Contouring of arbitrarily-shaped text with Fourier Series

Irene González i Fernández

Resum– Per detectar text en formes arbitràries a una imatge un ha de donar una representació de l'espai que aquest ocupa que permeti a una xarxa neuronal aprendre les variacions del text. Els enfocaments més comuns són dónar una màscara de píxels on es troba el text o un contorn de punts que l'envoltin. Aquests dos enfocaments tenen els seus dèficits, com la manca de continuïtat. Una solució a això és modelar aquests espais com l'interior d'una corba tancada, determinada pels seus coeficients de Fourier. Aquest enfocament té el benefici de ser capaç de representar text en formes altament curvilínies, a la vegada de tenir una signatura molt lleugera. Construïm una xarxa neuronal per estimar aquestes sèries de Fourier, determinada pels seus coeficients de Fourier, que corresponen a la seva signatura.

Paraules clau– Sèries de Fourier, xarxes neuronals, detecció de text en escena, contorn de text arbitrari, Total-Text dataset

Abstract– To detect arbitrarily-shaped text one must first design a text instance representation that enables a neural network to learn text variances. The most usual approach is to model the space that text occupies in an image via masks or contours of points, which have their clear deficits, such as non-continuity. A solution to this is to model these spaces as the inside of a curve, determined by its Fourier coefficients. This approach has the advantage of being able to represent highly-curved shapes as well as being able to do so with a small signature. We construct a neural network to estimate such Fourier series by calculating appropriate Fourier coefficients, which correspond to its signature.

Keywords– Fourier Series, neural networks, scene text detection, arbitrary text contouring, Total-Text dataset

1 Introduction

When developing an OCR application, it is helpful to first identify the position and orientation of the text in the image, in order to simplify the problem of transforming the image of text into actual plain text.

To ease this initial step it is often assumed that the text follows some convenient properties, such as being written in a straight line and even being presented following a horizontal orientation.

Of course, this is not always the case with scene text, where an intermediate step to straighten the image of the text could be required.

To extract positional information of scene text following arbitrary shapes, it is often useful to consider contours or masks to flexibly indicate such properties. Examples of these methods include

1. Using *contour points* to define a polygon surrounding the text. This has the benefit of being a lightweight solution, although it is important to remember to preserve the ordering of the points, as different orderings of the same points may generate vastly different polygons.

   This method also presents the benefit of providing closed, well defined, polygons. Note however that an increasing amount of points are needed to represent more complex shapes, specially in the case of curved text, present for example in Fig. 1.

2. Using *masks* to signal which pixels correspond to text in the scene. This solution requires more memory to store its results, but has pixel-perfect precision capabilities, not matched by most other formats.

   It has drawbacks such as presenting an output format where individual connected components are

• Correspondence address: irene.gonzalezf@e-campus.uab.cat
• Minor: Enginyeria de Computació
• Tutored by: Dimosthenis Karatzas (Computer Science Department)
• Class of 2020/21

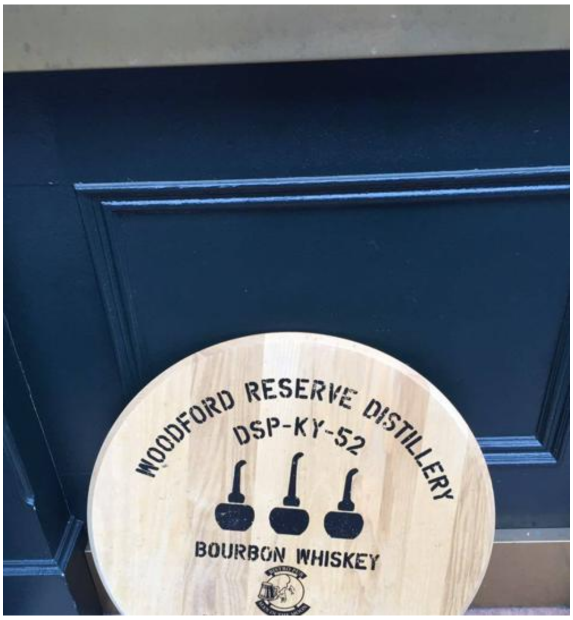

Fig. 1: Example of an image with a non-trivial text contour

hard to identify or having a tendency to lead to stray pixels, both false positives and false negatives, if no explicit care is taken. It may also be difficult to express orientation of text with this format (such as being upside-down or mirrored).

These solutions are not perfect, and as such the problem of extracting the contour of texts in an image is an open problem, with a very active field of research focused on it.

This document proposes another solution attempt for this problem. We aim to build a neural network that allows us to find the contour of text in an image and store it in a format different to the previously exposed, this being as a Fourier series of coefficients corresponding to a closed curve around the image.

A Fourier series is a series of real coefficients that uniquely determine a continuous closed curve as a parametrized function. Finding this curve from a Fourier series is as easy as a plugging the coefficients in a formula which we will describe in section 4.

In the conclusions, after having discussed the methodology and properties of the Fourier series, we will discuss the benefits of such approach.

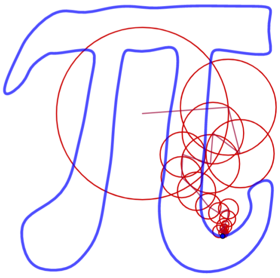

Fig. 2: Example of a how a Fourier series can construct a closed, continuous curve on the plane through the Fourier coefficients of the curve.

In this document we develop a neural network that approximates the contour of texts that appear in an image through the use of Fourier series. The idea of obtaining the contour of a shape or object with a Fourier series is applicable to many fields, *e.g.* object detection, but we choose the contour of texts as a proof-of-concept.

The first steps of the project have been in a more general setting, detecting synthetic shapes in images as a first internal trial, to then adapt the knowledge obtained and work in a non-synthetic case.

Additionally, we expanded our tentative goals to attempt to define a closed curve through its Fourier coefficients that tightly surrounds text in an image.

Later in this text we specify the goals more precisely, with the methodology and planning.

2 State-of-the-art

The subject of detection and description of shapes sees much modern research at current times, and as such is a growing and changing technical landscape.

The IAPR's Technical Committee 10 on Graphics Recognition, TC-10, organises a workshop every odd year since 1995, which enjoy strong participation from researchers in both industry and academia. These activities helps push the frontiers of computer vision.

On one hand, for pixel-based methods score masks are first obtained using a segmentation framework, and then they are grouped according to the text components to obtain a series of masks [6, 10, 11]. In order to increase the performance, for example [11] generates candidate text parts via linking neighbour pixels with a deep direction field.

On the other hand, regression-based methods aim to handle complex geometric variances by adopting the direct shape modelling of text instances [4, 7, 12, 13, 14, 8]. They are often much simpler to train, although their constrained representation capability may limit the performance of the network.

Coincidentally, since the start of this thesis, a paper greatly related to our work has been published, namely [15]. This helped reassure that the approach we are taking can indeed produce positive results, and does so by giving an example, which we use as source of inspiration when planning the topology to our network.

3 Goals and approach

3.1 Synthetic closed curves

The first objective is to obtain the discrete Fourier coefficients of a closed curve.

More specifically, we want to begin from the rendering of closed, differential curve and obtain though a neural network the discrete Fourier coefficients that will allow us to reproduce it in an approximate manner.

To accomplish this goal it is also necessary to create a dataset of images and, at the same time, to obtain the discrete Fourier coefficients through the formulas given in definition 4.

Also, in parallel to the dataset creation, we must obtain a loss function, since we now have the necessary inputs and labels for it.

3.2 Arbitrary text contouring

Finally, as a tentative goal, we want to build a neural network to calculate the Fourier coefficients of a curve that will minimally border the shape of text in an image.

The challenge in the goal is that we do not assume that the text sits in a straight line.

This can be used to better read text from an scene, as we can deduce the orientation of the letters to adapt their reading.

Additionally, since we only need to store the Fourier coefficients that generate the enclosing curve, instead of information about it.

4 Mathematical notions

We start by providing the needed mathematical background.

Definition 1. Formally, a *closed curve* is defined as a smooth function $c : [0,1] \longrightarrow \mathbb{R}^2$, such that $c(0) = c(1)$.

Intuitively, a *closed curve* in the plane is a curve with no endpoints and which completely encloses an area.

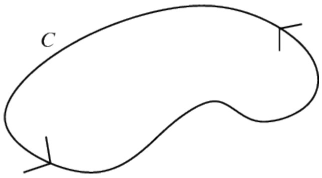

Fig. 3: Example of a closed curve

Definition 2. We define the *Fourier series* of a function $f : [0,1] \longrightarrow \mathbb{C}$ as the infinite sum

$$Sf(x) = \sum_{n=-\infty}^{\infty} \hat{f}_n \mathrm{e}^{2\pi \mathrm{i} n x},$$

where

$$\hat{f}_n = \int_0^1 f(x)\mathrm{e}^{-2\pi \mathrm{i} n x}\mathrm{d}x$$

is the n-th *Fourier coefficient* of f.

Under the right conditions, such as with our closed curves, we will have $Sf(x) = f(x)$, and the *Fourier series* will allow us to decompose our function as a combination of circles, by stacking them on top of each other.

The mathematical definition for Fourier series requires us to perform an infinite sum involving integrals of a function, which are both computationally non-viable. To circumvent this issue a discretization of both concepts is used, which relies on considering a sample.

Definition 3. We define a *sampled curve* as an ordered collection of points $\{X_k\}_{k=0}^{M}$.

In practice, these points will be equidistant points that lie on our curve.

Definition 4. We define the *Fourier series* of a sampled curve $\{X_k\}_{k=0}^{M}$ as the finite sum

$$S_N f(x) = \sum_{n=-N}^{N} \hat{f}_n \mathrm{e}^{2\pi \mathrm{i} n x},$$

where

$$\hat{f}_n = \sum_{k=0}^{M} X_k \mathrm{e}^{-2\pi \mathrm{i} n k/M}$$

are the *Fourier coefficients* of our sample.

Additionally, we define a type of curve that we will use to generate inputs for our training.

Definition 5. We define a (cubic) *Bézier curve* to be a curve parametrized as

$$\begin{aligned} B(t) = {} & (1-t)^3 P_0 + 3(1-t)^2 t P_1 \\ & + 3(1-t)t^2 P_2 + t^3 P_3, \end{aligned}$$

where P_0, P_1, P_2, P_3 are four arbitrary control points, and $0 \leq t \leq 1$.

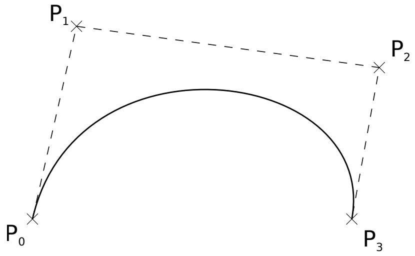

Fig. 4: Example of a Bézier curve

These are the most common type of Bézier curve, and the type that we will use.

5 Planning & methodology

As explained in the previous section, the methodology was based on an iterative process. So, the chosen methodology was Extreme Programming.

Since we obtained function versions at each iteration, we chose this methodology as it advocates for frequent releases in short development cycles, and we believed it would improve productivity.

The communication methodology with the TFG advisor was based on Agile, in which we made weekly meetings to define the progress and new small goals to achieve or to present some intermediate results.

The planning was not strict, but there would be some milestones on selected dates. If the expected project status was not reached in the milestones, it would indicate a delay. The milestones are outlined in table 1.

Milestones	Project status
March 14	Dataset for the neural network of the closed curves
April 18	A neural network to obtain the Fourier coefficients of closed curves
June 13	A neural network capable of calculating Fourier coefficients to surround scene text

TAULA 1: Milestones of the project

The milestones were achieved if before that date we had the project on that status, with results and an analysis from its from which we could conclude a positive conclusion.

The different phases of every iterative objective were generation or obtaining of the base of images, the design of the loss function for the neural network, and the implementation and analysis of the neural network. We estimated that half of the time would be split between the two first phases and the other half for the final phase.

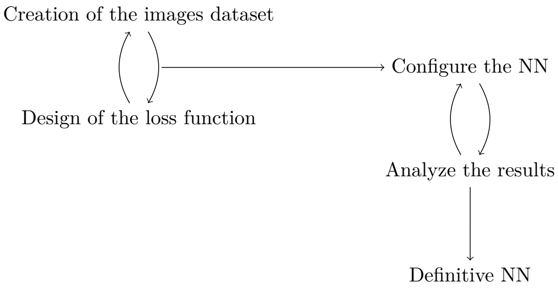

Fig. 5: Graph of the phases of each iterative objective

6 Synthetic Closed Curves

6.1 Dataset

We generate our dataset of closed curves by concatenating Bézier curves in a closed manner.

By choosing the right control points we can get a differentiable closed curve with lots of variation and, through the parametrization given by the formula in definition 5, we can get a sampling of the curve to calculate the discrete Fourier coefficients to use as labels.

To ensure that two Bézier curves B, B' are linked on the end of B and the beginning of B' in a differentiable manner we require [3] that

$$P_0' = P_3, \qquad \text{and} \qquad P_1' = 2P_3 - P_2,$$

where P_0, P_1, P_2, P_3 and P_0', P_1', P_2', P_3' are the control points for B and B', respectively.

We can generate a random sample of points, under the mentioned restrictions, to build a random curve of concatenated Bézier curves.

All the curves are plotted in black-white scale. We have 1000 training images with closed curves and another 400 testing images. All of them are on the same resolution with 1000 × 1000 pixels.

Examples with and without the initial random points are the following:

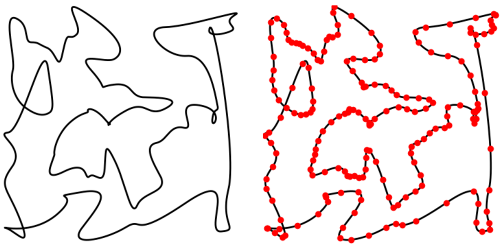

Fig. 6: Example of an image of the dataset.

6.2 Topology of the neural network

For synthetic closed curves, the topology of the neural network has been obtained through various iterations of the model. The model consists of 5 layers of convolutions, with a RELU as the activation function, and finally three linear layers.

6.3 Loss function

The choice of the loss function was between two systems. The first was through the coefficients of the groundtruth and those obtained through the network, evaluate the curve to obtain the points that form it and make MSELoss of the two samplings. The second was to directly calculate the MSELoss of the coefficients directly.

We chose the first version because the second version gave equal importance to all coefficients, and that is not a good idea since coefficients with low degrees determine the curve more.

6.4 Experimental results

In this situation, we have noticed that the quality of the result varies greatly on the value of K. To obtain positive results we found out we require a K twice as big as the number of points.

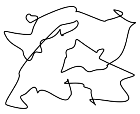

Fig. 7: Original curve of the following example

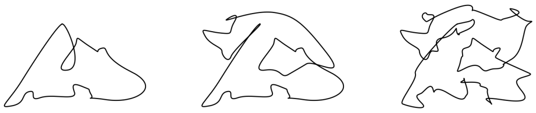

Fig. 8: Example of the Fourier approximation for K=56, 96 and 192.

This is probably due to the high complexity of the curves found in the pictures.

7 Scene Text Detection

7.1 Dataset

The selected dataset is not generated like the one with closed curves. We choose the Total-Text Dataset [2] because is one of the text dataset with multiple orientations and curved words. The dataset contains 1555 images, and there are more than 10000 words in its images.

The dataset includes a plain-text file with word-level polygon annotations.

7.2 Topology of the neural network

The model of the scene text detection is based on [15] mostly. But instead of using a pyramid network, we will use a fully-convolutional network with a ResNet-50, because we get inspired in other detection networks.

So, the model consists on a fully-convolutional network with a ResNet-50, that then branch into a classifier and a regressor.

The classification branch predicts the text classification score map. And the regression branch that predicts the Fourier coefficients of each pixel in the image.

The branches are joined to select only the Fourier coefficients of the text pixels. And then a non-maximum supression is done.

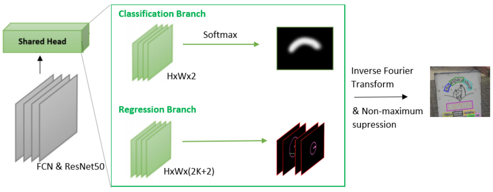

Fig. 9: Model of the Scene Text Detection.

More precisely, the Fourier coefficients with inverse Fourier transformation gives the reconstructed contour of the text. And with the reconstructed contour and the classification scores, we can get the final Fourier coefficients of the detected texts with non-maximum suppression.

7.3 Loss functions

The global loss function is the sum of the classification loss and the regression loss.

The classification loss wants to evaluate the probabilities that the model gives to each class (background or text region). So the classification loss is a cross entropy loss with two classes. In the cross entropy loss the weights are 0.5 for background and 1 for text region.

The regression loss wants to compare the signatures of the Fourier serie of the model and the groundtruth Fourier serie. So we need to evaluate the Fourier coefficients for both series with the inverse Fourier transform and compare them with a L1Loss function.

So the regression loss can be expressed as:

$$\mathbb{L}_{reg} = \frac{1}{N} \sum_{pixel \in I} \sum_{n=1}^{N} w_i L_1(F^{-1}(\frac{n}{N}, c_i), F^{-1}(\frac{n}{N}, \hat{c_i}))$$

where c_i are the Fourier coefficients calculated from the ground-truth and $\hat{c_i}$ are the Fourier coefficients obtained by the model. N is the sampling number of the text contour. I are all image-pixels. And w_i is 0 if the pixel is not in the text region or text centre region, 0.5 in the text region and 1 in the text centre region. And F^{-1} is the inverse Fourier transform. With this regression loss we only give importance to the pixels in the text regions and more importance to the centre ones.

7.4 Experimental results

. In this experiment we noticed a value of $K = 5$ is preferable, as the shapes are much simpler and the neural network training speed is greatly reduced by keeping K a low value.

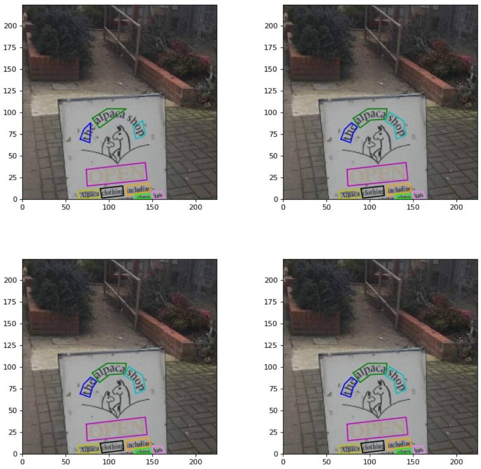

Fig. 10: Example of text detection with K=3, 5, 7 and 9

We also noticed that it is much more common for the neural network to produce false negatives than false positives, since its default behaviour, without training, is to classify all pixels as not containing text.

8 Conclusions

Although the field of scene text detection remains open, we have made significant progress on the problem by designing and implementing a neural network that would detect the text and define an outline using a Fourier series. This has been achieved by using the proposed model consisting of a classifier and a regressor.

Using Fourier series instead of most other common methods, like contour points or masks, also brings some significant benefits to the table *i.e.*, guarantee of a closed and continuous outlines, and small storage and unambiguous footprint. This combined with the fact that the resulting network tends to produce false negatives instead of false positives, leads to the conclusion that this neural network, or other similar to it, are suitable for combination with other networks, as is robust output format can, in most cases, be directly used as input by other models.

As such, a future line of research could be to use this model and add an OCR stage to it. One could also attempt to apply a similar model to other fields such as object detection and recognition, as these are very active and open fields.

References

[1] J. Bruna and J. Cufí. *Anàlisi Complexa.* Manuals UAB, 2008.

[2] Chee Kheng Ch'ng, Chee Seng Chan, and Chenglin Liu. "Total-Text: Towards Orientation Robustness in Scene Text Detection". In: *International Journal on Document Analysis and Recognition (IJDAR)* 23 (2020), pp. 31–52. DOI: 10.1007/s10032-019-00334-z.

[3] Neil Dodgson. *Bezier curves.* Last visited on 24/04/2021. URL: https://www.cl.cam.ac.uk/teaching/2000/AGraphHCI/SMEG/node3.html.

[4] Minghui Liao, Baoguang Shi, and Xiang Bai. "TextBoxes++: A Single-Shot Oriented Scene Text Detector". In: *IEEE Transactions on Image Processing* 27.8 (Aug. 2018), pp. 3676–3690. ISSN: 1941-0042. DOI: 10.1109/tip.2018.2825107. URL: http://dx.doi.org/10.1109/TIP.2018.2825107.

[5] Khalid Sayood. "Chapter 12 - Mathematical Preliminaries for Transforms, Subbands, and Wavelets". In: *Introduction to Data Compression.* Ed. by Khalid Sayood. Fifth Edition. Morgan Kaufmann, 2018. ISBN: 978-0-12-809474-7. DOI: https://doi.org/10.1016/B978-0-12-809474-7.00012-4.

[6] Zhuotao Tian et al. "Learning Shape-Aware Embedding for Scene Text Detection". In: June 2019, pp. 4229–4238. DOI: 10.1109/CVPR.2019.00436.

[7] Fangfang Wang et al. *TextRay: Contour-based Geometric Modeling for Arbitrary-shaped Scene Text Detection.* 2020. arXiv: 2008.04851 [cs.CV].

[8] Pengfei Wang et al. "A Single-Shot Arbitrarily-Shaped Text Detector Based on Context Attended Multi-Task Learning". In: *Proceedings of the 27th ACM International Conference on Multimedia.* MM '19. Nice, France: Association for Computing Machinery, 2019, pp. 1277–1285. ISBN: 9781450368896. DOI: 10.1145/3343031.3350988. URL: https://doi.org/10.1145/3343031.3350988.

[9] Eric W. Weisstein. *Closed Curve. From MathWorld—A Wolfram Web Resource.* Last visited on 13/03/2021. URL: https://mathworld.wolfram.com/ClosedCurve.html.

[10] Enze Xie et al. *Scene Text Detection with Supervised Pyramid Context Network.* 2018. arXiv: 1811.08605 [cs.CV].

[11] Yongchao Xu et al. "TextField: Learning a Deep Direction Field for Irregular Scene Text Detection". In: *IEEE Transactions on Image Processing* 28.11 (Nov. 2019), pp. 5566–5579. ISSN: 1941-0042. DOI: 10.1109/tip.2019.2900589. URL: http://dx.doi.org/10.1109/TIP.2019.2900589.

[12] Chuhui Xue, Shijian Lu, and Wei Zhang. *MSR: Multi-Scale Shape Regression for Scene Text Detection.* 2019. arXiv: 1901.02596 [cs.CV].

[13] Chengquan Zhang et al. *Look More Than Once: An Accurate Detector for Text of Arbitrary Shapes.* 2019. arXiv: 1904.06535 [cs.CV].

[14] Shi-Xue Zhang et al. *Deep Relational Reasoning Graph Network for Arbitrary Shape Text Detection.* 2020. arXiv: 2003.07493 [cs.CV].

[15] Yiqin Zhu et al. "Fourier Contour Embedding for Arbitrary-Shaped Text Detection". In: *CoRR* abs/2104.10442 (2021). arXiv: 2104.10442. URL: https://arxiv.org/abs/2104.10442.