A New Approach to Robust, Quality-Independent Scene Text Processing: Using Bi-level overlapped binning and Delta Images

# ABSTRACT

Extracting textual information from scene text images has been an active field of research with lots of work in done in areas of localization, recognition and binarization. These papers provide new features or new approaches to preserve text and remove non-text as much as possible and new features, techniques or machine learning models come out almost every few months that can beat the existing state-of-the-art. Though most of these papers have novel approaches, the first processing step on the raw image has remained nearly unchanged for a past decade. MSERs or one of it’s modified forms remain widely as the most used processing step for extracting regions from a raw scene text. The new techniques are then applied on these extracted regions. Though MSERs are very useful, it often suffers from problems of accurate parameters and thresholding. Also, MSERs focus on extracting a single stable region in the image space which often leads to problems in low quality images where a stable region might not correspond to a text region but overlap with both text and a large portion of the background. In this paper, we present a novel new approach for raw scene image processing called two-level overlapped binning which instead of giving a single solution for a homogenous region in a scene, outputs several solutions in the form of different connected components of binary images which we call Delta Images. These connected components represent regions in the scene text with different deviations of RGB values. It’s a Divide and Conquer based solution and we process each of the connected components individually. Finally, we recombine these solutions to achieve our desired results. In this paper we show the working of this new approach in an active field of research – Text Binarization and give our results on several standard datasets.The technique is able to extract text regions even from low quality images and has shown to be a very robust processing step for raw scene images

# The Overview

1. **Foreground – Background Pixels split**: The first step involves identifying and splitting the pixels of the image into sets of candidate foreground and candidate background pixels. We used a bi-level overlapped binning approach which gives different sets of candidate foreground and background splits. This is the critical step as the next set of steps depends on the fact that there is at least one split where the text pixels fall into one set of pixels and with very little background pixels in the same set.
2. **Homogeneous region detection:** Step 1 gives several sets of candidate foreground and background pixels. This step involves identifying clusters of connected pixels which represent a homogenous region in the given input image. A homogeneous region is defined as a set of connected pixels in an image whose deviation in RGB values is less than the deviation from the surrounding pixels of that cluster.
3. **Text, non-Text Separation:** The homogeneous regions extracted in the form of connected set of pixels from Step 2 are now classified into text and non-text using a supervised learning model. Regions classified as non-text are removed and candidate text regions are retained for combination
4. **Probabilistic Combination:** Step 3 gives us multiple sets of candidate text pixels across multiple bin sizes used in the differential binning step. We compute the probability of a pixel being a text pixel based on its occurrence across the different delta images. We label Pixels in the final binarized image based on a probability outcome
5. **Foreground – Background Pixels split: Bi-level Binning**

The first target is to identify sets of foreground and background pixels and split them. We use two-level overlapped binning and adjacent bin combination which gives us new set of binary images as output called Delta Images which have a unique property of segmenting the raw image into regions based on deviation of the region.

1. **Bi-Level Overlapped Binning**

For a given Bin Size S, we divide the entire range of RGB Euclidian distances into two levels of Bins

*For a given Bin size S:*

**Level 1** (Bin number 1 to m)

*{(0 to s-1), (s to 2\*s – 1), (2\*s to 3\*s – 1) …., (m-1\*s to m\*s)}*

*OR*

**Level 2** (Bin number m+1 to 2\*m -1)

*{(0.5\*s to 1.5\*s – 1), (1.5\*s to 2.5\*s –1), …..., ((m-1.5) \*s to (m-0.5) \*s)}*

*OR*

*where an element (a to b) denotes a bin containing all values from a to b both inclusive*

*Where where is the maximum value possible for a pixel. Level 1 ranges from 0 to and Level 2 ranges from )*

Each Bin is represented as a binary image matrix which is the kth bin of size S and ()ij is the value of the ith row and jth column of

Where

R(i,j) = Red Value of pixel at ith row and jth column of image

G(i,j) = Green Value of pixel at ith row and jth column of image

B(i,j) = Blue Value of pixel at ith row and jth column of image

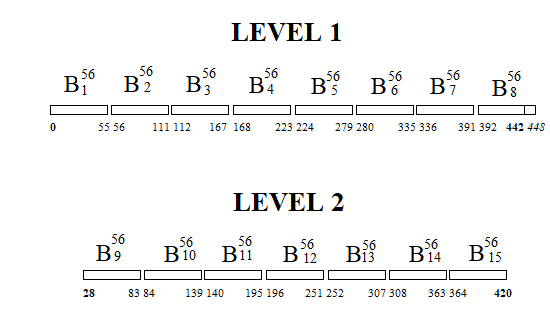


Fig 1. Bins for size = 56. The range corresponding to each Bin denotes the pixel of the corresponding binary image for the bin will be 1. If the value falls in that range then the binary image will have 1 for that pixel, otherwise it is 0.

For a given bin size s, there are total bins in both levels together where ,each represented by a binary image where the first images are the Level 1 bins and the next images are Level 2 bins. If we look at Fig 1 for a given bin at level 1, the bin at Level 2 overlapping with this bin on the left half is and with the right half is .

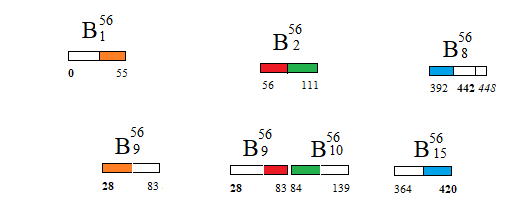


Fig 2. Diagram depicts the bins that overlap with the level 1 bins to the left and right. For bin no. 2 the right overlapping bin is and the left overlapping bin is . We see Bin no 9 overlapping to the left half and bin no. 10 overlapping to the right half of bin no. 2

In order to achieve robustness and quality independence, we use *Decremented Interval Binning*– using multiple bin sizes with the difference between corresponding bin sizes decreasing the higher we go. At this stage, let it be clear that when we say foreground or background pixel, it is for convenience and we do not label any specific set of pixels as foreground or background. Any one of the sets can correspond to either foreground or a background the next steps will do the work of eliminating the sets of background pixels.



Fig 3. A graph of different binary images to show how connected components behave on increasing bin size. The edges connect the subset bins to their superset bins at the immediate next bin size. The bin size denotes the size of the original Bins and the Range is the new size after the Adjacent Bin combination step

The lines in Fig 3 connect subset bins to their superset bins. For example, bin (374 to 406) is a subset of bin (367 to 420) in terms of range of pixel values and is connected by a line. Later, this relationship will be used to determine stable regions and extract features for text non-text separation.

1. **Adjacent bin combination**

Consider a Homogenous region Rwith PIXEL-VALUEs in the range of 30 to 60. We want all regions whose deviation is less than S to fall into at least one of the bins of size S such that the R occurs has a single connected component in a binary image and can be used for proper evaluation of structure and geometry for text-non-text separation. If suppose bin size is 32, then we want it to fall entirely in one of the bin. However, Let’s look at the pixel values that the first few bins of both the levels in two-level binning accumulates

Level 1

{(0 to 31), (32 to 63), (64 to 95), (96 to 127) ….}

Level 2

{(16 to 47), (48 to 79), (80 to 111) …}

We see in both the levels, the region R fails to fall into any of the bins despite having a pixel value deviation less than the particular bin size.

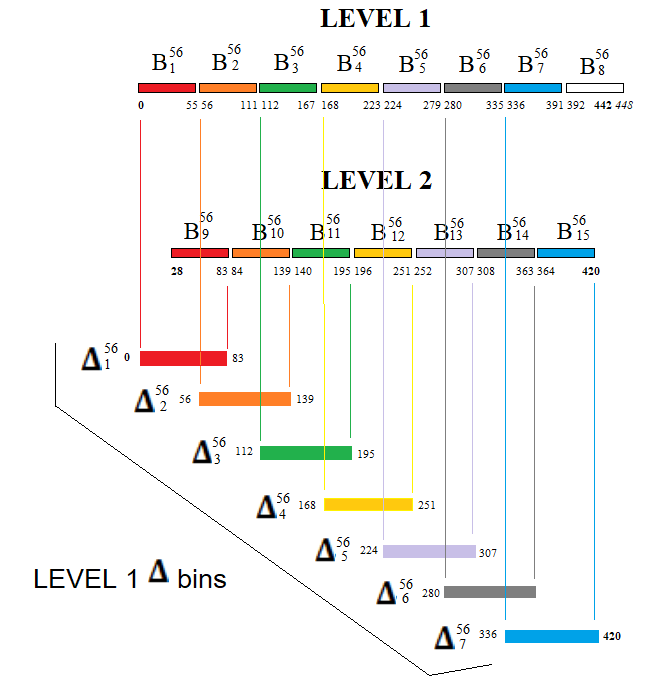
To solve this problem, we create a new set of recombined bins called as Δ bins, which are represented as Delta Images.

**LEVEL 1** Δ

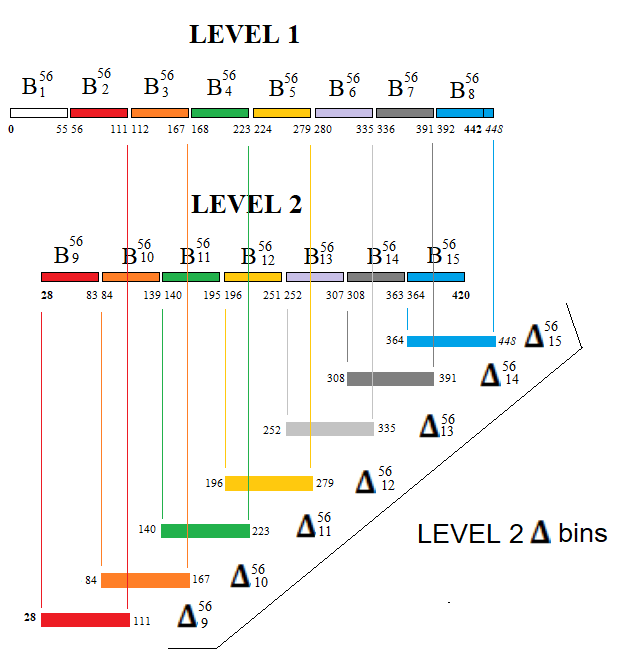
**LEVEL 2**

Where, +1

*Note: + operator while adding bins is pixel-wise logical OR operator of binary images. The resulting is another binary image called Delta Image.*



*Fig 4. The Level 1 Delta bins. The bins overlap in such a way such that all regions with deviation less than equal to 56 will fall into one of the colour ranges entirely(thus falling into one of the bins). Note*  does not have a Delta bin in level 1 but has a Delta bin in level 2 of the next figure.



*Fig 5. Illustration of Level 2 Delta bins for size 56. The range (colour block) corresponds to each Delta bin and denotes for what PixelValue of point (i,j) the binary image will be 1 at point (i,j). All other points of the binary image for a Delta bin will be 0 if it does not fall in the corresponding range*

Considering the previous example, Let us see the first few bins for bin size 32

LEVEL 1

{(0 to 47), (32 to 79), (64 to 111) ...…}

LEVEL 2

{(16 to 63), (48 to 95), (80 to 127) ….}

*Region R falls in 1st bin of Level 2*

The delta bins have an important property which makes it useful for our task. Given a Delta Bin of recombined from bins of size S (from henceforth will be referred as size of delta bin) a region in the image whose deviation is less than equal to S will occur as part of a single connected component in the same spatial coordinates as the region

We state the following lemma and its proof for any region R. *A region is a set of points on an image that is connected*

Lemma : *If the difference between the pixel with the highest PixelValue and lowest PixelValue in Region Ris D and let S be any bin size that is selected such that then if , ()ij = 1 for one value of positive integer*  +1

Proof: Let the minimum pixel value be for pixel (xa,ya) and maximum be for pixel (xb,yb) such that .

Let +1 ….. ***Def 1***

k is the value of the bin at level 1 where falls

then from the definition of ()ij ,

Now

Since,

***…… Eqn 1***

Also,

***..….. Eqn 2***

Thus, from *Eqn 1* and *Eqn 2* ***……… Eqn 3***

Since is an integer. Implies that the pixels with value will fall in the same bin or the immediate next bin. Since Deviation is less than size of each bin the bins where the region R is spread over is limited to a single bin or two adjacent bins

Case 1: ***…… Eqn 4***

In that case, both from *Def 1* and from *Eqn 4*.

For any ,

Since is an integer and L.H.S is an integer. Only possible solution

ij

Now,

Since, ij for all

ij  = 1 for all

Case 2 : ***…… Eqn 5***

We have to consider two different sub cases depending on whether lies to the left half of the bin overlapping with the right overlapping bin of level 2 or the right half of the bin thereby overlapping with the right overlapping bin of level 2

Case 2.1 : ***…… InEqn 1***

Let us consider bin no. , which is the bin that overlaps with the right half of bin k. +1 from lemma statement

Now,

From condition of *Eqn 6*,

**…… InEqn 2**

And from *Eqn 5* :

**…… InEqn 3**

***……. InEqn 4***

From Definition of ()ij

()XbYb = 1

Bin k has values ranging from and From *InEqn 4* Bin has the values

*Inference 1: Bin k and Bin k+m combined has all the pixel values P such that*

The pixel ranges Bin k and Bin m captures are overlapping. Since Pmin is in bin k and Pmax in bin m ,for any the following holds

**….InEqn 5**

From definition of

From *Inference 1* and *InEqn 5* we thus prove

ij  = 1 for all

case 2.2 :

, Since

Thus, ***……. InEqn 6***

+1

***……InEqn 7***

From *Def 1*,

***…….InEqn 8***

From *InEqn 6* and *InEqn 7*,

From definition of ()ij

()XbYb = 1

Bin k+1 has values ranging from and from Bin k+m has the values

*Inference 2: Bin k+1 and Bin m combined has all the pixel values P such that*

The PixelValue ranges of Bin k+1 and Bin k+m are overlapping. Since Pmin is in bin m and Pmax in bin k+1,for any the following holds

**..InEqn 9**

From definition of

From Inference 2 and the *InEqn 9* we thus prove

ij  = 1 for all

*Thus for all the cases the lemma gets proved.*

During the proof we draw two important inferences, each of which gives us the range for a bin. The range of pixel-values is .

The Lemma tells us that if there is a text region and if the range of PixelValues in the text is D, all the pixels will occur as a part of a connected component in a bin of size D (or more). Moreover, for the bin where it will occur, if it’s bin size is close to D as possible (best if it is D) then the chances of background pixels occurring in the same bin is also very less assuming that the deviation in PixelValues within a Text (or a homogenous region) is much less than the deviation between the Text pixels and the background pixels immediately surrounding the Text region. We find this to be true in nearly all the scene text images we come across.

Thus, the bins represented as binary images capture several homogenous regions for several different bin sizes. A connected component in one binary image of bin size S will have the same or more pixels for the next higher bin size.

1. **Homogenous Region Detection: Identifying stable regions using classifiers on connected components**

Step 1 gives several sets of connected components. All of the components do not correspond to homogenous regions in the original image. Bin sizes smaller than the deviation in a homogenous region will break up the connected components into pieces. Fig 1 shows how the size of the connected component increases with increasing bin size and how a homogenous region fragments into several bins when bin size reduces. For a particular connected component, after a particular bin size, the size will stop increasing too rapidly on increasing the bin size to the next value and rate of increase of size of connected component will be minimum.

This property is used to extract un-fragmented homogenous regions. We compare rate of increase of the size of connected components when the range of PixelValues in a binary image increases. The principle over here is similar to MSER techniques when trying to find stable extremal regions.

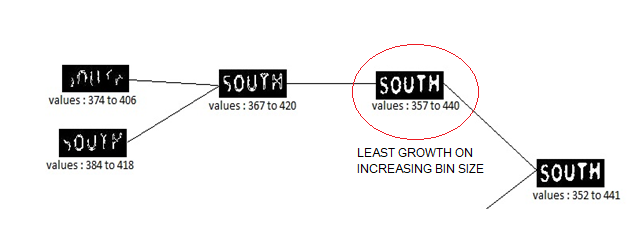


Fig 2: A part of the graph from figure 1. Along this path we see the components in the binary image representing bin (357 to 440) having the growth on increasing bin size. Note, analysis is done component wise and not bin wise (or binary image wise). Here incidentally one binary image has nearly all its component stable. This is a frequently occurring case since most real world text does have nearly the same deviation for each letter

However, unlike MSER we do not find one stable region along a path of a tree[?]. We evaluate stability of each connected components in a bin against two increases of the pixel-value ranges. The images compared against are called *Lower Range Incremented (*LRI) *Image* and *Higher Range Incremented* (HRI) *Image* where

Where +1

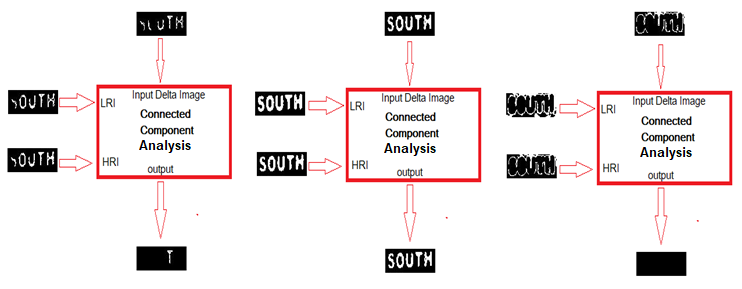


Fig 4. During connected component analysis we extract features from each connected component and decide whether it’s a stable text component or not. It preserves the connected component if it is a stable text region otherwise changes all the pixels to 0 for that component. During Analysis we use classification which takes both stability features of section 2 and text features of section 3

For each connected component of the Delta Image (the binary image corresponding to a bin) we extract the following features and use it to classify stable and unstable regions. For each connected component C in the Delta image, we denote LRI(C) as the largest connected component of the LRI image that overlaps with C assuming the same 2D space for both images and HRI(C) as the largest connected component that overlaps with C in the HRI image

Features Set for Stable region detection used for classification

1.

2.

3.

4.

5.

6.

7.

8.

9.

10. Bin Size of Delta bin

From our training samples we find stable connected components will have small values for features 2, 3, 5, 6, 8, 9. Features 1, 4, 7, 10 have high to mid-range values in their respective domain space.

1. **Text/Non-Text Separation**

Features described in section 2 when used to train a classifier and used for connected component analysis removes nearly most of the unwanted connected components and connected components which do not cover a complete homogenous region in the image. In this step, we remove those homogenous components which are non-text (example the background, lines across image, fragmented connected components which appear stable across LRI and HRI, etc.). We use the following features from [?]

The Feature Set for Text/Non-Text separation used in the classifier

1. **SW(Stroke Width):** Stroke width is a widely used characterness cue. We use SW of the connected components which is measured as

E*(l)* and Var*(l)* are SW mean and variance

respectively

1. **Edge Histogram of Gradients(eHOG):** eHOG measures gradient orientation at the edges of the histogram. It exploits a feature that text edge pixels occur in pairs with opposite gradient directions. Since we have connected components in a binary image, we do not need any edge detector method. The edge pixels are the pixels with values 1 which are adjacent to 0’s. For each connected component, the largest continuous chain of connected edge pixels is taken as the skeleton. The pixels on the skeleton are then divided into 4 types depending on the direction(or angle) between the pixel and the next pixel

*Type 1*: 0 *< θ* ≤ *π/*4 or 7*π/*4 *< θ* ≤ 2*π*, *Type 2*: *π/*4 *< θ* ≤ 3*π/*4,

*Type 3*: 3*π/*4 *< θ* ≤ 5*π/*4, and *Type 4*: 5*π/*4 *< θ* ≤ 7*π/*4.

For Text, the number of edge pixels in Type 1 will be close to that in Type 3, and so for Type 2 and Type 4. eHOG(r) is then measured as

where *wi (r )* denotes the number of edge pixels in Type *i* within region *r*

We also observed that if a connected component C is a text component, HRI(C) and LRI(C) are also text components with similar SW and eHOG values. We therefore have the following additional features

We extract all the features mentioned in Section 2 and Section 3 from a connected component and pass it through a binary classifier which gives us 1 for text and 0 for non-text and fragmented text components. The connected components that are classified as 0 are removed from the binary (the pixel values for the component are inverted). We used Ensemble of RUS Boosted Decision trees for our classification. Section 5 goes into how to train the classifier.

1. **Probabilistic Recombination: Combining across several bins to generate the final binarized image**

The connected component analysis performed at section 3 and 4 removes nearly all the non-text and fragmented components of text regions from the Delta binary images. At this stage most of the binary images are blank with no connected components left and there are only some images left with text components mostly spread across different bin sizes and a few images having fragments of the background or text. In our experiments we observed bin sizes closer to to have the maximum probability of being a text component. We denote as the probability that a data pixel in the delta image of is a text pixel.

We next create a *probability image* (P.I.) which gives us a value for

Each pixel. The dimensions are the same as the input image and each

Point in P.I. has a value. The higher the value, the higher the probability

Of that pixel being a text pixel.

denotes a multiplication of the scalar with each

pixels of matrix to generate a new matrix

The final Output Image is a binary image which we get by bi-level thresholding

The P.I. The threshold is set at half the median of the values in the P.I.

1. **Training the classifier**

For training the classifier we need pixel level annotations which will be used to label the connected components obtained from the bins as text or others. Pixel level annotations are provided by Kumar et al.[?] for

1. Street View Text 2010 – 647 Images
2. ICDAR 2003 – 1110 Images
3. ICDAR 2011 Born Digital Images - 918 Images
4. ICDAR 2011 Scene Text Images – 716 Images

We split the dataset and keep half the datasets for training and the other half for

Testing and evaluation with existing techniques. We take the images for training

And create the Delta images first. We use the following algorithm for quick

Generation of the delta images, the LRI and HRI

1. *BinSizes* 1-D array of Bin Sizes to use
2. *Img* Image Matrix (Height x width x 3 channels)
3. Initialize *BinImgs* to a Matrix of size HeightxWidthxN where N is the number of Bin Images that can be generated from given *BinSizes*
4. for row 1 to height
5. do for col 1 to width
6. do Val Img(row,height,:)
7. // From RGB values, generate a single pixel value using a
8. // a function of RGB. We use Euclidean distances
9. *PixValue*
10. // Depending on the function used set MAX\_VALUE to the
11. // maximum value possible for a point
12. MAX\_VALUE *MAXIMUM VALUE POSSIBLE FOR PixValue*
13. Offset = 0
14. for n 1 to length(*BinSizes)*
15. do bin *BinSizes(n)*
16. Level\_1\_bins = floor(MAX\_VALUE/bin)+1
17. x floor(PixValue/bin)+1
18. BinImgs(row,col,Offset +x) 1
19. if PixValue ceil(bin/2) &
20. PixValue MAX\_VALUE - ceil(b/2)
21. do y floor(PixValue – ceil(bin/2)/bin)+1
22. BinImgs(row,col,Offset +Level\_1\_bins+y)
23. Offset = Offset + (2\*Level\_1\_bins -1)
24. end for n
25. end for col
26. end for row
28. //After Creating the overlapped bins we next create delta images