深 | 圳 | 理 | 工 | 大 | 学
中国科学院深圳先进技术研究院
SHENZHEN INSTITUTE OF ADVANCED TECHNOLOGY
CHINESE ACADEMY OF SCIENCES

JUNE 18-22, 2023
CVPR
VANCOUVER, CANADA

THU-PM-175

Neural Transformation Fields for Arbitrary-Styled Font Generation

Bin Fu[1], Junjun He[2], Jianjun Wang[1], Yu Qiao[1,2]

[1]ShenZhen Key Lab of Computer Vision and Pattern Recognition,
Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences
[2]Shanghai Artificial Intelligence Laboratory

Codes

If you have any questions,
please contact: bin.fu@siat.ac.cn

Introduction and motivation

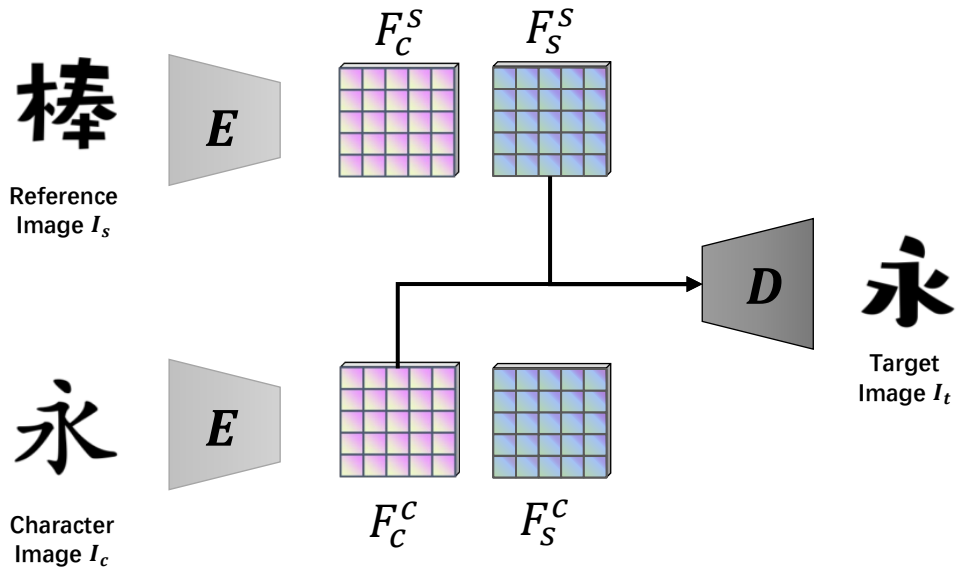

Fig.1 Style-content disentanglement approach.

Style-content disentanglement approach: combine the content representations of source characters and the style codes of reference samples to generate font images.

Recent methods explore powerful style representations, which ignore the spatial transformations in transferring font styles.

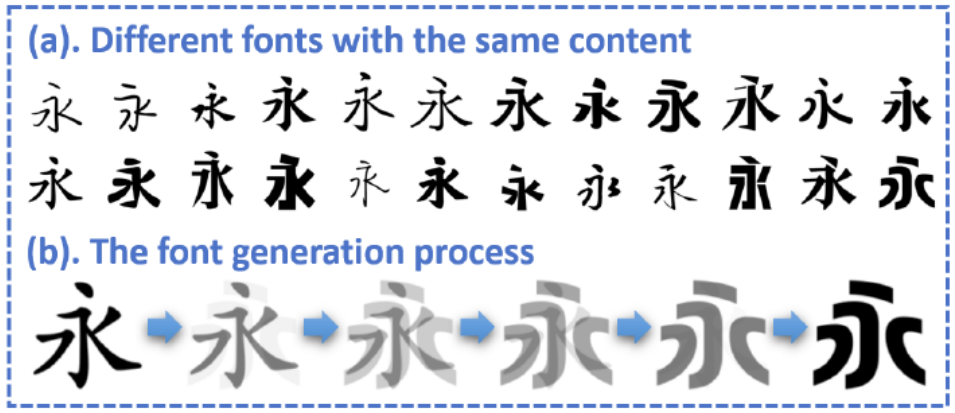

Fig.2 The motivation of our NTF.

New Solution

We model font generation as a continuous transformation process from the source character image to the target font image via the creation and dissipation of font pixels.

New Model

Inspired by advances in Neural Radiance Field (NeRF), we embed the corresponding transformations into a Neural Transformation Field (NTF).

Representing Font Generation as the Transformation Process of Font Pixels

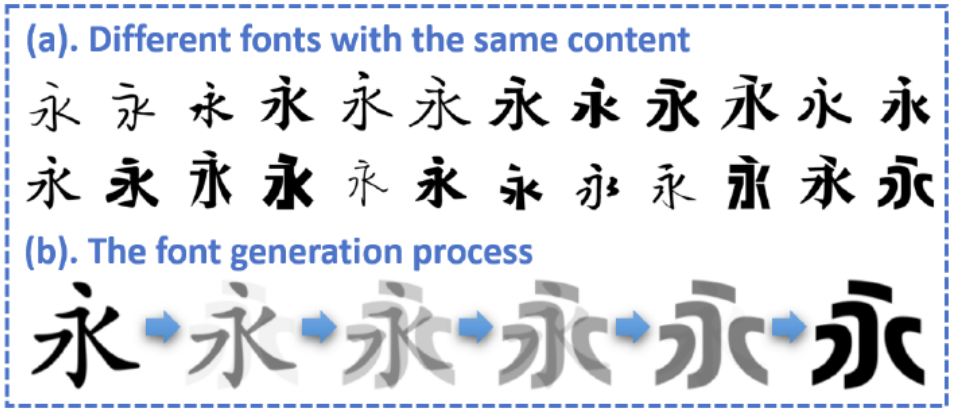

Fig.2 The motivation of our NTF.

New Solution

The transformations of different font styles can be divided into two categories, the font pixel generation process and dissipation process.

- creation intensity φ: represents the non-font pixel changing to font pixel or the font pixel enhancing its intensity.
- dissipation rate τ: represents a font pixel weakening its intensity with the rate τ.

New Model

We embed the corresponding transformations into a Neural Transformation Field (NTF), namely $(\varphi, \tau) = F_\Theta(\theta|F_c)$.

θ: font style; θ_0: standard font

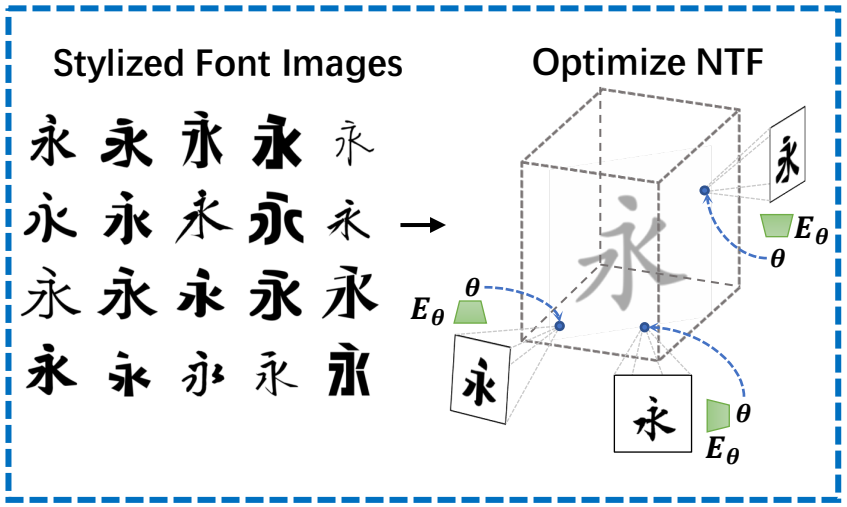

Fig.3 The methodology of our NTF.

Locations in NTF: each font style is represented by a specific location θ

Style estimator E_θ: predicts the locations θ of each font style

Font generation: a transformation process of font pixels from the original point to the target location

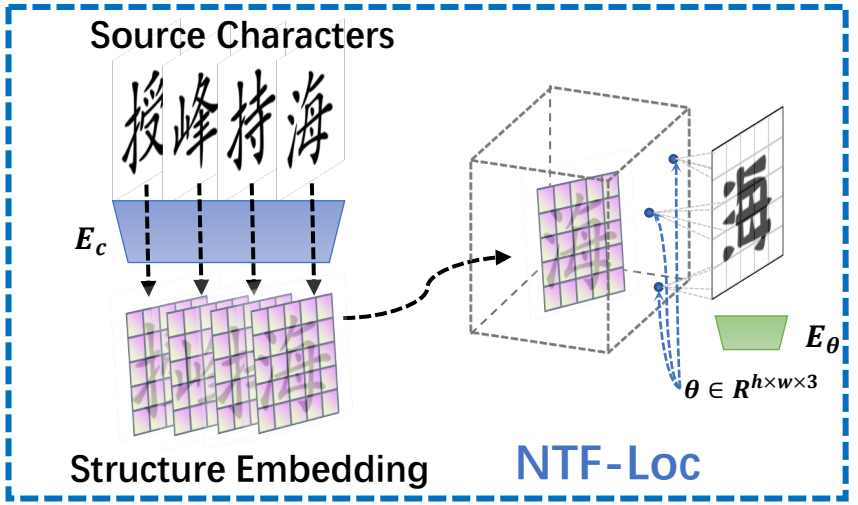

Fig.4 Generalize our NTF to multi-char and localized style representation cases.

Representing Font Generation as the Transformation Process of Font Pixels

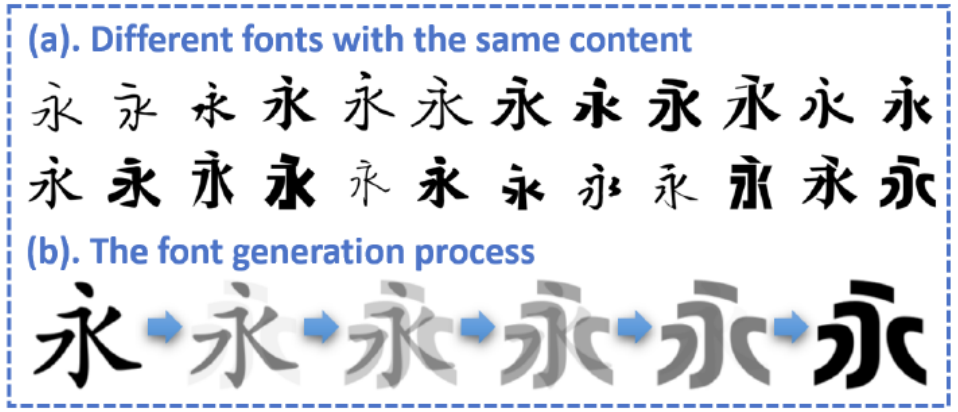

Fig.2 The motivation of our NTF.

New Solution

The transformations of different font styles can be divided into two categories, the font pixel generation process and dissipation process.

- creation intensity φ: represents the non-font pixel changing to font pixel or the font pixel enhancing its intensity.
- dissipation rate τ: represents a font pixel weakening its intensity with the rate τ.

New Model

We embed the corresponding transformations into a Neural Transformation Field (NTF), namely $(\varphi, \tau) = F_\Theta(\theta|F_c)$.

θ: font style; θ_0: standard font

Once we obtain the creation intensity and dissipation rate along the transformation path, we can collect these intermediate transformation results and formulate the font generation as a font rendering process in Neural Transformation Field.

Locations in NTF: each font style is represented by a specific location θ

Style estimator E_θ: predicts the locations θ of each font style

Font generation: a transformation process of font pixels from the original point to the target location

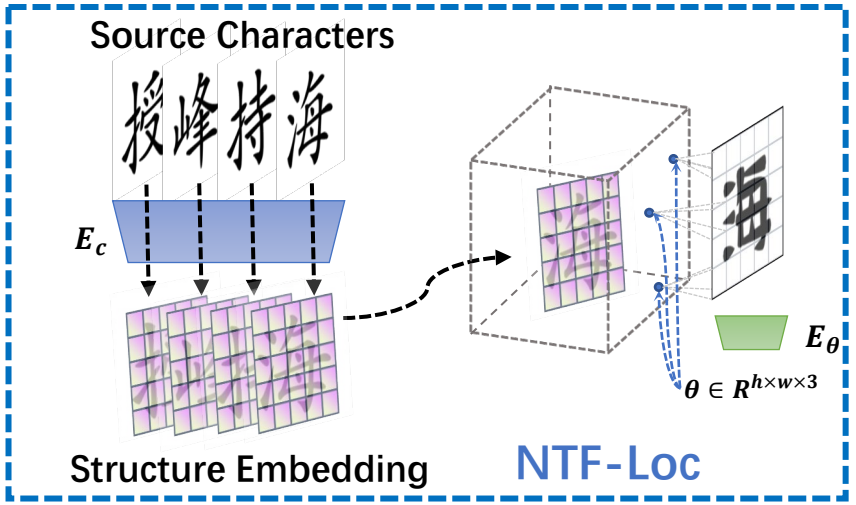

Fig.4 Generalize our NTF to multi-char and localized style representation cases.

Font Rendering with Neural Transformation Field

- The transformed intensity at location ω can be expressed as:

$$\frac{dI(\omega)}{d\omega} = \varphi(\omega)\tau(\omega) - \tau(\omega)I(\omega) \quad \text{(Eq.1)}$$

The first term models the creation process while the second term models the dissipation process of font pixels.

- The solution to this differential equation are:

$$I(\theta) = \int_0^{\theta} \varphi(\omega)\tau(\omega)T(\omega)d\omega \quad T(\omega) = \exp\left(-\int_{\omega}^{\theta} \tau(x)dx\right) \quad \text{(Eq.2)}$$

This equation is the font rendering formula in NTF, and $T(\omega)$ represents the accumulated transformations along the path from the location ω to θ.

- In practice, we estimate this continuous integral numerically.

The interval from the original point to location θ is partitioned into N evenly-spaced segments with the length $\xi = \frac{1}{N}\theta$, and we draw one sample in each segment i at the location $\theta_i = i\xi$.

$$I = \sum_{i=1}^{N} T_i \left(1 - \exp\left(-\tau_i \xi\right)\right) \varphi_i \qquad T_i = \exp\left(-\sum_{j=i+1}^{N} \tau_i \xi\right) \quad \text{(Eq.3)}$$

Algorithm 1: Font Rendering Process for Localized Style Representation

Data: the estimated location θ, the structure embedding F_c, the number of sampling points N
Result: the target font image I_t

$\xi \leftarrow \frac{1}{N}\theta$;
for $i \leftarrow N$ **to** 1 **do**
  $\theta_i \leftarrow i\xi$;
  $(\varphi_i, \tau_i) \leftarrow NTF(F_c \copyright \theta_i)$;
  if $i = N$ **then**
    $T_i \leftarrow 1$;
    $\tilde{T}_i \leftarrow \exp\left(-\tau_i \xi\right)$;
    $I_t \leftarrow T_i \left(1 - \tilde{T}_i\right) \varphi_i$;
  else
    $T_i \leftarrow T_{i+1}\tilde{T}_{i+1}$;
    $\tilde{T}_i \leftarrow \exp\left(-\tau_i \xi\right)$;
    $I_t \leftarrow I_t + T_i \left(1 - \tilde{T}_i\right) \varphi_i$;
  end
end

Fig.5 The visualization results of the intermediate rendering process.

Font Rendering with Neural Transformation Field

- The transformed intensity at location ω can be expressed as:

$$\frac{dI(\omega)}{d\omega} = \varphi(\omega)\tau(\omega) - \tau(\omega)I(\omega) \quad \text{(Eq.1)}$$

The first term models the creation process while the second term models the dissipation process of font pixels.

- The solution to this differential equation are:

$$I(\theta) = \int_0^{\theta} \varphi(\omega)\tau(\omega)T(\omega)d\omega \quad T(\omega) = \exp\left(-\int_{\omega}^{\theta} \tau(x)dx\right) \quad \text{(Eq.2)}$$

This equation is the font rendering formula in NTF, and $T(\omega)$ represents the accumulated transformations along the path from the location ω to θ.

- In practice, we estimate this continuous integral numerically.

The interval from the original point to location θ is partitioned into N evenly-spaced segments with the length $\xi = \frac{1}{N}\theta$, and we draw one sample in each segment i at the location $\theta_i = i\xi$.

$$I = \sum_{i=1}^{N} T_i \left(1 - \exp\left(-\tau_i \xi\right)\right) \varphi_i \qquad T_i = \exp\left(-\sum_{j=i+1}^{N} \tau_i \xi\right) \quad \text{(Eq.3)}$$

The generated font image gradually transforms from the source style to the target style:

- the unrelated font pixels in source images are dissipated
- other pixels of target font are gradually created

For the creation process:

- the skeleton of the font is first generated,
- then the fine-grained details are added, and the structures of characters become smooth and complete

Fig.5 The visualization results of the intermediate rendering process.

Overall Framework

Our font generation method contains four parts, including a style estimator, a structure encoder, a neural transformation field (NTF), and a discriminator.

1. Given a style image as the reference image and a source image, the style estimator estimates the 3D location from the reference image while the structure encoder network embeds the structure information from the source character.
2. Then we generate N sampling points along the estimated path, and obtain the creation intensity and dissipation rate at each sampling point from NTF.
3. Finally, we collect the intermediate results and generate font images by font rendering formula.

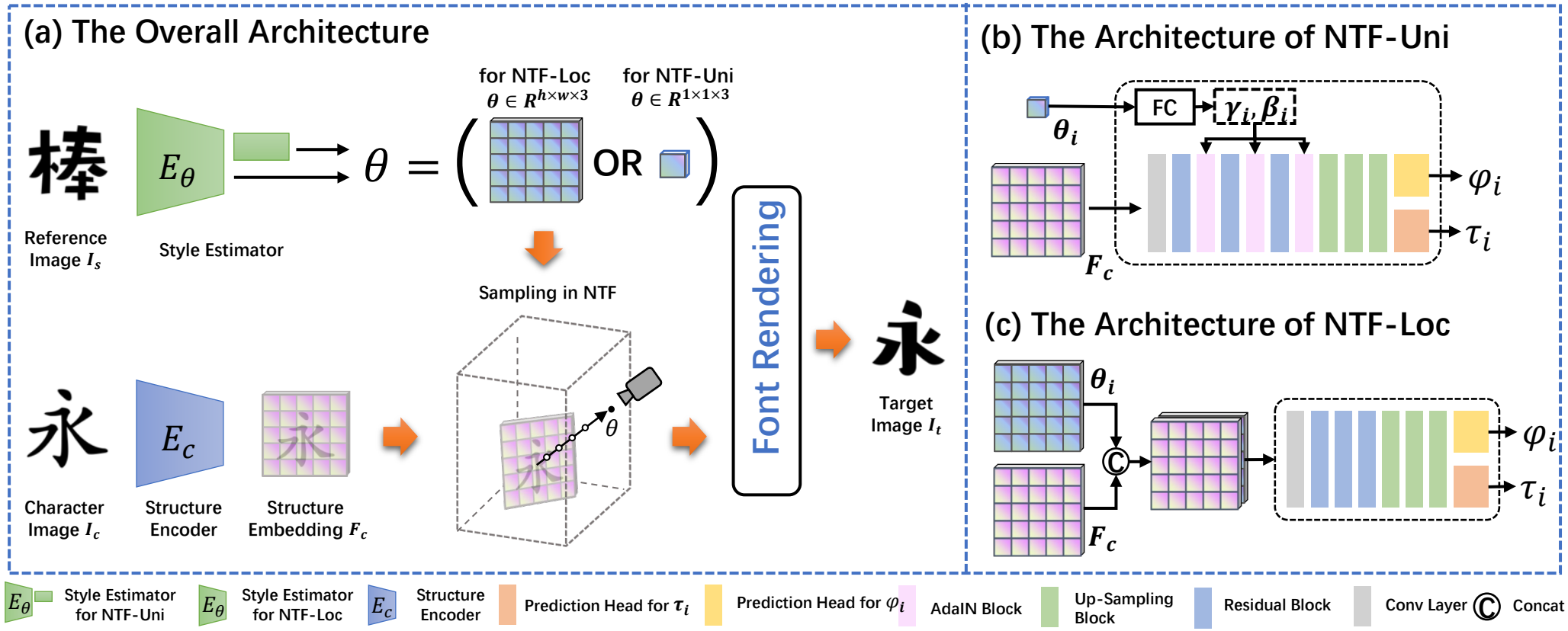

Fig.6 The overall network of our font generation method.

Qualitative and Quantitative Comparisons

Fig.7 Qualitative comparisons of our NTF with other state-of-the-art methods.

Tab. 1 Quantitative comparison on few-shot font generation task.

Methods	SSIM↑	ms-SSIM↑	LPIPS↓	FID↓
Unseen Fonts and **Unseen** Contents				
FUNIT [19]	0.6174	0.2651	0.1505	19.29
DG-font [36]	0.6433	0.3924	0.1293	70.90
LF-font [26]	0.6466	0.3052	0.1277	41.21
FSFont [32]	0.6463	0.4051	0.1188	66.49
MX-font [27]	0.6368	0.3676	0.1075	29.34
NTF-Uni (Ours)	0.6331	0.3468	0.1283	33.12
NTF-Loc (Ours)	**0.6533**	**0.4187**	**0.1019**	**15.67**

Tab. 2 Quantitative comparison in inference running time, number of parameters, and computation complexity.

Methods	FPS	N_p	MACs
FUNIT [3]	175	29.77M	21.01G
DG-font [6]	128	16.25M	23.99G
LF-font [4]	107	7.92M	24.79G
MX-font [5]	45	22.76M	51.12G
STF-Loc (N=5)	112	9.07M	24.78G
STF-Loc (N=15)	91	9.07M	43.64G

Contribution and Limitations

Contributions:

- We regard font generation as a continuous transformation process by the creation and dissipation of font pixels along the transformation path, and embed such transformations into the neural transformation field (NTF).
- A differentiable font rendering procedure is developed to accumulate the intermediate transformations into the target font image.
- Experimental results show that our method outperforms the state-of-the-art methods in the few-shot font generation task, which demonstrate the effectiveness of our proposed method.

Limitations:

- The computation complexity will increase and the inference running time will decrease, when we utilize more sampling numbers to perform font rendering.
- Since only few reference samples are provided and does not cover all strokes, few-shot font generation task is still a difficult task in computer vision community. Some local details of generated images are imperfect.

Codes

If you have any questions, please contact: bin.fu@siat.ac.cn