Research Article

Petr Babkin and Nikita Shubin

Position Informed Convolution for Multi-Agent Curve Detection

https://doi.org/10.1515/sample-YYYY-XXXX
Received Month DD, YYYY; revised Month DD, YYYY; accepted Month DD, YYYY

Abstract: This paper introduces a novel approach for robust curve detection in images, a task often hindered by the high degree of parametric freedom in curve representation. Our method utilizes a multi-agent system, wherein individual agents employ the Hough transform to identify potential curve segments. To effectively consolidate these partial detections and account for their respective spatial contexts, we propose the Positional Informed Convolution (PIC) layer. This novel layer extends traditional convolutional operations by explicitly encoding the spatial location of input feature maps, thus enabling a more sophisticated and contextually aware aggregation of agents' outputs. The effectiveness of our proposed approach is validated through experiments on a custom-built synthetic dataset, where we demonstrate significant improvements in curve detection accuracy and robustness compared to conventional methods.

Keywords: curve detection, Hough transform, convolution, likelihood

1 Introduction

Curve detection in images is a fundamental problem in computer vision, with applications spanning various domains, including medical image analysis, autonomous driving, and robotics [1, 2]. While human vision excels at rapidly identifying and interpreting curves, the automated extraction of such structures from images remains a challenging task. This difficulty arises from the inherent variability in the shape of the curves, which requires complex parameterizations, making traditional detection methods often inadequate [3–5].

The Hough transform (HT) [3] is a classical method for detecting lines and curves, and has been widely applied for decades [3, 4]. It functions by mapping image features to a parameter space, where the peaks indicate the presence of curves. Various extensions and optimizations of HT have been proposed to improve its performance and robustness, including randomized versions [5] and deep learning-based approaches [6]. However, these methods are limited in cases where the curves are highly complex and fragmented. Furthermore, early patterns recognition work has already proposed solutions for complex structures using related algorithms [7].

Line detection, a specific instance of curve detection, has received significant attention, particularly in the context of autonomous driving and infrastructure monitoring [8–10]. Recent approaches include end-to-end deep learning solutions based on convolutional neural networks (CNNs) [10] and improvements to classic HT algorithms [9]. While such methods achieve impressive results on certain benchmark datasets, they often struggle with complex and noisy environments.

Multi-agent systems (MAS) have emerged as a promising paradigm for solving complex tasks by distributing work among multiple autonomous agents, offering increased robustness and flexibility [11]. Such systems have been explored in many computer vision problems, including object detection and recognition

Petr Babkin, Moscow Institute of Physics and Technology, Intelligent Systems, Moscow, Russia, e-mail: babkin.pk@phystech.edu
Nikita Shubin, Moscow Institute of Physics and Technology, Phystech School of Applied Mathematics and Computer Science, Ryazan, Russia

[12]. Combining the power of MAS with robust curve detection algorithms, as in [13], presents an opportunity to overcome the limitations of single agent systems.

In this work, we propose a novel approach for curve detection based on a multiagent framework where individual agents are equipped with the Hough transform. Our key contribution lies in the introduction of the "Positional Informed Convolution" (PIC) layer, designed to effectively aggregate and refine curve proposals by explicitly considering their spatial origin, providing a solution to limitations in the existing methodologies. Through rigorous experimentation using a synthetic dataset, we demonstrate the effectiveness and efficiency of our approach, outperforming state-of-the-art techniques.

Contributions

- **Multi-Agent Curve Detection Framework for Reduced Parametric Complexity:** We introduce a novel multi-agent system for curve detection, where individual agents operate on local regions of the input image. This framework effectively decomposes the complex curve detection problem into a series of simpler sub-problems, allowing each agent to focus on detecting potential curve segments in a smaller, more manageable space. By distributing the search process among multiple agents, the approach reduces the dimensionality of the parameter space for each agent, thereby simplifying the overall detection task and making it more robust and scalable.
- **Positional Informed Convolution (PIC) Layer for Spatially-Aware Aggregation:** We propose a novel convolutional layer, termed the "Positional Informed Convolution" (PIC) layer, for aggregating the outputs of the individual agents. Unlike standard convolutional layers, the PIC layer explicitly takes into account the spatial origin of each agent's output, allowing for a more nuanced and contextually-aware aggregation of the detected curve segments.

2 Main Part

In this section, we detail the core methodology of our proposed approach. We begin by describing the deep Hough transform, which forms the basis of our feature extraction process. Subsequently, we describe the aggregation technique employed to combine the outputs of multiple agents, providing specific details about our proposed aggregation layer. Finally, we present the loss function utilized to train our model, highlighting its key characteristics and justification within the framework of our methodology.

Notation We use the following notation throughout our paper: x denotes scalars, $\mathbf{x}$ denotes vectors (bold text), and x_i denotes the i-th element of $\mathbf{x}$. π stands for permutation function and $\varepsilon(i)$ is a neighborhood of pixel i.

2.1 Deep Hough Transform

The Deep Hough Transform (DHT) is a method for transforming deep convolutional neural network (CNN) features into a parametric space. Given input features

$$\mathbf{X} \in \mathbb{R}^{C \times H \times W}$$

extracted by an encoder network, where C represents the number of channels and H and W are the spatial dimensions, DHT maps these features into a transformed space

$$\mathbf{Y} \in \mathbb{R}^{C \times \Theta \times R}$$

. The dimensions Θ and R are determined by the quantization intervals for the line parameters. For each line l in the image, characterized by parameters (θ_l, r_l), DHT aggregates the features of all pixels along that line and maps them to the corresponding position $(\hat{\theta}_l, \hat{r}_l)$ in the parametric space $\mathbf{Y}$, calculated according to Equation (4). The line parameters (θ_l, r_l) are quantized into discrete grid values $(\hat{\theta}_l, \hat{r}_l)$ using quantization levels Θ and R, effectively creating $\Theta \cdot R$ unique line candidates. Notably, DHT is order-agnostic in both the feature and parametric spaces, enabling highly parallelizable computation. This process aggregates information from features along candidate lines to a corresponding location in the parametric space, effectively converting pixel-level features into line-level representations.

2.2 Positional Exchange of Information

After extracting information from individual agents, aggregating their responses becomes crucial. This can be framed as a maximum likelihood estimation problem, aiming to find the optimal collective output $\mathbf{y}$ given the agent-specific inputs $\mathbf{x}$. Formally, we seek to maximize the joint likelihood, expressed as:

$$\Psi(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} \psi_i(\mathbf{y}_i|\mathbf{x}_i, \mathbf{x}_j, j \in \varepsilon(i)) \to \max_{\mathbf{y}}, \tag{1}$$

where $\varepsilon(i)$ denotes the set of neighboring agents of agent i, $\mathbf{x}_i$ represents the output of agent i, and $\mathbf{x}$ is the concatenation of all agent outputs. This formulation implies that the optimal collective output $\mathbf{y}$ is the one that maximizes the probability of observing the individual outputs $\mathbf{y}_i$ given the agent-specific features.

To simplify the problem, we assume that the conditional likelihood of an agent's output, given its input and its neighbors' inputs, is position-independent, expressed as:

$$\forall i \quad \psi_i(\mathbf{y}_i|\mathbf{x}_i, \mathbf{x}_j, j \in \varepsilon(i)) = \psi(\mathbf{y}_i|\mathbf{x}_i, \mathbf{x}_j, j \in \varepsilon(i)). \tag{2}$$

This assumption allows us to learn a single function that aggregates agent outputs, irrespective of the agent's location within the network. Our goal is to find a parameterized function f_θ that approximates this conditional likelihood:

$$\forall i \quad \arg\max_{\mathbf{y}} \psi(\mathbf{y}|\mathbf{x_i}, \mathbf{x}_j, j \in \varepsilon(i)) = f_\theta(\mathbf{x}_i, \mathbf{x}_j, j \in \varepsilon(i)). \tag{3}$$

This equation signifies that f_θ should predict the optimal output $\mathbf{y}$ based on the agent's input and the inputs of its neighbors. To train this parameterized function, we transform the maximization problem into a minimization problem by employing the negative log-likelihood, leading to the following loss function:

$$\mathbb{E}_{\mathbf{x},\mathbf{y} \in \mathcal{D}}[-\log \Psi(\mathbf{y}|\mathbf{x})] = \mathbb{E}_{\mathbf{x},\mathbf{y} \in \mathcal{D}} \left[\sum_i \mathcal{L}\left(f_\theta(\mathbf{x}_i, \mathbf{x}_j, j \in \varepsilon(i)), \mathbf{y}_i\right) \right] \to \min_\theta, \tag{4}$$

where $\mathcal{D}$ is the dataset of agent inputs and optimal outputs.

A critical aspect lies in choosing the appropriate form for the aggregation function f_θ. A naive approach using a standard convolution operation as a first step leads to an order-agnostic behavior, where the spatial relationship between neighboring agents is lost:

$$\mathrm{Conv}_\theta(\mathbf{x}_1, \ldots, \mathbf{x}_{|\varepsilon(i)|+1}) = \underbrace{\varphi_{\theta_1}(x_1) + \ldots + \varphi_{\theta_{|\varepsilon(i)|+1}}(x_{|\varepsilon(i)|+1})}_{\text{Order agnostic}}. \tag{5}$$

This formulation, where messages received from agents are simply summed, discards valuable information about the position of the agent transmitting the message. This is a significant limitation for our task, as it is crucial to discern from which side a line signal is being received. Therefore, we need a more sophisticated aggregation method to preserve spatial information.

2.3 Position Informed Convolution

As discussed in relation to Equation (5), a simple convolutional layer does not account for the spatial location of the contributing agents, making it order-agnostic and therefore unsuitable for our problem.

To address this limitation, we propose a function that retains the positional information of each agent's message. Specifically, instead of summing the contributions as in Equation (5), we aim for the following form:

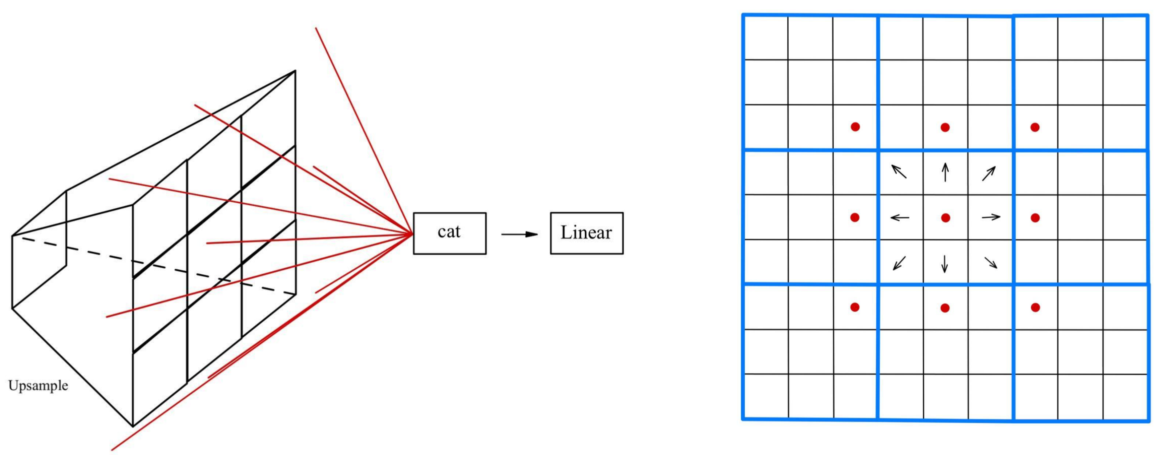
$$f_\theta(\mathbf{x}_1, \ldots, \mathbf{x}_{|\varepsilon(i)|+1}) = [\varphi_{\theta_1}(x_1), \ldots, \varphi_{\theta_{|\varepsilon(i)|+1}}(x_{|\varepsilon(i)|+1})], \tag{6}$$

where each φ_{θ_k} is a feature transformation specific to the k-th agent. Critically, this formulation should be sensitive to permutations, that is,

$$f_\theta(\mathbf{x}_1, \ldots, \mathbf{x}_{|\varepsilon(i)|+1}) \neq f_\theta(\mathbf{x}_{\pi(1)}, \ldots, \mathbf{x}_{\pi(|\varepsilon(i)|+1)}), \tag{7}$$

for any permutation π. This implies that the order of input received from neighboring agents is critical and should influence the aggregated result.

To achieve this, we propose a Position-Informed Convolution layer (PIC). The PIC layer explicitly aggregates information based on the position from which the message originated, thereby capturing crucial spatial dependencies that are essential for our task. This design ensures that we do not lose valuable directional information when aggregating the agent responses, allowing for a more robust representation.



(a) A side-view illustration of the PIC layer is depicted. Initially, the input undergoes upsampling. Subsequently, messages from the central node and its neighbors are aggregated through concatenation, followed by a linear transformation. This design allows the layer to differentiate between messages based on their source and to apply varying transformations accordingly.

(b) A view of the layer's slice following upsampling operation. Blue squares delineate the boundaries of the outputs from individual agents. Red dots indicate the messages selected by the central agent. Arrows represent the direction of information flow from the central agent within the depicted cell.

Fig. 1: This figure illustrates the Positional Informed Convolution (PIC) layer. Picture 1a depicts the complete operational flow of the layer, while picture 1b presents a cross-sectional view, highlighting the information exchange between agents.

It should be emphasized that the proposed layer facilitates control over the neighborhood structure, denoted as $\varepsilon(\cdot)$. Through the precise selection of kernel size, stride, and dilation, the layer can be deployed for operation within an expansive variety of neighborhood definitions.

The aforementioned layer facilitates the development of a network training algorithm. The complete pipeline of this algorithm is presented in Figure 2.

The processing pipeline begins by dividing the input image into non-overlapping patches, with each patch assigned to an independent agent. Each patch then goes through a series of operations involving a convolutional neural network, a deep Hough transform, and a context-aware line detector, to extract a set of features. The resulting features are subsequently aggregated using our novel Positional Informed Convolution (PIC) layer.

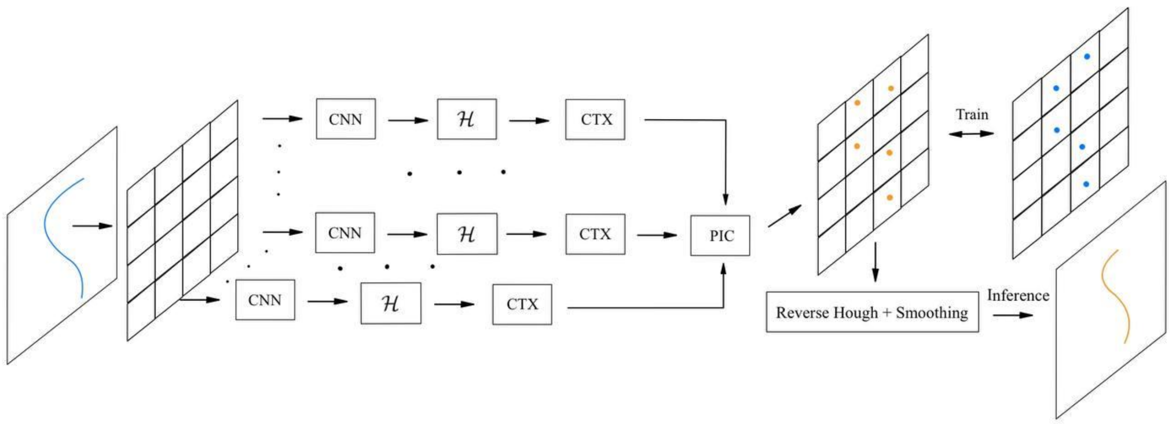


Fig. 2: Illustration of

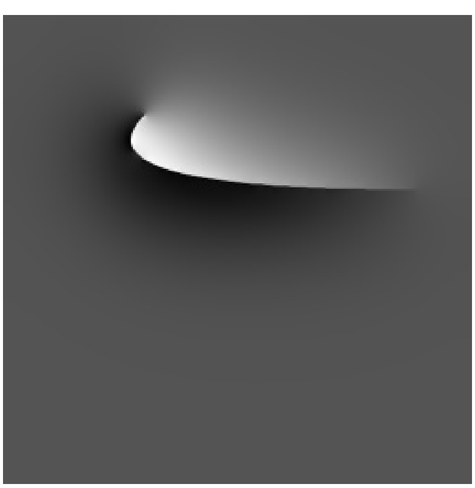
This procedure culminates in final network predictions, which are employed distinctively for training and inference tasks. For training, the predictions are assessed against the provided ground truth labels. For inference, the collection of line segments obtained from the agents is subjected to spline-based smoothing.

2.4 Losses

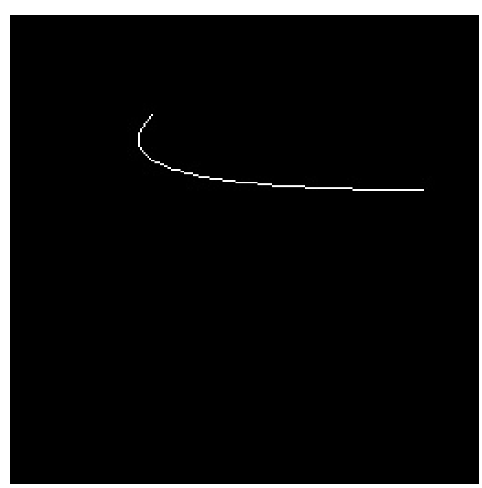
3 Experiments

3.1 Dataset

Due to the unavailability of a publicly accessible dataset with the required curved line annotations, we have generated a custom dataset to address this limitation. A sample of the generated data can be observed in Figure 3.



(a) Sample from our dataset.



(b) Label for a sample 3a

Fig. 3: Sample from newly generated dataset for curve detection on images.

4 Conclusion

References

[1] Yun Liu, Ming-Ming Cheng, Xiaowei Zhang, Jiashi Tan, Jianming Huang, and Philip HS Torr. Richer convolutional features for edge detection. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[2] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*, 2015.

[3] Richard O Duda and Peter E Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972.

[4] Dana H Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122, 1981.

[5] Lei Xu and Erkki Oja. A new curve detection method: randomized hough transform (rht). *Pattern recognition*, 26(11): 1713–1725, 1990.

[6] Yuhui Zhang, Zhaopeng Liu, Jia Xiao, Chao Zheng, Zhe Yuan, and Xin Wei. Deep hough transform for semantic line detection. *Neurocomputing*, 450:320–330, 2021.

[7] Azriel Rosenfeld. Method and means for recognizing complex patterns. *US Patent 3,069,654*, 1962.

[8] Mohamed Khamis, Mohamed Abdelhady, Mohamed Abdelmaguid, Mohamed Elhoseny, and Ibrahim Mohamed. Axial-unet++ power line detection network based on gated axial attention mechanism. *Remote Sensing*, 16(23):4585, 2024.

[9] Lihong Guo, Gang Xu, Minqiang Li, Yingmin He, and Nan Zheng. Real-time line detection through an improved hough transform voting scheme. *Pattern Recognition*, 41(6):1859–1871, 2008.

[10] Kai Zhang, Rui Wu, Jianwei Wang, Wei Liu, Yefeng Zhang, and Xiaojun Chen. Line-cnn: End-to-end traffic line detection with line proposal unit. *IEEE Transactions on Intelligent Transportation Systems*, 21(9):3632–3644, 2020.

[11] Michael Wooldridge. Multi-agent systems: A survey. *Advances in artificial intelligence*, 36(1):21, 2018.

[12] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[13] Kirill Samoilov, Alexey Zhuravlev, and Alexey Kuptsov. Multi-agent system for line detection on images. *Proceedings of the International Conference of Russian Young Researchers in Electrical and Electronic Engineering*, pages 542–544, 2016.

A Missing Proofs

B Wide Experiments