute-Force matcher is simple. It takes the descriptor of one feature in first set and is matched with all other features in second set using some distance calculation. And the closest one is returned.

For BF matcher, first we have to create the BFMatcher object using [**cv.BFMatcher()**](https://docs.opencv.org/master/d3/da1/classcv_1_1BFMatcher.html). It takes two optional params. First one is normType. It specifies the distance measurement to be used. By default, it is [**cv.NORM\_L2**](https://docs.opencv.org/master/d2/de8/group__core__array.html#ggad12cefbcb5291cf958a85b4b67b6149fa7bacbe84d400336a8f26297d8e80e3a2). It is good for SIFT, SURF etc ([**cv.NORM\_L1**](https://docs.opencv.org/master/d2/de8/group__core__array.html#ggad12cefbcb5291cf958a85b4b67b6149fab55c78ff204a979026c026ea19de65c9) is also there). For binary string based descriptors like ORB, BRIEF, BRISK etc, [**cv.NORM\_HAMMING**](https://docs.opencv.org/master/d2/de8/group__core__array.html#ggad12cefbcb5291cf958a85b4b67b6149fa4b063afd04aebb8dd07085a1207da727) should be used, which used Hamming distance as measurement. If ORB is using WTA\_K == 3 or 4, [**cv.NORM\_HAMMING2**](https://docs.opencv.org/master/d2/de8/group__core__array.html#ggad12cefbcb5291cf958a85b4b67b6149fa7fab9cda83e79380cd273c49de8e3231) should be used.

Second param is boolean variable, crossCheck which is false by default. If it is true, Matcher returns only those matches with value (i,j) such that i-th descriptor in set A has j-th descriptor in set B as the best match and vice-versa. That is, the two features in both sets should match each other. It provides consistent result, and is a good alternative to ratio test proposed by D.Lowe in SIFT paper.

Once it is created, two important methods are BFMatcher.match() and BFMatcher.knnMatch(). First one returns the best match. Second method returns k best matches where k is specified by the user. It may be useful when we need to do additional work on that.

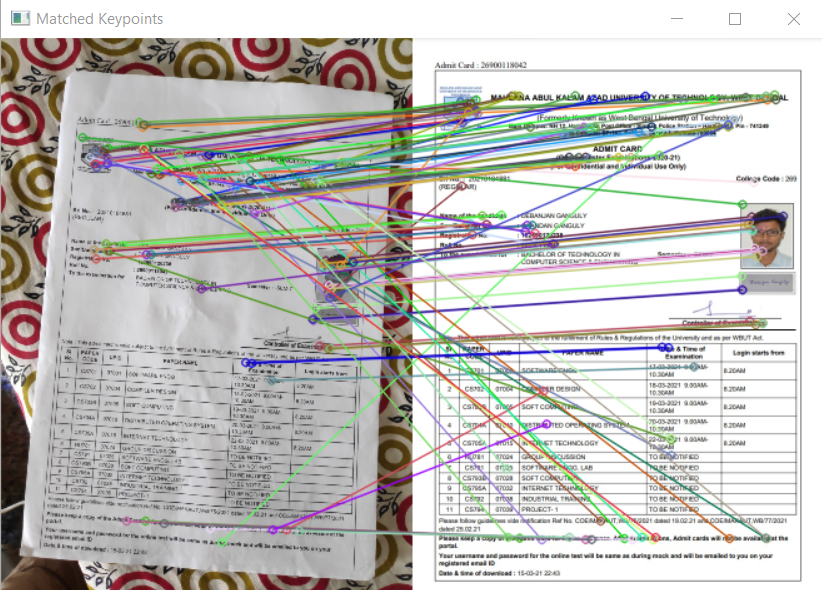
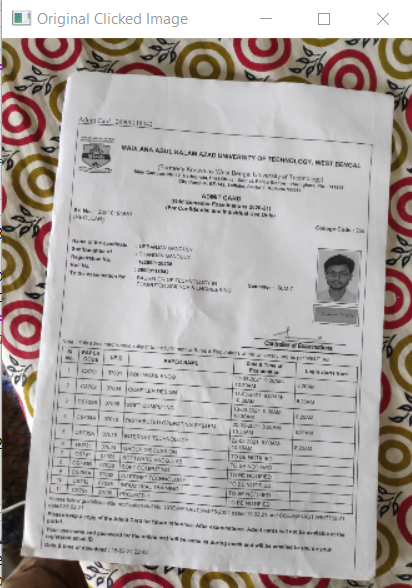
Like we used [**cv.drawKeypoints()**](https://docs.opencv.org/master/d4/d5d/group__features2d__draw.html#ga5d2bafe8c1c45289bc3403a40fb88920) to draw keypoints, [**cv.drawMatches()**](https://docs.opencv.org/master/d4/d5d/group__features2d__draw.html#gad8f463ccaf0dc6f61083abd8717c261a) helps us to draw the matches. It stacks two images horizontally and draw lines from first image to second image showing best matches. There is also **cv.drawMatchesKnn** which draws all the k best matches. If k=2, it will draw two match-lines for each keypoint. So we have to pass a mask if we want to selectively draw it.

Fig : Matched Keypoints with original Image

1.3 : Transform Image with Extracted Feature : As we’ve already extracted our required features, this information is sufficient to us to match the template with user uploaded image. Then the image should undergo geometrical transformations of 2D images. They do not change the image content but deform the pixel grid and map this deformed grid to the destination image. In fact, to avoid sampling artifacts, the mapping is done in the reverse order, from destination to the source. That is, for each pixel (x,y) of the destination image, the functions compute coordinates of the corresponding "donor" pixel in the source image and copy the pixel value:

dst(x,y)=src(fx(x,y),fy(x,y))

In case when you specify the forward mapping ⟨gx,gy⟩:src→dst, the OpenCV functions first compute thecorresponding inverse mapping ⟨fx,fy⟩:dst→src and then use the above formula.

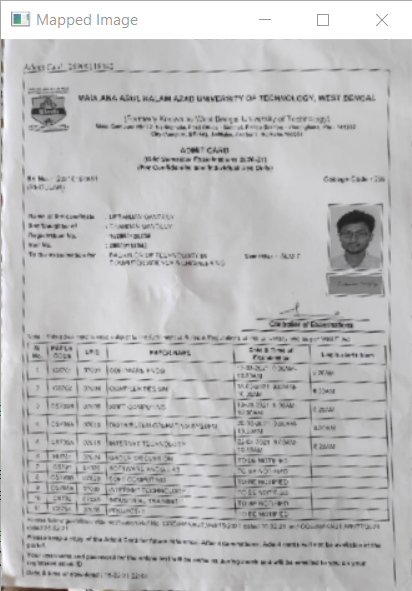


Fig: Original Clicked Image Fig: Mapped Image with Template

**Limitation**

**Limitations of using OCR for file classification**

* **Font Size**-OCR may not convert characters with very large or very small font sizes. This can make the most important characters and words unavailable for text-based systems.
* **Uni-Dimensional**-With OCR, individual words have one dimension, they’re either before or after other words. OCR does not catalog page coordinate information for characters even though page coordinates can be quite useful for classification and extracting attributes.
* **Sequential Editing**- OCR errors typically have to be corrected sequentially with the same errors being repeatedly being edited. Global spell checking can introduce other errors.
* **Case Sensitivity for Editing**- The use of spell checking to correct OCR text will typically not permit the case of the letters to be considered, e.g., cat and CAT will be treated alike.
* **Languages-** Many languages have special characters, and unless the correct OCR software is loaded, those characters can be lost or incorrectly recognized.
* **Non-Symmetrical DPI for Faxes**- Faxes are often stored in files where the number of dots per inch horizontally is not the same as the DPI vertically, and OCR engines can have difficulty with this non-symmetrical DPI.
* **Partial Text**-Document authors often incorporate graphics that have visible text. However, the OCR software may detect some text, assume that OCR is not needed, and skip processing the document leaving the text in the images invisible for text-only searching or analysis. A similar phenomenon can happen when textual headers, footers, or legends are added to previously image-only PDFs. OCR systems may detect the presence of a text layer and not attempt to convert the image layer, even though it may have the most important content.
* **Non-Textual Glyphs**- Many times there are important non-textual characters or glyphs that do not get converted to characters by OCR, leaving them invisible for text analytics or text-based retrieval, e.g., logos, or map symbols.
* **Inferring the Obvious**- Graphical elements often provide the most obvious clues as to how a file or document should be classified, e.g., placement and size of logos or text blocks. Because those graphical elements may not be directly accessible by text-restricted systems, they are left to try to infer what is most obvious to anyone just looking at the files.
* **Inferring the Obvious**- Graphical elements often provide the most obvious clues as to how a file or document should be classified, e.g., placement and size of logos or text blocks. Because those graphical elements may not be directly accessible by text-restricted systems, they are left to try to infer what is most obvious to anyone just looking at the files.
* **Incorrect Document Boundaries**-Image-only files often contain multiple documents per file and OCR does not provide a way to correct document boundaries. This causes downstream problems with systems which classify files based on comparing the words that are used within documents. Embedded documents can be missed and the ones that are classified can be misclassified. There can be similar issues for single-page TIFs where document boundaries are not obvious. For more information, see blog posting on Basic Assumptions Gone Wrong: ECM and Document Unitization, and Information Governance Lessons from 4 AFEs and a Daily Drilling Report.

**Scope of Improvements**

* **Improve Accuracy**

When it comes to manual data-entry, one of the major problems is the inaccuracy of data input. OCR, leveraged together other emerging technologies such as AI (Artificial Intelligence) & RPA (Robotic Process Automation), provides the business an effective method to automate data-entry, resulting in a reduction of errors and inaccuracies.

Minimizing human involvement solves the problem of accidentally typing in the wrong information and some other administrative errors, ultimately improve efficiency.

* **Saving Costs**

OCR is a great technology to leverage for a paperless approach. OCR helps replaces physical papers, which takes up time and spaces, as well as incur maintenance and storage costs. This means data can now be extracted and stored in digital format within business systems. As a result, businesses can reduce overheads on storage and maintenance, as well as other costs such as filing, photocopying and printing etc.

Moreover, OCR also eliminates the expensive cost of data loss due to misplacement and disasters.

* **Enhance data protection**

Data stored in the form of hard-copy papers are highly prone to become lost due to human errors, and/or natural disaster, as well as harsh weather conditions. And once those hard copies gone, it’s virtually impossible to retrieve.

OCR technology, coupled with advanced Machine Learning and other technologies, enables quick and automated extraction of data from scanned photos, consequently analyze and store them into the business system. This helps avoid the risk of data loss and mishandling – also allowing the option of recovery in case of hard-copy loss or human-error.

* **Onboarding Process Improvement**

Many financial institutions and start-ups, who provide digital baking services allowing their users to carry out banking and financial activities online, are required to comply with KYC (Know-your-customer) and AML (Anti-Money-Laundering) regulations.

Traditionally, customers are required to type in their personal information from their ID cards/passport which are susceptible to human errors. Again OCR and Machine Learning technologies enable automatic data extraction through the scanned ID/passport, which eliminates manual actions and reduce data entry errors.

With this, the onboarding process becomes much more efficient, the processing time is significantly reduced for end-users, thereby greatly improving their experience.

* **Manage OCR errors and fails**

Despite all that optimization, automatic character recognition can fail and OCR errors like wrong recognized words or names occur.

Therefore there are integrated tools for manual handling and management of OCR failures on document level for single documents or on meta level for all documents:

**Handle OCR failures by collaborative tagging and annotation**

For single documents with OCR failures you can add annotations or tags with the words that were recognized wrong by the OCR engine, so the search engine can find them despite this OCR errors because of the tags or annotations written correct.

**Manage OCR failures by thesaurus (Hidden labels)**

Manage common OCR failures for all documents by Thesaurus entries for management of OCR errors (Hidden labels)

Since you handle this OCR failure on meta level, the correction can be applied automatically to new documents with the same wrong recognized words or names.

The recommender can analyse the corpus for typos/OCR errors of a thesaurus entry and recommends such misspellings for adding to the thesaurus as hidden label by one click.

**Combining OCR results of multiple OCR tools**

No OCR engine is perfect.

So in some projects we used for example Abby Finereader to OCR the images in PDFs and additionally to the integrated Tesseract OCR. Each of them recognized words or names, on which the other OCR software failed. Because that combined and indexed both OCR results for the same document, we could find many documents more.

The Open Semantic ETL framework is able to combine or unify and index analysis results of multiple analysis or OCR tools or OCR parameters for the same document or image.

Use Case

[OCR](https://www.klippa.com/en/ocr/) (Optical Character Recognition) engine is highly skilled in capturing data from paper and digital documents. It takes our [OCR](https://www.klippa.com/en/ocr/) engine a mere couple of seconds to process a sales order or an invoice and extract all the information. Over the past years, we have come across many different use cases for data capturing. Below you can see some of the most common examples:

* **B2C Customer Services**Nowadays, customers are accustomed to fast response times. With the right technology, shorten the wait time for incoming queries and automate the administrative process. Segregating the incoming queries and sorting them in order, you can make sure that they reach the correct agent for much quicker action. With [OCR](https://www.klippa.com/en/ocr/)  quickly streamline the incoming queries and reduce the waiting time for clients.
* [**Accounts Payable – Invoice Processing**](https://www.klippa.com/en/invoice-processing/)Invoices and accounts payable (AP): the repetitiveness of recurring invoices and their large volumes lead to many hours of dull tasks for employees. People are prone to making mistakes, especially when they are bored. Computers are quite different. An OCR RPA automatically processes invoices when they come in. All the important information is parsed and placed in the respective fields in the ERP/accounting software.l. This minimizes human input, making the process less error-prone.
* **Purchase/sales orders**  
  Data should be consistent and coherent across all enterprise systems. Sales representatives spend a significant amount of time to track down all the client data and enter it into the CRM and the ERP system.Using OCR RPA, you can perform sales activities end-to-end by automating tasks such as sales order entry, invoicing, etc. This will leave you with a a clean database and improved customer experience.
* **Payslips and Payrolls**If you ask the HR department, they will surely confirm that one of the most repetitive and time-consuming tasks is processing payrolls and payslips. With OCR RPA you have consistent employee data across all different enterprise systems, and you can validate timesheets and quickly load and update earnings and deductions.
* **Storage of Client Related Information**  
  When using [OCR](https://www.klippa.com/en/ocr/) you have the ability to sort and store all the related to customer information via RPA in an organized manner. Based on the data that the engine has extracted, the system can now categorize and filter different data, such as phone numbers, addresses, purchase history, place of birth, etc. This data is automatically sent to the HR department, sales force and relevant employees. This allows employees to skip manual data entry and worries about the accuracy of results. With OCR the time it takes to do manual data entry can be reduced by 70%.
* **Refunds**Getting 5-star reviews depends highly on how quickly you processes refunds. However, repair requests, refund forms, and complaints are data-heavy and can actually take a significant amount of time to process. With [OCR](https://www.klippa.com/en/ocr/) you can increase the refund speed and the response time. Customer experience will improve, as will reviews on search engines.
* **Resumes (CVs) and recruitment processes**  
  Using [OCR](https://www.klippa.com/en/ocr/) in combination with RPA enables you to filter and categorize incoming CVs. Through a setup of keywords, you can access the value of applicants, get rid of unwanted applications or do any other task. This process streamlines recruitment and give recruiters better applicant insights.  Not to mention that with OCR even the onboarding process can go a lot faster. New employees need to send over identification documents and OCR RPA will enable you to extract all that data and place it directly into your HR software system.

**Future scope**

There have been numerous cases of computer glitches, errors in content, and security lapses reported in online examination .So in the near future the so –called software could be made more secure and reliable.While electronic glitches are rare, they have been known to occur, for instance when computer crashes voided the efforts of thousands of student. There are also cases in which the correction software has corrupted scores.  so the software can be programmed well so as to avoid corrupted scores.

**CONCLUSION**

The On line test System is developed using visual basic and sql fully meets the objectives of the system for which it has been developed. The system is operated at a high level of efficiency and all the teachers and user associated with the system understands its advantage. The system solves the problem. It was intended to solve as requirement specification.