Journal of Engineering Science and Technology
Vol. 13, No. 2 (2018) 540 - 557
© School of Engineering, Taylor's University

DOCUMENT TEXT DETECTION IN VIDEO FRAMES ACQUIRED BY A SMARTPHONE BASED ON LINE SEGMENT DETECTOR AND DBSCAN CLUSTERING

HASSAN EL BAHI*, ABDELKARIM ZATNI

Metrology and Information Processing Laboratory (LMTI)
Ibn Zohr University, Agadir, Morocco
*Corresponding Author: hassan.elbahi@edu.uiz.ac.ma

Abstract

Automatic document text detection in video is an important task and a prerequisite for video retrieval, annotation, recognition, indexing and content analysis. In this paper, we present an effective and efficient model for detecting the page outlines within frames of video clip acquired by a Smartphone. The model consists of four stages: In the first stage, all line segments of each video frame are detected by LSD method. In the second stage, the line segments are grouped into clusters using the DBSCAN clustering algorithm, and then a prior knowledge is used in order to discover the cluster of page document from the background. In the third and fourth stages, a length and an angle filtering processes are performed respectively on the cluster of line segments. Finally a sorting operation is applied in order to detect the quadrilateral coordinates of the document page in the input video frame. The proposed model is evaluated on the ICDAR 2015 Smartphone Capture OCR dataset. Experimental results and comparative study show that our model can achieve encouraging and useful results and works efficiently even under different classes of documents.

Keywords: Document text detection, video text segmentation, Smartphone, line segment detector, DBSCAN clustering.

1. Introduction

The detection task is one of the most important research areas that gained an increasing attention in image processing and machine vision. This task aims to find automatically the location and scales of the regions of interest presented in an image or in video. These regions could be: traffic sign, license plate, car, animal, person, text, or human faces, etc.

Nomenclatures

d	The length of a line segment.
Eps	Maximum distance neighbourhood between points in a cluster
L_i	A line segment.
m_i	The slope of a line segment.
MinPts	Minimum size of points necessary to form a cluster
N	The total number of line segments that exist in a frame
RA	Line-support region angle
TAHight	The high angle threshold.
TALow	The low angle threshold.
Tlength	The threshold length of a line segment.

Greek Symbols

θ_{ij}	The angle between the two lines segments.

Abbreviations

CC	Connected Component
DBSCAN	Density-based spatial clustering of applications with noise
ICDAR	International Conference on Document Analysis and Recognition
JI	Jaccard index
LSD	Line segment detector
OCR	Optical character recognition

Therefore, automatic detection has applied in many applications, among these applications, we mention: Such as: biometrics, medical, driver assistance, visual surveillance, security, human machine interface and robotics and so on. In recent years, a large number of videos are uploaded and shared every day on YouTube, social networks and TV Channels. Consequently, the automatic indexing and extracting of information from these videos is an issue of great importance and an indispensable process. This is why many companies and research labs contribute and invest in developing efficient algorithms and systems to ensure the segmentation and detection task.

In this work, we focus only on the operation of detecting the page outlines within frames. This operation aims to separate quadrilateral shape of the document page from the background of the image. The relevant text and information can be presented in different types of document image such as: datasheet, maps, newspapers, letter, mail envelopes and magazine pages. However, the large changes in text fonts, colors, styles, scales, and the poor contrast between the text region and the background of the image, in addition to the fact that the document text may comprise several types of text information (word, sentence, paragraph, title, subtitle, list, etc.), frequently make text extraction operation more challenging.

For many decades, text segmentation research has mainly focused on scanned documents. Detecting and segmenting text region in video frames acquired by a smartphone has received relatively low attention. However, segmenting the document region from this kind of video clips that obtained via mobile camera presents a challenge because of many difficulties: variety of text formats, document classes, complex background, perspective distortion, illumination variation, poor focusing and motion blurs [1].

In order to cope with the above-mentioned challenges, in this paper, a new model for detecting and extracting the text region from video that was recorded using a smartphone is proposed. This model will allow us to extract the quadrilateral coordinates of the page outlines of each frame of a video clip. The proposed model begins by detecting the line segments in each frame of the video clip. Then an unsupervised clustering algorithm has been adapted to discover the group of the line segments which belonging to the document region. After that, the final candidate of the quadrangle of the page outline is determined after applying many length and angle filters on the line segments. In experiments, the proposed model is evaluated using the dataset from ICDAR 2015 Smartphone Capture OCR [2].

The remaining of this paper is organized as follows: After an overview of the state of the art (Section 2), we describe the proposed model for detecting the contours of the page outlines in the video frames. The experimental results and comparative studies are presented in Section 3. Finally, a conclusion is drawn in Section 4.

2. Related Work

A great number of methods have been proposed in the literature for text region detection in video frames, they can be categorized into three categories, namely: 1) connected component based methods (CCs), 2) texture based methods and 3) gradient based method.

The methods based on the connected components analysis attempt to exploit the properties of text components in video frames for the detection and separation of the text region from the background of the frame. Koo and al [3], proposes a text-line detection method for camera-captured document images. It allows the extraction of CCs based on maximally stable extremal region technique, after the scale and the orientation of each CC are determined using projection profiles technique. Mittal and al [4], introduced a new method for detecting text in video frames. It begins with frame enhancement using super resolution approach, next, the histogram of oriented moments descriptor is employed in order to extract features from CC. Finally, the classifier of support vector machine is trained to identify text/non-text regions. Yi et al. [5] proposed a text detection algorithm by using layout analysis based on character candidate localization and text classification based on extracting character structure features. Yin et al. [6] proposed an accurate and robust method for the detection of the text in natural scene images by using MSER, clustering algorithm and character classifier. However, this type of methods is sensitive and cannot yield encouraging results in the case of the presence of a complex background or low contrast

To address the problems mentioned in the previous category, many texture-based techniques have been proposed. This class uses the fact that text regions have independent textural characteristics, which distinguish them from the background. Shekar et al. [7] introduced a new hybrid method for text localization in image and video frame by using both discrete wavelet transform and gradient difference. Shivakumara et al. [8] presented a new method for the detection and localization of text in video frames based on wavelet-moments. First, text frame classification is performed based on wavelet and median moments. After that, the angle projection boundary growing approach is used to detect multi-oriented text in video frames. Sudir and al [9], presented a new framework that combines an edge and a texture based approach in order to detect curved text in video frame. Liang et al. [10] introduced an approach based on using Laplacian and wavelet high frequency sub-bands through fusion at multi-level for localization and detection of arbitrarily-oriented text in video frames. However, determining the texture characteristics for each text component is a difficult task. Therefore, the disadvantage of this category is that it is consuming a lot of time in the calculation

The last category of methods is based on the gradient. It assumes that the region of the text has a high contrast and a symmetric gradient with respect to the region of the background. Therefore, they are very popular compared to methods based on related components and texture because of its simplicity and efficiency. Liu et al. [11] proposed a new technique for detecting and extracting caption text from video frames. It used a new stroke-like edge detector that relies on the contours, which allows eliminating the non-caption edges from background. Next, inter-frame information and a post-processing step are performed in order to improve the accuracy of caption detection and segmentation. Zhang and al. [12] introduced a new system that combines character and link energies in order to detect text regions in mages and video frames. Shivakumara et al. [13] proposed a new technique to detect multi-oriented text in video scene. It starts with the application of Laplacian and Sobel operations to improve text regions in videos. After that, the Bayesian classifier and a boundary growing method are used to locale and detect the text pixels in video frames. Huang et al. [14] presented a new method based on stroke information and Laplacian to detect text regions in TV news videos. Nevertheless, the main weakness of this category is the use of the geometric properties of regions in images or frames, which affects their performance in the case of background variation and contrast.

3. Proposed Model

In this section, a new model for detecting the quadrilateral coordinates of the page outlines within frames acquired using a Smartphone will be presented. The overall architecture of the proposed model is presented in Fig. 1. The model consists of four major steps. Firstly, we start by detecting the line segments in each frame by using the LSD method [15]. Then, we apply the DBSCAN clustering algorithm in order to find the lines of the paper sheet region. Finally, the quadrilateral coordinates of the page outlines are determined after applying length and angle filtering of line segments. In the following, we will explain the proposed model in detail.

3.1. Line segment detector

The line segments constitute significant geometric information of an input frame, particularly when the frame contains many shapes. Detection methods are generally composed of two different stages: candidate selection and validation process. Obviously, in the first stage, it is important not to have the false negatives (a shape is present in the image, but it was not detected). Thereafter, the validation process should minimize the number of false positives (an object is not present in the image, but the method has detected one).

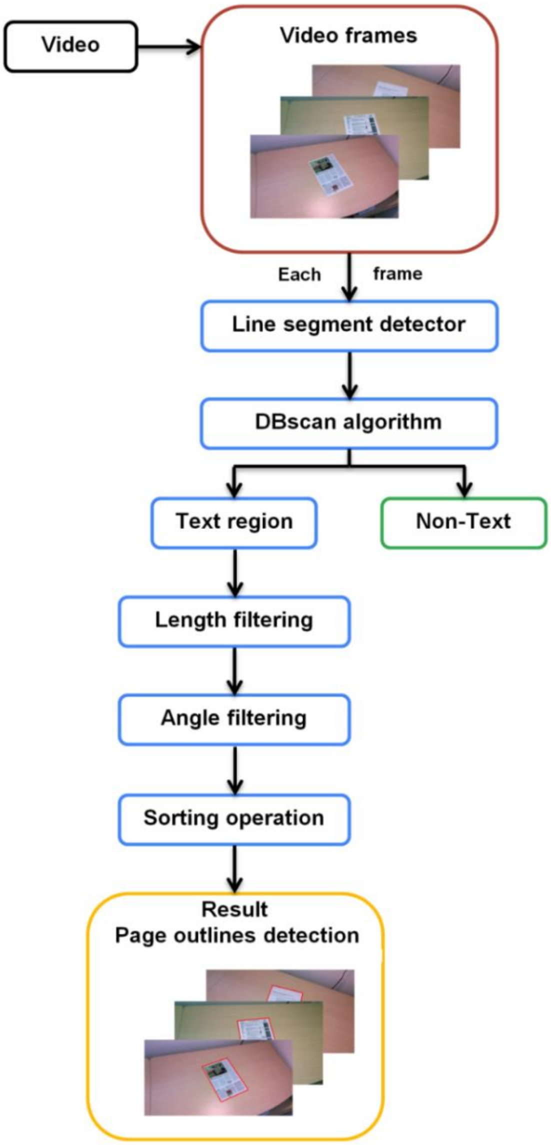


Fig. 1. The flowchart of our proposed model.

Classic approaches apply the Canny edge detector [16] and after that apply the Hough Transform in order to find all lines in an image or a frame, but the results are not encouraging both in terms of calculation time and accuracy rate. In this work, we use a new Line Segment Detector (LSD) method proposed by R. Von Gioi [15] that based on the methodology of candidate selection and validation process that allows us to find all the lines in all frames of a video obtained with mobile devices. The LSD method has three main stages:

- Division of each image (frame) into line-support regions by assembling the connected pixels that have a similar gradient orientation up to a chosen tolerance angle.
- Associate a rectangular approximation for each line-support region in order to determine its line segment.
- Validate or not each line segment by using a *contrario* model.

Stage 1 and 2 are inspired from [17] and some improvements were performed, while stage 3 is fully different and based on [18] method.

In the first stage the method starts by selecting one pixel which having the highest gradient magnitude and the region angle were fixed to the level-line angle at that pixel (the angle orthogonal to the gradient angle). Afterwards, a region growing algorithm is implemented and each pixel neighbour to line-support region is tested: the pixels that have a level-line orientation similar to the whole region angle up to some tolerance are added to line-support region. After adding each new pixel into the region, the line-support region angle (RA) is changed to:

$$RA = arctan\frac{\sum_i \sin(ang_i)}{\sum_i \cos(ang_i)} \tag{1}$$

where ang denotes the angle between two segments. This operation is repeated until no new pixel can be added to the present line-support region.

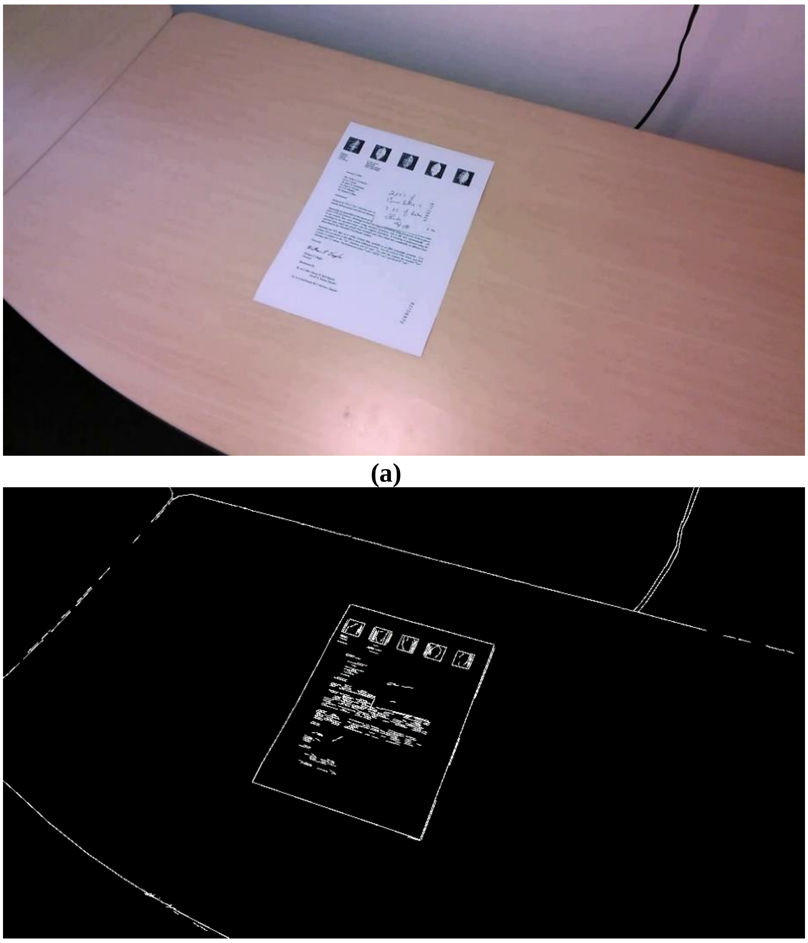
In the second stage, each line-support region is associated to a rectangle in order to approximate a candidate line segment. Normally, a rectangle was defined by its size, width centre and orientation. The center and the orientation are determined respectively by the centre of mass and the principal inertial axis of the line-support region. The length and the width of the rectangle are determined in such a way that all pixels in a line-support region are covered.

In the last stage, each line segment candidate on the image (frame) is tested with a view to classify it into a valid line segment or not. To accomplish this task the LSD method uses a contrario validation strategy [18, 19], which considerate the detection of line segment as a simplified hypothesis testing problem. The strategy is based on counting the number of pixels in each rectangle, and the number of aligned pixels (pixels with a level-line orientation similar to the line segment angle up to some tolerance); these two numbers are used as a threshold criterion to preserve a line segment as a valid detection.

Before applying the LSD detector, we must first convert the input frame from RGB color image format to gray scale intensity format. Figure 2 shows the result of applying the LSD method on an input video frame.

3.2. DBSCAN algorithm

The clustering algorithms belong to the automatic unsupervised segmentation approaches, which try to find a connected cluster without any prior information about the cluster. Generally, it allows discovering regions of high density that are separated by regions of spares areas in a dataset. Various clustering algorithms are proposed in the literature, in this work we will use the density-based spatial clustering of applications with noise algorithm [20, 21], habitually named DBSCAN.



(a)

(b)

Fig. 2. Line segments detection. (a): Input frame. (b): Detection result.

Contrary to other clustering algorithms which demanding many parameters, the DBSCAN algorithm takes two input user-defined parameters: the minimum size of points necessary to form a cluster (*MinPts*) and the maximum distance neighbourhood between points in a cluster (*Eps*). After that, DBSCAN allows classifying the data points into three types: core points, border points and noise points. A point *p* is labelled as a core point if it has the number of points in its neighbours higher than the minimum value of *MinPts* within the Eps distance. A point *q* is labelled as a border point if it has less than *MinPts* points, but it's a neighbour of a core point. A point *o* is labelled as a noise point if it has a distance (*Eps*) further from all core points, or have a few neighbours' points less than *MinPts* points.

The DBSscan algorithm begins by initializing the identification number of clusters by 0, and selecting a random initial point that has not been attributed to a cluster or being considered as a noise, and then compute the number of all the neighbouring points within *Eps* distance of the initial selecting point. If this number is higher than or equal to *MinPts*, a cluster will be created around this point, if not, this point will be labelled as an outlier. The cluster created will contain the initial point and its neighbours, and the initial point is designated as visited point. Subsequently, the DBSCAN algorithm repeats this step for all the neighbours until getting the case in which the number of neighbours of a point is less than *MinPts*, therefore this point is designated as noise. Once the cluster is formed completely, the identification number of clusters will be incremented, and the DBSCAN proceeds to select a new point among the remaining unvisited points in the dataset to begin creating a new cluster. These steps will be repeated until all the points in the dataset will be either attributed to a cluster or considered as a noise. Figure 3 illustrates the principle idea of the DBSCAN clustering algorithm.

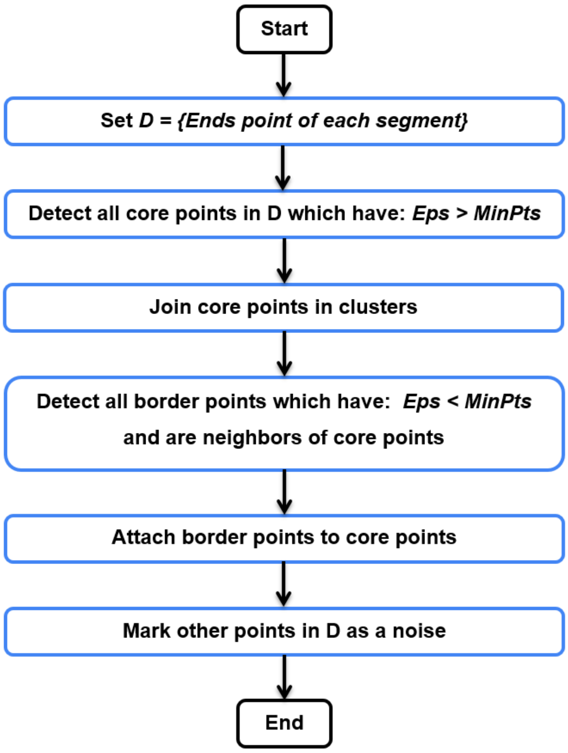
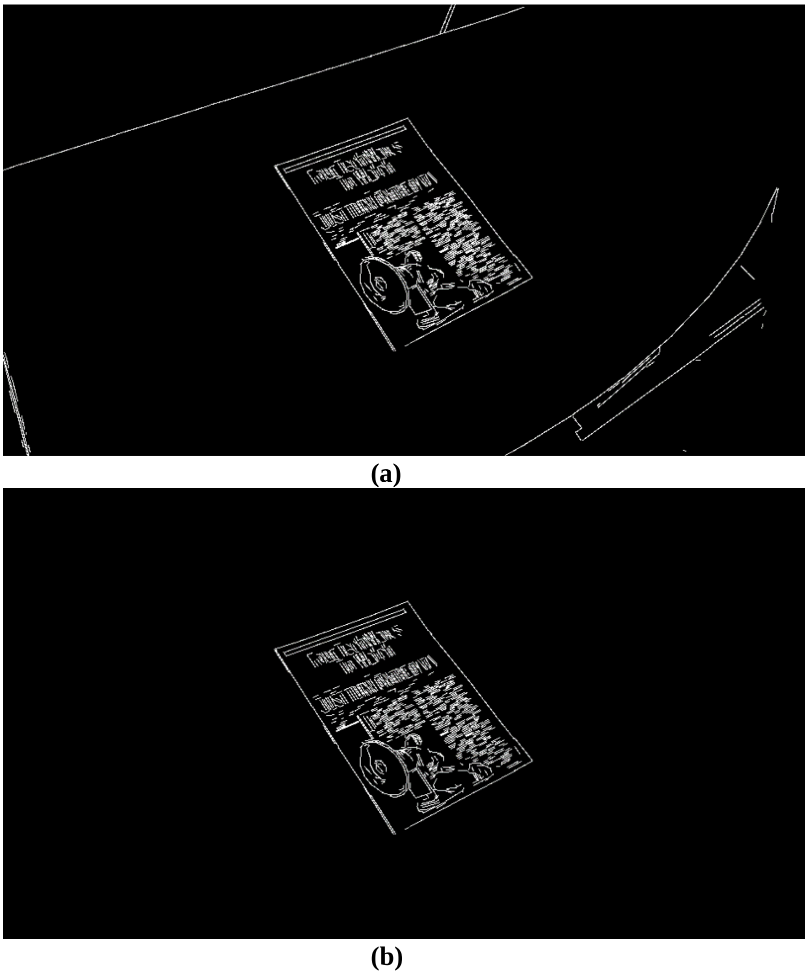

Start
Set *D = {Ends point of each segment}*
Detect all core points in D which have: *Eps > MinPts*
Join core points in clusters
Detect all border points which have: *Eps < MinPts*
and are neighbors of core points
Attach border points to core points
Mark other points in D as a noise
End

Fig 3. Flowchart of the DBSCAN algorithm.

In this work, the DBSCAN clustering algorithm is used to find all connected clusters in each frame, one of these clusters is the region of interest that represent the document text in the frame. As mentioned earlier, the reason of employing DBSCAN algorithm is that it does not demand any prior information about the number of regions in the frame; also it allows us to remove the noise data. This noise is habitually appearing in regions surrounding the document text region. In other word, after obtaining all line segments in each frame from the previous step of our model, the next step preserves only the line segments that represent the edge and the content of the document text in the frame. If *N* represents the total number of line segments that exist in the frame, then we take the two endpoints for each segment in order to form a dataset of points for the current frame, hence we get a total of *2*N* points. After that, this dataset will be presented as an input to the DBSCAN algorithm. Here the use of the DBSCAN will automatically give us all the clusters present in the dataset formed.

The last step is to find the cluster of interest. For this reason, we use a priori knowledge about the structural properties of text area to build a way to select the document text cluster. The prior knowledge used is that the main content in all the frames in each video is a text document, as well as the line segments of text region are placed in a compact way and disposed in a cluster. So, after many experiences, we have concluded that the cluster of the document text is the cluster that contains the largest number of points. This makes it possible to remove all small clusters around the text cluster. Figure 4 presents the result of applying the DBSCAN clustering algorithm.



(a)

(b)

Fig. 4. Application of the DBSCAN clustering algorithm.
(a): Initial frames. (b): Result of DBSCAN algorithm.

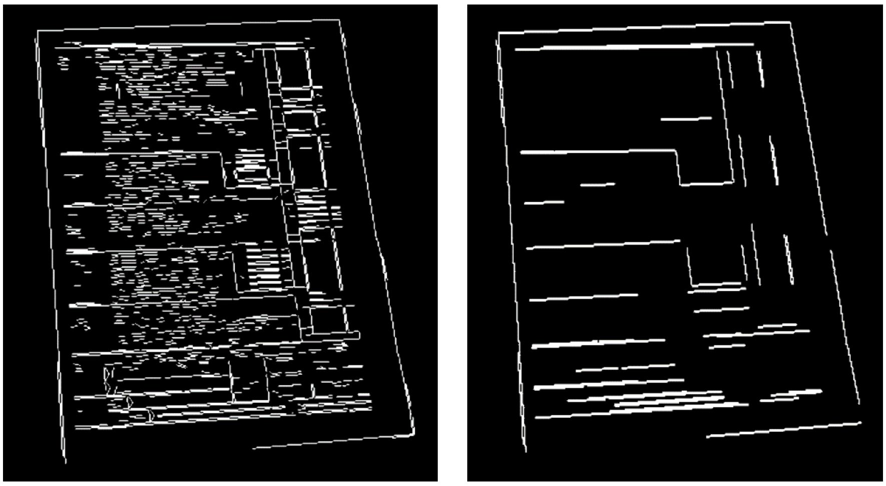
3.3. Length filtering

Our goal in this step is to remove the small line segments from the cluster of line which extracted in the previous step. Generally, these small lines are located in the region of document text, and represent the content of the document (text, image, table or a shape). The removing of these small unnecessary line segments allows us, on the one hand, to reduce the number of lines, therefore minimizes the computation time. On the other hand, facilitate the process of finding the four line segments that form the edge surrounding the text area.

In order to carry out this step, we take the coordinates of the two endpoints of each line segment, and then the length of the line segment is computed by the Euclidean distance between the two endpoints. Assuming that *p*(x_1,y_1), *q*(x_2,y_2) the coordinates of endpoints of a line segment the length *d(p,q)* of this segment is determined as follows:

$$d(p,q) = \sqrt{(x_1 - x_2)^2 + +(y_1 - y_2)^2} \tag{2}$$

After estimating the length of each line, we keep the lines which have a length greater than a threshold *Tlength*. Figure 5 shows the result before and after the application of the length filtering operation.



(a) **(b)**

Fig. 5. Length filtering operation. (a): Initial frame. (b): Frame result.

3.4. Angle filtering

In this step, we use a prior knowledge about geometry of the paper sheet; normally the external edges of the paper sheet consist of four lines which are grouped into two horizontal and vertical lines, and each group is perpendicular to the other. Based on this knowledge, we carry out an angle filtering of each pair of line segments to preserve all the perpendicular lines in each frame of video.

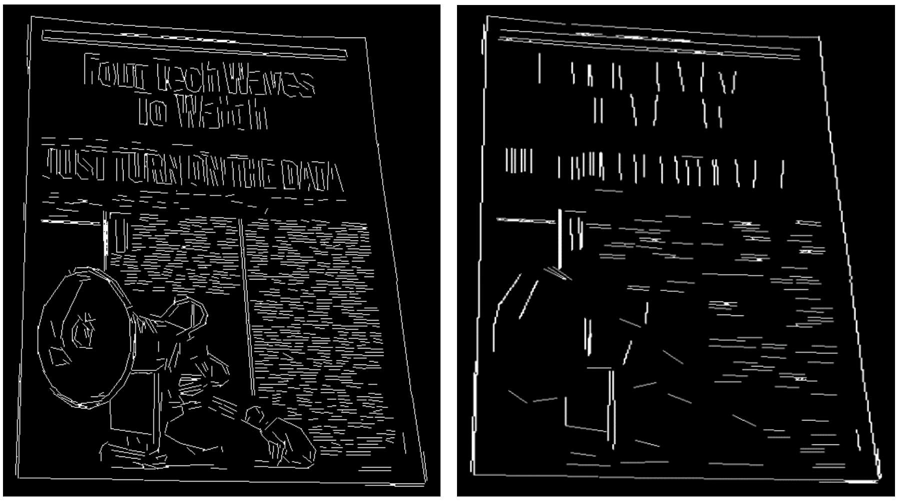
To accomplish this part of our model, we propose the following steps:

- Estimate the slope $m_i = \frac{y_2 - y_1}{x_2 - x_1}$ of each line segment $L_i(p_i, q_i)$.

- Eliminate a pair of lines, if the lines have the same slope (parallel lines).
- Preserve the pair of lines, if the product of its slopes equal to -1 (perpendicular lines).
- For the other pair of lines L_i and L_j, we calculate the angle θ_{ij} between the two lines by the following equation: $\theta_{ij} = tan^{-1}\left|\frac{m_j - m_i}{1+m_i m_j}\right|$
- The pair of lines L_i and L_j is preserved, if the angle θ_{ij} satisfies the following conditions: $TALow < \theta_{ij} < TAHigh$

  where *TALow* and *TAHigh* designate the low and high angle threshold.
- Affect each line segment selected into vertical or horizontal groups based on its slope. When the slope value $|m_i|$ of a line L_i is greater than 1, the line L_i is added into horizontal group. Otherwise the line is added to vertical group.

Like the length filtering the angle filtering allows us to decrease the processing time. Figure 6 illustrates the result obtained after the angle filtering process and the operation of grouping the lines.



(a) (b)

Fig. 6. Angle filtering operation. (a): Initial frame. (b): Frame result.

3.5. Candidate line segments selection

After obtaining two groups of line segments, we aim in this step to find the four corners of the quadrangle formed by four line segments. Therefore, in a first phase, we will try to select the four line segments from the horizontal and vertical groups, and in the second phase we will estimate the four corners of the line segments selected.

In order to find the top and the bottom horizontal lines of paper sheet, we start by sorting the lines of the horizontal group by Y-coordinate of its first endpoint, and then we can determine the two edges of paper sheet by selecting the first and the last line segments. Afterward, we follow the same process to determine the left and the right vertical lines of paper sheet, but this time the sorting is based on X-coordinate of the first endpoint of each line of vertical group lines.

After selecting the four candidate line segments, we can estimate the coordinates the four corners through the following way: Normally, the four corners are equivalent to the four intersection points of the candidate lines. So, we can easily calculate the coordinates of each point of intersection by using the line equations of each pair of candidate lines.

4. Experimental Results

This section is devoted to the presentation of the experimental results related to our proposed model. We start by giving an overview of the used database and the metric of evaluation. Then, we present the experimental results and a comparative study with other systems available in the literatures.

4.1. Datasets and evaluation measures

In order to demonstrate the performance of our model for detecting the quadrilateral coordinates of the page outlines within frames, we perform the experiments using the dataset from ICDAR2015 Smartphone document Capture [3]. In this work, we have used 30 video clips (Background 1) that include around 6180 frames. The documents have different content and layout pages that saved from patent documents, medical, scientific papers, magazine pages, tax forms and typewritten letters. All the documents have the same size (A4 paper format). After the generation of PDF documents with vector fonts, a color LaserJet printed them. Finally, videos are captured by a Google Nexus 7 tablet, about 10 seconds for each document with a Full HD 1920x1080 resolution. Videos are captured in various conditions: lighting, blur, occlusions, perspective and motion.

All our experiments were implemented and tested on the environment platform of Ubuntu 14.04 LTS (64 bit) distribution on a desktop computer equipped with Intel Core i7 − 4770 processor (3, 4 Go), 8Go RAM and NVIDIA GeForce GTX 980 4GO GPU. All the steps of our model have been implemented in C++ with the OpenCV (Open Computer Vision) library [23].

The performance evaluation of our model is done by calculating the Jaccard index (JI) [223]. The JI measures the pairwise similarity by computing the size of the intersection of two shapes relative to the size of their union. In our work, the two shapes are represented by the quadrilateral A formed by the four corners that we have detected with our model, and its quadrilateral ground-truth G. Therefore, for each frame f, we calculate the Jaccard index that measures the part of spatial overlap between two quadrilaterals by using the following equation:

$$IJ(A,G) = \frac{area(A \cap G)}{area(A \cup G)} \tag{3}$$

where $A \cap G$ denote the result of the intersection of the two quadrilaterals, and $A \cup G$ denote their union. The Jaccard index has a value vary from 0 (no overlap) to 1 (full overlap).

4.2. Parameters selection

To build our model of detection and segmentation of the page outlines in each frame, three parameters are to be set up. The first is *Tlength,* that represent the threshold length of each line segment in the length filtering step. The second is referring to the two angles *TALow* and *TAHigh* used in angle filtering step. The last is *Eps* that denotes the maximum distance neighborhood of the DBSCAN clustering algorithm.

In the first experiment, we have studied the influence of the parameter *Tlength*, for this reason; first we fix the value of *Eps* to 140 and the values of the angles [*TALow-TAHigh*] to [86, 94]. Then we evaluate the *Tlength* by running a series of experiments using different lengths 5, 10, 15, 20 and 30. The obtained results, presented in Table 1, indicate that the best performance is reached with a length of 5. Thus, all of our next experiments will be carried out using a *Tlength* of 5.

To study the impact of the angle of filtering on the page outlines segmentation, we set the *Tlength* to 5, *Eps* to 140 and we try various intervals of angles [*TALow-TAHigh*] ([88-92], [86-94], [84-96], [82-98], [80-100]). The result of the effect of angle of filtering is illustrated in Table 2. The best result was obtained by a model which uses the angles [80-100], with no significant improvement in the case where we expand the interval angle. This interval will be retained for the next experience.

The principal purpose of the last experiment is to study the influence of the maximum distance neighbourhood of the DBSCAN algorithm (*Eps*). Therefore, keeping the parameters *Tlength* to 5 and angles to [90-100], the distance *Eps* was varied between 100, 120, 140, 160 and 180. The result of detecting and segmenting of the page outlines with different *Eps* distance is represented in Table 3. It is clear from the table, that the value 140 of the *Eps* distance is the most appropriate.

Table 1. The impact of the length of line segments.

Tlength	Jaccard Index
5	0.9366
10	0.9325
15	0.9032
20	0.8573
30	0.7463

Table 2. The impact of the angle of filtreing.

[*TAngleLow,TAngleHigh*]	Jaccard Index
[88-92]	0.9279
[86-94]	0.9366
[84-96]	0.9406
[82-98]	0.9422
[80-100]	0.9425

Table 3. The impact of the maximum distance neighborhood (*Eps*) of the DBSCAN algorithm.

Eps	Jaccard Index
100	0.9187
120	0.9360
140	0.9422
160	0.9272
180	0.8470

4.3. Evaluation

After the selection of the optimal values for each parameter of our model, we compare the result of detection and segmentation of page outlines in terms of each document class (see Table 4). We conducted this comparison by using the same module that was evaluated in the previous experiments. The dataset [2] contains 6 document classes: Datasheet, Letter, Magazine, Paper, Patent and Tax.

Table 4. Average performance of our model for each document class.

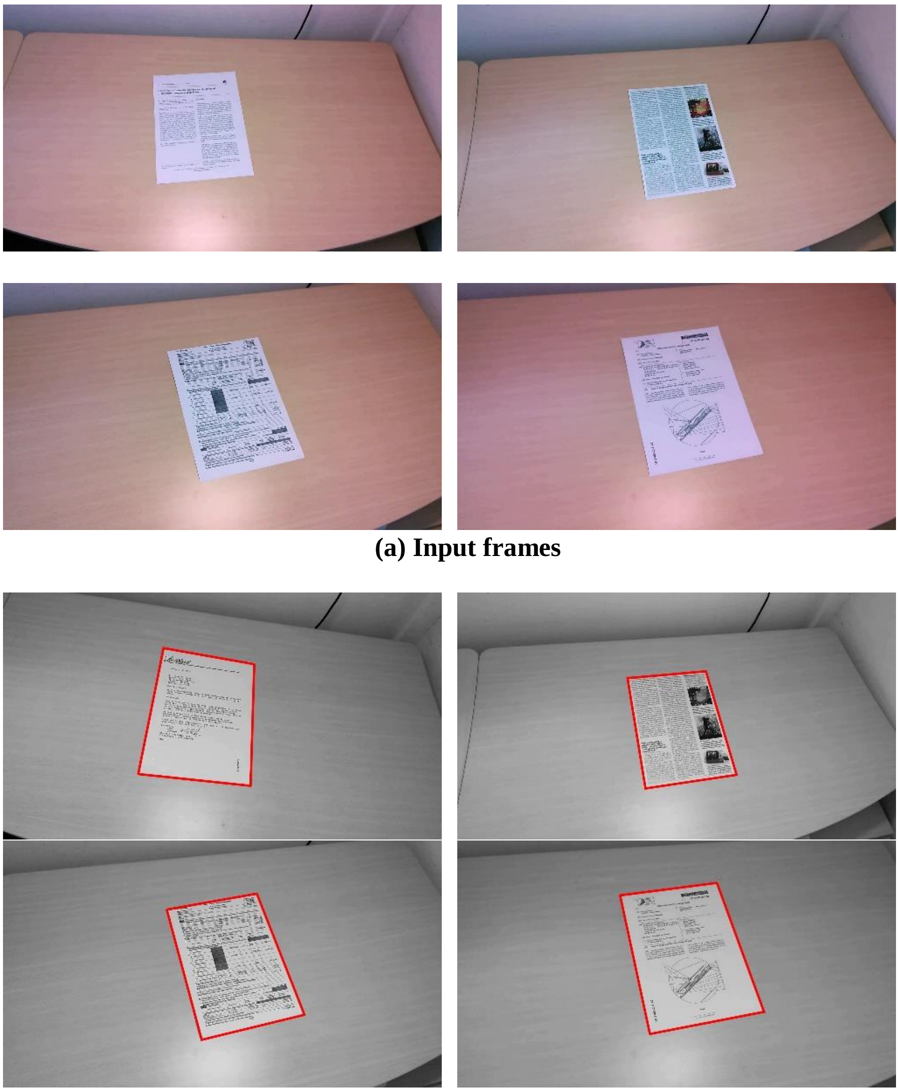
Document class	Jaccard Index
Datasheet	0.9201
Letter	0.9508
Magazine	0.9500
Paper	0.9100
Patent	0.9625
Tax	0.9598
All	0.9422

It is deduced from the obtained results in Table 4 that our model works well on the classes: Letter, Magazine, Patent, and Tax. This performance can be explained by the content of these classes, in which we can find tables, figures, graphics or a compact text box with large fonts and size. The result reached on the Datasheet and Paper document classes, can be interpreted by the content that was distributed on the paper sheet and included a text with a small size, in addition, there are cases where the text is printed only on a part of paper sheet. Figure 7 illustrates some successful examples of detecting the page outlines of different document classes.

In the last part of this section, we still assess the capacity and robustness of our proposed model by comparing it to the results of other models of the state of the art, who participated in the competition ICDAR2015 Smartphone OCR document Capture [2]. We performed this comparison using the same module that was evaluated in the previous experiment. The results of our model as well as the participated models are listed in Table 5.

RPPDI-UPE model begins by filtering the hue channel of the HSV color space to stand the background from the document page. Next, the Canny edge detector are applied, and then the Hough transform is used to extract the quadrangle candidate. The Canny edge detector is first used to estimate the document position in the SEECS-NUST model, then a priori knowledge about the contrast in the color channels and the Hough Transform are employed to find the text region. A2iA-2 model applied first the Canny edge detector, next Bezier curves is used to interpolate the detected con-tours. After that, the contrast is employed to select the quadrangles. LRDE model is based on the Tree of Shapes representation, which starts by calculating the energy on the tree, and then determine the shape most resembles to paper sheet based on their energy.

It is noticed from the Table 5 that the models LRDE and A2iA-2 provide best accuracies of the page outlines detection compared to our model. The accuracies for the two models as well as our proposed model are 0.9869, 0.9597 and 0.9422 respectively. We can also say that our model is more precise compared to the SEECS-NUST and RPPDI-UPE models.



(a) Input frames

(b) Result frames

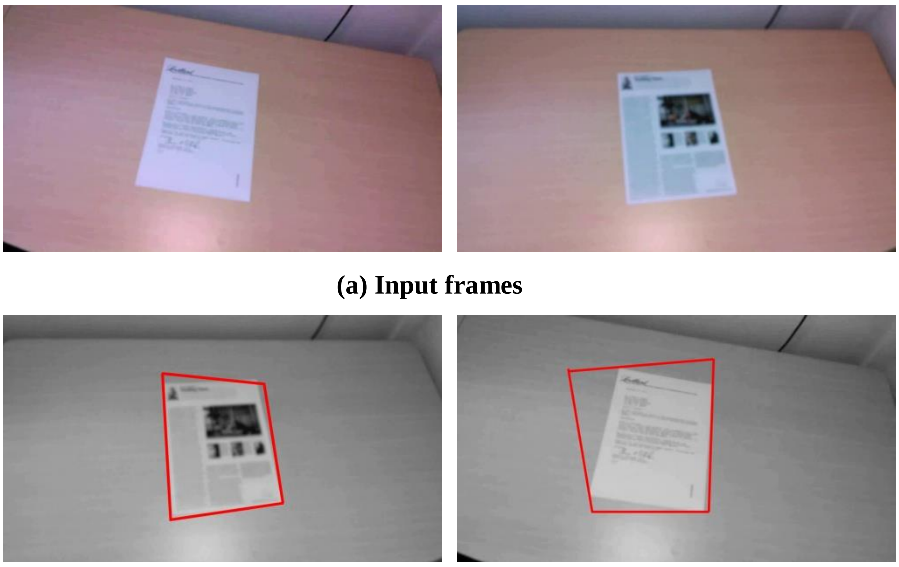
Fig. 7. Example of the page outlines detection results. The input frames taken from different document classes are positioned in the top and the corresponding frame result in the bottom. The detected regions are indicated by a red quadrilateral.

Table 5. Comparison of the proposed model and existing models.

Document class	Jaccard Index
RPPDI-UPE	0.8274
SEECS-NUST	0.8875
LRDE	0.9869
A2iA-2	0.9597
Our model	0.9422

In order to know the limitation of this work, Fig. 8 shows two examples of frames in which our model failed to accurately detect the document page. In this figure, the input frames are affected by the blurring caused by moving the Smartphone camera. Therefore, accuracy detection errors occurred due to degradation of text quality.

The last part of this section discusses the average run time of the proposed model. As shown in Table 6, the average processing time of a frame is 0.42 seconds. It's obvious that execution time varies from one step to another, whereas the two steps of clustering of the segments and the filtering of the angles consume more time (0.36s).



(a) Input frames

(b) Result frames

Fig 8. Example of frames where the system fails to accurately detect the page of document.

Table 6. Comparison of the proposed model and existing models.

Step	Execution time (s)
Line segment detection	0.06
Line segment clustering	0.23
Length filtering	0.02×10^{-3}
Angle filtering	0.13
Line segments selection	0.05×10^{-3}
Total	0.42

5. Conclusions

An efficient model for detecting the page outlines within frames of video clip obtained by a Smartphone was introduced in this paper. The key concept of our model is to detect the document text region without losing any part of the document, which was performed by using the LSD method, followed by DBSCAN clustering algorithm in order to locate the document text region. Next, the length and the angle filter operations have been applied to find the page outlines within each video frame. Experimental results on the ICDAR 2015 Smartphone Capture OCR dataset show that the proposed model can achieve an encouraging result and provides competitive detection performance with the state-of-the-art model.

For our future work, we will try to improve the proposed model by dealing with the problem of blur and the motion distortion posed by unfocused camera-phone, and the case of the presence of a complex background. Furthermore, we will focus on the text segmentation and text recognition based on the results of our proposed model.

References

1. Liang, J.; Doermann, D.; and Li, H. (2005). Camera-based analysis of text and documents: a survey. *International Journal of Document Analysis and Recognition (IJDAR)*, 7(2-3), 84-104.
2. Burie, J.C.; Chazalon, J.; Coustaty, M.; Eskenazi, S.; Luqman, M. M.; Mehri, M.; and Rusiñol, M. (2015). ICDAR2015 competition on smartphone document capture and OCR (SmartDoc). *In Proceedings of the 13th International Conference on Document Analysis and Recognition,* Tunis, Tunisia, 1161-1165.
3. Koo, H.I. (2016). Text-line detection in camera-captured document images using the state estimation of connected components. *IEEE Transactions on Image Processing*, 25(11), 5358-5368.
4. Mittal, A.; Roy, P.P.; Singh, P.; and Raman, B. (2017). Rotation and script independent text detection from video frames using sub pixel mapping. *Journal of Visual Communication and Image Representation*, 46, 187-198.
5. Yi, C.; and Tian, Y. (2013). Text extraction from scene images by character appearance and structure modeling. *Computer Vision and Image Understanding,* 117(2), 182-194.
6. Yin, X.C.; Yin, X.; Huang, K.; and Hao, H.W. (2014). Robust text detection in natural scene images. *IEEE transactions on pattern analysis and machine intelligence*, 36(5), 970-983.
7. Shekar, B.H.; Smitha, M.L.; and Shivakumara, P. (2014). Discrete wavelet transform and gradient difference based approach for text localization in videos. *In Proceedings of the 5th International Conference on Signal and Image Processing*. Bangalore, Karnataka, India. 280-284.
8. Shivakumara, P.; Dutta, A.; Tan, C.L. ; and Pal, U. (2014). Multi-oriented scene text detection in video based on wavelet and angle projection boundary growing. *Multimedia tools and applications*, 72(1), 515-539.
9. Sudir, P.; and Ravishankar, M. (2015). An effective edge and texture based approach towards curved videotext detection and extraction. *International Journal of System Dynamics Applications (IJSDA)*, 4(3), 1-29.
10. Liang, G.; Shivakumara, P.; Lu, T.; and Tan, C.L. (2015). Multi-spectral fusion based approach for arbitrarily oriented scene text detection in video images. *IEEE Transactions on Image Processing*, 24(11), 4488-4501.

11. Liu, X.; and Wang, W. (2012). Robustly extracting captions in videos based on stroke-like edges and spatio-temporal analysis. *IEEE transactions on multimedia*, 14(2), 482-489.
12. Zhang, J.; and Kasturi, R. (2014). A novel text detection system based on character and link energies. *IEEE Transactions on Image Processing*, 23(9), 4187-4198.
13. Shivakumara, P.; Sreedhar, R.P.; Phan, T.Q.; Lu, S.; and Tan, C.L. (2012). Multioriented video scene text detection through Bayesian classification and boundary growing. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(8), 1227-1235.
14. Huang, X.; Wang, Q.; Zhu, L.; and Liu, K. (2013). A new video text extraction method based on stroke. *In Proceedings of the 6th International Congress on Image and Signal Processing*, Hangzhou, China, 75-80.
15. Von Gioi, R.G.; Jakubowicz, J.; Morel, J.M.; and Randall, G. (2010). LSD: A fast line segment detector with a false detection control. *IEEE Transactions on pattern analysis and machine intelligence*, 32(4), 722-732.
16. Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence,* 8(6), 679-698.
17. Burns, J.B.; Hanson, A.R.; and Riseman, E.M. (1986). Extracting straight lines. *IEEE transactions on pattern analysis and machine intelligence*, 8(4), 425-455.
18. Desolneux, A.; Moisan, L.; and Morel, J.M. (2000). Meaningful alignments. *International Journal of Computer Vision*, 40(1), 7-23.
19. Desolneux, A.; Moisan, L.; and Morel, J.M. (2007). *From gestalt theory to image analysis: A probabilistic approach.* Springer Science & Business Media.
20. Ester, M.; Kriegel, H.P.; Sander, J.; and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining,* Portland, Oregon, USA, 226-231.
21. Ester, M.; Kriegel, H.P.; Sander, J.; Wimmer, M.; and Xu, X. (1998). Incremental clustering for mining in a data warehousing environment. *In Proceedings of the 24th International Conference on Very Large Data Bases*, New York, USA, 323-333.
22. Bradski, G.; and Kaehler, A. (2008). *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, Inc.
23. Everingham, M.; Van Gool, L.; Williams, C.K.; Winn, J.; and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2), 303-338.