TPSNet: Reverse Thinking of Thin Plate Splines for Arbitrary Shape Scene Text Representation

Wei Wang  
wangweig3456@iie.ac.cn  
Institute of Information Engineering, Chinese Academy of Sciences  
School of Cyber Security, University of Chinese Academy of Sciences

Yu Zhou  
zhouyu@iie.ac.cn  
Institute of Information Engineering, Chinese Academy of Sciences  
School of Cyber Security, University of Chinese Academy of Sciences

Jiahao Lv  
lvjihaok21@mails.ucas.ac.cn  
Institute of Information Engineering, Chinese Academy of Sciences  
School of Cyber Security, University of Chinese Academy of Sciences

Dayan Wu  
wudayan@iie.ac.cn  
Institute of Information Engineering, Chinese Academy of Sciences

Guoqing Zhao  
guoqing.zhao02@msxf.com  
Mashang Consumer Finance Co., Ltd

Ning Jiang  
ning.jiang02@msxf.com  
Mashang Consumer Finance Co., Ltd

Weiping Wang  
wangweiping@iie.ac.cn  
Institute of Information Engineering, Chinese Academy of Sciences

ABSTRACT

The research focus of scene text detection and recognition has shifted to arbitrary shape text in recent years, where the text shape representation is a fundamental problem. An ideal representation should be compact, complete, efficient, and reusable for subsequent recognition in our opinion. However, previous representations have flaws in one or more aspects. Thin-Plate-Spline (TPS) transformation has achieved great success in scene text recognition. Inspired by this, we reversely think of its usage and sophisticationally take TPS as an exquisite representation for arbitrary shape text representation. The TPS representation is compact, complete, and efficient. With the predicted TPS parameters, the detected text region can be directly rectified to a near-horizontal one to assist the subsequent recognition. To further exploit the potential of the TPS representation, the Border Alignment Loss is proposed. Based on these designs, we implement the text detector TPSNet, which can be extended to a text spotter conveniently. Extensive evaluation and ablation of several public benchmarks demonstrate the effectiveness and superiority of the proposed method for text representation and spotting. Particularly, TPSNet achieves the detection F-Measure improvement of 4.4% (78.4% vs. 74.0%) on Art dataset and the end-to-end spotting F-Measure improvement of 5.0% (78.5% vs. 73.5%) on Total-Text, which are large margins with no bells and whistles.

CC BY

This work is licensed under a Creative Commons Attribution International 4.0 License.

KEYWORDS

Scene Text Representation; Scene Text Detection; Scene Text Spotting

ACM Reference Format:

Wei Wang, Yu Zhou, Jiahao Lv, Dayan Wu, Guoqing Zhao, Ning Jiang, and Weiping Wang. 2022. TPSNet: Reverse Thinking of Thin Plate Splines for Arbitrary Shape Scene Text Representation. In Proceedings of the 30th ACM International Conference on Multimedia (MM '22), October 10–14, 2022, Lisbon, Portugal. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3503161.3547882

1 INTRODUCTION

Scene text detection and recognition have become increasing prevalent topics in computer vision, due to their various applications in document image analysis [81], scene understanding [85], autonomous driving [95], etc. Arbitrary shape text is one of the main challenges for detection and recognition, and lots of profound approaches have been proposed. For handling various text shapes, the fundamental problem is how to represent the shapes.

Text shape representations can be classified into two types: segmentation-based methods and regression-based methods. Segmentation-based methods represent text regions with pixel-level classification masks that allow flexibility for arbitrary shapes, but they have drawbacks such as computationally intensive post-processing and the lack of noise resistance. The regression-based methods regress the text boundaries directly, making the prediction process much more straightforward. For horizontal and multi-oriented straight text, regressing the quadrilateral is sufficient to represent the text shape [16, 17, 22, 25, 30, 91]. However,
Apart from directly increasing the number of points to represent
TPS. The red lines are the fitting curves and the green lines
basic shape of text, and it is adaptive to the characteristics of large
presentation. Firstly, the TPS representation takes the rectangle as the
shape embedding and meet all the three criteria for ideal text repre-
dinate on the source arbitrary shape. Since the target rectangle can
 correspond to the control points, with which the source shape can
text for recognition, the TPS parameters are solved based on the
and simple representation for arbitrary shape text. When rectifying
representation to the best of our knowledge. If we think of
scene text recognition, it has not been directly applied for scene
the horizontal regular region so that the classical simple methods
sequently in tight and smoother contour curves, but some of them are not suitable for the
characteristics of text shapes as shown in Fig. 1 (b) and (c).

In our opinion, an ideal representation for arbitrary shape text
should: (1) be compact and complete so that the presence of back-
ground pixels and the missing text pixels are as few as possible; (2)
simple and efficient so that less time is consumed on detection; (3)
facilitate following recognition, making curved texts be recognized
simply and accurately. Although there are various representation
methods for texts, most of them can not satisfy all the criteria.

To better represent arbitrary shape text, we propose a new repre-
sentation: Thin-Plate-Spline (TPS) representation. TPS transforma-
tion [3] is typically applied in scene text recognition for rectification
[46, 52, 55, 56, 79, 86], where the irregular text region is rectified to the
horizontal regular region so that the classical simple methods
like CRNN [54] can recognize it well. Though TPS is effective in
scene text recognition, it has not been directly applied for scene
text representation to the best of our knowledge. If we think of
the usage of TPS reversely, as shown in Fig. 1 (a), it can be a novel
and simple representation for arbitrary shape text. When rectifying
text for recognition, the TPS parameters are solved based on the
 corresponding control points, with which the source shape can
be transformed into the target rectangular shape. Considering the
reverse process of rectification, if getting the TPS parameters, every
coordinate on the target rectangle can be transformed to the coor-
dinate on the source arbitrary shape. Since the target rectangle can
be fixed, the TPS parameter vector can be taken as an appropriate
shape embedding and meet all the three criteria for ideal text repre-
sentation. Firstly, the TPS representation takes the rectangle as the
basic shape of text, and it is adaptive to the characteristics of large
aspect ratios and right angles in the corners, as shown in Fig. 1. Sec-
ondly, arbitrary shapes are encoded to the low-dimensional vectors
that can be regressed directly, and decoding is also efficient. Thirdly,
the TPS parameters can naturally rectify curved texts, leveraging
the predicted shape for accurate recognition.

Generally, the TPS parameters should be derived from the coor-
dinates of control points. However, there is another problem that
the boundary of the arbitrary shape text is casually annotated by
sparse points without strict rules. Even the number of points is
different. As a result, the ground truth of the control points lacks
reliable definition, and the prediction for sparse points is not robust
and hard to be optimized. So we choose to regress the TPS parame-
ters directly and propose the Border Alignment Loss to supervise
it, which abandons the hard point matching and exploits shape
alignment to make the TPS parameters regression more accurate.

The contributions of this work are summarized as follows:

- An exquisite representation - TPS representation is first pro-
 posed for arbitrary shape text detection. The inspiration is
 from the TPS transformation used in text recognition ex-
tensively and sophisticated reverse thinking. It is compact,
 complete, efficient, and can be reused in recognition.
- To address the ambiguity of the boundary annotation and
 improve the supervision of the text shape, we design the
Border Alignment Loss to exploit the potential of the TPS
representation to obtain robust performance.
- TPSNet equipped with TPS representation and proposed loss
function is presented, and it is extended to a text spotter with
a simple recognizer. TPSNet is evaluated on several scene text
detection and spotting benchmarks, and the performance is
superior to previous counterparts.

2 RELATED WORKS

2.1 Segmentation-based Text Representation

As a special kind of object, the scene text needs to be represented
appropriately for accurate detection. Segmentation mask is the
common representation and has been widely used [2, 5, 9, 13, 23, 24,
27, 33, 36, 37, 39, 45, 48, 50, 51, 58, 61, 62, 68, 70, 73, 74, 76, 77, 80, 82,
92, 94]. The mask can naturally represent arbitrary shape text, but
it has the limitation of confusing different text instances and suffers
from the lack of noise resistance and computationally intensive
post-processing. Some methods represent the text with a set of
text components [4, 10, 11, 41, 53, 57, 59, 88], which also belong
to segmentation-based methods, but the units are text segments
rather than pixels.

2.2 Regression-based Text Representation

Regressing the geometry of the text shape and position is an-
other kind of representation. For the horizontal and multi-oriented
straight text, a rectangle [15, 16, 18, 26, 32, 49, 64, 90, 91] or quadri-
lateral [25, 28, 40, 77, 83] is sufficient. For curved texts, the represen-
tation becomes complicated. TextSnake [38] and MSR [78] regress
the distance to the text boundary, which is similar to segmentation-
based methods. Methods like [8, 65, 71, 89] directly regress the
contour points as the text boundary. The points regression is more
efficient than segmentation since there is no complicated post-
processing. However, the position of every independent point on
the boundary is not well defined when it comes to a variety of text shapes, making the points prediction not robust.

The better choice is to represent text with the parameter curves, which abstract the text shape into a vector. TextRay [63] employs the Chebyshev polynomials under the polar coordinate system to approximate the boundary, but the distribution of sampling points under the polar system is not homogeneous on the boundary, making it hard to fit the long and highly curved texts. However, although the Fourier curve is an excellent fitter, it struggles to fit text at right angle corners with relatively fewer parameters, resulting in incomplete shapes, as shown in Fig. 1 (c). All the representations above have the limitation that they can not directly rectify the irregular shape text for subsequent recognition. ABCNet [31, 35] formulates the long sides of the text containing distortion, while they are still deformed from the basic rectangle, mostly retaining the characteristics of right-angle corners and large aspect ratios. From the perspective of deformation, we try to establish a mapping between arbitrary text shapes to regular rectangles, thus enabling arbitrary shape text representations.

Our proposed TPS representation employs a fundamentally different fitting formula to provide another choice, which meets all three criteria and has more flexibility for the variations of the text shape. Furthermore, TPS representation is also superior to Bezier representation in the case of arbitrary-shape text with severe perspective distortion, which is frequently encountered in real applications though is scarce in public benchmark datasets.

3 METHODOLOGY

3.1 Thin-Plate-Spline Representation

Text is a special kind of object. In standard cases such as document texts, the shape of a word or a text line is generally a (rotated) rectangle. In scene images, texts may be presented in various shapes containing distortion, while they are still deformed from the basic rectangle, mostly retaining the characteristics of right-angle corners and large aspect ratios. From the perspective of deformation, we try to establish a mapping between arbitrary text shapes to regular rectangles, thus enabling arbitrary shape text representations.

TPS [3] has been widely used as the non-rigid transformation model in image alignment and shape matching. We apply TPS as the basic model to implement the deformation from an arbitrary shape to a regular rectangle, which requires obtaining the correspondence of every coordinate from the rectangle to an arbitrary shape. We formulate the “target” shape as the rectangle A and the “source” text shape as S, and the rectangle A is also called fiducial shape. According to TPS [3], the corresponding point of (x, y) ∈ A on S can be calculated by

\[
x' = \Phi_x(x, y) = c_x + a_{1x}x + a_{2x}y + \sum_{i=1}^{k} w_{ix}r(d_i)
\]

\[
y' = \Phi_y(x, y) = c_y + a_{1y}x + a_{2y}y + \sum_{i=1}^{k} w_{iy}r(d_i)
\]

where \( r \) is the radial basis function

\[
r(d) = \begin{cases} 
0, & \text{if } d = 0 \\
\frac{1}{d^2}, & \text{otherwise} 
\end{cases}
\]

\( d_i = ||(x, y) - (x_i, y_i)||_2 \) is the distance from (x, y) to (x_i, y_i). \{(x_1, y_1), (x_2, y_2), ..., (x_k, y_k)\} are the fixed points on the fiducial shape A, called fiducial points, and \( k \) is the number of fiducial points. Given fiducial points, the basis function is defined as

\[
\phi(x, y) = \begin{bmatrix} 1 & x & y & r(d_1) & ... & r(d_k)\end{bmatrix}^T
\]

then the TPS transform function is determined by the parameters

\[
T = \begin{bmatrix}
c_x & a_{1x} & a_{2x} & w_{1x} & ... & w_{kx} \\
c_y & a_{1y} & a_{2y} & w_{1y} & ... & w_{ky}
\end{bmatrix}
\]

with the shape of \( 2 \times (k + 3) \).

With the TPS transform function, the grids (x, y) on A can be transformed into the corresponding points on S, where the text...
boundaries are naturally obtained, as shown in Fig. 2. Note that, the grids on $A$ are predefined, so the $\phi(x, y)$ is also calculated in advance. The TPS parameters $T$ can be decoded quickly to text shape with a matrix multiplication

$$
(x', y')^T = T\phi(x, y)
$$

(5)

Any set of points not lying on the same line can be taken as the fiducial points to define a basis function $\phi(x, y)$, which determines the fitting ability of the TPS parameters. For simplicity and efficiency, the number of fiducial points $k$ is set to 8, thus the dimension of TPS parameters $T$ is 22, which is enough to fit nearly all the text shapes. There are different options for the distribution of the fiducial points, as shown in Fig. 3. Except for the four corner points that should be fixed to determine the base location, the other fiducial points can locate not only on the edges like (b) but also inside the shape like (c) cross and (d) center. The corresponding points on text shape $S$ will distribute in the same relative position. This is different from Bezier [31], which uses two sets of control points to fit two edges separately. With more points located along the width, the longer curved text can be fitted, while with more points on the edge, the text boundary will be more accurate. Visualizations of fitting results with different distributions can be found in Appendix A.2. The ablation study is conducted to decide the optimal distribution.

We emphasize that, unlike previous detection or rectification methods, we do not attempt to predict control points on the text shape $S$ because the locations of control points on arbitrary shapes are not well-defined, and we directly regress the TPS parameters $T$ with the neural network instead.

To verify the fitting ability of TPS, we solve the TPS parameters $T$ with the standard least square method from equation (5). The visualization of the shape fitting is shown in the Fig. 1, and the quantitative evaluation can be found in Appendix A.1. The TextRay [63] fails on highly-curved shapes, and both TextRay [63] and Fourier [93] fail on extreme aspect ratio cases and miss the text corners, and they cannot rectify irregular text for subsequent recognition either.

The Bezier [31, 35] representation is the closest one to TPS, but it employs a different fitting formula:

$$(x', y')^T = \left[ \begin{array}{c} w_{0x} \\ w_{0y} \\ \vdots \\ w_{nx} \\ w_{ny} \end{array} \right] \left[ \begin{array}{cccc} B_0,0(t) & \cdots & B_{n,0}(t) \\ \vdots & & \vdots \\ B_0,n(t) & \cdots & B_{n,n}(t) \end{array} \right]$$

where $(w_{ix}, w_{iy})$ is the $i$th control points and $B_{i,n}(t)$ represents the Bernstein basis polynomials [31]. Our TPS representation Equ (5) can be rewritten as:

$$
(x', y')^T = \left[ \begin{array}{cccc} c_x & a_{1x} & a_{2x} & \cdots & a_{nx} \\ c_y & a_{1y} & a_{2y} & \cdots & a_{ny} \end{array} \right] \left[ \begin{array}{c} 1 \ x \ y \ \vdots \ r(d_1) \ \cdots \ r(d_k) \end{array} \right]$$

(7)

The Bezier formula is similar to the second part of the TPS formula, which reveals the local shapes. However, it misses the first part, which represents the global affine transformation. In addition, the basis function of Bezier $B_{i,n}(t)$ is one-dimensional function, while that of TPS $\phi(x, y)$ is two-dimensional. These differences allow our proposed TPS representation to handle more variations of text shapes. As shown in Fig. 4, Bezier and TPS with the same number of control points are used to fit the text in a perspective image. As (a) and (b) show, the TPS fits better than Bezier. In (c) and (d), when the perspective degree of the image is enlarged, Bezier fails to fit this perspective curved text, and the rectification of the text is incomplete while our TPS still works well.

### 3.2 Border Alignment Loss

With the TPS transformation, the geometric shape is abstracted to the TPS parameter $T$, and coordinates in the shape space are determined by all elements of $T$ in parameter space. Directly optimizing $T$ in parameter space with distance loss like mean absolute error treats each element independently and neglects the intra-parameter correlations [63]. To keep the inherent properties of the TPS transformation, $T$ should be decoded into the shape space first, and the loss can be calculated as the distance of boundary point pairs between the decoded shape and its ground truth.

As shown in Fig. 5 (a), the text boundaries are annotated with sparse points. These limited number of points roughly describe the boundary, and because of the lack of strict definition, the positions of these points are not unique. The number of the annotation point is even different across datasets. For example, CTW1500 uses 14 points while Total-Text uses ten or less. In other words, the ground truth of the boundary points is ambiguous and noisy. Directly regressing points to the ground truth points will bring noise and ambiguity to the network optimization. As shown in Fig. 5 (b), the prediction (red points) have already located on the text boundary and can describe this text instance well, but some points are still far from the noisy ground truth (green points), the undesired loss will be backward to the network, disrupting its convergence.

To address this problem, we propose the Border Alignment Loss (BA-Loss). As shown in Fig. 5 (c), Firstly, the noisy polygon annotation is smoothed with cubic spline interpolation[42], and then the
Points regression

\[ M' \] …

Bilinear Grid Sample

\[ L_{bbox} = \sum_{p \in B} (1 - \text{Sample}(M', p)) \]

Boundary Gradients

\[ L_{border} = \sum_{p \in B} (1 - \text{Sample}(M', p)) \]

\[ L_{reg} = \frac{1}{N} \sum_{t=0}^{N} \frac{1}{| \Omega_t |} (L_{BA}(B_t) + L_{cor}(B_t)) \]

The annotations of the corner are relatively robust, so \( L_{cor} \) will not cause ambiguity.

The total regression loss is:

\[ L_{reg} = \frac{1}{N} \sum_{t=0}^{N} \frac{1}{| \Omega_t |} (L_{BA}(B_t) + L_{cor}(B_t)) \]

where \( | \Omega_t | \) denotes the area of text instance \( t \), \( B_t \) means the boundary of \( t \), and \( N \) is the total number of text instances.

3.3 TPSNet

3.3.1 Text Detection. Following previous regression-based text detection network [63, 91, 93], we adopt a compact one-stage fully-convolutional framework. As shown in Fig.2, for an input image, multi-scale features are extracted with the backbone and Feature Pyramid Network (FPN) [29]. The detection head takes each level feature as input and uses convolution layers to predict text scores and TPS parameters maps. Following Textsnake [38], the text scores consist of the per-pixel masks of Text Region (TR) and Text Center (TC) classifications, and these two masks are multiplied as the confidence of the detection for every position. For every feature bin located inside of TR, the TPS parameters vector \( T \) is regressed for the corresponding text instance. Then \( T \) is decoded to text shape grids by TPS transform with Equ (5), where the fiducial shape is a pre-defined grid. From the text shape, the text boundary can be obtained directly, and it is also convenient to rectify the text by sampling on the input image. Duplicated detection results are removed with Non-Maximum Suppression (NMS).

For the TC classification, the Text Center Region(TCR) in previous methods are replaced with our proposed Gaussian Text Center(GTC), details can be found in Appendix B.2.

The optimization objectives of the classification branch and regression branch are \( L_{cls} \) and \( L_{reg} \) respectively, and the TPSNet for detection is optimized by:

\[ L = L_{cls} + L_{reg} \]

The classification loss consists of the Text Region loss \( L_{TR} \) and Text Center Region loss \( L_{TC} \):

\[ L_{cls} = L_{TR} + L_{TC} \]

Both \( L_{TR} \) and \( L_{TC} \) are cross entropy losses. To solve the sample imbalance problem, OHEM [93] is adopted for \( L_{TR} \).

3.3.2 Text Spotting. The detection model can be extended to an end-to-end text spotter with ease because it is convenient for the TPS representation to align the feature of the irregular shape text. The feature vector at each text shape grid point on the feature map is obtained with grid sampling as Spatial Transformer Network (STN) [19] to compose the text feature.
Any recognition model can be applied for the recognition. For simplicity, we take the same recognition module as ABCNetV2 [35], which consists of 6 convolutional layers, one bidirectional LSTM layer, and an attention-based decoder. For end-to-end training, the whole loss function is

\[ L = L_{\text{cls}} + L_{\text{reg}} + L_{\text{rec}} \]  

where \( L_{\text{rec}} \) is the Cross Entropy Loss for the recognition as in ABCNetV2 [35].

4 EXPERIMENTS

In this section, we evaluate our proposed TPSNet by CTW1500, Total-Text, ICDAR2015 and Art datasets to validate its effectiveness. We first conduct some ablation studies to demonstrate the advantages of proposed designs and the setting of hyper-parameters. Then we compare the detection performance of our model with previous state-of-the-art methods. Finally, the spotting performance on arbitrary shape scene text datasets are evaluated.

4.1 Datasets

**Total-Text** [6] includes curved, horizontal, and multi-oriented text. It consists of 1, 255 training images and 300 test images. All text annotations are word-level. Text areas are annotated with polygons. **SUCT-CTW1500** [84] is a dataset for the curved text. It contains 1,000 training images and 500 test images. Text is represented by polygons with 14 points at text-line level. **ICDAR2015** [20] is a multi-oriented text detection dataset only for English, which includes 1000 training images and 500 testing images. The text regions are annotated with quadrilaterals. **ArT** [7] is a large-scale multi-lingual arbitrary shape scene text detection dataset and is one of most complex datasets. It includes 5,603 training images and 4,563 testing images. The text regions are annotated by the polygons with adaptive number of key points. **SynthText-150K** [31] is the synthetic datasets generated based on the method from [14], and it includes nearly 150k images that contains straight and curved texts.

4.2 Implementation Details

We implement our TPSNet based on MMOCR [21] with Pytorch [43] library. The backbone is ResNet50 pretrained on ImageNet with DCN in stage 2, 3 and 4, followed by FPN. Feature maps of P3, P4 and P5 are used in classification and regression branch, and P2, P3, P4 are used for recognition. 4 convolutional layers of 3 x 3 are applied for the text region and text center classification and the TPS regression. Text instances are assigned into different feature maps according to its scale ratio (instance scale/image scale).

The training images are resized to 800 × 800, and data augmentation strategies are applied, including ColorJitter, RandomCrop, and RandomRotate. The training batch size is set to 8. Stochastic gradient descent (SGD) is adopted as an optimizer with a weight decay of 0.001 and a momentum of 0.9. For the detection-only training, the learning rate is initialized at 0.001 and adjusted by the Poly policy with a power of 0.9. The TPSNet is pretrained on Synthtext-150K datasets for 150k iteration, and the numbers of the finetune epochs are 250, 500, 500, and 200 for Total-Text, CTW1500, ICDAR2015, and ArT separately. For end-to-end text spotter, the training schedule is set totally according to ABCNet V2 [35], it is first pretrained on Synthtext-150K and MLT17 (English-only) for 260k iteration, and then finetuned 10k iterations for Total-Text and 80k for CTW1500. In the test stage, the short sides of the test image are set to 736, 736, 1080, 1600 for Total-Text, CTW1500, ICDAR2015, and ArT, while the long sides are resized to keep the original aspect ratio. All experiments are conducted on a single NVIDIA RTX3090 GPU.

### Table 1: Ablation study about representations and supervisions on Total-Text. “Rep” means Representation, including directly regressing points or TPS parameters. \( L(B) \) means matching predicted boundary points to ground truth points as Fig. 5 (b). TCR means the text center region, GTC means the gaussian text center. \( L_{\text{cor}}, L_{\text{BA}} \) and Border Relax are our proposed losses in Section 3.2.

| Points | \( L(B) \) | TCR | GTC | \( L_{\text{cor}} \) | \( L_{\text{BA}} \) | Border Relax | H |
|--------|-------------|-----|-----|----------------|----------------|-------------|---|
| ✓      | ✓           | ✓   | ✓   | ✓             | ✓              | ✓           | 81.3 |
| ✓      | ✓           | ✓   | ✓   | ✓             | ✓              | ✓           | 83.5 |
| ✓      | ✓           | ✓   | ✓   | ✓             | ✓              | ✓           | 84.7 |
| ✓      | ✓           | ✓   | ✓   | ✓             | ✓              | ✓           | 56.7 |
| ✓      | ✓           | ✓   | ✓   | ✓             | ✓              | ✓           | 85.8 |
| ✓      | ✓           | ✓   | ✓   | ✓             | ✓              | ✓           | 86.6 |

is set totally according to ABCNet V2 [35], it is first pretrained on Synthtext-150K and MLT17 (English-only) for 260k iteration, and then finetuned 10k iterations for Total-Text and 80k for CTW1500.

4.3 Ablation Study

The ablation study is conducted on the Total-Text dataset without pretraining. As shown in Table 1, to verify the advantage of the TPS representation, we set a baseline model that directly regresses the boundary points and uses the simple boundary distance loss \( L(B) \) as Fig. 5 (b) and TCR for classification. Compared with the points representation, TPS representation brings 2.2% improvements on Hmean, proving abstracting geometric shapes to TPS parameter space can represent text better. The Gaussian Text Center (GTC) performs 1.2% better than the Text Center Region (TCR). The boundary distance loss \( L(B) \) is not a good choice to optimize the TPS regression as we have discussed above, so we replace it with our proposed Border Alignment Loss. The corner loss \( L_{\text{cor}} \) alone can only locate the basic position of text but not fit curved texts, while \( L_{\text{BA}} \) makes this part, they work together to bring 1.1% improvements on Hmean, and the border relaxation brings another 0.8% improvement. The relax threshold \( t_r \) is set to 0.8, of which the ablation is present in Appendix B.1. Note that the Border Assignment Loss leverage the continuity of TPS representation that dense boundary points can be decoded from it, so \( L_{\text{BA}} \) can not apply to the direct points regression. In other words, the TPS representation brings totally the 5.3% improvement on Hmean. The ablation study about the distribution of the fiducial points is present in Appendix A.2, the Cross is selected as the default setting.

4.4 Detection Evaluation

#### 4.4.1 Compared with Bezier

Bezier is another impressive representation for the arbitrary shape text, and it employs a different formula from TPS. To compare with Bezier fairly, we replace the TPS representation with the Bezier representation of 8 control points in our framework, and both models of TPS and Bezier are trained with the same loss functions on the training set of CTW1500. Following [93], the highly curved are sampled from the test set of CTW1500. Nearly all the images in the dataset are focused on the text with no perspective, but in the real world, scene texts are often...
Table 2: Comparison on highly curved text subset of CTW1500 test set with variations on the degree of perspective. The Performances are Hmeans under IOU constant at 0.7. The perspective augmentation is applied to training set, and the test set is perspective with an angle at 0\(^\circ\) (original), 45\(^\circ\) and 70\(^\circ\) for evaluation.

| Reps | Perspective Augmentation | 0\(^\circ\) | 45\(^\circ\) | 70\(^\circ\) |
|------|--------------------------|----------|----------|----------|
| Bezier | ✓ | 71.6 | 64.8 | 54.7 |
| TPS | ✓ | 72.1 | 66.2 | 56.2 |

Figure 6: Visualization of the detection results of Bezier representation (top) and TPS representation (bottom) on highly curved text. From left to right, the image is transformed to larger degrees of perspective.

viewed sideways. To simulate this situation, we apply a perspective transformation to the images: Rotate the image along the left edge and then reproject the image onto the plane to get the perspective image. The perspective transformation can be applied as a data augmentation to the training process, where the rotation angle is randomly selected from the range of 0\(^\circ\) - 70\(^\circ\), and the highly curved test subset is transformed with the rotation angle at 0\(^\circ\), 45\(^\circ\) and 70\(^\circ\) to evaluate the TPS and Bezier representation. The results are shown in Table 2. Without perspective augmentation, our TPSNet performs slightly better than Bezier. With perspective augmentation, our TPSNet is improved much more than Bezier, especially on a perspective angle at 70\(^\circ\) (9.0 vs. 5.3). The visualization results are shown in Fig. 6. As discussed in Section 3.1, our TPS representation has more flexibility to fit the highly curved text with perspective, while the Bezier with limited control points can not. The results prove that the TPS better represents the arbitrary shape scene text.

4.4.2 Evaluation with TIOU Metric. TIOU [34] is a well-known evaluation protocol for scene text detection. TIOU-Recall and TIOU-Precision can separately quantify the completeness and compactness of the detection results, and the TIOU-Hmean quantifies the overall tightness of the matching degree. Tighter detection means less miss of character and less presence of background, which is a stricter metric than the VOC metric. As shown in Table 3, our TPSNet achieves the best performance on all three metrics. The FCENet [93] is the most competitive method with our method on

| Methods | TIOU-R | TIOU-P | TIOU-H |
|---------|--------|--------|--------|
| FCENet [93] | 58.6 | 69.8 | 63.7 |
| PCR [8] | 58.0 | 69.9 | 63.4 |
| BPN [89] | 61.5 | 69.2 | 65.1 |
| TPSNet (ours) | 62.9 | 72.4 | 67.3 |

Figure 7: Qualitative comparisons with previous methods on selected challenging samples in CTW1500.

Table 3: Evaluation with TIOU [34] metric on CTW1500. Detection-only training is applied.

the common VOC metric, but on the TIOU metric, TPSNet outperforms it by 3.6\%, which means that the detection results from TPSNet are tighter than previous methods. Qualitative comparison is shown in Fig. 7. ABCNetV2 [31] and PCR [8] fail in long and dense texts, while TextRay [63], FCENet [93] and BPN [89] prefer to missing the corners of long text, which is not conducive to the subsequent recognition. By comparison, our proposed TPSNet obtained the most compact and complete detection.

4.4.3 Evaluation with Benchmark Metric. We evaluate our TPSNet on benchmark datasets and compare with previous detection methods as shown in Table 4 and 5. Previous methods are divided into three categories: segmentation-based, regression-based, and hybrid-based methods that use both segmentation and regression for detection. As a regression-based method, our TPSNet can outperform previous methods on curved scene text detection datasets CTW1500 and Total-Text. For multi-oriented scene text in ICDAR2015, our proposed TPSNet can also achieve the best performance because the TPS representation adopts the rectangular fiducial shape, and when it is applied for multi-oriented text, the TPS parameters \(T\) will collapse to \([c_x, a_1, a_2; c_y, a_1, a_2]\) that denotes an affine transformation that is still effective. In addition, even without extra data for pretraining, the TPSNet is still comparable with or even better than previous methods with extra data, which proves that the TPSNet
**Table 4:** Comparison with previous detection methods on CTW1500, Total-Text and ICDAR2015 test sets. "Ext" means using the external dataset to pretrain the model. * denotes the results based on end-to-end text spotting training.

| Methods       | CTW1500 | Total-Text | ICDAR2015 |
|---------------|---------|------------|-----------|
|               | R      | P        | H        | FPS   | R     | P       | H       | FPS   |
| TextSnake     | ✓       | 85.3     | 67.9     | 75.6   | –     | 74.5    | 82.7    | 78.4   | –     | 80.4    | 84.9    | 82.6   | 1.1   |
| PSENet [68]   | ✓       | 79.7     | 84.8     | 82.2   | 3.9   | 78.0    | 84.0    | 80.9   | 3.9   | 84.5    | 86.9    | 85.7   | 1.6   |
| CRAFT [1]     | ✓       | 81.1     | 86.0     | 83.5   | –     | 79.9    | 87.6    | 83.6   | –     | 84.3    | 89.8    | 86.9   | 8.6   |
| SAE [61]      | ✓       | 77.8     | 82.7     | 80.1   | 4.3   | 77.8    | 82.7    | 80.1   | 4.3   | 85.0    | 88.3    | 86.6   | –     |
| MSR [78]      | ✓       | 78.3     | 85.0     | 81.5   | 4.3   | 74.8    | 83.8    | 79.0   | 4.3   | 78.4    | 86.6    | 82.3   | –     |
| PAN [70]      | ✓       | 81.2     | 86.4     | 83.7   | 39.8  | 81.0    | 89.3    | 85.0   | 39.6  | 81.9    | 84.0    | 82.9   | 26.1  |
| SAST [66]     | ✓       | 77.1     | 85.3     | 81.0   | –     | 76.9    | 83.8    | 80.2   | –     | 87.1    | 86.7    | 86.9   | –     |
| DB [27]       | ✓       | 80.2     | 86.9     | 83.4   | 22.0  | 82.5    | 87.1    | 84.7   | 32.0  | 83.2    | 91.8    | 87.5   | 13.2  |
| DRGRN [68]    | ✓       | 83.0     | 85.9     | 84.5   | –     | 84.9    | 86.5    | 85.7   | –     | 84.7    | 88.5    | 86.6   | 3.5   |

**Table 6:** Text spotting performance on Total-Text and CTW1500 datasets. "None" means lexicon-free, and "Full" represents using all the words appeared in the test set as lexicon. * denotes results based on multi-scale test.

| Methods       | Total-Text | CTW1500 |
|---------------|------------|---------|
|               | None      | Full | FPS | None | Full | FPS |
| TextDragon [10] | 44.8 | 74.8 | 2.6 | 39.7 | 72.4 |
| Unconstrained [47] | 67.8 | – | 4.8 | – | – |
| CharNet [75]   | 66.6 | – | 1.2 | – | – |
| ABCNet [31]    | 64.2 | 75.7 | 17.9 | 45.2 | 74.1 |
| Boundary [65]  | 65.0 | 76.1 | – | – | – |
| TextPerceptron [45] | 69.7 | 78.3 | – | 57.0 | – |
| PNet [67]      | 60.5 | – | 40.5 | – | – |
| MANGO [44]     | 72.9 | 83.6 | 4.3 | 58.9 | 78.7 |
| MaskTextSpotterv3 [24] | 71.2 | 78.4 | 2.5 | – | – |
| PAN++ [69]     | 68.6 | 78.6 | 21.1 | – | – |
| ABCNetV2 [35]  | 70.4 | 78.1 | 10.0 | 57.5 | 77.2 |
| ABCNetV2* [35] | 73.5 | 80.7 | – | 58.4 | 79.0 |

**Table 5:** Comparison with previous methods on Art. "Ext" means using the external dataset to pretrain the model.

| Methods     | Ext | R   | P   | H   | FPS |
|-------------|-----|-----|-----|-----|-----|
| ContourNet  | ✓   | 84.1 | 83.7 | 83.9 | 4.5 |
| Mask-TTD    | ✓   | 79.0 | 79.7 | 79.4 | –   |
| MaskTextSpotter* [23] | ✓ | – | – | – | 82.4 |
| ATRR [71]   | ✓   | 80.2 | 81.0 | 80.1 | 10.0 |
| Boundary*   | ✓   | 80.5 | 82.0 | 80.8 | 9.3 |
| TextRay     | ✓   | 80.4 | 82.8 | 81.6 | –   |
| FCENet      | ✓   | 83.4 | 87.6 | 85.5 | –   |
| PCR [8]     | ✓   | 82.3 | 87.2 | 84.7 | 11.8 |
| MOST [16]   | ✓   | 83.6 | 86.5 | 85.0 | 12.2 |
| BPN [89]    | ✓   | 83.8 | 85.8 | 84.7 | –   |
| ABCNetV2*   | ✓   | 83.7 | 88.1 | 85.9 | 17.9 |
| TPSNet(ours)| ✓   | 85.1 | 87.7 | 86.4 | 17.9 |
| TPSNet(ours)* | ✓ | 86.3 | 88.7 | 87.5 | 17.9 |

5 CONCLUSION

In this paper, we propose a novel TPS representation for arbitrary shape text, which comes from the sophisticated reverse thinking of Thin Plate Splines. The TPS representation is compact, complete, efficient, and reusable for subsequent recognition. To further exploit the potential of the TPS representation, the Border Alignment Loss is designed. Based on the representation and loss function, we implement the TPSNet and extend it to a text spotter. The TPSNet is evaluated on CTW1500, Total-Text, ICDAR2015, and Art datasets for text detection and spotting, and it outperforms previous counterparts with large margins.

**ACKNOWLEDGMENTS**

Supported by the Beijing Municipal Science & Technology Commission (Z191100007119002), the Key Research Program of Frontier Sciences, CAS, Grant NO ZDBS-LY-7024.
## A TPS REPRESENTATION

### A.1 Evaluation about fitting abilities

To verify the fitting ability of TPS, we solve the TPS parameters $T$ with the standard least square method from equation (5). The fitting boundary can be derived with equation (1), then evaluated the Tightness-IOU (TIOU) [34] comparing with the ground truth annotations. The TIOU value reveals the compactness and completeness of the fitting. The fitting results of different representations in several previous methods are demonstrated in Table 7, where TextRay and Fourier use 22-dimensional parameters, and both Bezier and TPS use 8 control points.

### Table 7: Comparison of text shape representation. TIOU [34] means the tightness-IOU between the fitted shapes and ground truth. “Corner”, “Rectify” means whether the representation can keep the corner, can rectify irregular text for recognition.

| Method    | TIOU-H | Corner | Rectify |
|-----------|--------|--------|---------|
| TextRay [63] | 82.5   | ×      | ×       |
| Fourier [93]    | 90.6   | ×      |         |
| Bezier [31, 35] | 96.8   | ✓      | ✓       |
| TPS (Ours)     | 97.1   | ✓      | ✓       |

### A.2 Distributions of Fiducial Points

The fiducial points with three different distributions can define different basis functions $\phi(x, y)$. To illustrate their differences, we use the standard least square method to solve the parameters for Bezier and TPS to fit the polygon annotations, and the results are shown in Figure 8. For Bezier, the solved parameters are the coordinates of the control points, and for TPS, the parameter $T$ is solved. As the blue points shown in (b), (c), and (d), with $T$, the fiducial points can be transformed into corresponding points on the text shape, and they will locate at the similar relative positions as the fiducial points, which can be seen as the control points for TPS. Obviously, with the same number of control points, Bezier fails to fit this highly curved text, but TPS works well. Furthermore, the “Cross” distribution can achieve the best shape fitting and rectification results. With the “Cross” distribution, the fiducial points alternately distribute at the edge and center of the text, balancing the length of the text and the accuracy of the edges.

### Table 8: Ablation study about different distribution of the fiducial points on CTW1500. (b), (c) and (d) refer to the distributions in Fig. 3.

```
| Fiducial Distribution | H[IOU@0.5] | H[IOU@0.7] |
|-----------------------|-----------|-----------|
| (b) Edge              | 85.5      | 74.3      |
| (c) Cross             | 85.9      | 76.0      |
| (d) Center            | 85.6      | 75.1      |
```

For the distribution of the fiducial points, we conduct the ablation study on CTW1500 without pretraining. As shown in Table 8, three distributions of the fiducial points in Fig. 3 are applied to construct the TPS representation. The performance is evaluated with IOU constraints at 0.5 and 0.7. When evaluated with IOU@0.5, there are only slight differences between these three distributions. But when evaluated with IOU@0.7, the second distribution Cross is clearly better than the others, indicating the crossed distribution of fiducial points at the edge and center allows for the better shape representation of the TPS.

### B TPSNET

#### B.1 Relaxation Threshold

The ablation study about the border relaxation threshold $t_r$ is conducted on Total-Text, and the results are shown in Table 9. $t_r = 1.0$ means no relaxation is applied to the border mask ground truth, and smaller $t_r$ leads to larger relaxation range. From the experiments, the border relaxation with $t_r = 0.8$ achieves the best performance, and more relaxation is harm for the model optimization.

### Table 9: The ablation study about the relax threshold $t_r$ on Total-Text. Detection-only training is applied without pre-training.

```
| $t_r$ | R     | P     | H     |
|-------|-------|-------|-------|
| 1.0   | 84.6  | 87.1  | 85.8  |
| 0.9   | 84.6  | 88.1  | 86.3  |
| 0.8   | 84.0  | 89.2  | 86.6  |
| 0.7   | 83.6  | 87.4  | 85.4  |
```

#### B.2 Gaussian Text Center

In the previous work FCENet [93], TC is the shrink region from the text region as shown in Fig. 10 (a), which is nearly as long as the text. However, this design is not suitable for a one-stage regression-based model because the position near one end of the long text can hardly perceive the shape of the other end, so only the prediction around the center point should be reserved. Centerness [60] is designed...
Visualizations of spotting results on test samples of CTW1500 are shown in Figure 9. For curved texts, our TPSNet can detect the shape of the text completely and compactly and identify the text content with the rectification accurately.

Figure 9: Visualizations of spotting results on test samples of CTW1500. Green lines are detection results, and the recognition results are marked nearby.

Figure 10: Illustration of the Text Center [38, 93] and our proposed Gaussian Text Center.

(a) Text Center Region

(b) Illustration of the generation process of Gaussian Text Center.

for the general object detection to solve this problem, but it does not suit for the arbitrary shape text. To this end, we propose the Gaussian Text Center as shown in Fig. 10 (b). The ground truth of the Gaussian Text Center (GTC) is generated as the process shown in Figure 10 (b). Firstly, we generate a two-dimensional gaussian distribution map in a rectangle, then we use TPS transformation to transform the rectangle to the arbitrary text shape, which is exactly the reverse process of text rectification. Note that, the center map does not need high precision, so using the noise annotation to solve this transformation will not affect the performance. The mask value on GTC varies continuously in range $0 - 1$, so the classification loss for the text center $L_{TC}$ becomes soft cross entropy loss:

$$L_{TC} = - \sum_{i} y_i \log p_i + (1 - y_i) \log (1 - p_i)$$

where $y_i$ is the value of $i$th pixel on GTC map, and $p_i$ is the predicted confidence.

C VISUALIZATIONS OF SPOTTING RESULTS

Visualizations of spotting results on test samples of CTW1500 are shown in Figure 9. For curved texts, our TPSNet can detect the shape of the text completely and compactly and identify the text content with the rectification accurately.