Design of a Text Detection System via Hypothesis Generation and Verification

Sam S. Tsai*, Vasu Parameswaran†, Jerome Berclaz†,
Ramakrishna Vedantham†, Radek Grzeszczuk†, Bernd Girod*

*Department of Electrical Engineering, Stanford Univeristy, Stanford, CA
†Nokia Research Center, Nokia, Sunnyvale, CA **

Abstract. Text-detection is a well-established topic within the document recognition domain, but detecting text in natural scenes has proved to be considerably harder due to the large variability of text appearances, and due to dominating clutter. In this paper, we begin with a practical model for text appearance and use it to motivate the design of a text detection system consisting of two stages: a hypothesis generation stage and a hypothesis verification stage. For hypothesis generation, we adapt the Maximally Stable Extremal Regions (MSER) algorithm for the task of text detection by a combination of judicious parameter selection and a computationally efficient multi-scale analysis of MSER regions. For hypothesis verification, we design simple and fast image operations that produce features highly discriminative of text, which require minimal training, whose performance bounds can be specified, and which can be tuned in terms of intuitively understandable properties of text such as font, stroke-width, spacing, etc. Our final contribution is a new dataset of images of urban scenes annotated for text. The annotations include not only bounding boxes but also a richer set of text attributes. There are about 6000 text items in the dataset making it to our knowledge, the largest, most challenging, and most detailed text dataset for urban scenes[1]. We show improved performance than state-of-the-art on the well-known ICDAR datasets and results on our new dataset.

1 Introduction

While text detection is a mature topic within the document analysis domain, detecting text in unconstrained outdoor scenes is much more challenging due to two main difficulties. Firstly, the visual appearance of text is highly variable due to different possible fonts, styles, backgrounds, viewpoints, and camera parameters (geometric as well as radiometric). These infuencing parameters are typically unknown. Secondly, the text footprint is typically a small fraction of the image (often a few hundred pixels in an image of several megapixels). Hence the text detector needs to be especially robust to clutter. Text detection in unconstrained scenes has many applications including navigational assistance for the visually impaired, visual text translation, image indexing, digital map creation, etc. For these applications to be practical, the text detector needs to be not only accurate but also very fast.

** These authors are currently with Microsoft Corporation.
[1] http://msw3.stanford.edu/~sstsai/SanFranciscoText

In this paper, we make four novel contributions. *First*, we propose a model for text appearance in an image, and from it, motivate the design of a two-stage text detection system consisting of a hypothesis generation stage and a hypothesis verification stage. Our *second* contribution is a novel adaptation of the Maximally Stable Extremal Regions (MSER) algorithm [1]. We show how MSER parameters can be tuned in a principled way to best detect individual characters in a manner robust to image noise and contrast. Image blur is a frequent problem when using MSER to detect characters, causing them to get merged with each other or with the background. We tackle this problem by tuning MSER to return overlapping level sets in its output per real region, thereby better preserving the integrity of individual characters. We minimize the computational cost required to analyze the larger number of regions by performing a multi-scale histogram filtering of the MSER outputs. Our *third* contribution is the direct exploitation of our proposed model for the design of simple and fast filters that are highly discriminative of text at the word level, forming the text hypothesis verification stage. A key novelty in this paper is our demonstration that two well-motivated features are able to achieve a performance better than state-of-the-art using a simple Gaussian model, obviating the need for a large number of features or more sophisticated classifiers. Our *fourth* contribution is a new ground-truthed dataset of images containing text.

Prior Work: There have been two main approaches for text detection in natural scenes: *Classifier based approaches* apply a text-classifier to windows of different sizes and positions in the image. For example, Zhong et al. [2] detect text captions in video by exploiting the spatial periodicity of text. Ye et al. [3] apply wavelet filtering and Support Vector Machines (SVM) to classify text. Chen et al. [4, 5] design weak classifiers based on spatial gradient distributions, intensities, and edge-linking and use Adaboost to create a strong text classifier. Zhang et al. [6] extract feature from intensity and perform chinese character classification from the image. Classifier based approaches are helpful if high recall performance is desired, but a lot of training data is typically needed, and precision is often lower. *Connected-component* (CC) based approaches detect potential characters, link them together, and apply constraints to eliminate non-text regions. The top scoring entry reported in [7] uses adaptive binarization to detect CCs and groups them via geometric constraints. Epshtein et al. [8] detect text via an intermediate 'stroke-width-transform' image and connecting components with similar stroke-width. Shivakumara et al. [9] extract CCs via Fourier-Laplace filtering and K-means clustering and detect text lines via different constraints. Neuman et al. [10] use MSER and SVMs to detect characters based on features such as aspect ratio, number of holes, etc. and grouping them considering multiple word hypothesis. Chen et al. [11] use MSER and edge pruning for character detection, followed by character linking and text line formation to detect text. Other CC based methods include [12] and [13].

2 Motivating the Design Choices

Text Model: Searching for text in an image requires a model, either implicitly or explicitly. But it is difficult to define a model that captures all the variability

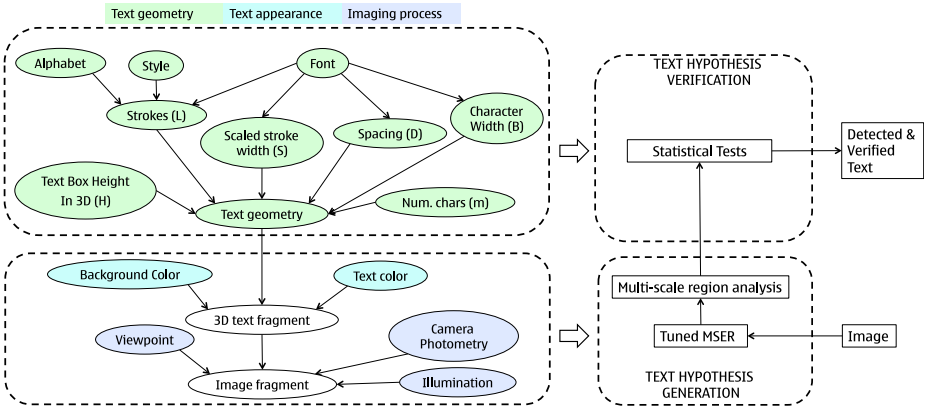


Fig. 1. Left: Bayesian Network for Text Appearance in an Image (best viewed in color), Right: System Design

of text such as fonts, styles, viewpoints, contrasts, etc. Nevertheless text in the real 3D world satisfies the following properties which could form the basis for a text detector:

1. For a given font, various text geometric attributes such as the stroke width, character height, and character spacing are fairly constant across the text.
2. Readability constrains stroke-width and spacing, given the text height.
3. The bounding box around the text is bimodal in intensity assuming some type of background.

We capture the main variables governing the generation process of 3D text and its 2D appearance in an image, via a Bayesian Network (BN) shown in Figure 1. Our use of the BN is to mainly identify the influencing hidden variables and not for recovering all of them, per se. Doing so allows us to make informed design choices for robustness to influencing variables where possible, and to make the assumptions and limitations of the system explicit, where it is not possible. We assume the following: (1) Out-of-plane rotations are negligible, (2) The minimum text/background contrast requirement is specified, and (3) The intensity noise distribution is specified (this could be specified or computed using methods such as [14]).

Referring to the BN in Figure 1, characters can be modeled as strokes generated by moving a 'pen' of given thickness (i.e. stroke width) along the characters skeleton. Let H be the height of a character, $L = \lambda H$ its 1D curved length (i.e. the 'stroke-length'), $B = \beta H$ the bounding-width of a character, $S = \psi H$ the stroke-width, $D = \delta H$ the inter-character spacing, and m the number of characters in a word. Note that the quantities $(\lambda, \beta, \psi, \delta)$ are random variables denoting respectively the character length, width, stroke width, and inter-character spacing relative to the text height. Given a choice of alphabet (e.g. English, Chinese etc.), style (i.e. bold, italic, regular), and font (e.g. Arial, etc.), the choice maps to probability distributions for λ, β, ψ, and δ. One chooses the number of characters, m, and draws m instances from the four distributions to create a set

of characters for the word[2]. Choices for text height, and colors for background and text generate the word in the 3D world. Choices for viewpoint, camera, and illumination, finally determine text appearance in the image.

System Design: We expect that the text properties outlined above will manifest themselves as correlations between the random variables $(\lambda, \beta, \psi, \delta)$. Accordingly, our design includes a *Text Hypothesis Verification* module that exploits such correlations. In section 4, we derive simple and fast image operations that correspond directly to statistcal tests encoding such correlations, remarkably, *without* needing to estimate all the random variables. Complementing this module, we design a *Text Hypothesis Generation* module that detects characters. Practically, any segmentation method can be used to detect characters. In natural scene images, views of text may be from different viewpoints and angles. Illumination along the text line may change, and the focus may not be exactly on the text, causing blur. The dataset we introduce includes panoramic snapshots taken periodically along a drive through San Francisco. The cameras are tuned to picture the whole scene, and so some text may not get adequate exposure and contain more noise. The blur and noise in these images is sometimes worse than in images from a camera phone. We note that MSER is known for robustness to different viewpoints, illuminations, and contrasts, addressing some of the remaining influencing variables in the BN. Text characters typically have a low-gradient interior and a high gradient boundary and so correspond well to stable regions Accordingly, we base the text hypothesis generation module on MSER. In contrast to past methods (e.g. [11][10]), our novelty is that we avoid using MSER as a black-box. Given our task of character detection and the imaging characteristics, we (1) tune MSER parameters to optimally detect characters given user-specified minimum contrast requirements and the image noise distribution, and (2) adapt it for robustness to image blur. The module is described in detail in Section 3). Note that MSER is a region detector and is agnostic to text geometric attributes.

In summary, the design includes two complementary modules for text hypothesis generation and verification, which, together with our assumptions, ensure that all the influencing variables in the BN are addressed.

3 Text Hypothesis Generation

3.1 Character Candidate Generation using MSER

MSER [1] is a region detector that finds stable regions in an image. Thresholding the image with a given threshold I_{th} produces regions having intensity below I_{th}. When the relative change in area of a given region is small when I_{th} is changed, the region is considered stable. A popular implementation of MSER [15] is configurable with three main parameters: *Delta* which controls the step-size for MSER thresholding, *maxVariation* which is a threshold on the maximum allowed area change, and *minDiversity* which controls the merging of overlapping MSER regions. The same values for *Delta* and *maxVariation* can produce very different regions under differing contrasts and image noise. In low contrast, a large *Delta*

[2] In this work we do not incorporate a language model.

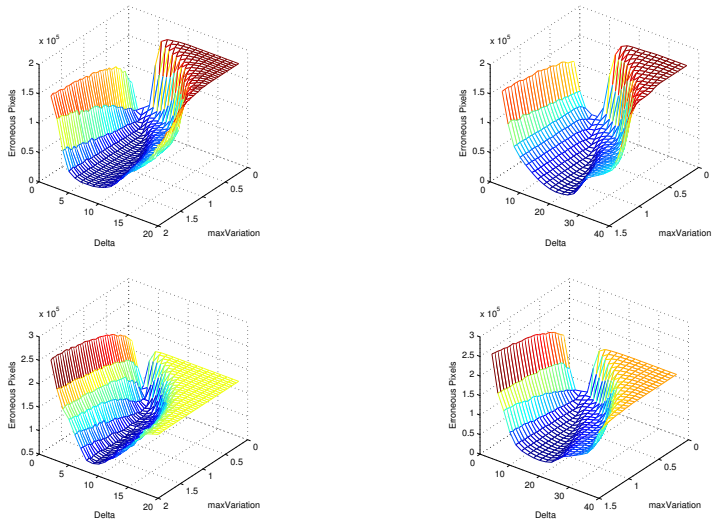


Fig. 2. Error (no. of pixel differences) as a function of $(Delta, maxVariation)$ for two different contrasts and noise levels on random text images. Left to right: low contrast, low noise($\Delta = 30$, $\sigma = 4$); high contrast, low noise($\Delta = 60$, $\sigma = 4$); low contrast, high noise($\Delta = 30$, $\sigma = 8$); high contrast, high noise($\Delta = 60$, $\sigma = 8$).

often leads to missing characters as they tend to merge into the background MSER regions. In heavy noise, a small *Delta* or large *maxVariation* tend to produce more false regions and over-segmenation of true regions. We note that a user of the system can specify the minimum contrast Δ of text to be detected. Image noise σ is often known apriori or can be estimated (e.g. [14]). Given these inputs, one can optimize *Delta* and *maxVariation* via simulations. Figure 2 shows the error as a function of MSER parameters for two different contrasts and normally distributed noise levels and reflects the above observations (see caption for more details). It can also be seen that setting parameters based on lowest contrast of expected text does not negatively affect larger contrast cases, allowing a larger range of parameters. We build a lookup table that maps (Δ, σ) to optimal (*Delta, maxVariation*) by generating synthetic text fragments for the specified contrast and corrupted by the specified noise. The lookup table can be built offline for a range of contrasts and noise levels and indexed at run-time.

The MSER regions extracted from this stage are passed through three filters designed to eliminate extreme outliers (thresholds for each filter were set empirically). First, we calculate the stroke-width of each region based on distance transforms (using the method of [11]) and reject regions that have too large a ratio of standard-deviation to mean. Second, regions of extreme aspect ratios are eliminated as these tend to be from long lines (e.g. poles, cables, parts of building facades, etc.). Thirdly, regions of very low density (defined as ratio of region size to upright bounding box size) are eliminated.

In conditions of blur, the character-background boundary is less significant, leading MSER in its default settings to produce incorrect character boundaries,

Fig. 3. Left to right: Input, Typical MSER regions using settings of [15] (several characters are missed), Our MSER regions.

and worse, merging of adjacent characters. The MSER parameter *minDiversity* which controls merging of overlapping regions, can be set to a low value (set to 0.1 in our experiments), such that several 'layered' MSER regions are returned per real region, often including the correct character boundary in one of them. The additional computational cost required is minimized by a spatial histogram analysis of the MSER layers. From here on, we refer to the MSER regions as character candidates. The character candidate selection process is shown in Algorithm 1 and described below.

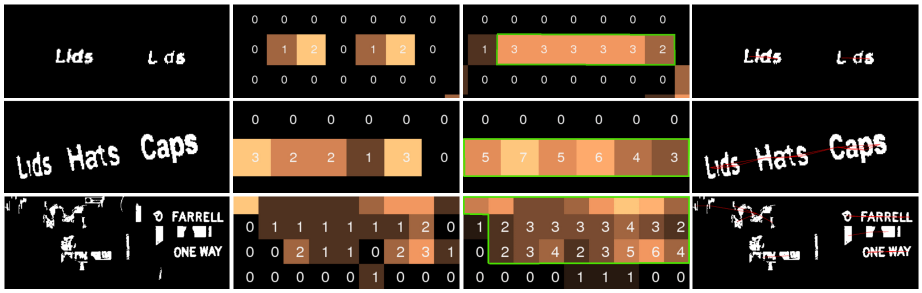
3.2 Character Candidate Selection

Input: Natural scene image Im, Lower/Upper scale ranges $(L_i, U_i), i = 1..N$
Output: Character candidates $\{CC\}$, Pairwise Links $\{l\}$
$M \leftarrow$ sort_by_size(extractMSER(Im));
for $i \leftarrow 1$ **to** N **do**
  $G_i \leftarrow \{m | m \in M, L_i \leq size(m) < U_i\}$;
  $M_i \leftarrow \{m | m \in G_i, pixels(m) \not\subset pixels(m_c), \forall m_c \in G_i,$
    $\nexists m_1, m_2 \in G_i, pixels(m_1) \subset pixels(m_c),$
    $pixels(m_2) \subset pixels(m_c), pixels(m_1) \bigcap pixels(m_2) = \emptyset\}$;
  $H_i \leftarrow$ GenerateHistogram(M_i);
  $CC_i, l_i \leftarrow$ PairwiseConnect(H_i, M_i);
end

Algorithm 1: Character candidate selection (high level view).

We process groups of character candidates determined based on their sizes (hence the call to `sort_by_size`). We configure MSER via a low *minDiversity* value, thus forcing it to return multiple regions (*level-sets*) per real region. The algorithm systematically considers each such region as it relates to other regions. The groups are divided from small to large, with logarithmic step changes and overlapping sizes.

For each group G_i, we retain character candidates that are largest and contain at most one smaller region. Character candidates that contain more than two other character candidates are ignored as these will be considered in the higher scales (this step corresponds to calculation of M_i in the algorithm). Figure 4 (a) shows examples of 3 sets of the character candidates of different sizes and locations generated from the image in Figure 3.



(a) (b) (c) (d)

Fig. 4. (*Best viewed in color*) Intermediate results of three different locations shown in Fig. 3. (a), detected MSER regions at three different scales. (b), the histogram at different scales. (c), the sum of adjacent histogram bins and the valid bins are within the green box. (d) The character candidates and links for the three locations.

We generate a spatial histogram of these character candidates, which simply gives the number of character candidates in each spatial location, as shown in Figure 4 (b). The bin size is set based on a user-specified minimum character size expected in the image (set to 20 in our experiments). Our use of the spatial histogram is to impose a constraint on the minimum number of characters in a word (currently set to 3). Specifically, for each bin in the histogram, we count the sum of the number of character candidates in the bin and horizontally adjacent bins. If the sum of the character candidates does not exceed the minimum threshold, we discard the character candidates in the current bin. Spatial bins satisfying the threshold are shown in Figure 4 (c).

The last line in the **for** loop forms pair-wise connections between candidates. We look at the remaining character candidates of the same bin and horizontally adjacent bins, and link them together if they overlap vertically. Shown in Figure 4 (d) are pairs of character candidates that are linked. These linked character candidates form connected sets of character candidates used for generating text line hypotheses.

3.3 Hypothesizing Text Lines

To verify whether character candidates are valid, we check if they can be constructed as characters of a text line. We assume that text lines are straight, which is true for the vast majority of text found in urban scenes. To hypothesize text lines, we fit straight lines to the centroids of character candidates of each connected set of character candidates. The text line that contains the largest number of characters form the hypothesized line and the characters are removed from the connected set. The process is iterated in each connected set until no text lines can be further extracted.

4 Text Hypothesis Verification

We design a word-level verification module that is as independent as possible from the previous stages. Our goal is to: (1) Derive constraints on text model variables that are expected to have low entropy, using the observations we made regarding constraints on the stroke-width, text-height and spacing in section

2, and (2) Derive measures in the image domain that mirror these constraints. Consider the binarized box of a text fragment of height H and containing m characters (please recall notation used in section 2). The area occupied by the text alone is:

$$A_T = \sum_{i=1}^{m} L_i S_i = H^2 \sum_{i=1}^{m} \lambda_i \psi_i = mH^2 \overline{(\lambda\psi)} \tag{1}$$

where $\overline{X}$ denotes the sample mean of $X, i = 1..m$. The bounding box area is:

$$A_B = HW = H^2 \sum_{i=1}^{m} \beta_i + \delta_i = mH^2 \left(\overline{\beta} + \overline{\delta}\right) \tag{2}$$

Taking the ratio of the two areas makes the measure invariant to the number of characters m and the word height H:

$$\omega = \frac{A_T}{A_B} = \frac{\overline{(\lambda\psi)}}{\overline{\beta} + \overline{\delta}} \tag{3}$$

ω can be computed trivially from the word hypothesis. If the stroke-width were known, we can divide ω by the relative stroke width sample mean $\overline{\psi}$ to obtain[3]:

$$\alpha = \frac{A_T}{\overline{\psi} A_B} = \frac{\overline{(\lambda\psi)}}{\overline{\psi}\left(\overline{\beta} + \overline{\delta}\right)} \approx \frac{\overline{\lambda}}{\overline{\beta} + \overline{\delta}} \tag{4}$$

Let's turn attention back to the text model variables in section 2. We synthesized about 31000 random text fragments from 200 English fonts with all combinations of plain, bold, italic, upper and lower cases. Figure 5 (left) shows a scatter-plot between the variables $\lambda, \psi, (\beta + \delta)$. While there is no correlation apparent, the subspace $(\lambda, \beta + \delta)$ (right) shows a strong linear correlation and hence the probability distribution of their ratio can be expected to have low entropy. Empirically, the signal-to-noise ratio (i.e. mean to standard-deviation) of the four text model variables λ,ψ,β, and δ, is less than half that for $\lambda/(\beta + \delta)$ (which has SNR of about 6.63). The quantity $\lambda/(\beta + \delta)$ precisely corresponds to the image feature α, which thus forms a feature highly discriminative of text. We can calculate α from the hypothesized box image only needing to estimate the relative stroke-width ψ. Interestingly it is possible to calculate α *without* having to recover *any* text model variable by borrowing from the field of stereology [16], which estimates higher dimensional quantities accurately, from rapidly computable lower dimensional statistics.

Given that stroke width is nearly constant, the curved length (λ) is approximately half the perimeter (as stroke width is a small fraction of the perimeter). One could obtain the perimeter via skeletonization, but skeletonization often does not produce the expected skeleton, especially for complex shapes. The perimeter can be estimated in a fast and efficient manner as follows: The

[3] $\overline{(\lambda\psi)} \approx \bar{\lambda}\bar{\psi}$ is reasonable because empirically, λ and ψ are weakly correlated (Pearson correlation of 0.4). Contrast this to λ and $\beta + \delta$ whose Pearson correlation is 0.94.

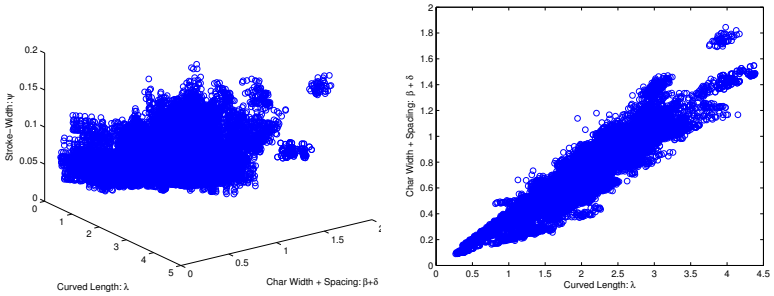


Fig. 5. Scatterplots: Left: $(\lambda, \psi, (\beta + \delta))$, Right: $(\lambda, \beta + \delta)$

binarized characters are intersected by a randomized set of parallel lines of spacing d, to obtain the number of intersections N. The total curved length of all characters is given by [16]:

$$m\overline{\lambda}H = \frac{\pi dN}{4} \tag{5}$$

We can now calculate α rather simply as follows:

$$\alpha = \frac{\overline{\lambda}}{\overline{\beta} + \overline{\delta}} = \frac{\left(m\overline{\lambda}H\right)H}{m\left(\overline{\beta} + \overline{\delta}\right)H^2} = \frac{\pi dNH}{4A_B} \tag{6}$$

We also maintain the lengths of the line segments intersecting the characters. We add a second feature, γ as the (standard-deviation/mean) of these lengths. Here we simply exploit the fact that text characters within a word are expected to have uniform stroke-widths, and hence have a low value for γ.

Fig. 6. Low α (Wide-Latin), High α (Onyx), Low γ (Arial-Narrow), High γ (Didot)

Figure 6 shows extremes in α and γ. Fonts with low α (e.g. Wide-Latin) have wider widths relative to stroke-length and vice-versa (e.g. Onyx), heights being equal. It can also be seen that the Didot font (high γ) has a higher stroke width variation than Arial-Narrow (low γ). Figure 7 (left) shows the (α, γ) scatterplot for about 3000 real-image text patches from our new dataset (positive class), and about 2000 random non-text patches from the same images (negative class). Fitting a 2D Gaussian distribution to the positive samples and applying increasing thresholds on the distance to the center, we obtain the ROC curve shown on the right. An equal-error operating point is $(0.09, 0.91)$ which is quite good, considering that there are only *two* easily computed features and that the positive class includes a large variety of text of different fonts and sizes. Furthermore, the user can easily tune this module based on expected fonts, needing only synthetic data, and tolerable miss detection and false alarm rates. Training for all experiments in this paper was done on synthetic data.

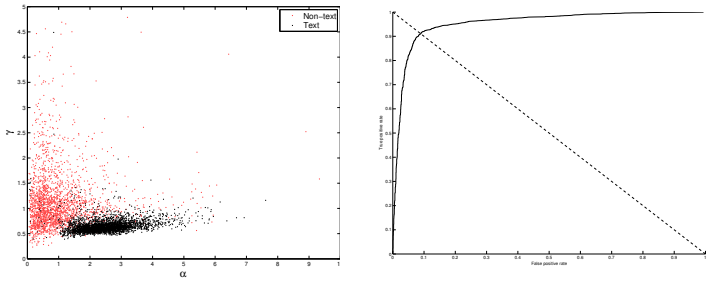


Fig. 7. (α, γ) scatterplot and ROC curve via 2D Gaussian (best viewed in color).

Some false positives escape the above tests due to features typical in an urban environment such as repeated windows and bricks which tend to mimic text. We deal with such cases by: (1) Comparing the characters to each other and rejecting the text box if half of the neighboring characters match each other. A match is defined in a template matching sense: if 85% of the pixels of the binarized characters match, then the characters are said to be matching. (2) Checking the solidity of each character, and rejecting the text box if a significant number of characters have a solidity (defined as the proportion of object pixels within its convex hull) that is greater than 0.95.

5 San Francisco Text Data Set

The most well known dataset for text detection in *natural* scenes is the ICDAR 2003 and 2011 dataset [7, 17] consisting of about two hundred test and training images. The images, hand picked to represent challenges in natural scene imagery, are mostly well focused but skewed towards diversity (e.g. text within or overlapping other text, hand drawn text, one-two character long text, multi-colored text etc.). While the datasets, largely containing the same images but with different annotation and scoring methods, provide a good starting point for validating text detection algorithms, they do not represent challenges of the kind and frequency typical in *urban* scenes, the main focus of our paper.

Other datasets include Epshtein et al.'s [8] street scene imagery, similar to ICDAR but harder, and Wang and Belongie's [18] Street View Text database containing images with annotations, but mainly for evaluating word-spotting algorithms given a lexicon, which is a different goal from this paper's goal, which is text detection without a lexicon.

We present a set of street scene images where all natural and legible text is annotated. The images are from several blocks from downtown San Francisco (SF) and capture all the challenges one encounters for text detection in *urban* scenes at their normal frequencies The images form four faces of a cubic panorama [19]. A few examples of the images with overlaid annotations are given in Figure 8. Each text annotation is labeled with a unique tag. For text that occurs repetitively in multiple images, the same tag is used to describe them.

Each annotation includes the following information: (1) *Location*, indicated by the four corners of an oriented bounding box surrounding the text, (2) *Text contents*, if English, with occluded characters identified, (3) *Physical appearance*, such as "*plain*", "*ornamental*", "*logo*", etc. (see Figure 9), (4) *Special character-*

Fig. 8. Sample SF images with annotations: Left to Right: Forward facing image, Right-side facing image, Backward facing image, Left-side facing image.

Fig. 9. Examples of "*plain*" (left) and "*ornamental*" (right) text.

istics, such as "*extruded*", "*reflective*", "*fragmented*", and "*reversed*" (see Figure 10), and (5) *Background*, such as "*wall*", "*ad sign*", "*billboard*", "*store sign*", "*store awning*", "*road surface*" and even "*no-background*" (see Figure 11).

6 Experimental Results

We use both the San Francisco data set and the ICDAR data sets for our experiments. While the San Francisco data set is a better representation of urban scenes taken in a systematic fashion, we experiment with the ICDAR data set mainly for validation and comparison.

ICDAR detection measure: For the ICDAR 2003 dataset, [20, 7], precision and recall is calculated as follows: For two boxes (e, t), the matching score is: $m_h(e,t) = Area(e \cap t)/Area(r(e,t))$, where $r(e,t)$ is the minimum sized horizontal box that encloses e and t. Let E be the set of estimated horizontal text boxes and T be the set of ground truth text boxes in an image. Then, the precision on a single image is $p = (\sum_{e \in E} max_{t \in T} m_h(e,t))/|E|$, while the recall is $r = (\sum_{t \in T} max_{e \in E} m_h(e,t))/|T|$. The precision and recall for the whole data set is given by the average of score per image. The f-score is defined as $f = 1/(\alpha/p + \alpha/r)$, with α typically set to 0.5. For the ICDAR 2011 dataset, the evaluation method in [21] is used as suggested by the text detection rules.

SF Text Data Set detection measure: In our newly presented data set, the text is annotated using oriented boxes. For two oriented boxes (e, t), the matching score is calculated as $m_o(e,t) = Area(e \cap t)/Area(e \cup t)$. Then, similar to the ICDAR measure, the precision of an image is calculated as $p = (\sum_{e \in E} max_{t \in T} m_o(e,t))/|E|$, while the recall is $r = (\sum_{t \in T} max_{e \in E} m_o(e,t))/|T|$, for a set of estimated boxes E and ground truth annotated boxes T.

Text detection results: Table 1 shows results on the ICDAR data sets. On ICDAR 2011, the proposed system improves recall and the f-score at practically the same precision, relative to recently reported results in [22]. Like [22], our method does not improve on all three measures of the top scoring (and *unpublished*) method in [17]. Furthermore, quoting Neumann and Matas [22], the

Fig. 10. Left to right:"*Extruded*", "*Reflective*", "*Non-Planar*", "*Fragmented*" text.

Fig. 11. Left to right: "*Ad sign*", "*Billboard*", "*Flag*", "*No-background*".

competition *"...was held in an open mode where authors supply only outputs of their methods on a previously published competition dataset"*. The proposed system improves performance on precision over the recently reported results of [23] at the same recall for ICDAR 2003. While the ICDAR dataset provides a good starting point for validating a text detection method, as noted in Section 5, there is a larger than normal proportion of outliers in the dataset making systematic exploration using principled methods difficult. Our method mainly misses text that is within text and text on transparent surfaces where different scene contents are blended together, etc. Since our focus is mainly on *urban* rather than *natural* scenes, our use of the ICDAR dataset was mainly for validation.

The SF dataset results (Table 2) demonstrate more clearly the strengths of the design choices we made in our system.

ICDAR 2011 Data set	p	r	f
Kim's method [17]	**0.83**	0.62	**0.71**
Proposed	0.73	**0.66**	0.69
Neumann et al [22]	0.73	0.65	0.69
Yi's method [24]	0.67	0.58	0.62

ICDAR 2003 Data set	p	r	f
Proposed	**0.74**	**0.61**	**0.67**
Minetto et al [23]	0.73	0.61	0.67
Chen et al [11]	0.73	0.60	0.66
Epshtein et al. [8]	0.73	0.60	0.66
Neumann et al [25]	0.65	0.64	0.63

Table 1. (p, r) on the ICDAR data sets

SF Data set	p	r, f (P)	r, f (O)	r, f (HP)
Proposed	**0.42**	**0.49,0.45**	0.42,**0.42**	**0.56,0.48**
Proposed (no verification)	0.32	0.49,0.39	0.43,0.37	0.56,0.41
Chen et al [11]	0.17	0.46,0.25	**0.45**,0.25	0.54,0.26

Table 2. (p, r) on the SF data set (right).

Note that precision is reported using all ground truth and all detections. Recall is reported for different categories to highlight the different challenges in the data set: "*plain*(P)", "*ornamental*(O)", "*horizontal plain*(HP)" text. Note that "*ornamental*" consists of text with special characteristics. The table also shows results from [11] on the dataset. The row labeled 'Proposed (no verification)' corresponds to results using only the MSER improvements described in section 3 which already improves signficantly over their method, which uses MSER as a black-box. The row labeled 'Proposed (no verification)' corresponds to results without use of the statistical tests from Section 4 showing the significant improvement that the verification tests bring for *negligible loss in recall*. Figure 12 shows sample detection results (please refer to caption for color codes). On the left is an instance where the system detects all text with no false alarms inspite of clutter. On the right is an instance where the system performs poorly due to heavy noise coupled with poor contrast, ornamental text with no character

Fig. 12. Examples of detected results where we perform well(left) and poorly(right). The red overlays are the annotated boxes, the purple overlays are the detected boxes, and the green overlays are the intersects of the boxes.

separation, and non-uniform text interior. There are two false positives detected on the tree due to leaves mimicking text. It can be noted that the overall lower performance on the SF data set relative to the ICDAR data set underscores its many challenges, which we hope will stimulate further research.

Computational complexity: The computation time of our system depends on imaging conditions (illumination, contrast, noise) and scene content (no. of regions). On ICDAR, our system takes roughly 1.8 seconds/image on a 2.53 GHz single core machine. On the SF dataset containing larger (4Mpix), non-well-focused, and noisier images, it takes on average 11 seconds/image. Most of the computation time is spent on generating the MSER regions. The multi-scale analysis, hypothesis generation and the verification steps are relatively fast. Note that our focus for this work was the principled design of a text detection system for best accuracy rather than speed, and these numbers are for unoptimized code.

7 Conclusions

We presented a text detection system for urban scenes comprised of modules for hypothesis generation and verification. We motivated the elements of the system via a practically useful model for text. We systematically tuned and presented a multi-scale adaptation of the MSER algorithm for detecting characters addressing challenges such as blur, noise, and image contrast variations typical in outdoor scenes. We exploited relationships between variables of the text model to derive simple, easily computable, fast, yet highly discriminative features for text. We introduced a new text detection dataset that is larger, richer, and more detailed than previous datasets, and which we hope will serve as a new benchmark dataset for text detection in urban scenes. We showed improved performance over the state-of-the-art on the ICDAR dataset as well as the new dataset. Future work is planned along several lines for improved performance: (a) Use of local noise estimates (rather than global) in the detection stage, (b) Exploiting constraints such as language models, distributions of number of characters per word etc. in the verification stage, and (c) Exploiting contextual cues typical in urban environments (e.g. lines from facades for partial scene geometry).

Acknowledgements: The authors would like to thank Xin Chen (Nokia Location & Commerce) for the SF text dataset used in this paper.

References

1. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide baseline stereo from maximally stable extremal regions. In: British Machine Vision Conference. (2002)
2. Zhong, Y., Zhang, H., Jain, A.: Automatic caption localization in compressed video. IEEE TPAMI. (2000)
3. Ye, Q., Huang, Q., Gao, W., Zhao, D.: Fast & robust text detection in images and video frames. IVC (2005)
4. Chen, X., Yuille, A.: Detecting and reading text in natural scenes. In: CVPR. (2004)
5. Chen, X., Yuille, A.: A time-efficient cascade for real-time object detection: With applications for the visually impaired. In: CVPR - Workshops. (2005)
6. Zhang, J., Hanneman, A., Yang, J., Waibel, A.: A robust approach for recognition of text embedded in natural scenes. In: ICPR. (2002)
7. Lucas, S.: ICDAR 2005 text locating competition results. In: ICDAR. (2005)
8. Epshtein, B., Ofek, E., Wexler, Y.: Detecting text in natural scenes with stroke width transform. In: CVPR. (2010)
9. Shivakumara, P., Phan, T.Q., Tan, C.L.: A laplacian approach to multi-oriented text detection in video. IEEE Trans. Pattern Anal. Mach. Intell. (2011)
10. Neumann, L., Matas, J.: A method for text localization and recognition in real-world images. ACCV (2010)
11. Chen, H., Tsai, S.S., Schroth, G., Chen., D.M., Vedantham, R., Grzeszczuk, R., Girod, B.: Robust text detection in natural images with edge-enhanced maximally stable extremal regions. In: IEEE ICIP. (2011)
12. Retornaz, T., Marcotegui: Scene text localization based on the ultimate opening. International Symposium on Mathematical Morphology (2007)
13. Leon, M., Mallo, S., Gasull, A.: A tree structure based caption text detection approach. IASTED Int. Conference on Visualization, Imaging, and Image Processing (2005)
14. Liu, C., Szeliski, R., Kang, S.B., Zitnick, C., Freeman, W.: Automatic estimation and removal of noise from a single image. IEEE PAMI (2008)
15. Vedaldi, A., Fulkerson, B.: VLFeat: An open and portable library of computer vision algorithms (2008) http://www.vlfeat.org/.
16. Huang, Y., Klette, R.: A comparison of property estimators in stereology and digital geometry. In: Combinatorial Image Analysis, Lecture Notes in Computer Science,. (2005)
17. Shahab, A., Shafait, F.; Dengel, A.: Icdar 2011 robust reading competition challenge 2: Reading text in scene images. In: ICDAR. (2011)
18. Wang, K., Belongie, S.: Word spotting in the wild. ECCV (2010)
19. Greene, N.: Environment mapping and other applications of world projections. Computer Graphics and Applications, IEEE (1986)
20. Lucas, S., Panaretos, A., Sosa, L., Tang, A., Wong, S., Young, R.: ICDAR 2003 robust reading competitions. In: ICDAR. (2003)
21. Wolf, C., Jolion, J.M.: Object count/area graphs for the evaluation of object detection and segmentation algorithms. IJDAR **8** (2006)
22. Neumann, L., Matas, J.: Real-time scene text localization and recognition. CVPR (2012)
23. Minetto, R., Thome, N., Cord, M., Stolfi, J., Precioso, F., Guyomard, J., Leite, N.: Text detection and recognition in urban scenes. In: Computer Vision for Remote Sensing of the Environment. (2011) 227–234
24. Yi, C., Tian, Y.: Text string detection from natural scenes by structure-based partition and grouping. IEEE Trans. Image Processing **20** (2011)
25. Neumann, L., Matas, J.: Text localization in real-world images using efficiently pruned exhaustive search. In: ICDAR. (2011) 687–691