Noname manuscript No.
(will be inserted by the editor)

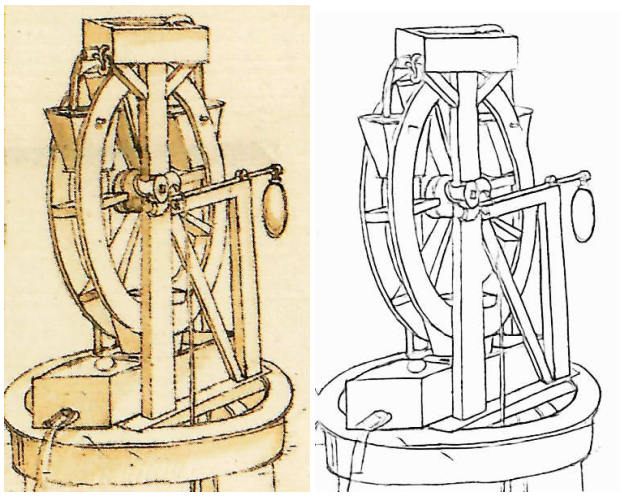
Learning to Restore Deteriorated Line Drawing

Kazuma Sasaki · Satoshi Iizuka · Edgar Simo-Serra · Hiroshi Ishikawa

Received: date / Accepted: date

Abstract We propose a fully automatic approach to restore aged old line drawings. We decompose the task into two subtasks: the line extraction subtask, which aims to extract line fragments and remove the paper texture background, and the restoration subtask, which fills in possible gaps and deterioration of the lines to produce a clean line drawing. Our approach is based on a convolutional neural network that consists of two sub-networks corresponding to the two subtasks. They are trained as part of a single framework in an end-to-end fashion. We also introduce a new dataset consisting of manually annotated sketches by Leonardo da Vinci which, in combination with a synthetic data generation approach, allows training the network to restore deteriorated line drawings. We evaluate our method on challenging 500-year-old sketches and compare with existing approaches with a user study, in which it is found that our approach is preferred 72.7% of the time.

Keywords Image Restoration · Line Drawings · Image Manipulation · Convolutional Neural Network



Input old sketch Proposed approach

Fig. 1 Example clean up by our proposed approach on an original sketch by Leonardo da Vinci and is in the public domain.

1 Introduction

Line drawings are a minimalistic way of representing shapes, emphasizing form and outline over color, shading, and texture. They play a pivotal role in expressing objects during sketching, and are widely used in many applications. Most line drawings are drawn using pencil on paper, which over time, will age and fade. Immortalizing these line drawings in digital form necessitates tedious manual annotation and resulting high cost. In this work, we propose a high-performance method for the automatic clean up of line drawings, amenable to obtaining high-quality digitalization of line drawings. This allows digitalization of not only recent line drawings, but also line drawings of historical importance such as those drawn by Leonardo da Vinci as shown in Fig. 1.

We base our approach on convolutional neural networks, which have shown great performance in many different image translation tasks applied to line draw-

K. Sasaki
E-mail: milky_kaid.lc@ruri.waseda.jp ·
S. Iizuka
E-mail: iizuka@aoni.waseda.jp ·
E. Simo-Serra
E-mail: esimo@aoni.waseda.jp ·
H. Ishikawa
E-mail: hfs@waseda.jp

Waseda University

ings, such as tone removal [16] and sketch simplification [25]. Unlike recent approaches, we focus on old, deteriorated line drawings, in which the paper has changed color and the lines have faded. Furthermore, aged line drawings commonly have other defects such as spots, holes, ink leakage from nearby pages, caused by humidity, insect infestations, and other poor conditions. Thus, not only is it necessary to remove the paper background and strengthen the lines, but it also becomes important to complete disconnected and broken lines. Instead of using a single fully convolutional network model for completing line drawings [23], we propose using two networks: one to extract the line segments, another to restore the line drawing, which we will show to perform significantly better than existing approaches.

We additionally propose a new dataset comprising deteriorated old sketches by Leonardo da Vinci that have suffered many different types of deterioration and their ground-truth line drawing annotations. We show that this dataset, in combination with manually and randomly generated synthetic data, allows for training of high performance line drawing restoration networks that are able to generalize to many different types of input sketches.

We validate our approach with a user study, in which we compare to a strong baseline based on [23] trained using our new dataset, and show that 72.7% of users prefer our approach.

In summary, our contributions are: (1) a new model that jointly extracts line segments and restores the line drawings, (2) a manually annotated dataset based on sketches by Leonardo da Vinci, and (3) a user study in which we evaluate our approach and compare against a strong baseline.

2 Related Work

2.1 Line Drawings

Traditionally, image patterns have been exploited for restoration of line drawings [4] with a focus of engineering drawings. Later, contour matching was used for skeleton generation [10]. On these lines of work, most of the research has focused on the vectorization of such line drawings by using the maximal inscribing circle algorithm [3] or morphological operators [14]. More recent approaches first perform a binarization and then optimize a graph to obtain a final vectorization [8]. However, all these approaches share the limitation that the input line drawing must be fairly clean, and are not effective on deteriorated and aged line drawings, which is our focus in this paper.

Recently, fully convolutional networks (FCNs) [17] have been applied to tasks related to line drawing. Simo-Serra et al. [26] proposed a sketch simplification model trained with supervised data that was later extended to the semi-supervised setting [25]. Li et al. [16] proposed simulating manga tones on line drawings to train a model to extract the structural lines. More similar to this work, Sasaki et al. [23] proposed joint detection and inpainting of line drawings. However, all these approaches focus on non-deteriorated line drawings, unlike our proposed approach.

2.2 Natural Image Restoration

Restoration of natural images and image denoising has always been an active research topic [19], with popular approaches being based on total variation [20,22], dictionary learning [29,9,7], and the BM3D algorithm [5]. Most recent approaches focus on using neural networks trained with large amounts of data [13], where auto-encoders have shown great results in restoring natural images [28,31,18]. While these approaches work for natural images, which contain rich image gradients, they perform significantly worse when images are sparse, as in the case of line drawings.

A related task to image restoration is that of image inpainting, in which regions of the input image are replaced, allowing removal of noise and other unwanted regions of the input image [2]. Typically, the areas to be removed are given as user-specified inputs. The dominant approaches have been patch-based [1,6, 15,24,30], in which the unwanted region of the image is inpainted using patches taken from the same image, allowing for realistic inpainting of a diversity of images. However, these approaches can only generate objects that already exist in the image, and thus cannot generalize to novel objects. Recent approaches [21,11] based on deep learning resolve this issue by training on large datasets, which allows them to generate objects not seen in the image. However, all these approaches require user input to specify the removed regions, unlike our approach.

3 Proposed Approach

We base our line drawing clean-up approach on convolutional neural networks trained jointly with old sketches and cleaner line drawings. Instead of training a single network, we use two sub-networks: one for extracting the lines and line segments from the deteriorated sketch, and one for restoring the missing regions and completing the line drawing. We employ three kinds of data

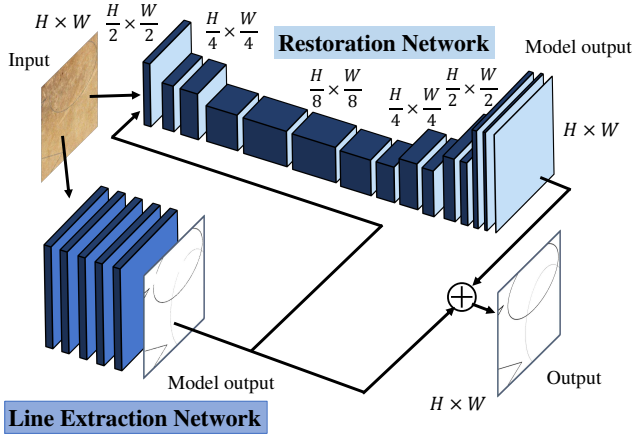


Fig. 2 Overview of our architecture for line drawing restoration. The approach consists of two sub-models: a line extraction network, and a restoration network. The output of the line extraction network is used as an input to the restoration network, and the output of both the line extraction network and restoration network is added component-wise to form the final output.

to train the whole network: the input images, which have deteriorated background and lines with gaps, the ground-truth extracted line drawings with clean background but that may still have gaps, and the ground-truth restored line drawings with completed gaps.

3.1 Model Architecture

The line cleanup model is based on two smaller networks: a line extraction network E and a restoration network R. First, an input image x is processed by the line extraction network, which eliminates the background and extracts lines from the input. Afterwards, the input image and the output of the line extraction network are fed to the restoration network, which completes the gaps in the lines. Finally, the output of both networks is added and used to generate the final output image. Thus, the output of our model F for an input image x can be written as:

$$F(x) = \max(0, \min(1, E(x) + R(x, E(x)))) \quad , \qquad (1)$$

which we note is clamped to the $[0, 1]$ range. An overview of the approach is shown in Fig. 2, and visualization of the output of the different sub-networks can be seen in Fig. 3.

The line extraction network is based on a fully convolutional neural network with 6 layers. All layers use 32 filters and have a stride of 1×1 pixels. The first layer uses an 11×11 pixel kernel for the convolutions, while the rest of the layers use 3×3 pixel kernels to minimize the network parameters [27]. In order not to change the output size, 0-value padding is used in all the layers. After each convolution layer, we use the Rectified Linear Unit (ReLU) activation function. The final layer uses a Sigmoid activation function to keep the output in the $[0, 1]$ range.

Table 1 Model architecture for our line restoration network. Up-sampling is done using nearest neighbours.

Layer type	Kernel	Strides	Output size
input	-	-	$2 \times H \times W$
convolution	5×5	2×2	$32 \times {}^{H}/_{2} \times {}^{W}/_{2}$
convolution	3×3	2×2	$64 \times {}^{H}/_{4} \times {}^{W}/_{4}$
convolution	3×3	1×1	$128 \times {}^{H}/_{4} \times {}^{W}/_{4}$
convolution	3×3	2×2	$256 \times {}^{H}/_{8} \times {}^{W}/_{8}$
convolution	3×3	1×1	$512 \times {}^{H}/_{8} \times {}^{W}/_{8}$
convolution	3×3	1×1	$512 \times {}^{H}/_{8} \times {}^{W}/_{8}$
convolution	3×3	1×1	$256 \times {}^{H}/_{8} \times {}^{W}/_{8}$
convolution	3×3	1×1	$128 \times {}^{H}/_{8} \times {}^{W}/_{8}$
up-sampling	-	-	$128 \times {}^{H}/_{4} \times {}^{W}/_{4}$
convolution	3×3	1×1	$128 \times {}^{H}/_{4} \times {}^{W}/_{4}$
convolution	3×3	1×1	$64 \times {}^{H}/_{4} \times {}^{W}/_{4}$
up-sampling	-	-	$64 \times {}^{H}/_{2} \times {}^{W}/_{2}$
convolution	3×3	1×1	$64 \times {}^{H}/_{2} \times {}^{W}/_{2}$
convolution	3×3	1×1	$32 \times {}^{H}/_{2} \times {}^{W}/_{2}$
up-sampling	-	-	$32 \times H \times W$
convolution	3×3	1×1	$16 \times H \times W$
convolution	3×3	1×1	$8 \times H \times W$
convolution	3×3	1×1	$1 \times H \times W$

The line restoration network consists of a 15-layer fully convolutional neural network. The input consists of a two-layer image where one is the input greyscale deteriorated sketch and the other is the output of the line extraction network. The network internally lowers the size down to $1/64$ of the original area in three steps, and then restores to the original size in another three steps. Down-sampling is done by using convolutions with 2×2 pixel strides, while up-sampling is done by using nearest-neighbour up-sampling. All layers except the first layer use 3×3 kernel convolutions, while the first layer uses a 5×5 kernel convolution. After each convolutional layer except the last one, we apply the ReLU activation function. The last layer uses a hyperbolic tangent activation function to output values in the $[-1, 1]$ range. Finally, the output is added to the output of the line extraction network and clamped to be in the $[0, 1]$ range. The full specification of the line restoration network architecture is given in Table 1.

3.2 Dataset

To train the two-part network, we propose a new dataset and data augmentation techniques for cleaning line draw-

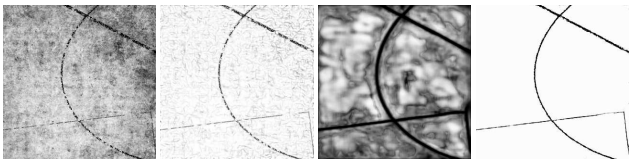


Fig. 3 Example of the outputs of the different networks. We note that the ranges of the pixel values differ for the different outputs. While x, $E(x)$, and $F(x)$ are in the $[0, 1]$ range, $R(x, E(x))$ is in the $[-1, 1]$ range.

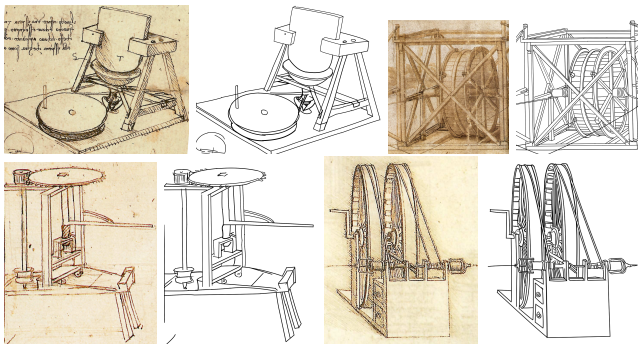


Fig. 4 Example of old sketches with annotation, from the Leonardo da Vinci dataset. Images are by Leonardo da Vinci and are in the public domain.

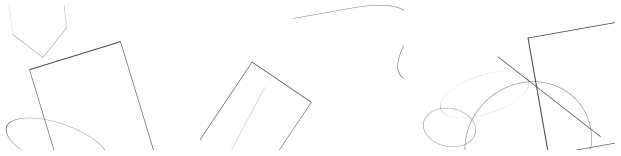


Fig. 5 Examples of randomly generated line drawing data composed by simple primitives. Due to the randomness, not all generated data is very challenging.

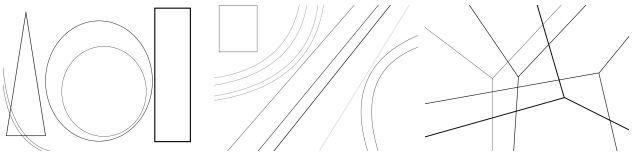


Fig. 6 Examples of manually generated line drawing data composed of simple primitives. The data is designed to provide challenging situations such as complex intersections and parallel lines.

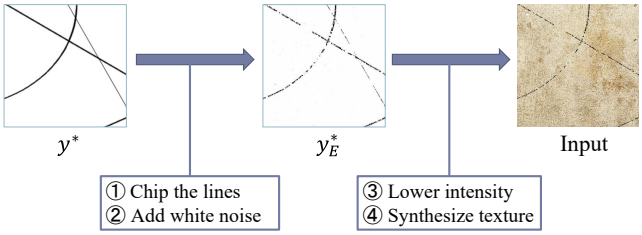


Fig. 7 Overview of our data creation approach to synthetic data. We use the synthetic data as a ground truth and proceed to white out large parts of it. Afterwards, we lower the intensity of the lines and apply realistic paper textures.

ings, based on Leonardo da Vinci's sketches. We complement this new dataset with synthetically generated images: both randomly generated and manually generated patterns.

3.2.1 Leonardo da Vinci dataset

We collect 71 of Leonard da Vinci's old sketch scans, and manually provide annotations of the underlying line drawing. The annotations are added by drawing on top of the old sketches using a pen tablet. Here, we choose those sketches with limited amount of shading that can be considered line drawings, avoiding artist drawings and the most heavily deteriorated sketches. We do not annotate dirt nor text as ground truth lines. Examples of the sketches and their annotations are shown in Fig. 4. Out of the 71 images, we reserve 10 for evaluation, leaving 61 images to be used for training.

3.2.2 Synthetic Data

We complement our main Leonardo da Vinci dataset with large amounts of synthetic data, and in particular focus on two sources: random data, and manually created data that are challenging to restore. We create the data from overlapping primitives, in particular, straight lines, curved lines, circles, triangles, rectangles, and hexagons, with line widths varying from 1 to 14 pixels.

For the random data, the different primitives are generated with sizes between 250 and 1700 pixels, and randomly rotated and translated on a 1200×840 pixel canvas. Furthermore, they also are scaled randomly by a factor of between 0.4 and 2 in both the X and the Y axis. For each canvas, the number of primitives to render is chosen from the binomial distribution $\binom{24}{0.15}$. Ten thousand canvases are rendered randomly offline and form the random dataset. Some examples are shown in Fig. 5.

As the random data procedure does not generate complicated situations such as parallel lines and multiple intersections very often, we complement the random images with 89 manually generated ones to include complex intersections, parallel lines, and tangent lines, as shown in Fig. 6.

3.2.3 Synthetic Training Data

Given the clean synthetic line data, we must generate the corresponding "dirty" rough sketch inputs to train

Fig. 8 Examples of some of the old paper textures used to augment our synthetic line data. These are multiplied component-wise with the line data to simulate many different type of paper textures, similar to those found in real deteriorated line drawings.

our models. For this purpose, we simulate many common types of distortions to try to appear as close to real data as possible. In particular, we simulate large holes, small noise, fading lines, and dirty paper textures. An overview of the approach is shown in Fig. 7.

For each of the images, we create hole distortions of three types: large holes, small holes, and white noise. For each image we randomly generate between 10 and 15 large holes distributed randomly around the image ranging from 10×10 pixels to 40×40 pixels in size. We also generate the 1500 to 2000 small holes ranging from 2×2 pixels to 6×6 pixels in size. This is all done on-the-fly during training. Finally, we randomly turn between 70% and 90% of the pixels white.

As old sketches tend to have fading ink, we simulate this by fading each pixel value to:

$$\frac{x+\alpha}{1+\alpha} \quad , \tag{2}$$

where x is the image, and $\alpha \in [0, 1]$ is a random value. This does not change the white pixels ($x = 1$), while it can whiten originally black pixels to up to 50% brightness.

The synthetic data has white backgrounds, while the old papers we wish to simulate are rarely completely white. Thus, we augment the images with textures of real old paper. In particular, we use 21 different scanned textures, in addition to the plain white background. We superimpose the textures by performing component-wise multiplication with the input image. Examples of the used textures are shown in Fig. 8.

3.3 Training

We train both the line extraction network E and the restoration network R as a single model in an end-to-end fashion using multiple losses. We train by minimizing:

$$\theta^* = \arg\min_{\theta} \|F(x;\theta_E,\theta_R) - y^*\|_2 + \lambda\|E(x;\theta_E) - y^*_E\|_2 \quad , \tag{3}$$

where $\theta = [\theta_E, \theta_R]$, with θ_E and θ_R being the parameters of the line extraction network E and the restoration network R, respectively; x is the input sketch image; y^* is the target line drawing; y^*_E is the target line drawing with gaps; and λ is a weighting parameter.

The objective function is a λ-weighted sum of two pixel-wise Mean Squared Error (MSE) loss functions. The first term is the loss for the joint output of both networks, to output clean restored line drawings. On the other hand, the second term is only for the output of the line extraction network, in order to extract clean lines, albeit with gaps.

Note that the second loss can only be used with the synthetic data, as we need the ground truth non-inpainted lines y^*_E. For the Leonardo da Vinci dataset, we have the y^* but not y^*_E; thus we use a single loss by setting $\lambda = 0$. An overview of the training approach can be seen in Fig. 9.

Training is done using the ADADELTA [32] variant of stochastic gradient descent, which does not require any hyperparameters. To allow training of such deep models, we apply batch normalization [12] layers between the convolutional layers and the ReLU activation functions.

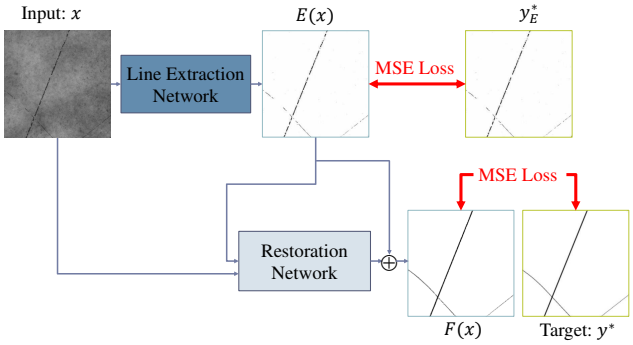
For data augmentation, we randomly scale the image by a random factor between 0.9 and 2.2, perform random rotation, and finally crop the image to a constant image patch size. Each training batch consists of patches extracted from either the Leonardo da Vinci dataset, random synthetic images, or manually created synthetic images, with images taken from the Leonardo da Vinci dataset being three times more likely than either of the synthetic datasets.

4 Results

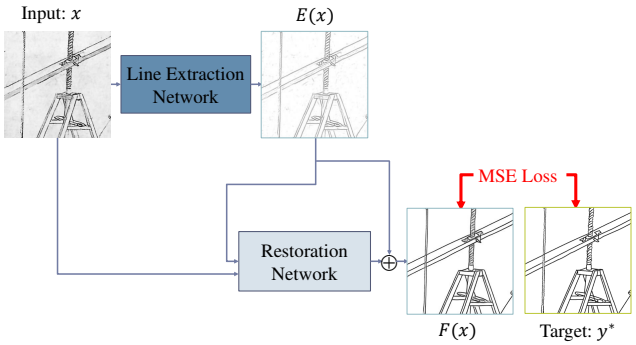
We train our model using the Leonardo da Vinci dataset and synthetic line data by using patches of 320×320 pixels in size and a batch size of 3. For our objective function, we use $\lambda = 1$ for synthetic data and $\lambda = 0$ for the Leonardo da Vinci dataset, and train for 560,000 iterations.

4.1 Comparison with Existing Approaches

We compare with existing approach of [23] on challenging Leonardo da Vinci images from our test set. Examples are shown in Fig. 10. We can see how our approach



Training flow (Synthetic data)



Training flow (Leonard da Vinci's data)

Fig. 9 Overview of our training approach. For synthetic data, we are able to apply two losses during training, while for real Leonardo da Vinci drawings, we resort to a single loss. Both the line extraction network and the line restoration network are trained jointly.

is able to extract clean line drawings from the images while [23] is unable to handle the challenging variation in background and shading. This is caused by being trained exclusively on synthetic data,

4.2 Perceptual User Study

In order to quantitatively evaluate our model, we perform a perceptual user study against a strong baseline. In particular, we retrain the model of [23] on our dataset as the baseline, which significantly outperforms the original model. We evaluate on the 10 images from our Leonardo da Vinci test set. Examples can be seen in Fig. 11.

For the user study, we show each user a random input sketch image and the result of processing it with both our approach and the baseline, and ask them to choose the better clean line drawing. The order in which each image is shown is randomized. Our approach is preferred 72.7% of the time over the baseline approach, which indicates the strength of splitting the problem into a line extraction and line restoration task. A total of 11 users participated in the study and the result is shown in Table 2.

Table 2 Results of our user study. Users were shown images processed by the baseline and our approach in random order and told to indicate which result was a better clean line drawing.

	Baseline	Ours
vs Baseline	-	27.3
vs Ours	72.7	-

4.3 Qualitative Evaluation

We provide additional qualitative examples in Fig. 12 on the challenging sketches taken from a diversity of sources. We can see how our approach is able to handle a diversity of complicated drawings with minimal error. In particular, it can process old schematic sketches drawn by Leonardo da Vinci, while at the same time obtaining good results on drawings with lines that are nearly invisible to humans, such as the water lilies or the Egyptian drawing.

4.4 Inpainting Line Drawings

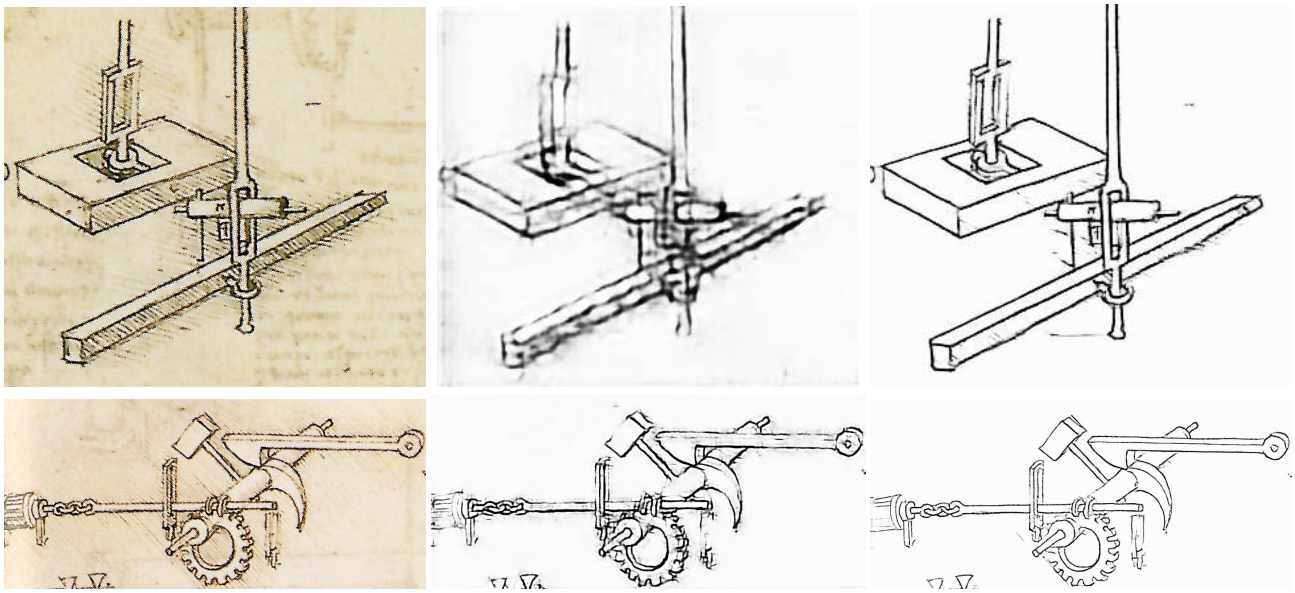
We also evaluate our approach on inpainting line drawings. In particular, we show results for several examples taken from [23]. Our approach an accurately complete the line drawings despite having large amounts of gaps, obtaining similar performance to that of [23].

4.5 Limitations

While we have shown our approach to handle a wide diversity of rough line drawings on paper, many historical line drawings have been done on other mediums, such as rock engravings. As our approach is trained on real rough sketches drawn on paper, in addition to synthetic data that simulates paper, it is unable to perform as well on the stone engravings as shown in Fig. 14. Simulating stone engravings and using annotated real stone engraving images would be a way to improve results in this case.

5 Conclusions and Discussion

We have presented an approach for restoration of deteriorated line drawings, in which the drawings are extracted and then restored, thus splitting the restoration



Input [23] Ours

Fig. 10 Comparison with the approach of [23] on challenging Leonardo da Vinci rough sketches. We can see how our model is able to extract more clean and well-defined lines. Images by Leonardo da Vinci and in the public domain.

problem into two simple sub-problems. We additionally have presented a new dataset based on sketches by Leonardo da Vinci with ground truth annotations that, in combination with our synthetic data, allows training our framework in an end-to-end fashion to obtain high performance on a diversity of images. We have compared our approach with a strong baseline trained on our data with a user study, and found that users prefer our approach 72.7% of the time.

Although we focus in this work on extracting clean raster line drawings from rough sketch inputs, it is possible, with further post-processing, to convert these into vector images. As an example, we show an output processed with the approach of [8] in Fig. 15. Converting the resulting clean images into vector graphics allows for both large compression and easier editing of the line drawings. We believe the next step would be to integrate the line restoration of our approach with vectorization approaches into a single framework.

6 Compliance with Ethical Standards

The authors declare no conflicts of interests. This work was partially supported by JST ACT-I Grant Number JPMJPR16U3, JST ACT-I Grant Number JPMJPR16UD, and JST CREST Grant No. JPMJCR14D1.

References

1. Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.B.: PatchMatch: A randomized correspondence algorithm for structural image editing. ACM Transactions on Graphics (Proc. of SIGGRAPH 2009) 28(3), 24:1–24:11 (2009)
2. Bertalmio, M., Sapiro, G., Caselles, V., Ballester, C.: Image inpainting. In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, pp. 417–424 (2000)
3. Chiang, J.Y., Tue, S., Leu, Y.: A new algorithm for line image vectorization. Pattern recognition 31(10), 1541–1549 (1998)
4. Cugini, U., Ferri, G., Mussio, P., Protti, M.: Pattern-directed restoration and vectorization of digitized engineering drawings. Computers & graphics 8(4), 337–350 (1984)
5. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image denoising by sparse 3-d transform-domain collaborative filtering. IEEE Transactions on image processing 16(8), 2080–2095 (2007)
6. Darabi, S., Shechtman, E., Barnes, C., Goldman, D.B., Sen, P.: Image Melding: Combining inconsistent images using patch-based synthesis. ACM Transactions on Graphics (Proc. of SIGGRAPH 2012) 31(4), 82:1–82:10 (2012)
7. Dong, C., Loy, C.C., He, K., Tang, X.: Learning a deep convolutional network for image super-resolution. In: European Conference on Computer Vision, pp. 184–199. Springer (2014)
8. Favreau, J.D., Lafarge, F., Bousseau, A.: Fidelity vs. simplicity: a global approach to line drawing vectorization. ACM Transactions on Graphics (TOG) 35(4), 120 (2016)
9. Gu, S., Zuo, W., Xie, Q., Meng, D., Feng, X., Zhang, L.: Convolutional sparse coding for image super-resolution. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1823–1831 (2015)
10. Han, C.C., Fan, K.C.: Skeleton generation of engineering drawings via contour matching. Pattern recognition 27(2), 261–275 (1994)

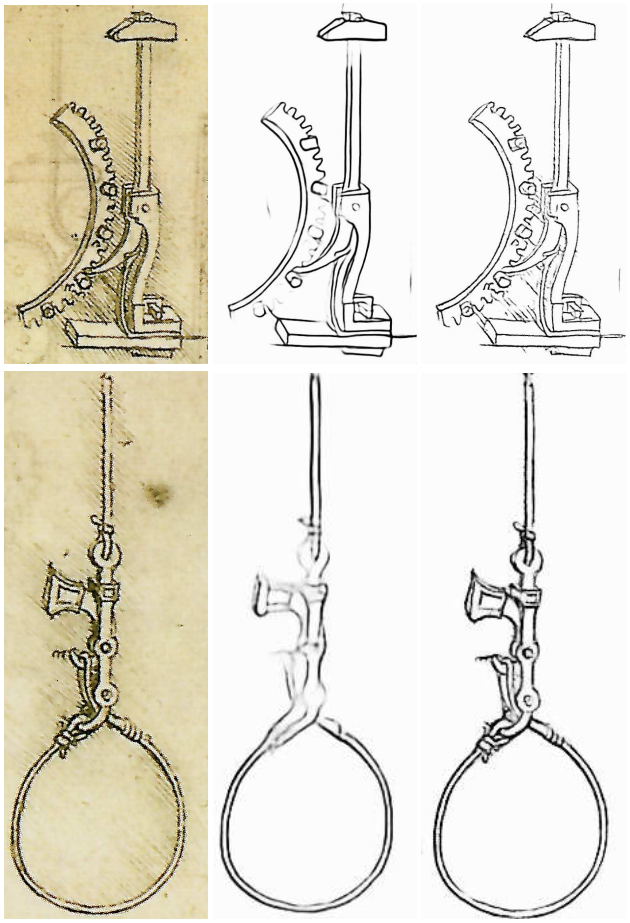


Fig. 11 Examples taken from our user study. The baseline consists of the model of [23] trained on our data. We can see that the baseline tends to erase and thus miss many important lines of the input images. Images by Leonardo da Vinci and in the public domain.

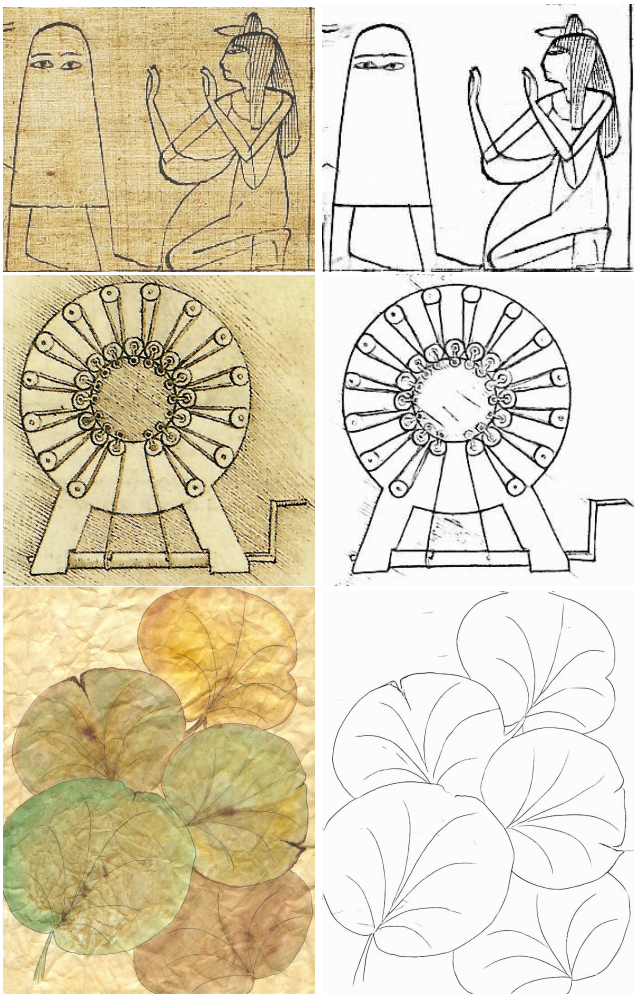


Fig. 12 Qualitative results on a diversity of images. The water lily drawing in the bottom-left is copyrighted by Alan (user sunsetsailor on Flickr) and licensed under CC-by 2.0. The other two images are in the public domain.

11. Iizuka, S., Simo-Serra, E., Ishikawa, H.: Globally and Locally Consistent Image Completion. ACM Transactions on Graphics (Proc. of SIGGRAPH 2017) 36(4), 107:1–107:14 (2017)
12. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning (2015)
13. Jain, V., Seung, S.: Natural image denoising with convolutional networks. In: Advances in Neural Information Processing Systems, pp. 769–776 (2009)
14. Janssen, R.D., Vossepoel, A.M.: Adaptive vectorization of line drawing images. Computer vision and image understanding 65(1), 38–56 (1997)
15. Komodakis, N., Tziritas, G.: Image completion using efficient belief propagation via priority scheduling and dynamic pruning. IEEE Transactions on Image Processing 16(11), 2649–2661 (2007)
16. Li, C., Liu, X., Wong, T.T.: Deep extraction of manga structural lines. ACM Trans. Graph. 36(4), 117:1–117:12 (2017)
17. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Computer Vision and Pattern Recognition, pp. 3431–3440 (2015)
18. Mao, X., Shen, C., Yang, Y.B.: Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In: Advances in neural information processing systems, pp. 2802–2810 (2016)
19. Milanfar, P.: A tour of modern image filtering: New insights and methods, both practical and theoretical. IEEE Signal Processing Magazine 30(1), 106–128 (2013)
20. Osher, S., Burger, M., Goldfarb, D., Xu, J., Yin, W.: An iterative regularization method for total variation-based image restoration. Multiscale Modeling & Simulation 4(2), 460–489 (2005)
21. Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., Efros, A.: Context encoders: Feature learning by inpainting. In: Computer Vision and Pattern Recognition (2016)
22. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D: nonlinear phenomena 60(1-4), 259–268 (1992)
23. Sasaki, K., Iizuka, S., Simo-Serra, E., Ishikawa, H.: Joint Gap Detection and Inpainting of Line Drawings. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

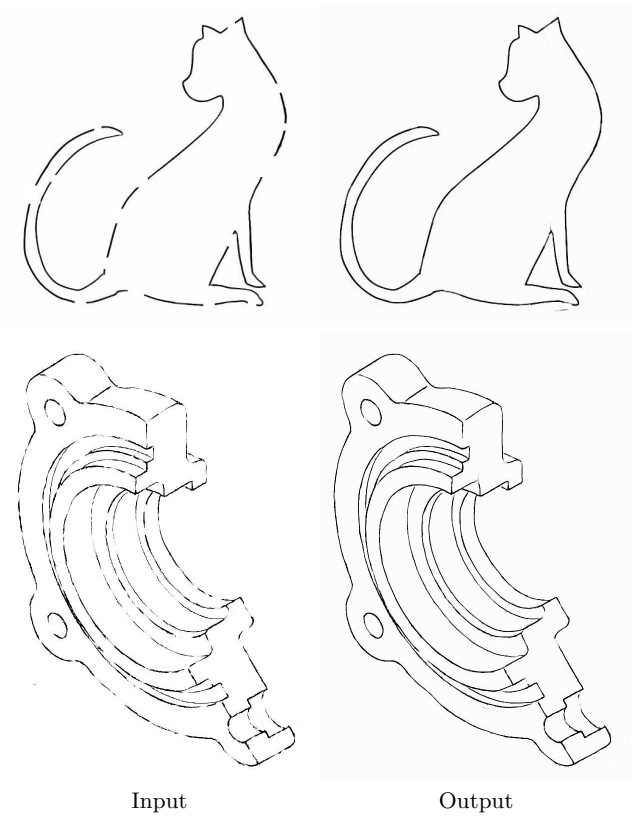


Fig. 13 Example results on the line inpainting task of [23]. The objective is to complete lines that have parts missing. Unlike the traditional inpainting problem, masks of the areas to complete are not provided: the algorithm must both detect and complete the missing segments.

Fig. 14 Failure case example of restoring stone engraved hieroglyphics. Due to the shadows cast by the engraving and deep cracks in the stone, important lines are lost, while unnecessary lines are conserved. The image is in the public domain.

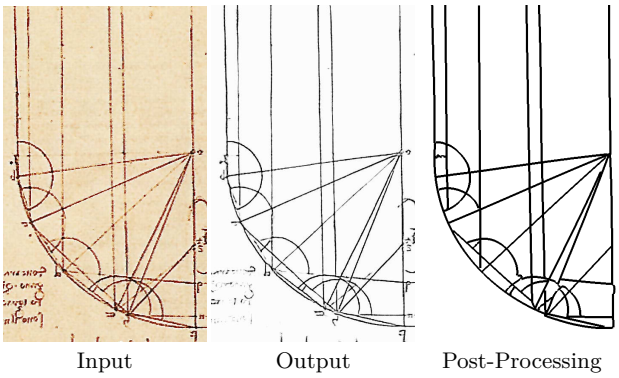
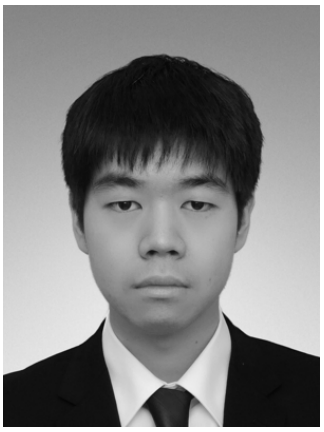


Fig. 15 Example of vectorization post-processing of our restored line drawing with the approach of [8]. Image is by Leonardo da Vinci and is in the public domain.



Kazuma Sasaki received his B.S. and M.S. degrees in Engineering from Waseda University, Japan, in 2016 and 2018. His research interests are image editing, illustration, and machine learning.

24. Simakov, D., Caspi, Y., Shechtman, E., Irani, M.: Summarizing visual data using bidirectional similarity. In: Computer Vision and Pattern Recognition, pp. 1–8 (2008)
25. Simo-Serra, E., Iizuka, S., Ishikawa, H.: Mastering Sketching: Adversarial Augmentation for Structured Prediction. Transactions on Graphics (TOG) 37(1) (2018)
26. Simo-Serra, E., Iizuka, S., Sasaki, K., Ishikawa, H.: Learning to Simplify: Fully Convolutional Networks for Rough Sketch Cleanup. ACM Transactions on Graphics (In Proc. of SIGGRAPH 2016) 35(4) (2016)
27. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations (2015)
28. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th international conference on Machine learning, pp. 1096–1103. ACM (2008)
29. Wang, Z., Yang, Y., Wang, Z., Chang, S., Yang, J., Huang, T.S.: Learning super-resolution jointly from external and internal examples. IEEE Transactions on Image Processing 24(11), 4359–4371 (2015)
30. Wexler, Y., Shechtman, E., Irani, M.: Space-time completion of video. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(3), 463–476 (2007)
31. Xie, J., Xu, L., Chen, E.: Image denoising and inpainting with deep neural networks. In: Advances in neural information processing systems, pp. 341–349 (2012)
32. Zeiler, M.D.: Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701 (2012)

Satoshi Iizuka received his B.S., M.S., and Ph.D. degrees in Engineering from the University of Tsukuba, Japan, in 2010, 2012, and 2015 respectively. Since April 2015, he has been a junior researcher (assistant professor) at Waseda University. His research interests are in computer graphics, computer vision, and machine learning. He received the Innovative Technologies 2016 Special Award for Culture from Japan's Ministry of Economy, Trade and Industry.

Edgar Simo-Serra is currently a junior researcher at Waseda University He obtained his Industrial Engineering degree from BarcelonaTech in 2011 and his Ph.D. in 2015 from the same university. His general research interests are in the intersection of computer vision, computer graphics, and machine learning with applications to large-scale real world problems.

Hiroshi Ishikawa is a full professor at Waseda University. He received the B.S. and M.S. degrees in mathematics from Kyoto University, and the Ph.D. degree in computer science from New York University. He received the IEEE Computer Society Japan Chapter Young Author Award and the MIRU Nagao Award. He is on the Editorial Board of IJCV and is an Associate Editor in Chief of IPSJ TCVA. He has also been an Associate Editor of IEEE TPAMI, a General Chair for IAPR MVA, and an Area Chair for CVPR, ICCV, and ACCV. He is also appointed a Program Chair for ACCV2020.