

Real-Time Data-Driven Interactive Rough Sketch Inking

Edgar Simo-Serra, Satoshi Iizuka, Hiroshi Ishikawa

Wednesday, August 15, 2018

Waseda University

Motivation



Input

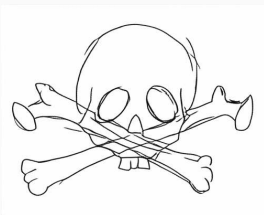


Previous Approach

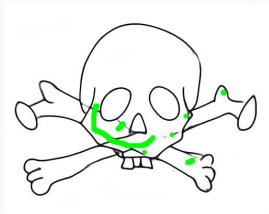
Motivation



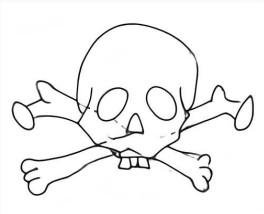
Input



Previous Approach



User Input



Standard Eraser

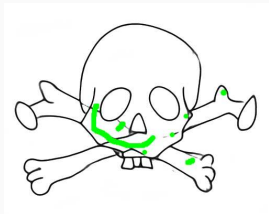
Motivation



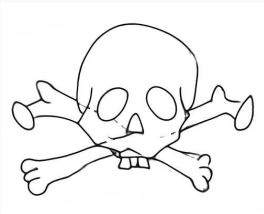
Input



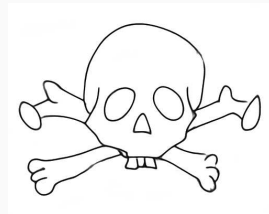
Previous Approach



User Input



Standard Eraser



Proposed

- “1. The inker’s main purpose is to translate the penciller’s graphite pencil lines into reproducible, black, ink lines.*
- 2. The inker must honor the penciller’s original intent while adjusting any obvious mistakes.*
- 3. The inker determines the look of the finished art.”*

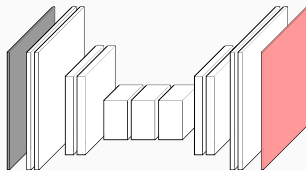
— Gary Martin, *The Art of Comic Book Inking* [1997]

Interactive Neural Networks

- Feed-forward fully convolutional neural network



Input



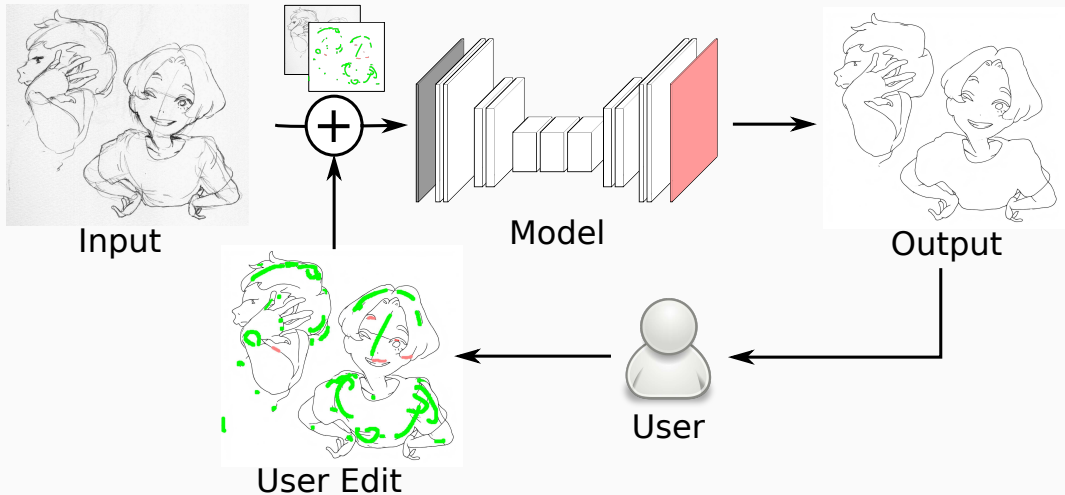
Model



Output

Interactive Neural Networks

- Feed-forward fully convolutional neural network
- Input rough sketch and user edit are concatenated channel-wise



Interactive Neural Networks - Related Work

- User input is treated as an additional image channel
- Training user input is sampled from ground truth
 - Grayscale image colorization [Sangkloy+ 2017, Zhang+ 2017]



Real-Time User-Guided Image Colorization with Learned Deep Priors. Richard Zhang et al. SIGGRAPH 2017

Interactive Neural Networks - Related Work

- User input is treated as an additional image channel
- Training user input is sampled from ground truth
 - Grayscale image colorization [Sangkloy+ 2017, Zhang+ 2017]
 - **Not directly applicable to the rough sketch inking problem**



Interactive Neural Networks - Related Work

- User input is treated as an additional image channel
- Training user input is sampled from ground truth
 - Grayscale image colorization [Sangkloy+ 2017, Zhang+ 2017]
 - Not directly applicable to the rough sketch inking problem
 - How to train an interactive network for inking?

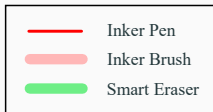


Proposed Framework

- **Main contributions**
 - Line width normalization
 - Simulation of user edits
- Three different smart tools
- Evaluation with a perceptual user study

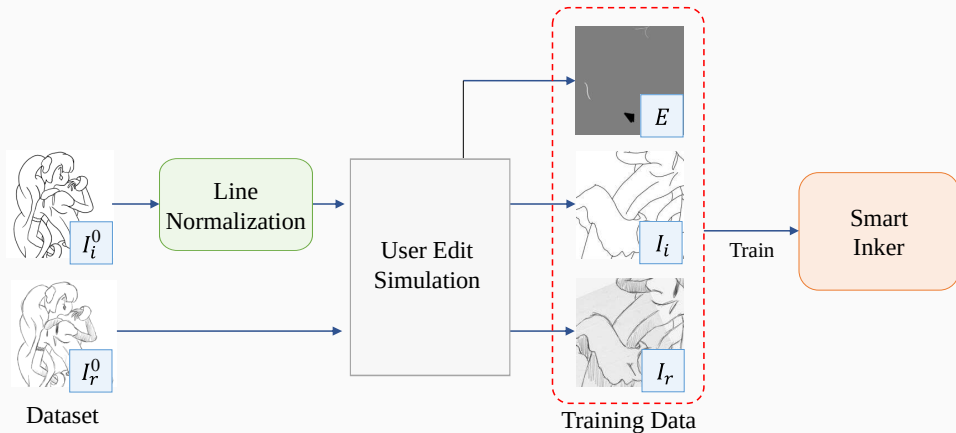


Input rough sketch



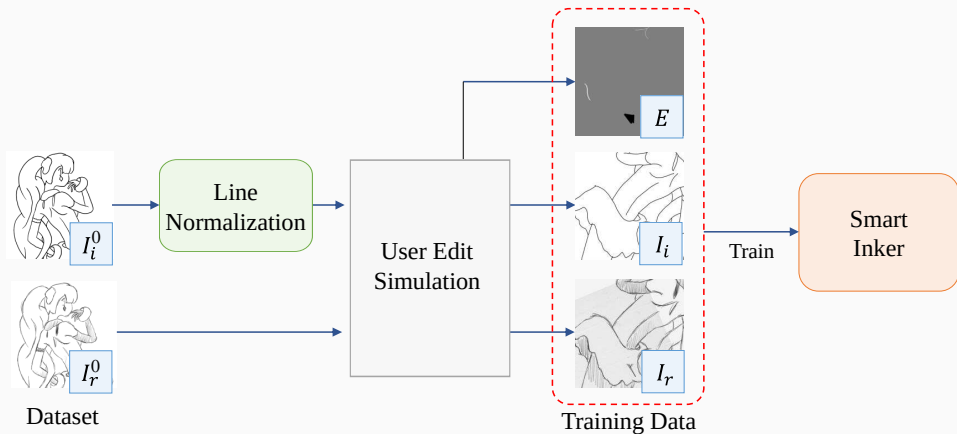
Training Framework

1. Line width normalization
2. Simulation of user edits

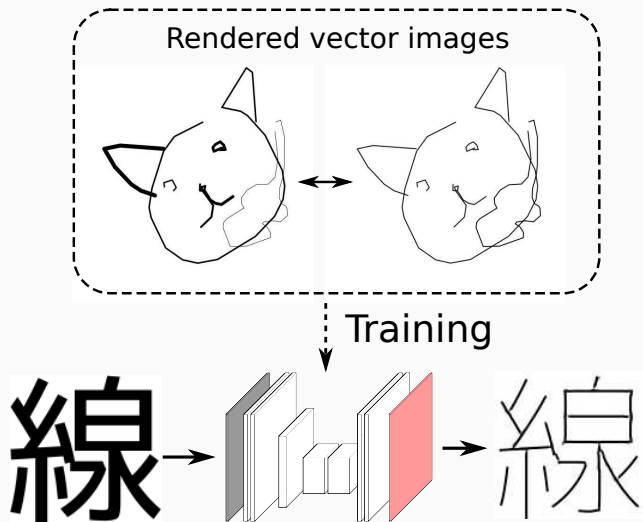


Training Framework - Line width normalization

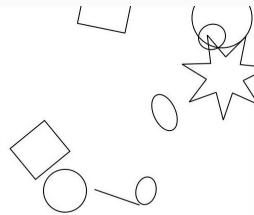
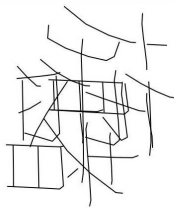
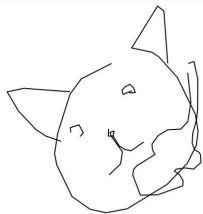
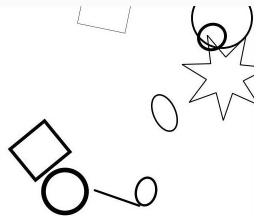
1. Line width normalization
2. Simulation of user edits



Training Framework - Line width normalization



Training Framework - Line width normalization



QuickDraw

TUD-Berlin

KanjiVG

Synthetic

Training Framework - Line width normalization



Input



[Zhang and Suen 1984]



Ours

Training Framework - Line width normalization



Input



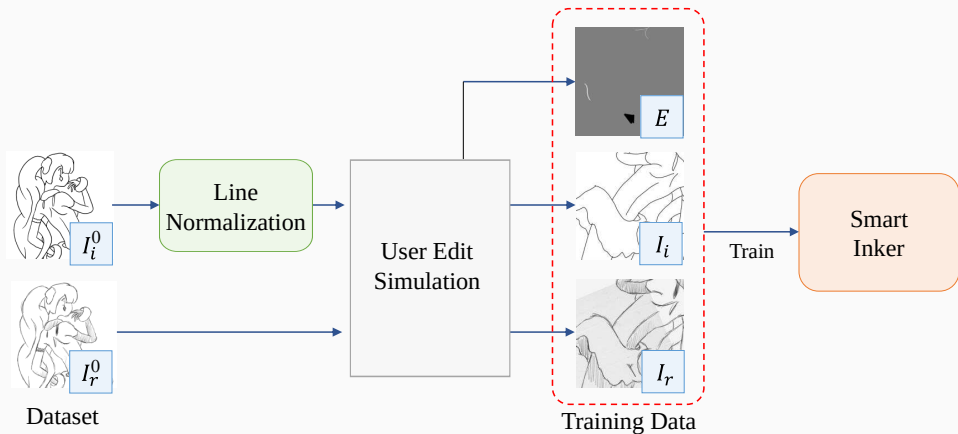
No normalization



Normalization

Training Framework - Simulation of user edits

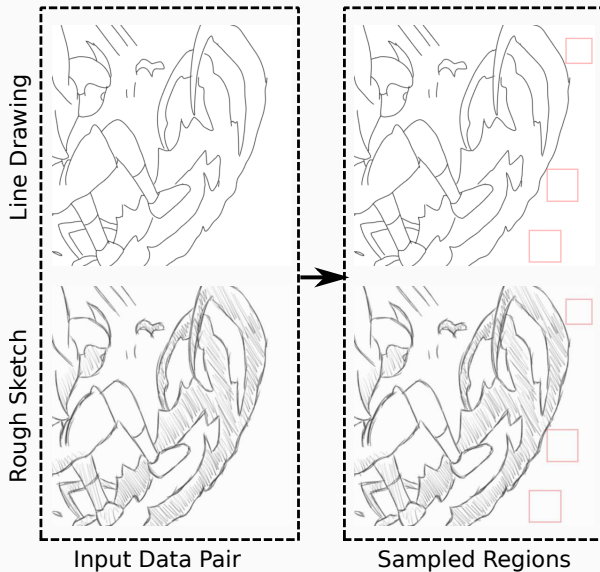
1. Line width normalization
2. Simulation of user edits



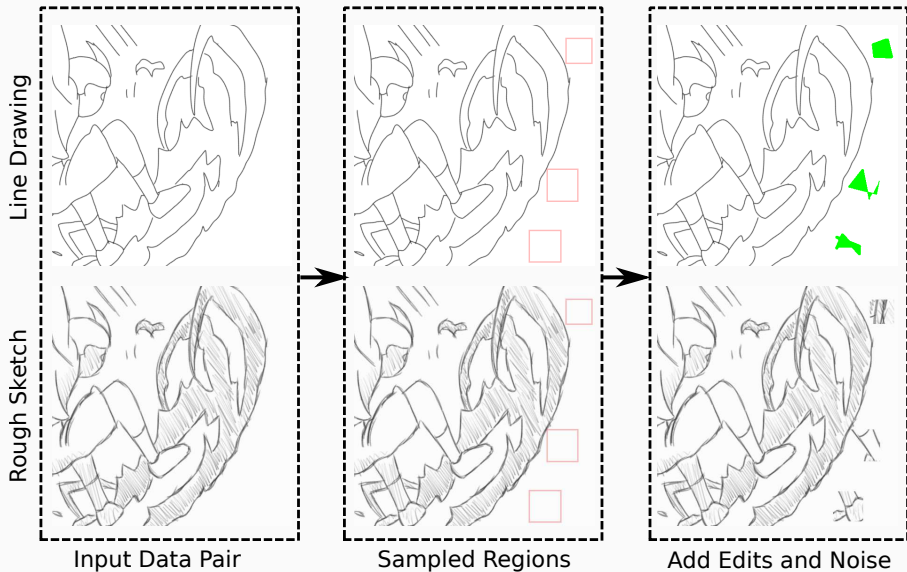
Training Framework - Simulation of user edits



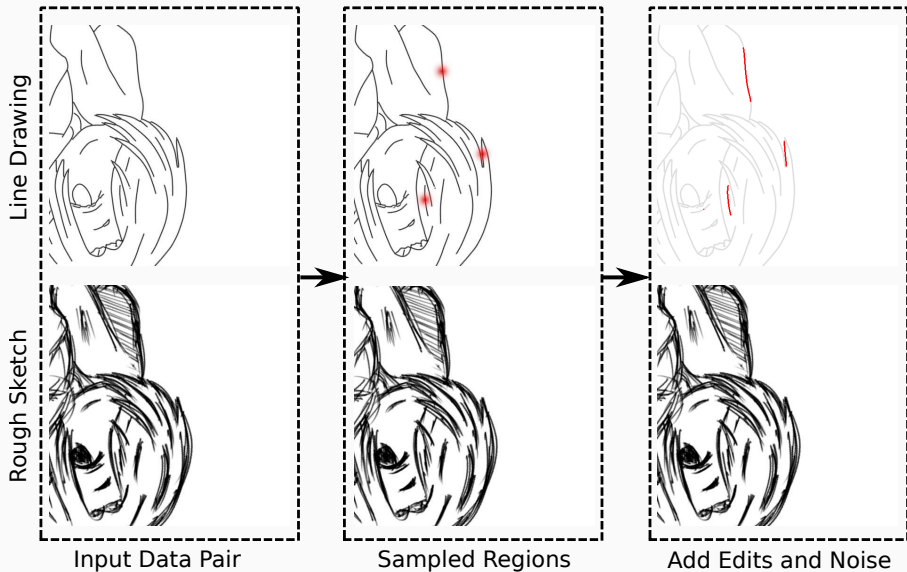
Training Framework - Simulation of user edits



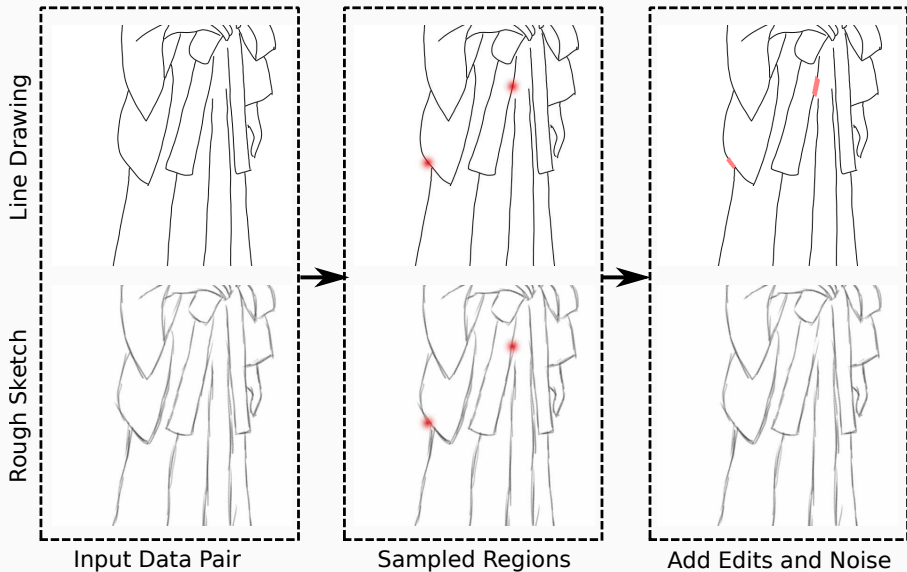
Training Framework - Simulation of user edits



Training Framework - Simulation of user edits

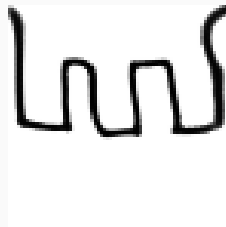
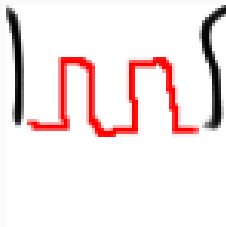
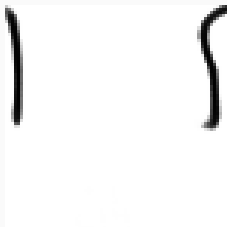


Training Framework - Simulation of user edits



Smart Tools

- Inker Pen
- Allows for fine-grained line control



Input

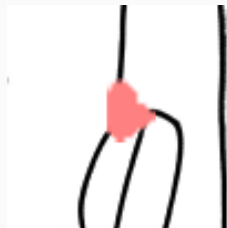
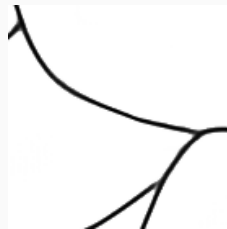
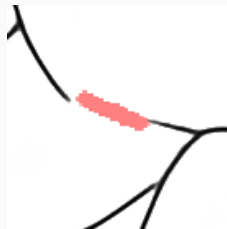
Automatic

Edit

Ours

Smart Tools

- Inker Brush
- Sloppy and fast line manipulation



Input

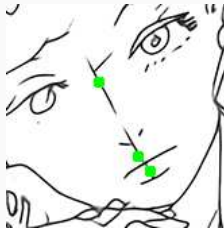
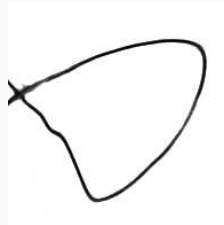
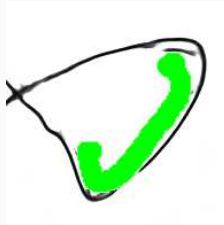
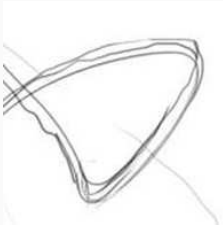
Automatic

Edit

Ours

Smart Tools

- Smart Eraser
- Takes into account rough sketch when erasing



Input

Automatic

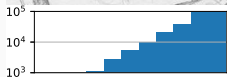
Edit

Ours

Training

$$L(y, y^*) = \underbrace{|(y - y^*)|}_{L_1 \text{ loss}} \odot \underbrace{(1 + \gamma(1 - y^*))}_{\text{Weight lines with } \gamma} |_1$$

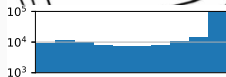
- Using L_1 loss
- Change weight of lines with γ



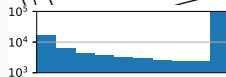
Input



[Simo-Serra+ 2016]



Baseline



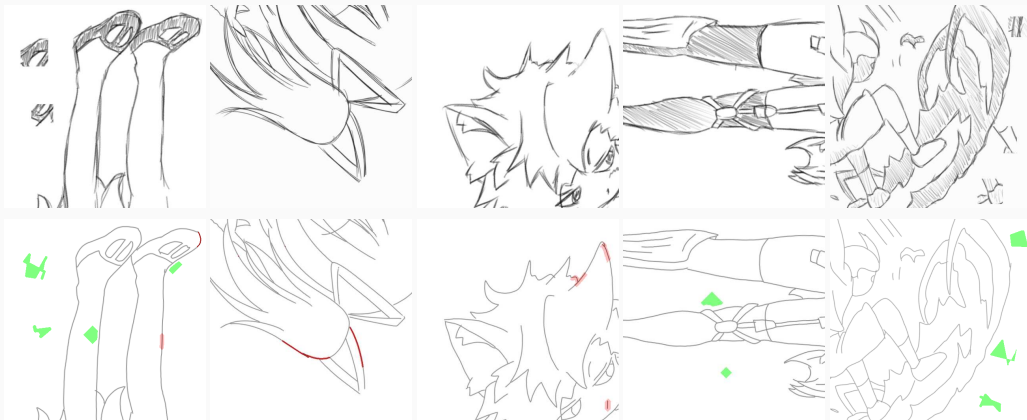
Ours

- Similar to model of [Simo-Serra+ 2016]
- 24 layer fully convolutional neural network
- Number of filters optimized for real-time performance
- Roughly three times the performance

	Approach	Parameters	1024 ² px	1512 ² px	2048 ² px	2560 ² px
[Simo-Serra+]	2016	44,551,425	238.8ms	562.4ms	984.7ms	1.59s
	Ours	12,795,169	89.9ms	225.5ms	382.7ms	592.9ms

Dataset

- 288 rough sketch and line drawing pairs
- More challenging than previous works



Real-Time Data-Driven Interactive Rough Sketch Inking

Edgar Simo-Serra Satoshi Iizuka Hiroshi Ishikawa
Waseda University



GENERATIONS / VANCOUVER
12-16 AUGUST
SIGGRAPH2018

Results



Input



Automatic



Edit



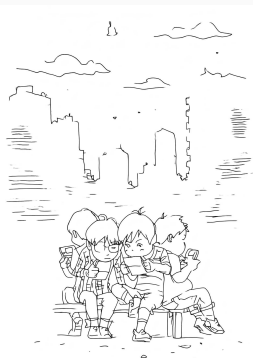
Ours

©Eisaku Kubonouchi

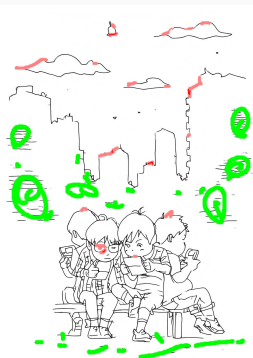
Results



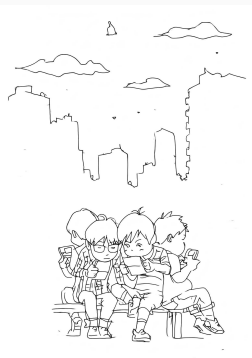
Input



Automatic



Edit



Ours

©Eisaku Kubonouchi

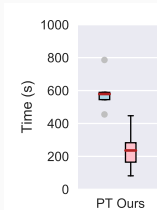
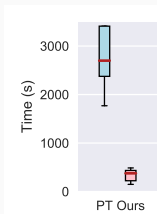
User Study

- Comparison (time) with proposed approach vs Clip Studio Pro
- Total of 10 users and 10 unique images
- Each user processes random 5 images with each tool
- Total average time of 2.8 hours per user
- Overall 1.8× speed-up with ours



Some of the images used in the user study

User Study



Time

Input

Amateur

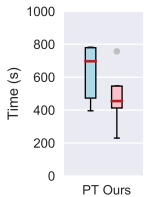
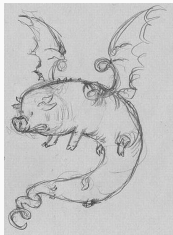
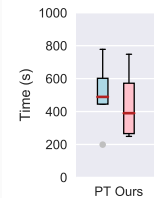
Experienced

Edit

Ours

User Study

©David Revoy www.davidrevoy.com



Time



Input



Amateur



Experienced



Edit



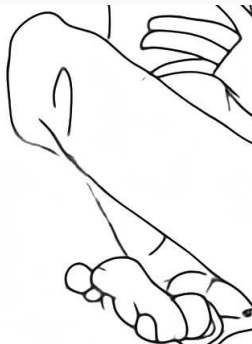
Ours

©Krenz Cushart

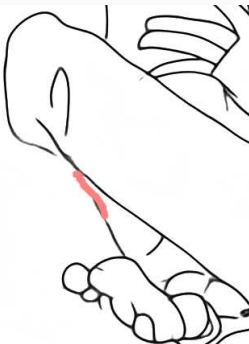
Limitation



Input



Automatic



Edit



Ours

To conclude

<http://hi.cs.waseda.ac.jp/~esimo/research/inking/>

- Interactive rough sketch inking framework
 - Line width normalization
 - User edit simulation

