Extraction de la valeur affichée par un compteur électrique au moyen d'une technique de reconnaissance optique des caractères

Un projet réalisé dans le cadre du cours de Traitement du Signal en 3e TI à l'EPHEC Louvain-la-Neuve par Castiaux Julien, Lambin Noé, Rousseau Mathieu et Van Waesberghe Christophe

Introduction

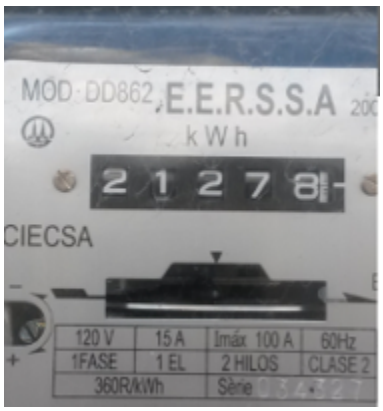
Récupérer de l'information utile depuis un domaine soumis au bruit est au coeur de notre cours de traitement du signal. Ce projet est un bon moyen pour mettre en pratique les outils qui nous ont été enseignés mais également pour nous ouvrir à un domaine de recherche scientifique.

Mise en application

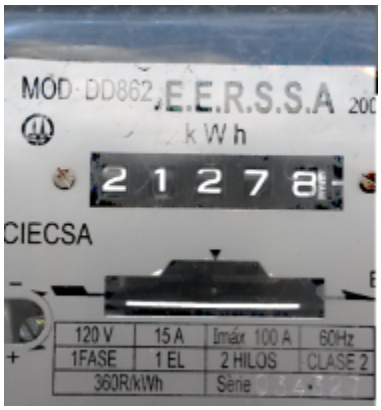
1. Récupération de l'image
2. Nettoyage de l'image :
    - Application d'un filtre de floutage afin d'atténuer les détails inutiles qui pourraient perturber la lecture
    - Conversion en niveau de gris
    - Binarisation de l'image selon un seuil paramétrable
3. Isolation de la zone d'affichage du compteur
    - Détection des contours
    - Sélection du plus grand élément ayant 4 cotés
4. Traitement de cette valeur via Pytesseract

Exemple

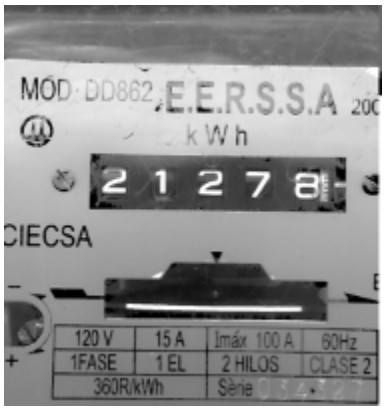
1. Récupération de l'image en couleur (le haut est bleu).



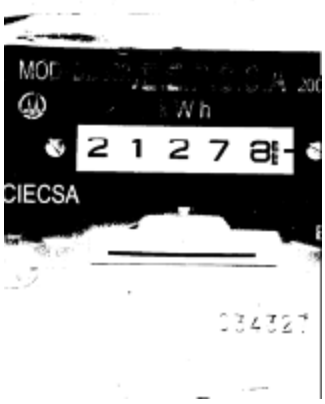
2. Premier traitement de l'image pour augmenter sa netteté.



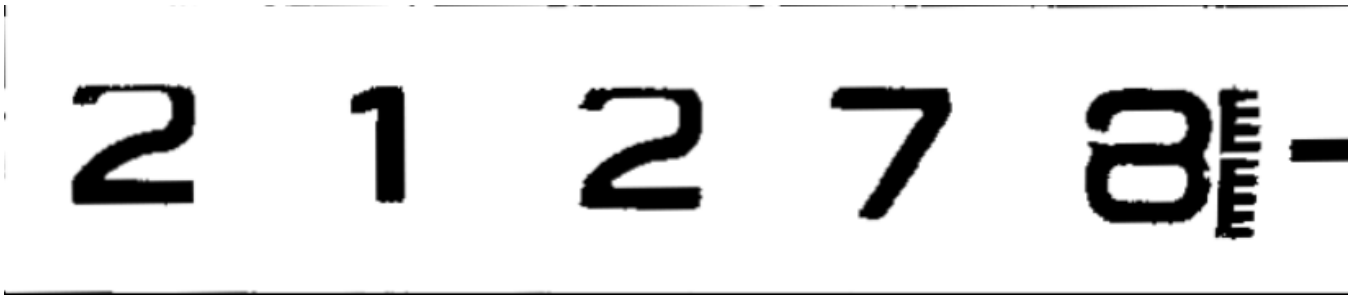
3. Conversion de l'image en RGB vers des niveaux de gris.



4. Binarisation de l'image selon un seuil fixe pour toute l'image.



5. Récupération de la zone d'affichage du compteur.



6. Traitement de cette zone d'affichage avec Pytesseract

Problèmes rencontrés et leurs solutions

Le premier objectif du projet était de pleinement utiliser les notions de produits de convolutions et de corrélations vues au cours en utilisant la librairie numpy.
Malheureusement nous avons rapidement rencontré à la fois des bugs internes à la librairie, mais également des difficultés lorsque les calculs matriciels devenaient plus complexes.
La solution fut donc d'utiliser Pytesseract, une librairie de détection et de lecture automatique de caractère dans une image.

Bien que très puissante, la librairie Pytesseract rencontre des difficultés pour lire les compteurs électriques car ceux-ci contiennent trop d'information inutile.
Il fallait donc augmenter la netteté de l'image et à la binariser pour supprimer les informations inutiles.

Ensuite, nous avons dû trouver un moyen de sélectionner la zone d'affichage du compteur.
Pour ce faire nous avons utilisé une méthode de détection de bord pour rechercher la zone rectangulaire de l'image contenant l'affichage du compteur.

Limitation et améliorations possibles

Actuellement, l'OCR ne fonctionne correctement qu'avec des images de bonne qualité et certains types de compteurs.
Une amélioration possible serait donc d'améliorer le traitement de l'image en implémentant des techniques de suppression/atténuation des reflets/parasites sur l'image.

Il serait également bien d'améliorer l'algorithme d'isolation de la zone d'affichage du compteur. En effet, se baser sur un "objet" de l'image qui a 4 sommets fonctionne qu'avec des images bien calibrées et ne contenant pas d'autres éléments pouvant être confondu avec cette zone d'affichage.

Pour le moment le seuil de binarisation est fixe pour toute l'image, une amélioration possible serait d'avoir un seuil adaptatif pour s'adapter au changement de lumière sur l'image.

Conclusions personnelles

Julien Castiaux

Il a été intéressant d'implémenter des fonctions de traitement d'image et d'avoir une réflexion pour trouver une méthode efficace pour rendre un compteur lisible par tesseract. Travailler avec git et des tests unitaires nous aura fait gagner beaucoup de temps.

À côté de l'aspect technique, je suis content d'avoir pu partager ma passion pour le langage Python avec mon groupe et de les sentir enthousiastes.

Noé Lambin

Ce projet a été pour moi l'occasion de découvrir et d'apprendre à maitriser Python.
Ce fut aussi l'occasion de tenter de mettre en pratique les techniques de produit de convolution et de corrélation vues lors du cours théorique.
J'ai aussi pu découvrir la partie mathématique qui se cache derrière le traitement d'image.

Mathieu Rousseau

Au démarrage de ce projet, je ne connaissais pas du tout comment fonctionnait un OCR.
Ce projet m'a permis d'apprendre les différentes techniques de traitements de l'image nécessaires à la bonne extraction des caractères de celle-ci (dans ce cas une valeur de consommation).
J'ai également réappris le langage Python.

Christophe Van Waesberghe

Entre ce défi et le projet d'intégration, python aura été au coeur de mon apprentissage ! Découvrir le fonctionnement d'un OCR et l'application d'un cas pratique de traitement du signal a été très intéressant tant dans la réflexion, le travail d'équipe et la persévérance !

Conclusion

Dans le domaine de la recherche scientifique, la programmation n'est plus une finalité comme dans une application ou un site web, elle ne devient plus qu'un outil mis au service du domaine de recherche. Cette approche différente de ce que nous avons l'habitude de faire, peut être déroutante ; on peut être très bon en python sans pour autant avoir facile à réaliser ce projet.

La solution n'est pas systématique, elle requiert un investissement en temps, en réflexion et en recherche.

Bibliographie

- Dewulf, A. (2017). *Les techniques de traitement du signal*. Syllabus, École Pratique des Hautes Études Commerciales, Louvain-la-Neuve.
- Anto59290. (2017). *Introduction au traitement d'image*. En ligne https://zestedesavoir.com/tutoriels/1557/introduction-au-traitement-dimage/
- Kompf, M. (2018). *OpenCV practice: OCR for the electricity meter*. En ligne sur le site de Martin Kompf https://www.mkompf.com/cplus/emeocv.html
- Goyal, M. genpfault. (2014). *Detect text region in image using Opencv*. En ligne https://stackoverflow.com/questions/24385714/detect-text-region-in-image-using-opencv consulté le 4 décembre 2017
- donpresente. (2016). *Python + OpenCV: OCR Image Segmentation*. En ligne https://stackoverflow.com/questions/40443988/python-opencv-ocr-image-segmentation consulté le 4 décembre 2017
- Legrand, F. (sd). *Détection des bords*. En ligne sur le site de Frédéric Legrand http://www.f-legrand.fr/scidoc/docmml/image/filtrage/bords/bords.html
- Legrand, F. (sd). *Filtrage d'une image par convolution*. En ligne sur le site de Frédéric Legrand http://www.f-legrand.fr/scidoc/docimg/image/filtrage/convolution/convolution.html
- Python Image Search. (2015). *Zero parameter automatic canny edge detection*. En ligne https://www.pyimagesearch.com/2015/04/06/zero-parameter-automatic-canny-edge-detection-with-python-and-opencv/

Bibliographie

- Python Image Search. (2017). *Recognizing digits with opencv dans python*. En ligne https://www.pyimagesearch.com/2017/02/13/recognizing-digits-with-opencv-and-python/
- Python Image Search. (2017). *Bank check ocr with opencv and python part I*. En ligne https://www.pyimagesearch.com/2017/07/24/bank-check-ocr-with-opencv-and-python-part-i/
- OpenCV documentation. (2017). *Image Thresholding*. En ligne https://docs.opencv.org/3.3.1/d7/d4d/tutorial_py_thresholding.html