SIGGRAPH ASIA 2018 TOKYO
CONFERENCE 4 – 7 December 2018
EXHIBITION 5 – 7 December 2018
Tokyo International Forum, Japan
SA2018.SIGGRAPH.ORG

Sponsored by

As-Compact-As-Possible Vectorization for Character Images

Zeqing Xia, Zhouhui Lian, Yingmin Tang, Jianguo Xiao

Institute of Computer Science and Technology
Peking University

北京大学
PEKING UNIVERSITY

OUTLINE

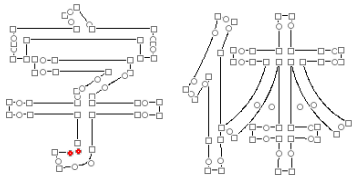
- Introduction
- Method Description
- Experimental Results
- Conclusion and Future Work

Introduction

BACKGROUND

- Raster Font → Vector Font
  - Type 1 & Type 3
  - Truetype
  - Opentype
  - ……

字体



BACKGROUND

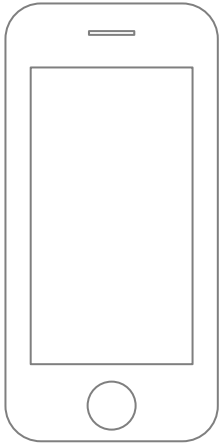
- English Fonts

!"#$%&'()*+,-./0123456
789:;<=>?@ABCDEFGHI
JKLMNOPQRSTUVWXYZ
[\]^_`abcdefghijklmnop
qrstuvwxyz{|}~

- Chinese Fonts

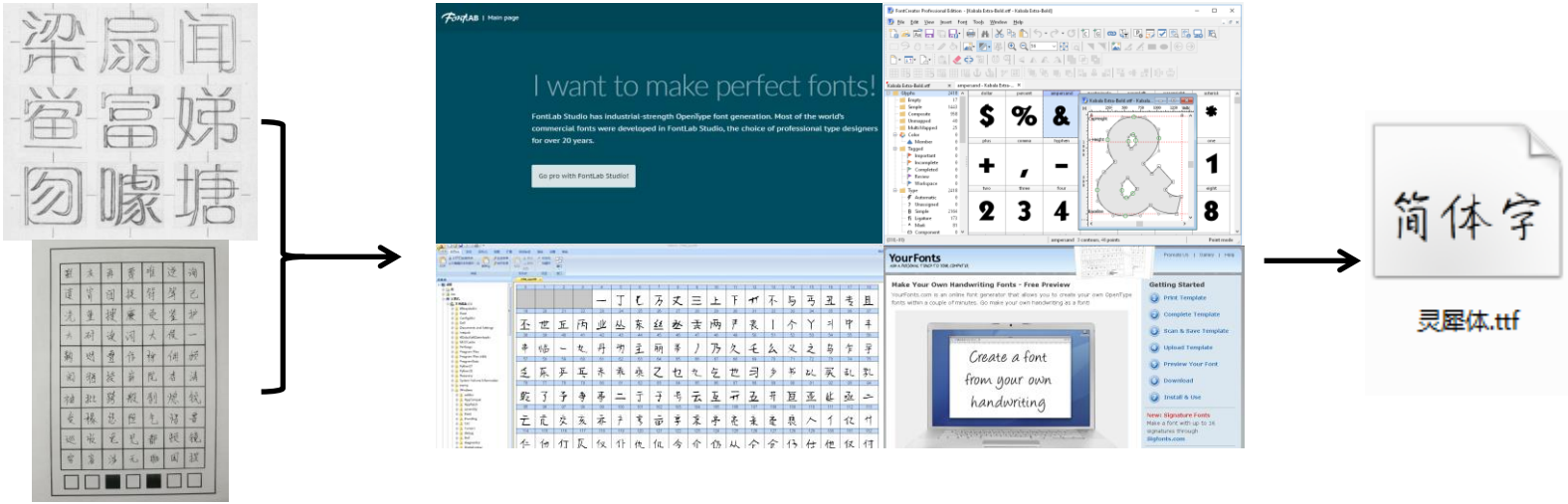
MOTIVATION

- Mobile Devices
  - Screen Size
  - Storage



MOTIVATION

- Human-based font design/vectorization

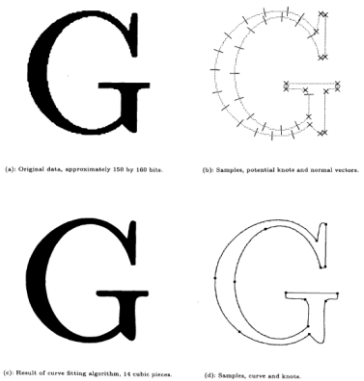


POTENTIAL SOLUTIONS

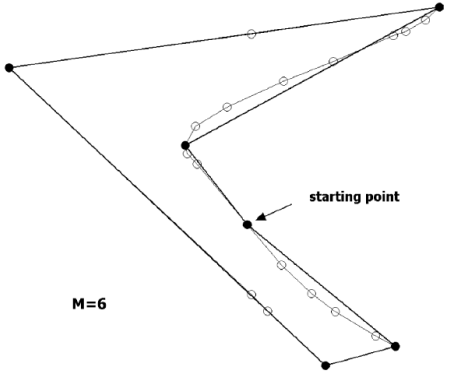
- Preserve significant detail
  - Use neural network
- Vectorization
  - With limited type of line segments

POTENTIAL SOLUTIONS

- Vectorization and simplification for curves



[Plass et al. 1983]

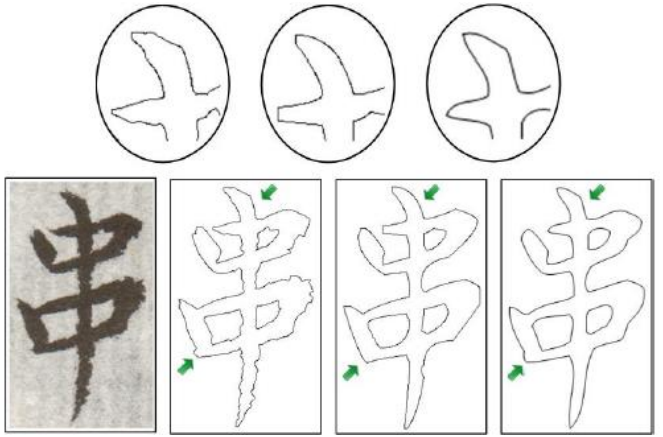


[Kolesnikov et al. 2007]

POTENTIAL SOLUTIONS

- Vectorization for Chinese character & calligraphy

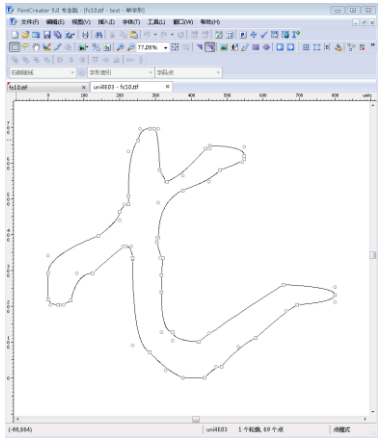
[Zhang et al. 2011]



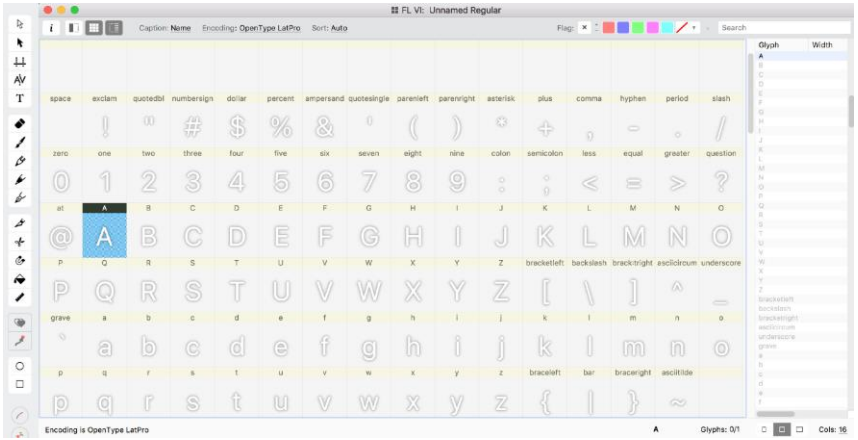
[Pan et al. 2014]

POTENTIAL SOLUTIONS

- Vectorization for Chinese character & calligraphy



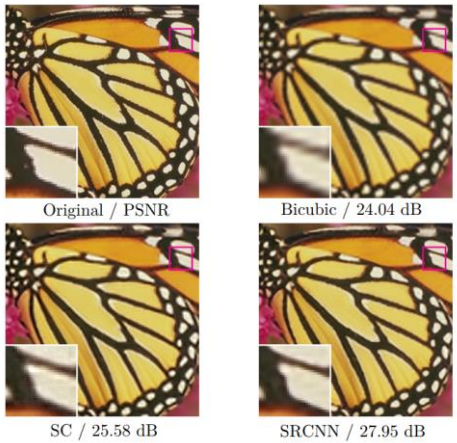
Font Creator



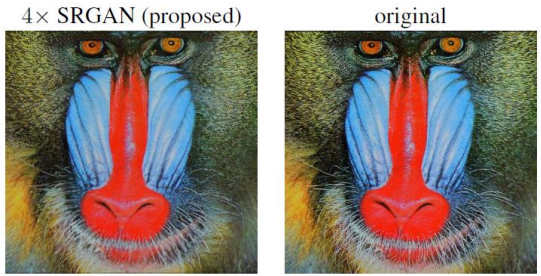
Font Lab

POTENTIAL SOLUTIONS

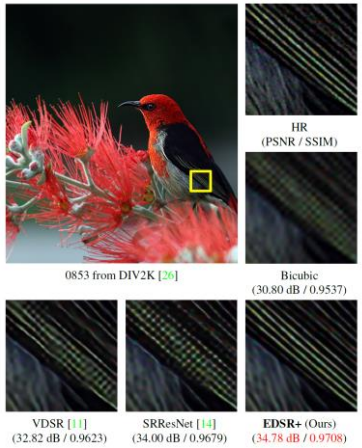
- Detail recovery



SRCNN
[Zhang et al. 2011]



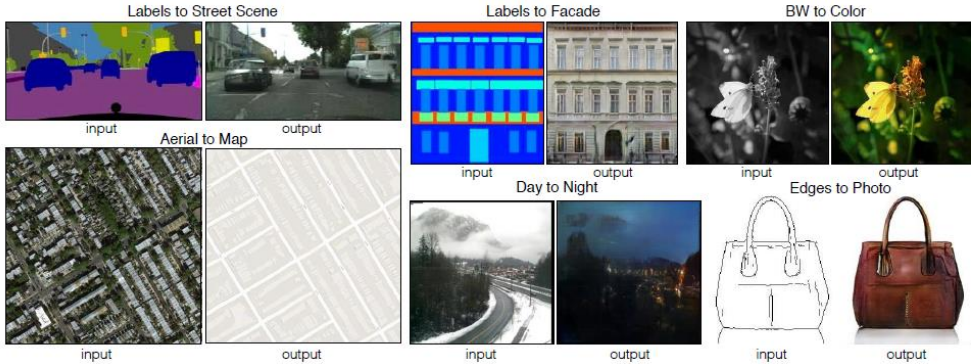
SRGAN
[Ledig et al. 2016]



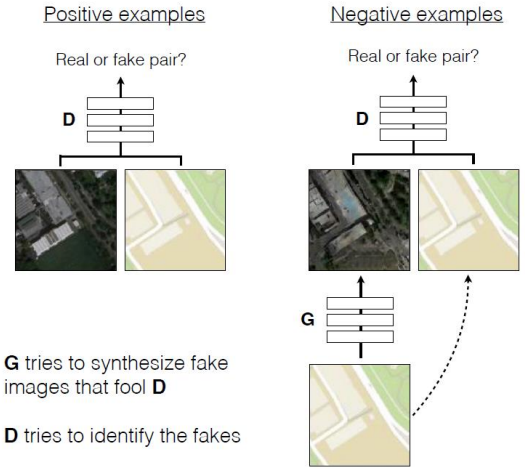
EDSR
[Lim et al. 2017]

POTENTIAL SOLUTIONS

- Image & Font translation



pix2pix
[Isola et al. 2016]



POTENTIAL SOLUTIONS

- Font translation

Rewrite
[Tian et al. 2016]

zi2zi
[Tian et al. 2017]

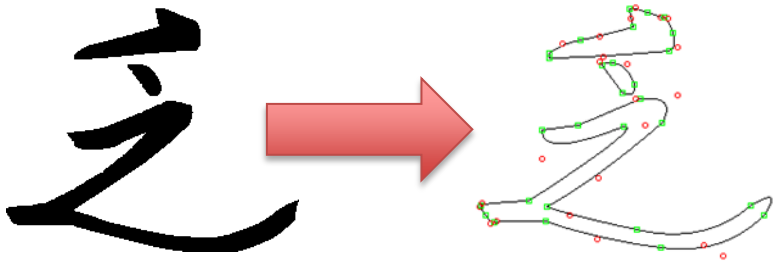
Method Description

PROBLEM DESCRIPTION

- Vectorization
  - Source: Image

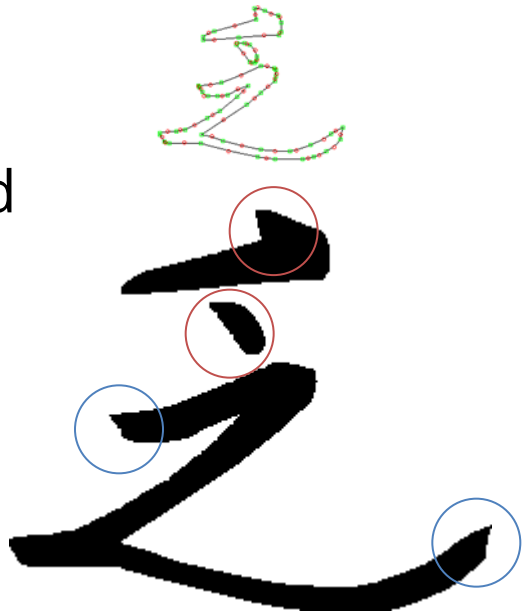
    (render from TTF/scan)
  - Target: as-compact-as-possible contour fit for TTF

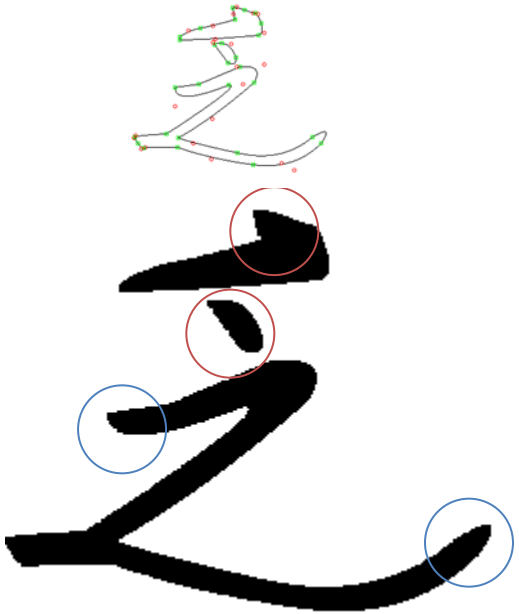
    (consist of Straight Line and quadratic Bezier Curve)



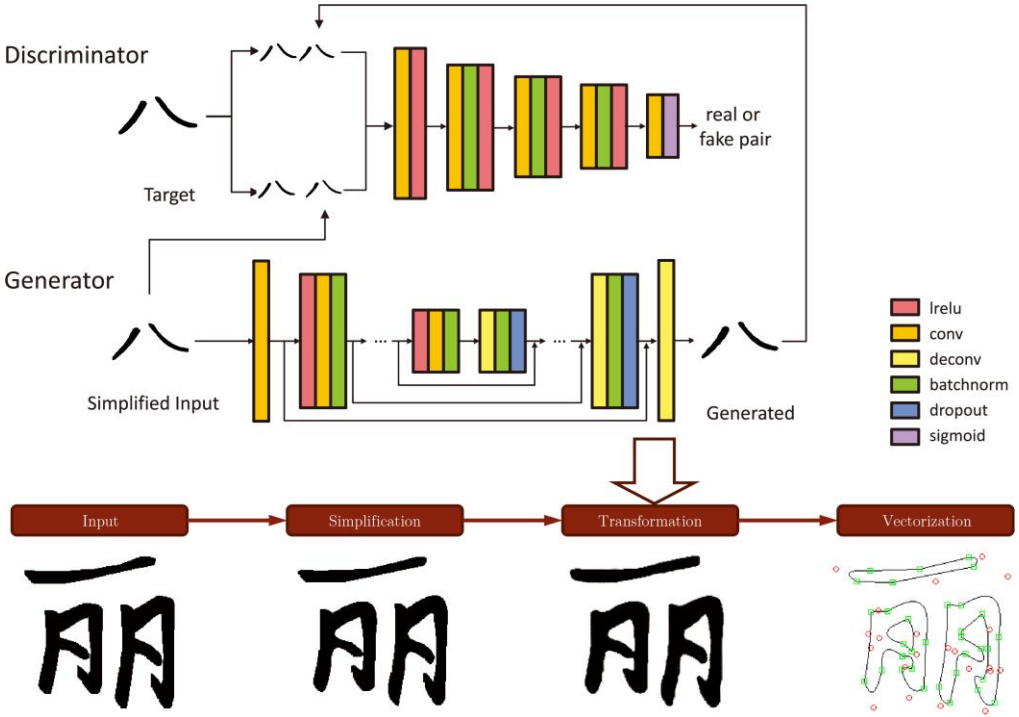
BEFORE & AFTER HUMAN ANNOTATION

- Number of points
- Unimportant detail and noise smoothed (blue)
- Important feature preserved (red)
- Seems almost same



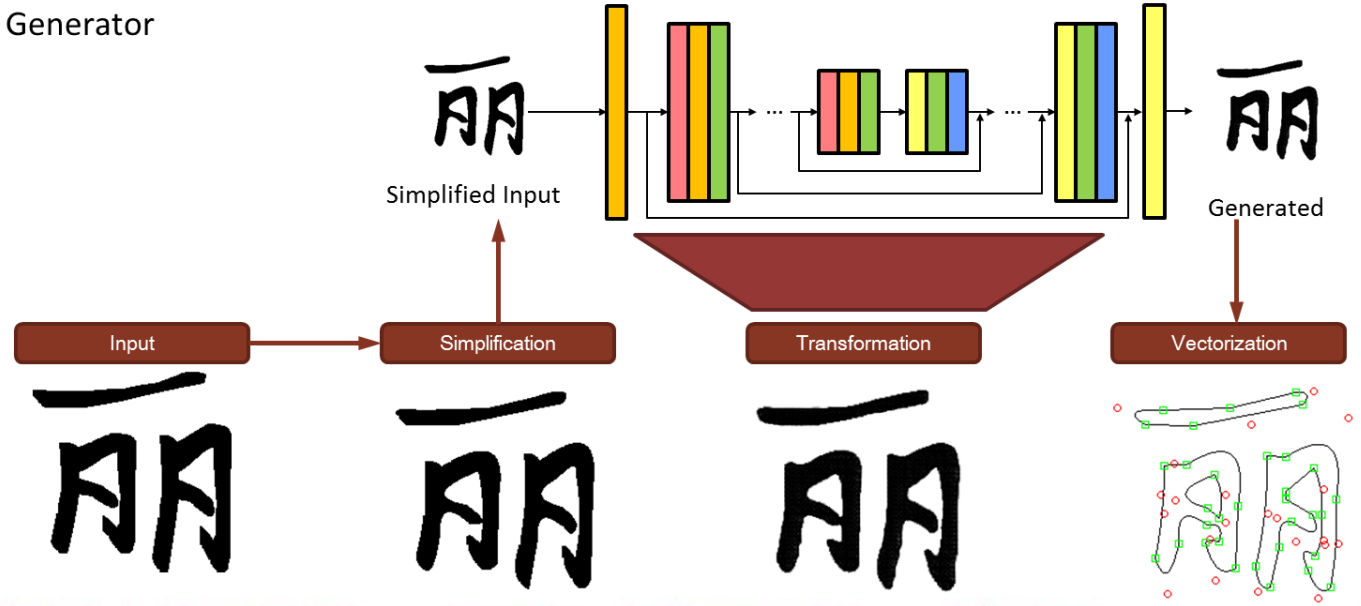


OVERVIEW


Discriminator
Target
real or fake pair
Generator
Simplified Input
Generated
lrelu
conv
deconv
batchnorm
dropout
sigmoid
Input
Simplification
Transformation
Vectorization

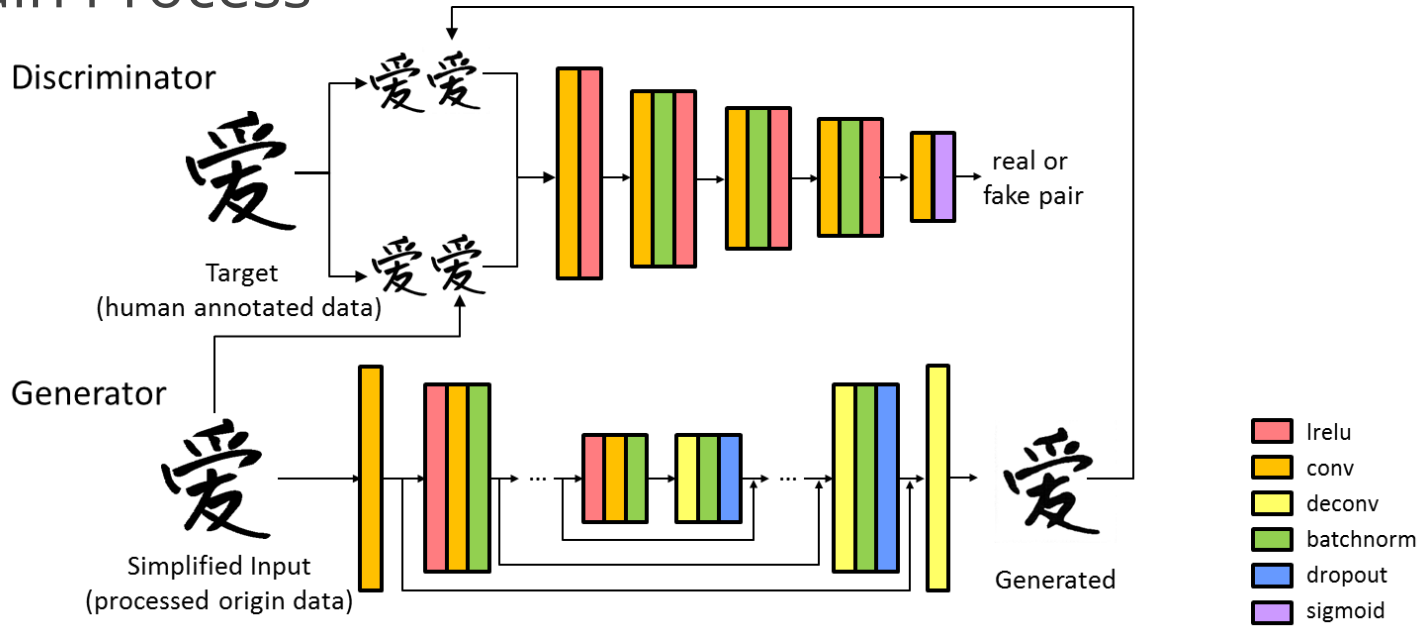
OVERVIEW

- Generate Process



OVERVIEW

- Train Process



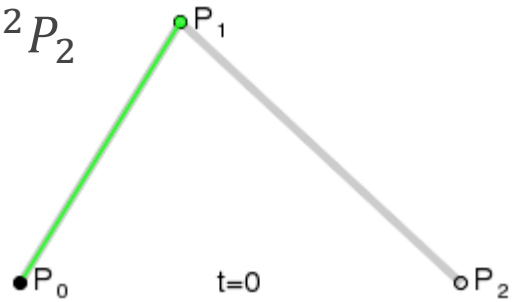
SIMPLIFICATION & VECTORIZATION

- Problem Description
  - Given Input Contour P, Find the polygonal curve Q with minimum number of curve segments where error $E(P)$ of each point is lower than a given bound ϵ.
  - For TTF font vectorization, the type of curves is limited to quadratic Bezier curve and straight line.

SIMPLIFICATION & VECTORIZATION

- Truetype
  - Developed by Apple and Microsoft in the late 1980s
  - To compete against Adobe's Type 1 fonts
- Quadratic Bezier curve

$$B(t) = (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2$$



SIMPLIFICATION & VECTORIZATION

- Error Calculation
  - Straight Line: Decided by edge point Q_0 and Q_1
    For each integer point (x, y) on input contour, its error can be describe as

    $$E(x,y) = \min_{0 \le t \le 1} \left\{ [x - S_x(t)]^2 + [y - S_y(t)]^2 \right\}$$

    where $S(t) = (1-t)Q_0 + tQ_1$
  - Bezier Curve: Except for edge point Q_0 and Q_2,
    Q_1 need to be estimated

SIMPLIFICATION & VECTORIZATION

- Estimation of middle point
  - Least Square Method
    - Use straight Line segments as initial “curve” and estimate their positions
      Given Input Contour Segment P_a to P_b, the estimate position of P_c should be

$$t_c = \frac{\sum_{i=a}^{c-1} |P_i P_{i+1}|}{\sum_{i=a}^{b-1} |P_i P_{i+1}|}$$

    - Getting each t_c, we can use Least Square Method to calculate the intermediate control point (also verbally called middle point)

$$M = \frac{\sum_{i=a}^{b} \left(2t_i(1-t_i)(P_i - (1-t_i)^2 P_a - \left(1-t_j\right)^2 P_b \right)}{\sum_{i=a}^{b} (2t_i(1-t_i))^2}$$

SIMPLIFICATION & VECTORIZATION

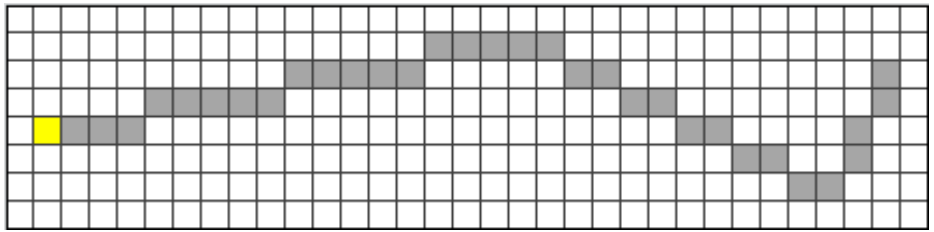
- Estimation of middle point
  - After getting middle point M, we come back to reestimate point positions

$$\widehat{t_c} = \underset{0 \le \widehat{t_c} \le 1}{\operatorname{argmin}} \left\{ \left\| (1-\widehat{t_c})^2 P_a + 2\widehat{t_c}(1-\widehat{t_c}) M + \widehat{t_c}^{\,2} P_b - P_c \right\|_2^2 \right\}$$

  - The solution of the above formula can be transformed into a cubic equation. Solving it, we can get a new $\widehat{t_c}$ to reestimate middle point M
  - It coverages in a few cycles.

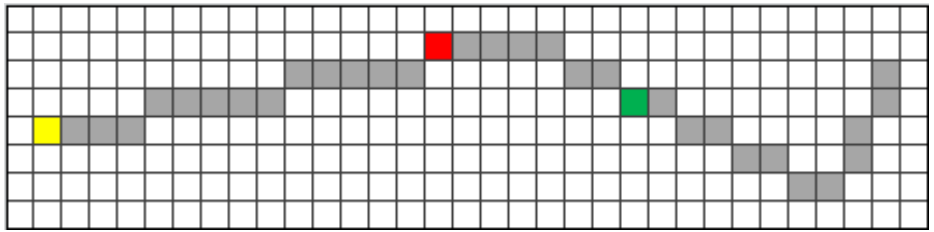
SIMPLIFICATION & VECTORIZATION

- Find Furthest Possible position: check together



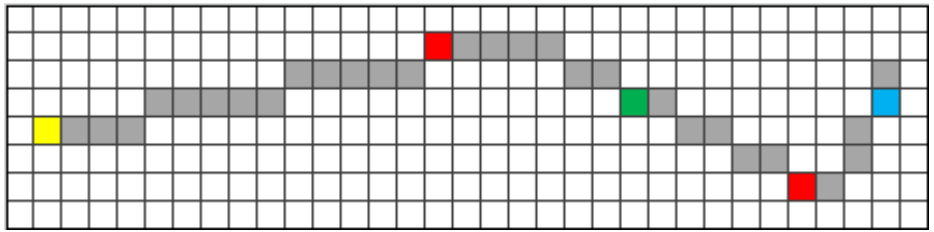
SIMPLIFICATION & VECTORIZATION

- Find Furthest Possible position: check together
  - Straight Line: If there exists one point 2ϵ away from the line between P_a and P_b



SIMPLIFICATION & VECTORIZATION

- Find Furthest Possible position: check together
  - Straight Line: If there exists one point 2ϵ away from the line between P_a and P_b
  - Quadratic Bezier curve: If there exists two points at the opposite side of the line between P_a and P_b, and at least 2ϵ away



SIMPLIFICATION & VECTORIZATION

- Check matches precisely
  - Render curve Q to integer points
  - Make sure Contour Segment P_a to P_b matches Q one by one

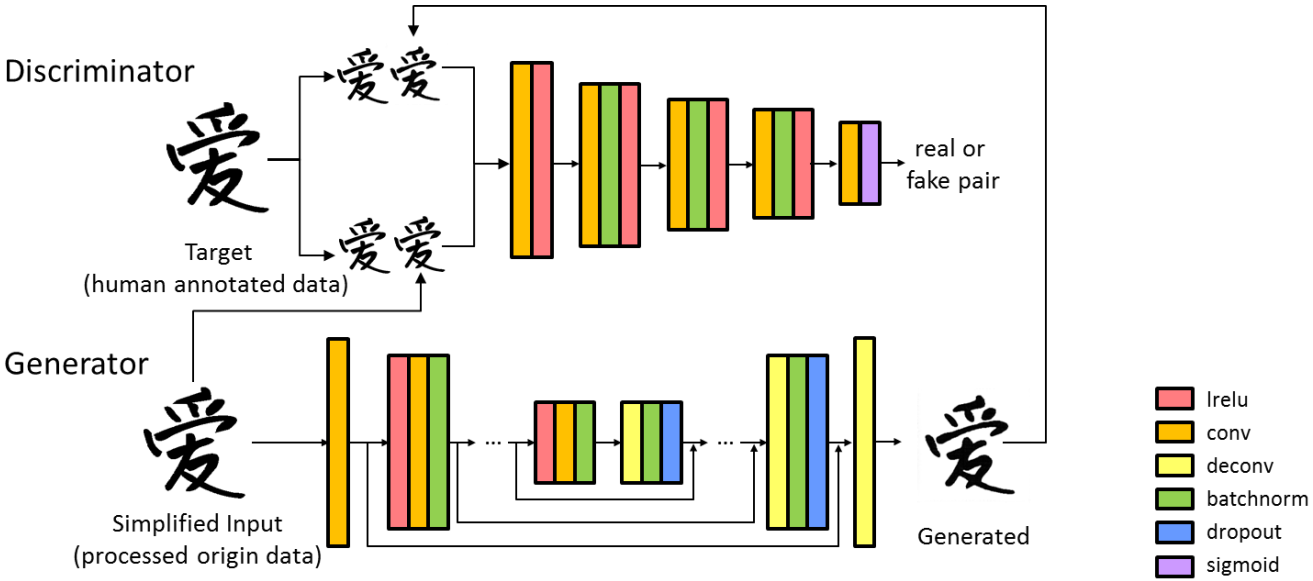
SIMPLIFICATION & VECTORIZATION

- Connect Segments
  - With algorithm proposed in [Kolesnikov et al. 2007]
    - Traverse start position
    - Calculate and update the minimal cost and backward pointer from near to far
    - Choose the position go around with least cost

TRAIN PROCESS

- Dataset
  - 6763 characters in total
  - To cover all strokes
  - Choose 775 character train dataset[Lian et al. 2016]

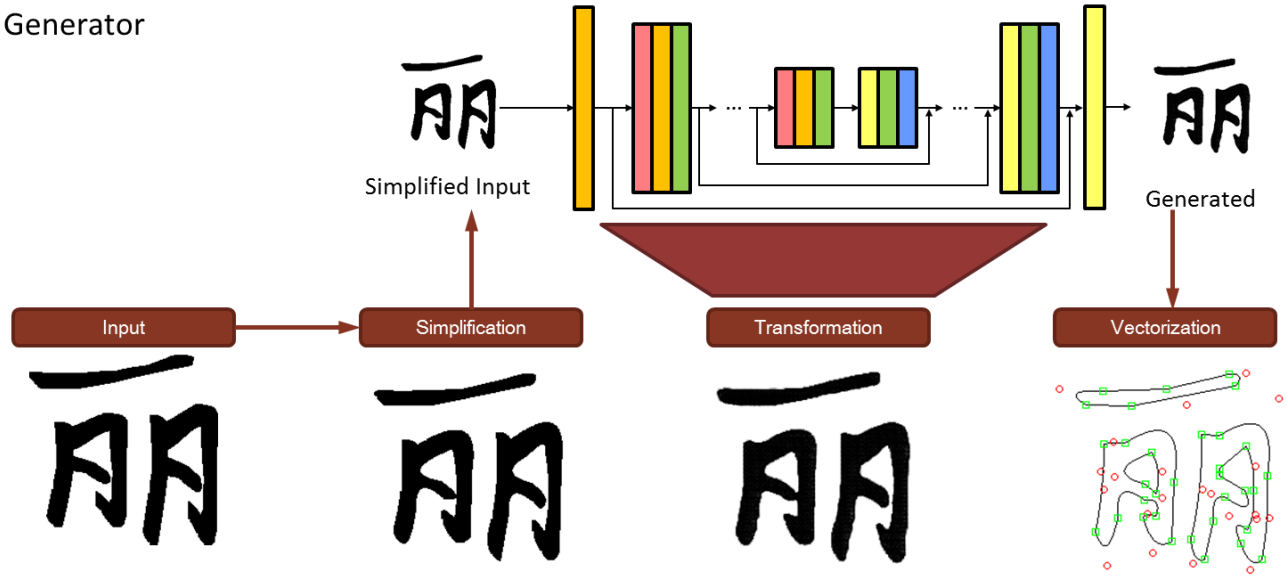
TRAIN PROCESS



TRAIN PROCESS

- Simplify Input data with a larger bound($\epsilon = 2$ px when image is 256×256)
- Choose 775 simplified input character image, pair with human annotated results.
- Train the GAN

GENERATE PROCESS

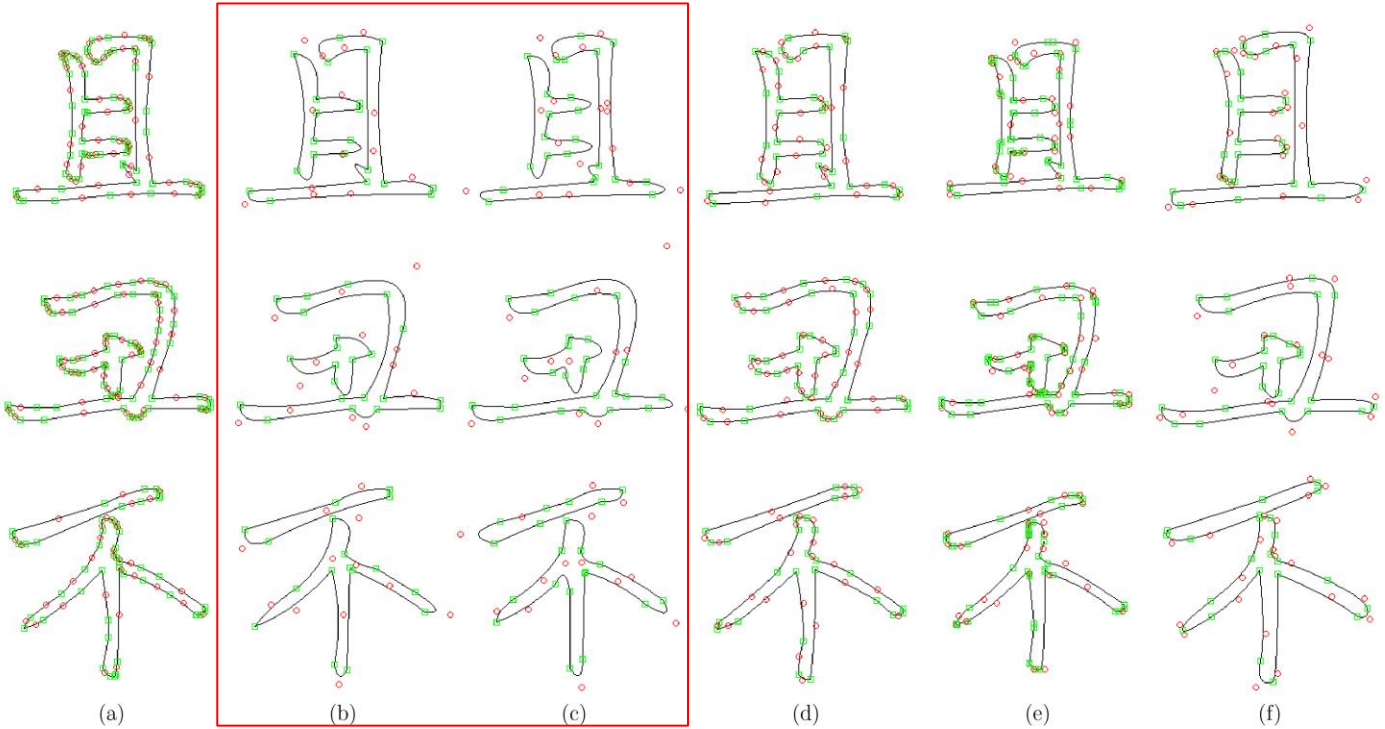


GENERATE PROCESS

- Generate the remaining simplified input character to vectorized results
  - Generate “recovered” Image
  - Vectorize the above image with smaller tolerance ($\epsilon = 1$ px)

Experimental Results

SIMILARITY



(a) Input; (b) Ours w/o NN; (c) Ours with NN; (d) Pan's method[Pan et al. 2014] ; (e) FontCreator; (f) Manual vectorization.

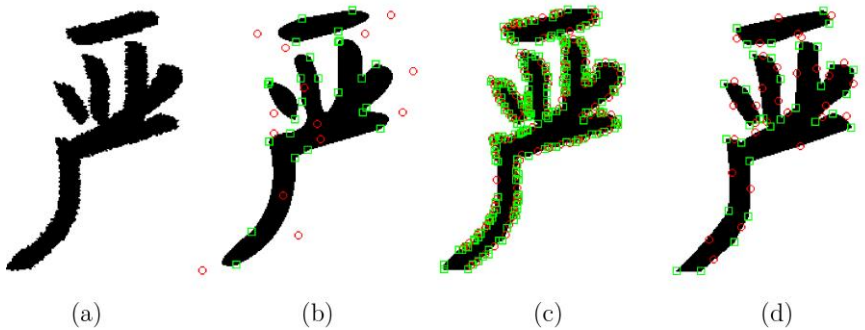
COMPRESSION

- Count the first 8 characters

Type/Method	Number of Control Points
Input	593
Font Creator	493
Pan's Method[Pan et al. 2014]	379
Our Method (w/o NN)	189
Our Method (with NN)	201
Human Annotation	290

DENOISING

- Denoising
  - Detail preserved
  - Less Points
  - smooth surface



(a) Input; (b) Our result, 50 control points; (c) Pan's method [Pan et al. 2014] with high resolution, 394 control points; (d) Pan's method [Pan et al. 2014] with low resolution, 79 control points.

Conclusion and Future Work

CONCLUSION

- Data-driven vectorization method
  - make use of font designer knowledge
- Less storage
  - while preserving salient details
- Robustness

FUTURE WORK

- End-to-end vectorization network
  (e.g. Polygon-RNN [Castrejon et al. 2017]
  Polygon-RNN++[Acuna et al. 2018])
- Auto generation of vectorized fonts
  (instead of just generate images)

Sponsored by

SIGGRAPH ASIA 2018 TOKYO

CONFERENCE 4 – 7 December 2018
EXHIBITION 5 – 7 December 2018
Tokyo International Forum, Japan
SA2018.SIGGRAPH.ORG

THANK YOU FOR YOUR ATTENTION!

zeqing.xia@pku.edu.cn