DanbooRegions: Illustration and Cartoon Region Dataset Annotated by Real-life Artists

- Appendix -

Anonymous ECCV submission

A Dataset Screenshots

We provide several screenshots of the dataset as in Fig. 1-3.

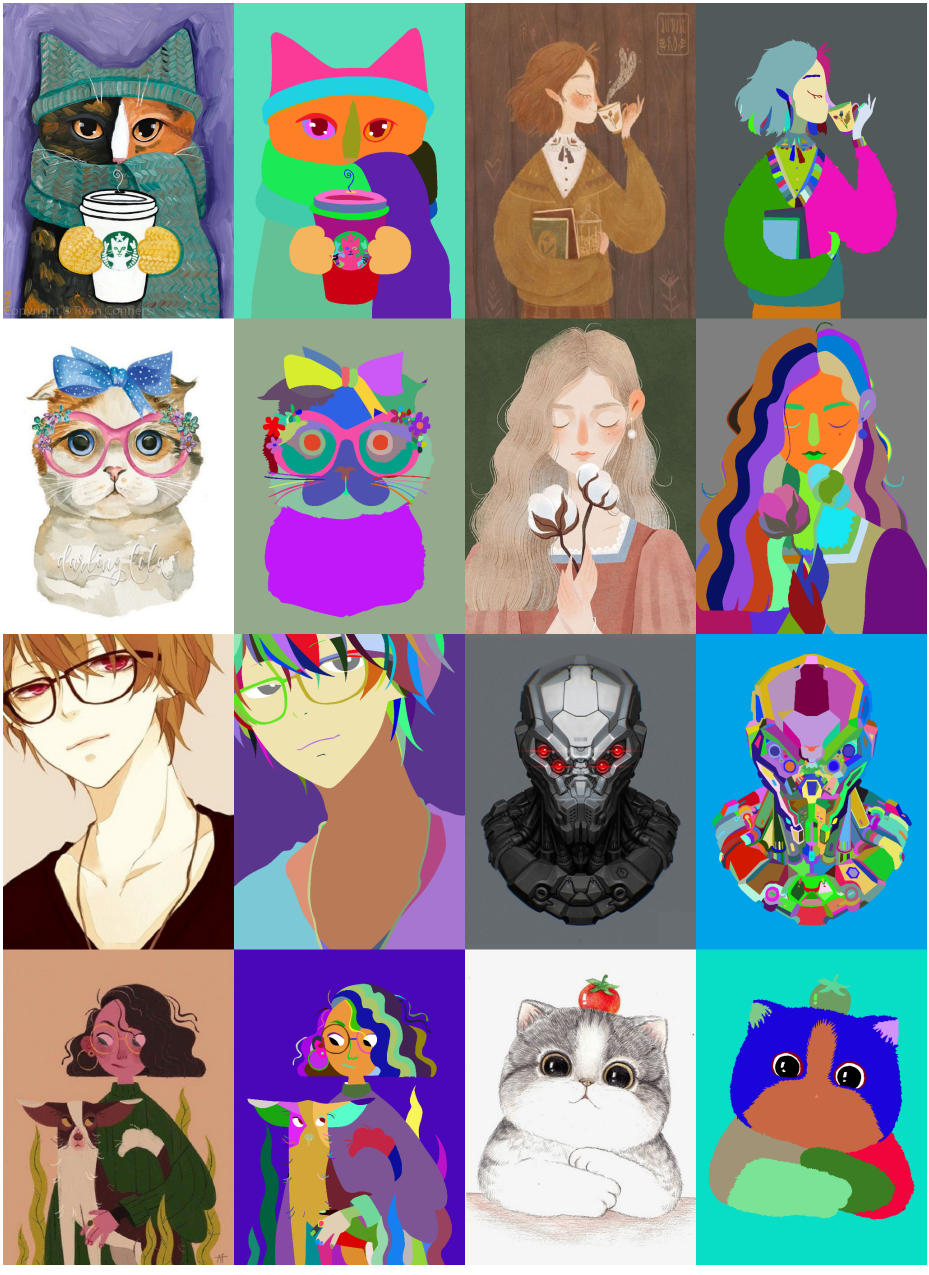

Fig. 1. Dataset screenshots.

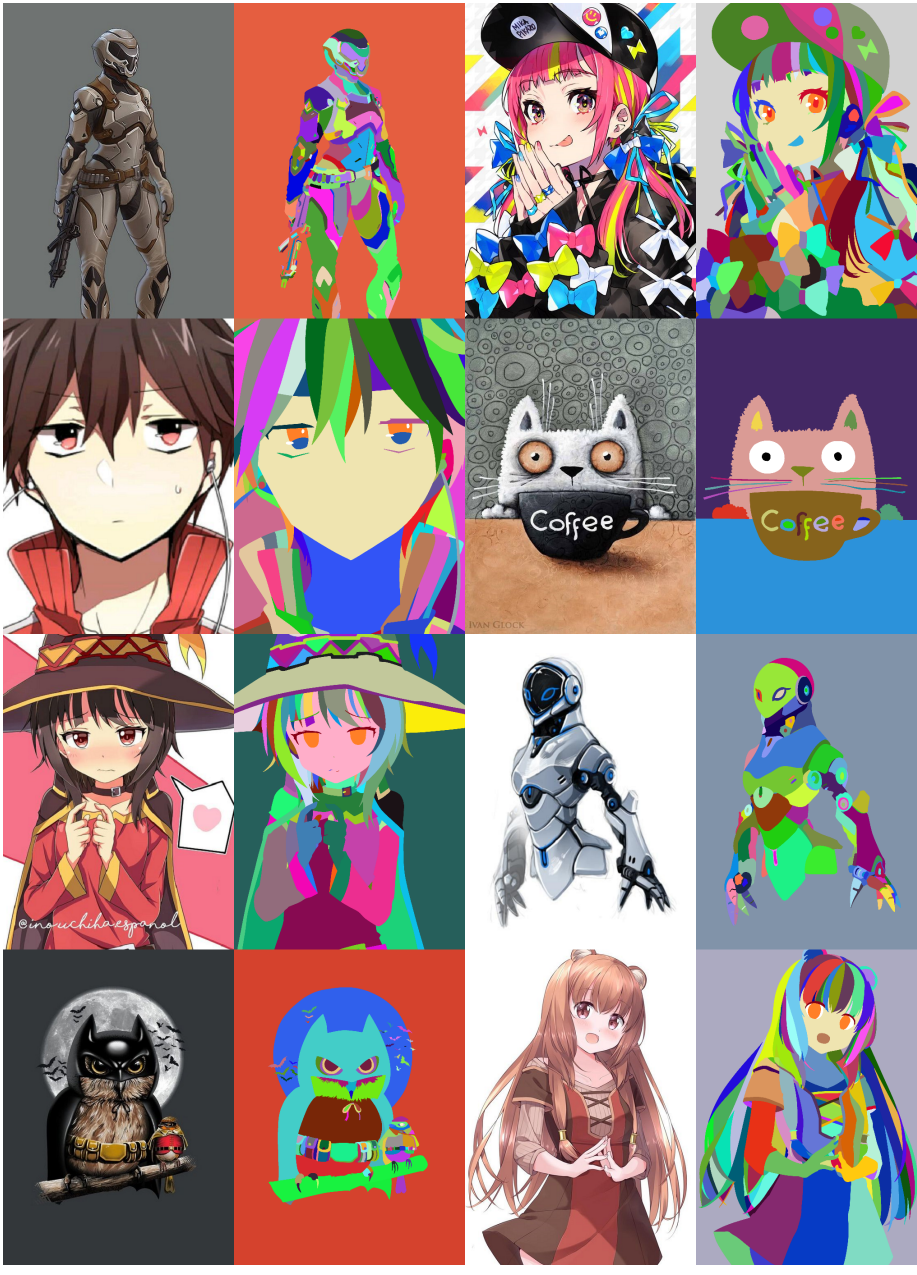

Fig. 2. Dataset screenshots.

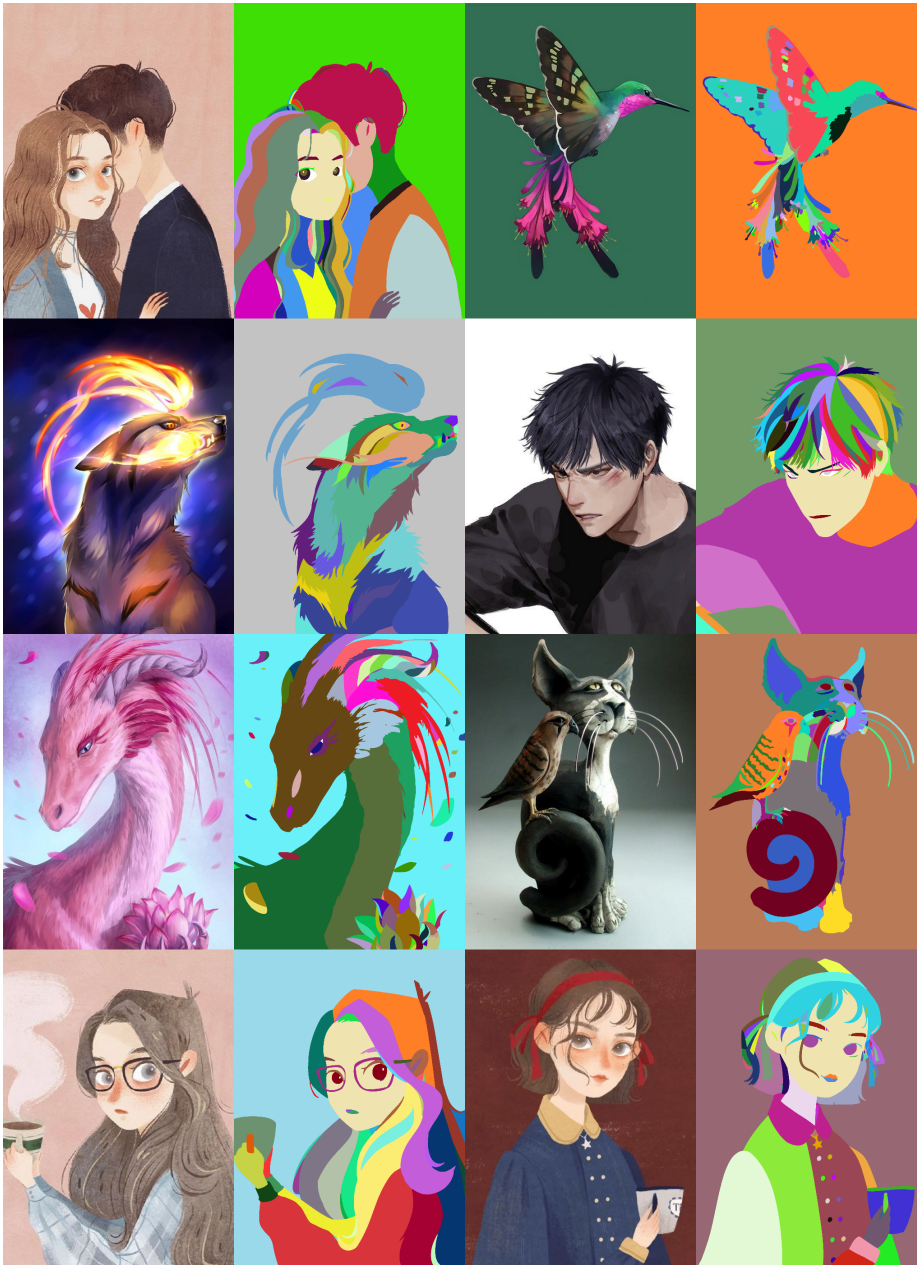

Fig. 3. Dataset screenshots.

B Implementation Details: Conversion between Region and Normal

We give a detailed description of the *normal-from-region* and *region-from-normal* conversion methodologies mentioned in the main paper. These two conversions are implemented using a set of widely used and technically rooted morphology transform algorithms. The below algorithms are not proposed in this paper. The below descriptions are prepared for readers who are not familiar with the translation between region maps and normal maps. The correctness, effectiveness, and applicability of the below algorithms are already discussed in related previous literatures. The below algorithms are also widely used in instance segmentation fields.

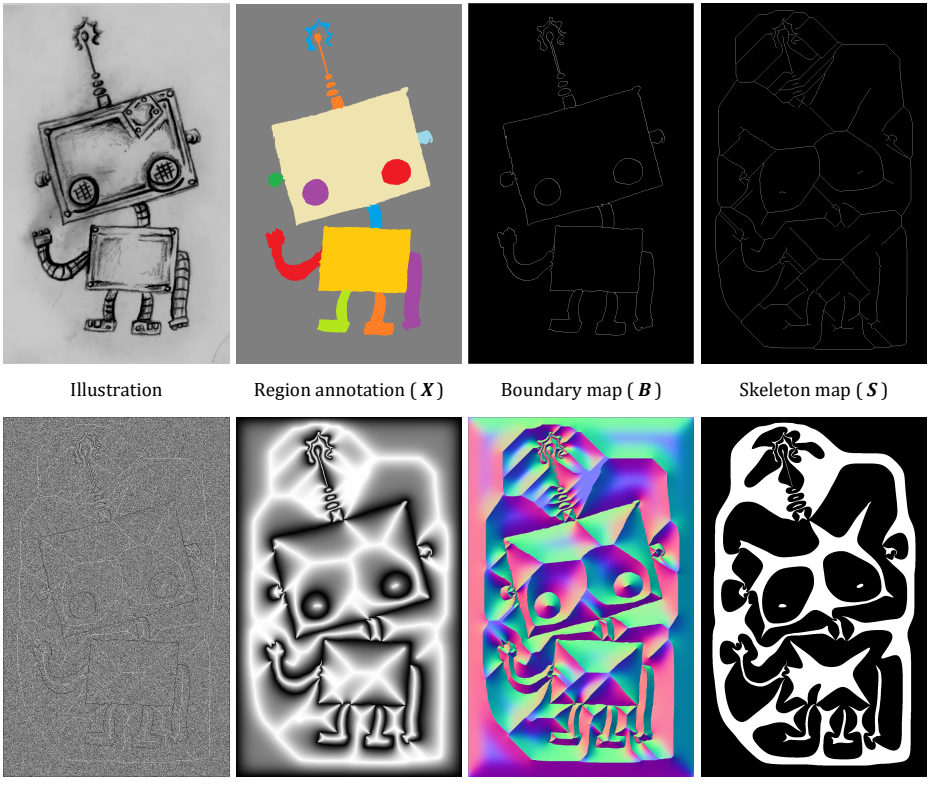

Fig. 4. Visualization of involved maps in the region-normal transform algorithms.

B.1 Normal-from-region transform

As shown in Fig. 4, given an input region annotation $\boldsymbol{X}$, the *normal-from-region* transform is aimed at produce a transformed map $\boldsymbol{Y}$, which is a concatenation of a normal map $\boldsymbol{N}$ and a watershed marker $\boldsymbol{W}$. The transform includes the following steps:

Step 1: Compute the boundary of the region annotation $\boldsymbol{X}$ to get the boundary map $\boldsymbol{B}$.

Step 2: Compute the Zhang-Sun-Skeleton [1] of the region annotation $\boldsymbol{X}$ to get the skeleton map $\boldsymbol{S}$.

Step 3: Initialize a random field $\boldsymbol{D_{rand}}$. If one pixel belongs to boundary in $\boldsymbol{B}$, that pixel will be marked as zero. If one pixel belongs to skeleton in $\boldsymbol{S}$, that pixel will be marked as one. The remaining pixels are be marked with random value sampled from random uniform distribution between zero and one.

Step 4: Optimize the random field $\boldsymbol{D_{rand}}$ to get the displacement map $\boldsymbol{D}$. We uses a routine Gaussian energy

$$\begin{aligned} E_{\text{displacement}} = & \sum_{p} ||(\boldsymbol{D_{rand}})_p - (g(\boldsymbol{D_{rand}}))_p||_1 \\ & + \sum_{i \in \{i | \boldsymbol{B}_i = 1\}} ||(\boldsymbol{D_{rand}})_i - 0||_1 \\ & + \sum_{j \in \{j | \boldsymbol{S_j} = 1\}} ||(\boldsymbol{D_{rand}})_j - 1||_1 \end{aligned} \tag{1}$$

where p, i, and j are possible pixel positions, $g(\cdot)$ is a Gaussian (sigma is 1.0) filter, and $||\cdot||_1$ is the L1 Euclidean distance. This energy can be flexibly solved by gradient descent.

Step 5: Compute the normal map $\boldsymbol{N}$ using the displacement map $\boldsymbol{D}$. We use a standard *normal-from-height* [2] algorithm to achieve the normal.

Step 6: Compute the watershed marker $\boldsymbol{W}$ by binarizing the displacement map $\boldsymbol{D}$. We use the threshold 0.618.

Step 7: Concatenate the normal map $\boldsymbol{N}$ and the watershed marker $\boldsymbol{W}$ into the final output $\boldsymbol{Y}$.

B.2 Region-from-normal transform

Given the concatenated $\boldsymbol{Y}$, we split it into the normal map $\boldsymbol{N}$ and the watershed marker $\boldsymbol{W}$. After that, we run the watershed [3] with the marker $\boldsymbol{W}$ filling the map $\boldsymbol{N}$. The results are the reconstructed regions. Note that when the marker $\boldsymbol{W}$ is predicted from neural networks, *e.g.*, the Coarse CNN, we may use morphology methods to remove some possible noise. In particular, we enforce all white regions in $\boldsymbol{W}$ to be bigger than 64 connected pixels.

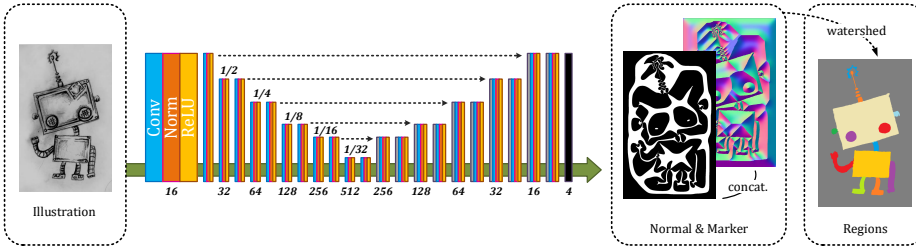

Fig. 5. Coarse CNN. Neural network architectures for generating coarse region annotations.

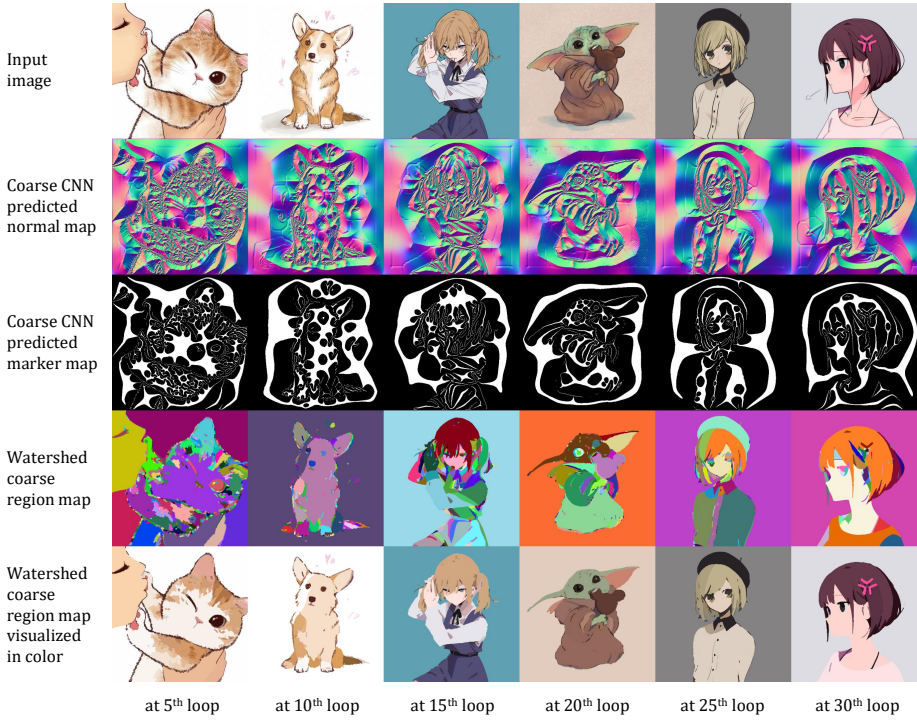

Fig. 6. More examples of the predictions from Coarse CNN and the generated coarse region maps during the annotating loops as mentioned in main paper.

C Implementation Details: Coarse CNN

We illustrate the coarse annotation generator architectures as shown in Fig. 5. All convolution layers use 3×3 filters. Down-sampling is achieved using average pooling and Up-sampling is achieved using bilinear interpolation. These neural networks are optimized using Adam ($lr = 1e{-}3$ and other parameters are default as in [4]). The inputs of the model is RGB images, and the outputs are the four channels of the concatenated normal map (three channels) and watershed marker (one channel). This model is trained with L2 loss (or called the mean squared error) and adversarial loss (using Pix2Pix's patch discriminator [5]). The predicted marker map is binarized with a threshold at 0.5. The outputs of these neural networks can be converted to regions using watershed as mentioned before.

This model can be trained on any {image, region map} pairs by translate these pairs into {original image, concatenated normal map and watershed marker map} pairs, and then the estimation can also be translated back to regions using the aforementioned *region-from-normal* method.

One notice is that this approach "translate the regions to watershed normals" is a routinely used approach in region processing and instance segmentation. The effectiveness, correctness, and performance of this approach have already be extensively discussed in many "normal + watershed" or "distance transform + watershed" instance segmentation literatures, *e.g.*, "Deep Watershed Transform for Instance Segmentation" [6].

More Estimated Coarse Region Maps We present more estimated region maps in Fig. 6. We include the visualization of the estimated normal map, estimated watershed marker, and generated coarse regions. These results are from different loops in our human-in-the-loop workflow as mention in main paper.

D Implementation Details: Cartoon Tracking

We detail the implementation of the Global Optimal Toon Tracking (GOTT) [7], as mentioned in the main paper. We do not make modifications to GOTT's net-flow correspondence matching algorithms. We only change its backend region segmentation method. Originally, GOTT uses a edge-detection-based region segmentation method. We replace that segmentation method with our leaning-based region segmentation backend.

Using the above *region-from-normal* approach, we translate our dataset into {original image, concatenated normal map and watershed marker map} pairs. These pairs can be used to train image-to-image translation methods. In particular, we directly train a Pix2PixHD [8] for the GOTT application. After the training, the estimated normal maps and watershed marker maps can be translated back to regions using the above *normal-from-region* approach.

One important notice is that, in the GOTT application, we use a special data augmentation method to augment the high-frequency domain of the images. This

is because the GOTT places high demands on the region closure and structure but cartoon images often have broken regions (as we have discussed in the main paper). Although our artists have already provided manually closed regions, we find that Pix2PixHD (and some other possible architectures) is not always able to learn those closure. In many cases, the CNN tends to over-fit to the high-frequency patterns in the input images, and whatever we train it, a broken region is always broken as long as the input illustration region is broken. We have tried many strategies, and fortunately find that we can use a high-frequency domain data augmentation method to get rid of this limitation. The implementation is very easy. We only need to apply a Bilateral Filter [9] to the training input image.

Also, we have a discovery that we should not use fixed parameters in the Bilateral Filter. This is because CNNs are very "powerful", and they can even solve the blurring kernels and thus invalid this data augmentation, causing over-fitting problems where broken regions are still broken. Therefore, when training the Pix2PixHD, we use randomized parameters in the Bilateral Filter to augment the input images. In particular, we apply a random number in $U(20, 1000)$ to the Bilateral spacial sigma, and $U(20, 1000)$ to the Bilateral color sigma. Some results are visualized in Fig. 7 (the low&high-frequency augmentation). In test time, we use a simple Gaussian filter (sigma is 7.0) to pre-process the input image in order to prevent the CNN from being "stuck" in the high-frequency constitutes (those high-frequency broken lines and regions).

After the training, we directly translate the Pix2PixHD predicted normal and marker into regions. Besides, we find that the GOTT's shape metrics can work a bit better if the region boundary is less "noisy" or a bit more smoother. In order to make the estimated region boundary a bit more smoother (regions produced by watershed is not very smooth in most cases), we use the method [10] to simplify the topology of our region boundary. Finally, our region is used as the initial segmentation in GOTT to replace its original segmentation backend.

It is notable that this detailed implementation is only a possible approach to make use of the dataset. We are not comparing this implementation "against" the original GOTT and any other methods. More importantly, this implementation is only a strong evidence to verify that our dataset can be used in this task. And, this implementation can achieve some beneficial results. It facilitates further researches to develop more reliable frameworks, conduct more adequate ablative study, and organize more rooted comparisons, so as to make better use of our presented dataset.

E Implementation Details: Cartoon Intrinsic Images

We detail our implementation of the L1 smoothing [11]. This implementation makes use of the dataset to remove illumination inferences from the input cartoon image. Note that although our dataset enables a reliable illumination separation, in particular, several training and implementations strategies are still needed to achieve a practically usable framework.

Fig. 7. Data augmentation methods that are used in mentioned applications.

The first strategy is to use an adversarial augmented generator. In particular, we translate our dataset into {original image, concatenated normal map and watershed marker map} pairs and train a Pix2PixHD [8] generator. In our study, using adversarial generator yields a bit better results. After the training, the estimated normal maps and watershed marker maps can be translated back to regions using the above *normal-from-region* approach.

The second strategy is to use the luminance-chroma augmentation. This is because eliminating the shadow boundary is very difficult for CNNs. In most cases, the trained CNN will always preserve all shadow boundaries. Even though our dataset have provided the data to eliminate the shadow edge, the CNN is not always able to learn these data. Sometimes the CNN can over-fit to the high-frequency cues of the input image, and the shadow regions tend to be preserved in the final region map. In order to avoid such over-fitting, we use a special data augmentation called luminance-chroma augmentation. This augmentation is very easy to implement. We only need to convert the RGB image into Lab image, and then randomly reduce the contrast of the L channel. In particular, we reduce the L contrast by random scalar $U(0.1, 0.9)$, and then translate the Lab image back to RGB image. In this way, the shadow edge becomes less salient in the CNN input. Some results are visualized in Fig. 7 (the Luminance-chroma augmentation). In test time, we use a fixed scalar 0.5 to reduce the L contrast.

The third strategy is to use a relatively higher piece-wise weight in L1 smoothing. Because our segmentation is more adequate than L1 smoothing original backend (the original super-pixel-based segmentation), we can set a relatively larger region-wise weight to the overall smoothing. In particular, we use a lambda of 5.0 (the original lambda is 0.5). This can result in more adequate region-wise smoothing. And, once the shadow edge is eliminated from the region boundary, those shadow can be then adequately removed.

The forth strategy is to enable the multiple channel intrinsic decomposition. The original L1 smoothing formulate the intrinsic problem as the multiplication between a multiple-channel albedo and a single-channel shading. This is not acceptable in artist image. Artists is not always drawing shadow with single color, and it is a must to enable the multiple-channel shading map. It can be implemented easily. We can flexibly compute the shading map using all channels of the L1 smoothed output.

It is notable that this detailed implementation is only a possible approach to make use of the dataset. We are not comparing this implementation "against" the original L1 smoothing and any other methods. More importantly, this implementation is only a strong evidence to verify that our dataset can be used in the cartoon intrinsic decomposition task. And, this implementation can achieve some practically usable results. It facilitates further researches to develop more reliable frameworks, conduct more adequate ablative study, and organize more rooted comparisons, so as to make better use of our presented dataset.

F Implementation Details: Flat Sketch Colorization and Color Cleaning-up

We detail our implementation for the flat sketch colorization strategy. This application is very easy to implement. Given the input sketch, we segment it into regions. Then, we sample the median color in the sketch colorization results for each regions, and then output the flat color map.

The region generator is a Pix2PixHD [8] trained on our region dataset. We translate our dataset into {original image, concatenated normal map and watershed marker map} pairs and train the Pix2PixHD. After the training, the estimated normal maps and watershed marker maps can be translated back to regions using the above *normal-from-region* approach.

For data augmentation, we use the aforementioned "low&high-frequency augmentation" (Fig. 7). This is to encourage the neural networks to learn to reconstruct broken or ambiguous regions, and avoid the neural networks to over-fit the high-frequency edges in the input image. Another strategy is that we use the method [10] to improve the topology of the region boundary. This will make the region boundary looks a bit more smoother. Nevertheless, we are aware of that the post-processing can cause the some possible "edge inconsistency" artifacts (see the paper of LazyBrush for details [12]). In these cases, we can remove this post processing for applicability.

We also allow users to merge some regions in the region map. As mentioned in the main paper, we allow users to draw some "dotted" lines to merge the regions in the neural network estimation. In this way, the region quality can be improved in some practical use cases.

It is notable that this detailed implementation is only a possible approach to achieve flat sketch colorization. We are not comparing this implementation "against" the LazyBrush, PaintsChainer, Style2Paints, GIMP, and any other methods. More importantly, this implementation is only a strong evidence to

verify that our dataset can be used in the flat sketch colorization task. And, this implementation can achieve some beneficial and usable results. It facilitates further researches to develop more reliable frameworks, conduct more adequate ablative study, and organize more rooted comparisons, so as to make better use of our presented dataset.

References

1. Zhang, T.Y., Suen, C.Y.: A fast parallel algorithm for thinning digital patterns. Communications of the ACM (1984)
2. Kender, J.R., Smith, E.M.: 3. In: Shape from Darkness: Deriving Surface Information from Dynamic Shadows. Jones and Bartlett Publishers, Inc., USA (1992) 378–385
3. Neubert, P., Protzel, P.: Compact watershed and preemptive slic: On improving trade-offs of superpixel segmentation algorithms. ICPR (2014)
4. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. Computer Science (2014)
5. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. CVPR (2017)
6. Bai, M., Urtasun, R.: Deep watershed transform for instance segmentation. CoRR **abs/1611.08303** (2016)
7. Zhu, H., Liu, X., Wong, T.T., Heng, P.A.: Globally optimal toon tracking. ACM Transactions on Graphics **35**(4) (July 2016) 75:1–75:10
8. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional gans. CVPR (2018)
9. Tomasi, C., Manduchi, R.: Bilateral filtering for gray and color images. In: Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271), Narosa Publishing House (1998)
10. Bessmeltsev, M., Solomon, J.: Vectorization of line drawings via polyvector fields. ACM Transactions on Graphics **38**(1) (jan 2019) 1–12
11. Bi, S., Han, X., Yu, Y.: An l1 image transform for edge-preserving smoothing and scene-level intrinsic decomposition. ACM Trans. Graph. **34**(4) (July 2015)
12. Sykora, D., Dingliana, J., Collins, S.: LazyBrush: Flexible painting tool for hand-drawn cartoons. Computer Graphics Forum **28**(2) (2009)