**TXT SCAN**

**PROJECT SYNOPSIS**

OF MINOR PROJECT

**BACHELOR OF TECHNOLOGY**

B-Tech CSE

SUBMITTED BY

HARNEET SINGH(16BCS1624)

FEBRUARY 2018

****

**CHANDIGARH UNIVERSITY**

**GHARUA**

**Introduction**

**Optical character recognition** (also **optical character reader**, **OCR**) is the [mechanical](https://en.wikipedia.org/wiki/Machine) or [electronic](https://en.wikipedia.org/wiki/Electronics" \o "Electronics)conversion of [images](https://en.wikipedia.org/wiki/Image) of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo (for example the text on signs and billboards in a landscape photo) or from subtitle text superimposed on an image (for example from a television broadcast).[[1]](https://en.wikipedia.org/wiki/Optical_character_recognition#cite_note-1) It is widely used as a form of information entry from printed paper data records, whether passport documents, invoices, bank statements, computerised receipts, business cards, mail, printouts of static-data, or any suitable documentation. It is a common method of digitising printed texts so that they can be electronically edited, searched, stored more compactly, displayed on-line, and used in machine processes such as [cognitive computing](https://en.wikipedia.org/wiki/Cognitive_computing), [machine translation](https://en.wikipedia.org/wiki/Machine_translation), (extracted) [text-to-speech](https://en.wikipedia.org/wiki/Text-to-speech), key data and [text mining](https://en.wikipedia.org/wiki/Text_mining). OCR is a field of research in [pattern recognition](https://en.wikipedia.org/wiki/Pattern_recognition), [artificial intelligence](https://en.wikipedia.org/wiki/Artificial_intelligence) and [computer vision](https://en.wikipedia.org/wiki/Computer_vision).

Early versions needed to be trained with images of each character, and worked on one font at a time. Advanced systems capable of producing a high degree of recognition accuracy for most fonts are now common, and with support for a variety of digital image file format inputs. Some systems are capable of reproducing formatted output that closely approximates the original page including images, columns, and other non-textual components.

**History-**

Early optical character recognition may be traced to technologies involving telegraphy and creating reading devices for the blind. In 1914, [Emanuel Goldberg](https://en.wikipedia.org/wiki/Emanuel_Goldberg) developed a machine that read characters and converted them into standard telegraph code.] Concurrently, Edmund Fournier d'Albe developed the [Optophone](https://en.wikipedia.org/wiki/Optophone" \o "Optophone), a handheld scanner that when moved across a printed page, produced tones that corresponded to specific letters or characters.

In the late 1920s and into the 1930s [Emanuel Goldberg](https://en.wikipedia.org/wiki/Emanuel_Goldberg) developed what he called a "Statistical Machine" for searching microfilm archives using an optical code recognition system. In 1931 he was granted USA Patent number 1,838,389 for the invention. The patent was acquired by [IBM](https://en.wikipedia.org/wiki/IBM).

With the advent of smart-phones and [smartglasses](https://en.wikipedia.org/wiki/Smartglasses" \o "Smartglasses), OCR can be used in internet connected mobile device applications that extract text captured using the device's camera. These devices that do not have OCR functionality built into the operating system will typically use an OCR [API](https://en.wikipedia.org/wiki/Application_programming_interface) to extract the text from the image file captured and provided by the device. The OCR API returns the extracted text, along with information about the location of the detected text in the original image back to the device app for further processing (such as text-to-speech) or display.

### Blind and visually impaired users

In 1974, [Ray Kurzweil](https://en.wikipedia.org/wiki/Ray_Kurzweil) started the company Kurzweil Computer Products, Inc. and continued development of omni-[font](https://en.wikipedia.org/wiki/Typeface) OCR, which could recognise text printed in virtually any font (Kurzweil is often credited with inventing omni-font OCR, but it was in use by companies, including CompuScan, in the late 1960s and 1970). Kurzweil decided that the best application of this technology would be to create a reading machine for the blind, which would allow blind people to have a computer read text to them out loud. This device required the invention of two enabling technologies – the [CCD](https://en.wikipedia.org/wiki/Charge-coupled_device) [flatbed scanner](https://en.wikipedia.org/wiki/Flatbed_scanner) and the text-to-speech synthesiser. On January 13, 1976, the successful finished product was unveiled during a widely reported news conference headed by Kurzweil and the leaders of the [National Federation of the Blind](https://en.wikipedia.org/wiki/National_Federation_of_the_Blind).In 1978, Kurzweil Computer Products began selling a commercial version of the optical character recognition computer program. [LexisNexis](https://en.wikipedia.org/wiki/LexisNexis) was one of the first customers, and bought the program to upload legal paper and news documents onto its nascent online databases. Two years later, Kurzweil sold his company to [Xerox](https://en.wikipedia.org/wiki/Xerox), which had an interest in further commercialising paper-to-computer text conversion. Xerox eventually spun it off as [Scansoft](https://en.wikipedia.org/wiki/Scansoft" \o "Scansoft), which merged with [Nuance Communications](https://en.wikipedia.org/wiki/Nuance_Communications).] The research group headed by [A. G. Ramakrishnan](https://en.wikipedia.org/wiki/A._G._Ramakrishnan) at the [Medical intelligence and language engineering lab](https://en.wikipedia.org/wiki/Medical_intelligence_and_language_engineering_lab), [Indian Institute of Science](https://en.wikipedia.org/wiki/Indian_Institute_of_Science), has developed PrintTo Braille tool, an open source GUI frontend that can be used by any OCR to convert scanned images of printed books to Braille books.

In the 2000s, OCR was made available online as a service (WebOCR), in a [cloud computing](https://en.wikipedia.org/wiki/Cloud_computing) environment, and in mobile applications like real-time translation of foreign-language signs on a [smartphone](https://en.wikipedia.org/wiki/Smartphone).

[Various commercial and open source OCR systems](https://en.wikipedia.org/wiki/Comparison_of_optical_character_recognition_software) are available for most common [writing systems](https://en.wikipedia.org/wiki/Writing_system), including Latin, Cyrillic, Arabic, Hebrew, Indic, Bengali (Bangla), Devanagari, Tamil, Chinese, Japanese, and Korean characters.

**UNICODE**

Characters to support OCR were added to the [Unicode](https://en.wikipedia.org/wiki/Unicode) Standard in June 1993, with the release of version 1.1.

Some of these characters are mapped from fonts specific to [MICR](https://en.wikipedia.org/wiki/MICR), [OCR-A](https://en.wikipedia.org/wiki/OCR-A) or [OCR-B](https://en.wikipedia.org/wiki/OCR-B).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**Optical Character Recognition**](https://en.wikipedia.org/wiki/Optical_Character_Recognition_(Unicode_block)) [Official Unicode Consortium code chart](http://www.unicode.org/charts/PDF/U2440.pdf) | | | | | | | | | | | | | | | | |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| U+244x | ⑀ | ⑁ | ⑂ | ⑃ | ⑄ | ⑅ | ⑆ | ⑇ | ⑈ | ⑉ | ⑊ |  |  |  |  |  |
| U+245x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | | | | | | | | | | | | | | | | |

**Applications-**

OCR engines have been developed into many kinds of domain-specific OCR applications, such as receipt OCR, invoice OCR, check OCR, legal billing document OCR.

They can be used for:

* [Data entry](https://en.wikipedia.org/wiki/Data_entry_clerk) for business documents, e.g. [check](https://en.wikipedia.org/wiki/Check_clearing), passport, invoice, bank statement and receipt
* [Automatic number plate recognition](https://en.wikipedia.org/wiki/Automatic_number_plate_recognition)
* Automatic insurance documents key information extraction
* Extracting business card information into a contact list
* More quickly make textual versions of printed documents, e.g. [book scanning](https://en.wikipedia.org/wiki/Book_scanning) for [Project Gutenberg](https://en.wikipedia.org/wiki/Project_Gutenberg)
* Make electronic images of printed documents searchable, e.g. [Google Books](https://en.wikipedia.org/wiki/Google_Books)
* Converting handwriting in real time to control a computer ([pen computing](https://en.wikipedia.org/wiki/Pen_computing))
* Defeating [CAPTCHA](https://en.wikipedia.org/wiki/CAPTCHA) anti-bot systems, though these are specifically designed to prevent OCR. The purpose can also be to test the robustness of CAPTCHA anti-bot systems.
* Assistive technology for blind and visually impaired users

**CONCEPTS AND TECHNOLOGY USED**

Android provides a rich application framework that allows you to build innovative apps and games for mobile devices in a Java language environment. The documents listed in the left navigation provide details about how to build apps using Android's various APIs.

If you're new to Android development, it's important that you understand the following fundamental concepts about the Android app framework:

#### Apps provide multiple entry points

Android apps are built as a combination of distinct components that can be invoked individually. For instance, an individual activity provides a single screen for a user interface, and a service independently performs work in the background.

From one component you can start another component using an intent. You can even start a component in a different app, such as an activity in a maps app to show an address. This model provides multiple entry points for a single app and allows any app to behave as a user's "default" for an action that other apps may invoke.

#### Apps adapt to different devices

Android provides an adaptive app framework that allows you to provide unique resources for different device configurations. For example, you can create different XML layout files for different screen sizes and the system determines which layout to apply based on the current device's screen size.

You can query the availability of device features at runtime if any app features require specific hardware such as a camera. If necessary, you can also declare features your app requires so app markets such as Google Play Store do not allow installation on devices that do not support that feature

**Feasibility Study**

OCR is very helpful app because you can

1. Scan the files and printed text formats and in real time you can view it on your application screen
2. The OCR is best for making the hard format files easily a soft copy
3. OCR can translates the text with a voice output
4. Its decreases the human work and decreasing the time bounds
5. Study online and offline tools of the character recognition.
6. Its need is to reduce the human work.
7. It decrease the human man-power
8. It increases the time saving.
9. It will help when the user in not watching the text or document and the application will read for it.
10. The user can edit the document through an picture.
11. The previous pictures will be saved in a database.
12. This is for the ease of doing the work.

**METHDOLOGY**

Everything listed below will be added step by step.

* Using the Google’s API
* We will modify the code and the API according to our requirements.
* The size format will be changed.
* The Text formatting is done.
* The new UI/UX is Introduced.
* The Addition of new application icon.
* The new features like images will be stored in database.
* The text reading features.
* The New things like using and database.
* We will make a new activity the will include all the modules .
* The app testing will be done.
* The app will be uploaded on the play store.

**HARDWARE AND SYSTEM REQUIREMENTS.**

* **HARDWARE REQUIREMNTS**
* Laptop with internet connectivity
* Mobile
* **SYSTEM REQUIREMNTS**
* Android OS
* Android studio
* notepad
* Emulator

**BIBLIOGRAPHY**

* 1. <https://developer.android.com/>
  2. <https://developers.google.com/vision/text-overview>
  3. <https://codelabs.developers.google.com/codelabs/mobile-vision-ocr>
  4. <https://github.com/>
  5. <https://tutorialspoint.com/android/>