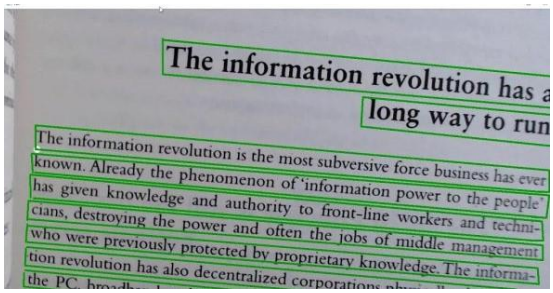
Lecture automatique du texte dans l'image d'un document



Etude bibliographique

Réalisé par :

Cao NAN
Najib IDZIM

Introduction

Le but de ce projet est de réaliser un programme informatique capable de lire du texte rédigé en français contenu dans une image. Cette dernière s'agit d'une photo d'un document prise d'un angle de vue libre. Le programme doit donc être en mesure de redresser l'angle de vue pour qu'il soit de face et ensuite d'en extraire les données. L'extraction se composera des étapes suivantes :

- La détection de la région contenant le texte
- La séparation du texte en lignes
- La séparation des lignes en mots
- La séparation des mots en caractères individuels

Le programme sera écrit en langage Python et se servira de la bibliothèque Open CV pour le traitement de l'image.

Traitement de l'image

Le traitement de l'image est décomposé en 6 étapes :

Prétraitement :

Comme son nom l'indique, cette étape préliminaire sert à préparer l'image pour le traitement qu'elle va subir dans la suite. Elle se décompose en 2 parties :

- Binarisation

La binarisation sera de convertir l'image en noir et blanc. Elle se présente comme un moyen simple et précis pour distinguer le texte de l'arrière-plan. Ce qui servira aussi à détecter les contours plus facilement.

- Filtrage

Pour filtrer les bruits et mieux détecter les contours nous utiliserons un filtre gaussien.

Etape 1 : Redressement de l'image

Le document présent sur la photo aura une forme rectangulaire. Donc le redressement de celui-ci consistera à localiser les 4 côtés du rectangle et retransformer l'image pour qu'elle soit de face. La détection des contours peut être faite avec l'outil **Canny.** Et pour redresser l'image suivant ces contours nous disposons de deux fonctions fournis par Open Cv : cv2.warpPerspective() et cv2.perspectiveTransform().

Etape 2 : Détection de la région contenant le texte

Pour cette photo, le contour plus large représente la partie avec le teste dans l'image.

Etape 3 : séparer les lignes

Il y a plusieurs façons pour séparer les lignes de teste. Comme l'image est bien redressé dans l'étape 1, on peut tracer les lignes droite horizontale sur l'image, si les mots sont en noir et l'arrière-plan est en blanc, chaque fois la ligne traverser le point en noir, on incrément une valeur correspondant. Donc pour l'espace entre les lignes de teste, on peut avoir un group de ligne qui contiens le valeur 0. Et les restes sont les lignes de teste.

Une autre façon est d'utiliser une fonction de openCV : cv2.dilate()

Etape 4 : séparer les mots

Comme la méthode on utiliser dans l'étape 3, on peut tracer les lignes verticale pour séparer les caractères.

Etape 5 : reconnaissance des caractères

Pour la détection des mots, le ROC(La reconnaissance optique des caractères, ou Optical Character Recognition – OCR en anglais) est la technologie on utilise aujourd'hui. C'est une conversion électronique d'images textuelles dactylographiées, manuscrites ou imprimées.

On a choisi d'utiliser 'tesseract' pour faire la reconnaissance des caractères. Il a bien l'autre méthodes par exemple, OCRopus, Ocular, SwiftOCR, etc. La dernière version de tesseract est basée sur LSTM(Long Short Term Memory networks) , est l'architecture de réseau de neurones récurrents(RNN) la plus utilisée en pratique qui permet de répondre au problème de disparition de gradient.

OCR : https://moov.ai/fr/blog/reconnaissance-optique-de-caracteres-ocr/
tesseract : https://github.com/tesseract-ocr/tesseract
LSTM : https://fr.wikipedia.org/wiki/R%C3%A9seau_de_neurones_r%C3%A9currents
Perspective Transform : https://www.pyimagesearch.com/2014/08/25/4-point-opencv-getperspective-transform-example/