Research Notes

**[1]: Text Extraction from Bills and Invoices**

* Used Tools: **OpenCV** and **Tesseract OCR**
* The Tesseract OCR engine has poor quality output if the input images are too noisy or contain unnecessary objects.
* Images have to be **preprocessed** before being passed to Tesseract
* Firstly, an **edge detection mechanism** is applied to the image and then contour tracking is performed.
* A four-point contour is searched; the **largest four-point contour is the required object** from the image; the detected object is cropped out from the original image and some filters are applied to it afterwards
* **Canny Edge Detection Algorithm**: uses a multi-stage algorithm to detect a wide range of edges in images – accurate and easy to use
* Segmentation process:
  + Line segmentation
  + Word segmentation
  + Character segementation
* Drawbacks:
  + works only for non-handwritten bills (since Tesseract uses the first character’s position and uses it to try to read the whole line)
  + will not work if the piece of paper is not rectangular

**[2]: Application of OCR systems to processing and digitization of paper documents**

* Documents with coloured or patterned backgrounds, marked with pens or crooked when scanned, can yield poor OCR results.
* Tests that were performed: recognition of invoice number and date
  + Total number of processed documents: 1000
  + 100% of invoice numbers were recognized correctly
  + 98% of dates were recognized correctly

**[4]: An overview of the Tesseract OCR engine**

* The first step is a connected component analysis in which outlines of the components are stored (computationally expensive step)
* Outlines are gathered together into Blobs

**[5]: Data Extraction From Invoices Using Computer Vision**

* Before text recognition is launched, the image is analyzed for light and dark areas to recognize every character or numeric digit
* Noisy Image -> Gaussian Filtering -> Otsu’s Thresholding
* Image is converted from **RGB to Grayscale**
* Afterwards, a thershold is applied; there are three types of thresholding:
  + Simple Thresholding
  + Adaptive Thresholding
  + **Otsu’s Thresholding**

When removing noise, **white noises are removed by erosion**, and afterwards, **dilation is applied for joining broken parts** of an object

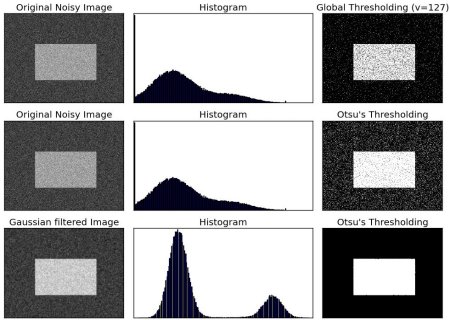
* Contours are found in the dilated image
* Individual contours are looped through, and rectangles are drawn
* After each rectangle is drawn, they are passed further to Tesseract for data extraction

**[7]: Investigation on the effect of a Gaussian Blur in image filtering and segmentation**

* If an image contains noise, Gaussian blur is recommended

**[8]: OpenCV24-Python-Tutorials**

* Images can be filtered using low-pass filters (LPF), in order to remove noise/edges or blur the image, or high-pass filters (HPF), in order to help find edges in images
* **Gaussian Filtering:**
  + Highly effective in removing Gaussian noise (e.g. noise caused by poor illumination) from an image
* **Simple Thresholding:**
  + If a pixel value is higher than a threshold value, it is assigned one value (e.g. white), else, it is assigned another value (e.g. black)
* **Otsu’s Binarization:**
  + **Image Histogram:** the graphical representation of the tonal distribution in a digital image (wikipedia)
  + **Bimodal image:** image whose histogram has two peaks
  + Otsu’s binarization calculates a threshold value from the image histogram for a bimodal image
  + THRESH\_BINARY + THRESH\_OTSU parameter



* **Morphological Transformations:** 
  + **Erosion:**
    - erodes away the boundaries of the foreground object
    - a pixel in the original image will be considered 1 only if all the pixels under the kernel have a value of 1; otherwise, it is eroded, i.e. made to 0
    - all the pixels nead the boundary will be discarded depending upon the size of the kernel -> the thickness of the foreground object decreases
    - useful for removing small white noises
    - \*\*\* (insert example image here) \*\*\*
  + **Dilation:**
    - the opposite of erosion, a pixel element will be considered 1 if at least one of the pixels under the kernel is 1
    - increases the white region in the image
    - useful in joining broken parts of an object
    - \*\*\* (insert example image here) \*\*\*
  + **Opening:**
    - erosion, followed by dilation
    - \*\*\* (insert example image here) \*\*\*
  + **Closing:**
    - dilation, followed by erosion
    - \*\*\* (insert example image here) \*\*\*

**[10]: Otsu's thresholding method based on gray level-gradient two-dimensional histogram**

* Image segmentation subdivides an image into subareas or objects
* Image thresholding plays an important role in the segmentation of an image
* **Otsu’s thresholding** produces better segmentation results for large targets, and has better noise-resistance
* The obtained regions are more consistent, and the edge of the obtained target is clearer

**[11]: Reviewing Otsu’s method for image thresholding**

* **Otsu’s thresholding** algorithm assumes that the image is composed of two basic classes: **Foreground and Background**
* It involves iterating through all the possible threshold values and calculating a measure of spread for the pixel levels each side of the threshold, i.e. the pixels that fall either in foreground or in background
* The aim is to find the threshold value where the sum of foreground and background spreads is at its minimum

[1]: H. Sidhwa, S. Kulshrestha, S. Malhotra and S. Virmani, "Text Extraction from Bills and Invoices," 2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), 2018, pp. 564-568, doi: 10.1109/ICACCCN.2018.8748309.

[2]: Stoliński, Sebastian, and Wojciech Bieniecki. "Application of OCR systems to processing and digitization of paper documents." *Information Systems in Management VIII* 102 (2011).

[3]: Jiju, Alan, Shaun Tuscano, and Chetana Badgujar. "OCR text extraction." *International Journal of Engineering and Management Research* 11.2 (2021): 83-86.

[4]: Smith, Ray. "An overview of the Tesseract OCR engine." *Ninth international conference on document analysis and recognition (ICDAR 2007)*. Vol. 2. IEEE, 2007.

[5]: M. S. Satav, T. Varade, D. Kothavale, S. Thombare and P. Lokhande, "Data Extraction From Invoices Using Computer Vision," *2020 IEEE 15th International Conference on Industrial and Information Systems (ICIIS)*, 2020, pp. 316-320, doi: 10.1109/ICIIS51140.2020.9342722.

[6]: Tsomko, E., H. J. Kim, and E. Izquierdo. "Linear Gaussian blur evolution for detection of blurry images." *IET image processing* 4.4 (2010): 302-312.

[7]: Gedraite, Estevão S., and Murielle Hadad. "Investigation on the effect of a Gaussian Blur in image filtering and segmentation." *Proceedings ELMAR-2011*. IEEE, 2011.

[8]: <https://opencv24-python-tutorials.readthedocs.io/>

[9]: https://docs.opencv.org/4.x/

[10]: Chen Yu, Chen Dian-ren, Li Yang and Chen Lei, "Otsu's thresholding method based on gray level-gradient two-dimensional histogram," *2010 2nd International Asia Conference on Informatics in Control, Automation and Robotics (CAR 2010)*, 2010, pp. 282-285, doi: 10.1109/CAR.2010.5456687.

[11]: BANGARE, Sunil L., et al. Reviewing Otsu’s method for image thresholding. *International Journal of Applied Engineering Research*, 2015, 10.9: 21777-21783.