Deep Vectorization of Technical Drawings

Vage Egiazarian[1*] [0000−0003−4444−9769], Oleg Voynov[1*] [0000−0002−3666−9166], Alexey Artemov[1] [0000−0001−5451−7492], Denis Volkhonskiy[1] [0000−0002−9463−7883], Aleksandr Safin[1] [0000−0002−5453−1101], Maria Taktasheva[1] [0000−0001−6907−5922], Denis Zorin[2,1], and Evgeny Burnaev[1] [0000−0001−8424−0690]

[1] Skolkovo Institute of Science and Technology, 3 Nobel Street, Skolkovo 143026, Russian Federation
[2] New York University, 70 Washington Square South, New York NY 10012, USA
{vage.egiazarian, oleg.voinov, a.artemov, denis.volkhonskiy, aleksandr.safin, maria.taktasheva }@skoltech.ru, dzorin@cs.nyu.edu, e.burnaev@skoltech.ru
adase.group/3ddl/projects/vectorization

We provide additional details on the neural network architecture and training process in Sections A and B. Details of our postprocessing can be found in Section C. We compare runtimes of the methods in Section D. In Section E we show the performance of [1,2,3] on small patches in comparison to whole images. We show example results of our system for cartoon drawings in Section F. We provide additional comparisons from the ablation study in Section H and additional comparisons with other methods in Section G. In Section I we describe our refinement algorithm in detail.

A Neural Networks architectures

For image cleaning we use U-net [7] encoder-decoder architecture. It consists of blocks of layers, each containing convolutional and batch normalization layers and ReLU activations. We use seven such blocks interleaved with MaxPool downsampling in the encoder, and seven blocks interleaved with nearest neighbor upsampling in the decoder. We connect the blocks of the encoder and decoder with the same resolution of feature maps with skip connections, as in the original U-net.

We build our primitive extraction network from two parts: the encoder consisting of ResNet18 blocks [4], which extracts features from the raster, and the decoder Transformer model [9], which estimates the primitive parameters. The architecture of our primitive extraction network is shown in Figure 1.

We use $n_{\text{res}} = 1$ ResNet18 block with $c = 64$ channels in each convolution. We use $n_{\text{dec}} = 8$ Transformer blocks with 4 heads of multi-head attention and 512 neurons in the last fully-connected layer.

We set the hidden dimensionality of the primitive representations in the Transformer part of the network d_{emb} equal to the number of primitive parameters, 6 for lines and 8 for curves: one for the width, one for the confidence value

* Equal contribution

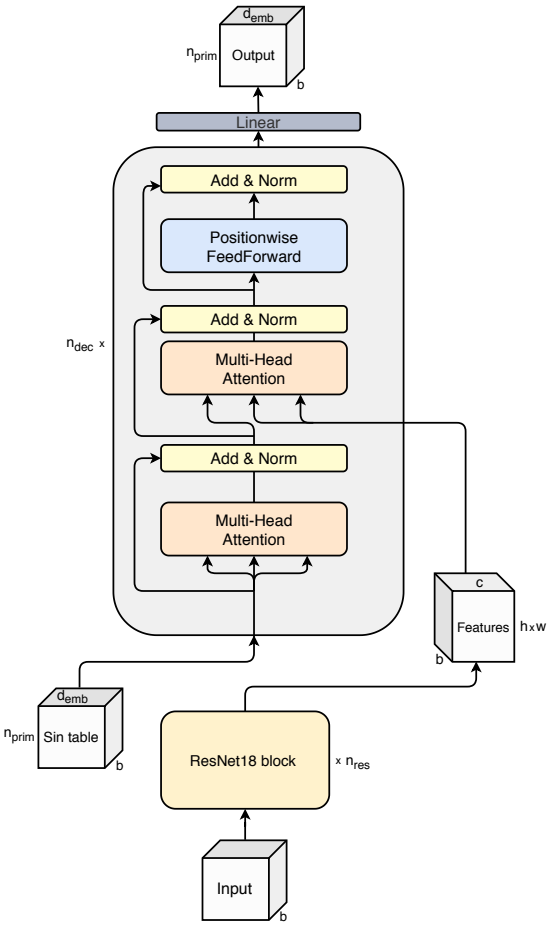

Fig. 1: Architecture of our vectorization network. A batch of b grayscale raster patches is first encoded with the sequence of n_{res} ResNet blocks. Then, c channel feature maps of size $h \times w$ are decoded with a sequence of n_{dec} Transformer blocks. Finally, the output of the last Transformer block is converted with a linear layer into n_{prim} sets of primitive parameters per sample in batch.

and the rest for coordinates of the control points. We keep the other hyper-parameter values the same for lines and curves.

B Training details

We used Pytorch 1.2 for GPU computations and model training [6] and GNU Parallel to speed up the metric calculations [8] for our methods. We trained our models for 15 epochs on ABC dataset and 17 epochs on PFP dataset. The batch size was 128. We used Adam [5] for optimization with a scheduler with

the same hyperparameters as in the original Transformer paper [9]. It took us approximately four days to train each model on a single Nvidia v100. To speed up the training, we pre-calculated all data augmentations, including cropping, and trained our model on this augmented data. First, we split original images into train, validation, and test sets. Then we cropped and augmented images to prevent overfitting.

In Figure 2 and Figure 3 we provide metrics on train and validation sets for patches with size 64×64 from ABC and PFP datasets correspondingly.

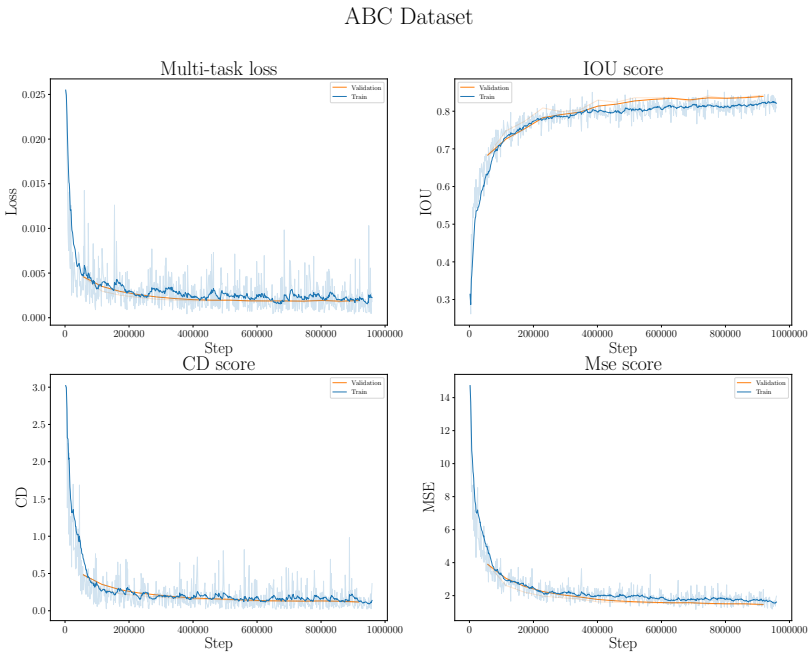

Fig. 2: Metrics and loss function for train and validation on ABC dataset. One step of X-axis represents calculations on a single batch.

C Merging algorithm

For lines we start by building a graph with the primitives as nodes and edges between nodes that correspond to a pair of lines that are close and collinear enough but not almost parallel (Figure 4 (a, b)). Then, we replace the lines in each connected component of the graph with a single least-squares line fit to their endpoints (Figure 4 (c, d)). Finally, we snap the endpoints of intersecting primitives by cutting down the "dangling" ends shorter than a few percent of the total length of the primitive. (Figure 4 (e)).

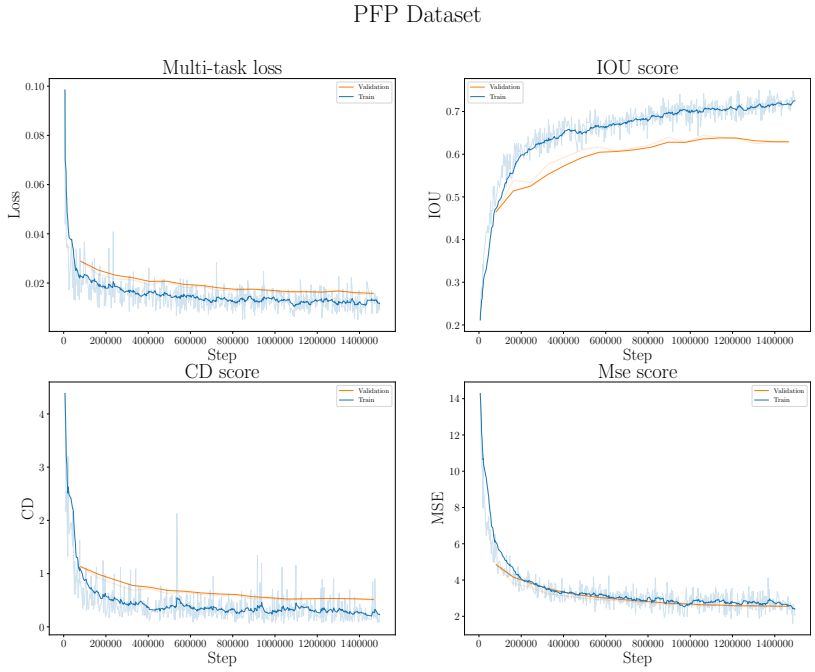

Fig. 3: Metrics and loss function for train and validation on PFP dataset. One step of the X-axis represents computations on a single batch.

For quadratic Bézier curves, we iteratively try to replace pairs of curves with a single one. For each pair of curves $\mathrm{P}(t)$, $t \in [0,1]$, $\mathrm{Q}(s)$, $s \in [0,1]$, we first check if their widths are close (Figure 5 (b)). Then, we check if the "midpoint" (Figure 5 (a)) and the endpoints of the second curve are close to the first one, as illustrated in Figure 5 (c). If all checks are passed, we find a new quadratic Bézier curve $\mathrm{R}(u)$, $u \in [0,1]$ as a least-squares fit to the endpoints and midpoints of the curves in the pair (Figure 5 (d)). Specifically, we minimize the distances between the points

$$\begin{aligned} \mathrm{P}_1 = \mathrm{P}(0), \quad \mathrm{P}_\mathrm{b} = \mathrm{P}(t_\mathrm{b}), \quad \mathrm{P}_3 = \mathrm{P}(1), \\ \mathrm{Q}_1 = \mathrm{Q}(0), \quad \mathrm{Q}_\mathrm{b} = \mathrm{Q}(s_\mathrm{b}), \quad \mathrm{Q}_3 = \mathrm{Q}(1) \end{aligned} \tag{1}$$

and the points on the new curve

$$\begin{aligned} \mathrm{R}(0), \quad \mathrm{R}(t_\mathrm{b} u_{\mathrm{q}1} / t_{\mathrm{q}1}), \quad \mathrm{R}(u_{\mathrm{q}1} / t_{\mathrm{q}1}), \\ \mathrm{R}(u_{\mathrm{q}1}), \quad \mathrm{R}(1 - (1 - s_\mathrm{b})(1 - u_{\mathrm{q}1})), \quad \mathrm{R}(1) \end{aligned} \tag{2}$$

respectively w.r.t. control points of the new curve. Here, t_b and $t_{\mathrm{q}1}$ are the parameter values of P_b and the projection of Q_1 on the first curve, s_b is the parameter value of Q_b on the second curve, $u_{\mathrm{q}1}$ is the parameter value of Q_1 on the new curve. We find the value of $u_{\mathrm{q}1}$ with brute-force search and take the

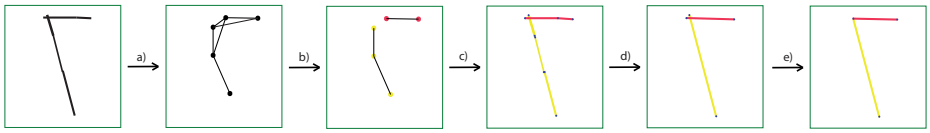

Fig. 4: Our algorithm of line merging: (a) we find close lines, (b, c) we join them in the connected components of the graph, (d) we fit the endpoints of the lines in each connected component with least squares, (e) and finally snap the endpoints of the lines.

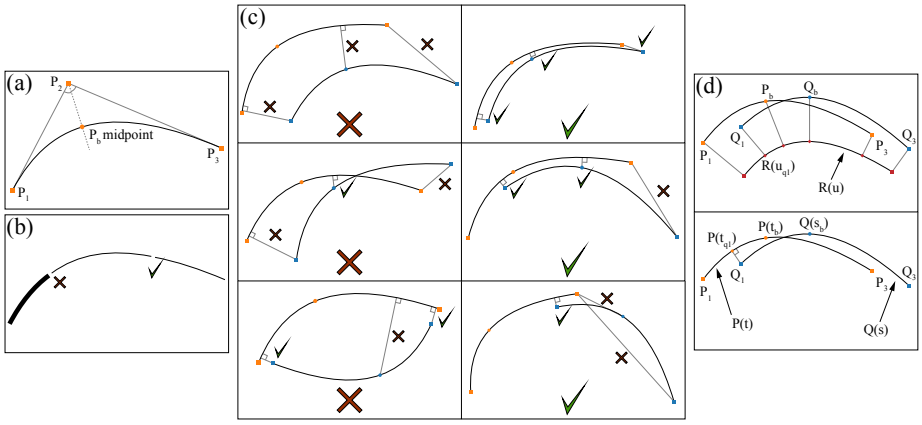

Fig. 5: (a) Our definition of the "midpoint" for quadratic Bézier curve, and (b-d) single step of our algorithm of curve merging: (b) we check that the widths are close, (c) we check that the curves are close, (d) we fit the endpoints and midpoints of the curves with least squares.

best fit. Finally, if the best fit is close enough, we replace the pair of the curves with the fit. We repeat this process until no more pairs allow for a close fit.

D Computation time

Our refinement step is iterative and allows trading longer computation times for more accurate results. In Table 1 we show example computation times for the prior work along with IoU values, and the computation times required by our system to reach similar IoU values.

Our system without the final merging step reaches the same IoU value as CHD [2] in a similar time, and the same IoU values as FvS [3] and PVF [1] in much less time. We note however that none of the methods were optimized for performance and that we run the methods in different environment because of technical requirements.

	IoU, %	Time	#P
CHD [2]	64	10 s	994
FvS [3]	74	17.5 m	433
PVF [1]	91	25 h	43k
Our, w/o final merging	68	35 s	2108
Our, w/o final merging	75	50 s	2106
Our, w/o final merging	91	5.5 m	1502
Our, w/o final merging, converged	92	12 m	1435
Our, with final merging	76	26 m	579

Table 1: IoU, computation time, and number of primitives for the results on the Globe (Figure 9) produced by the prior work, for intermediate results of our method with similar values of IoU, and for our final result.

E Prior work on patches

The main steps of our vectorization system, the primitive extraction network and refinement, operate on small patches of the image, while the methods that we compare with operate on whole images. To demonstrate that our method outperforms these ones not only because of this divide-and-merge strategy, in Figure 6 we show example outputs of these methods applied to small patches in comparison to the respective patches cut from the results on whole images.

The methods of [1,2] produce similar results on small patches and whole images, as expected since they use local operations. The method of [3] produces worse results on patches.

F Generalization to cartoon drawings

Figure 7 shows the results produced by our system on clean cartoon drawings. Here we used the version operating on curves, with the neural networks trained on technical drawings.

Our system produces reasonable results, although the predictions of the primitive extraction network are qualitatively less accurate than in case of technical drawings that we focused on. A proper extension of our system to a different kind of drawings would require (1) the corresponding training dataset for the primitive extraction network, and (2) in case of rough sketches, either a proper training dataset with clean targets for the preprocessing cleaning step, or significant changes of the refinement step.

G Additional results

In this section, we show more qualitative comparisons on test set for both PFP in Figure 8 and ABC in Figure 9 datasets and on real data in Figure 10.

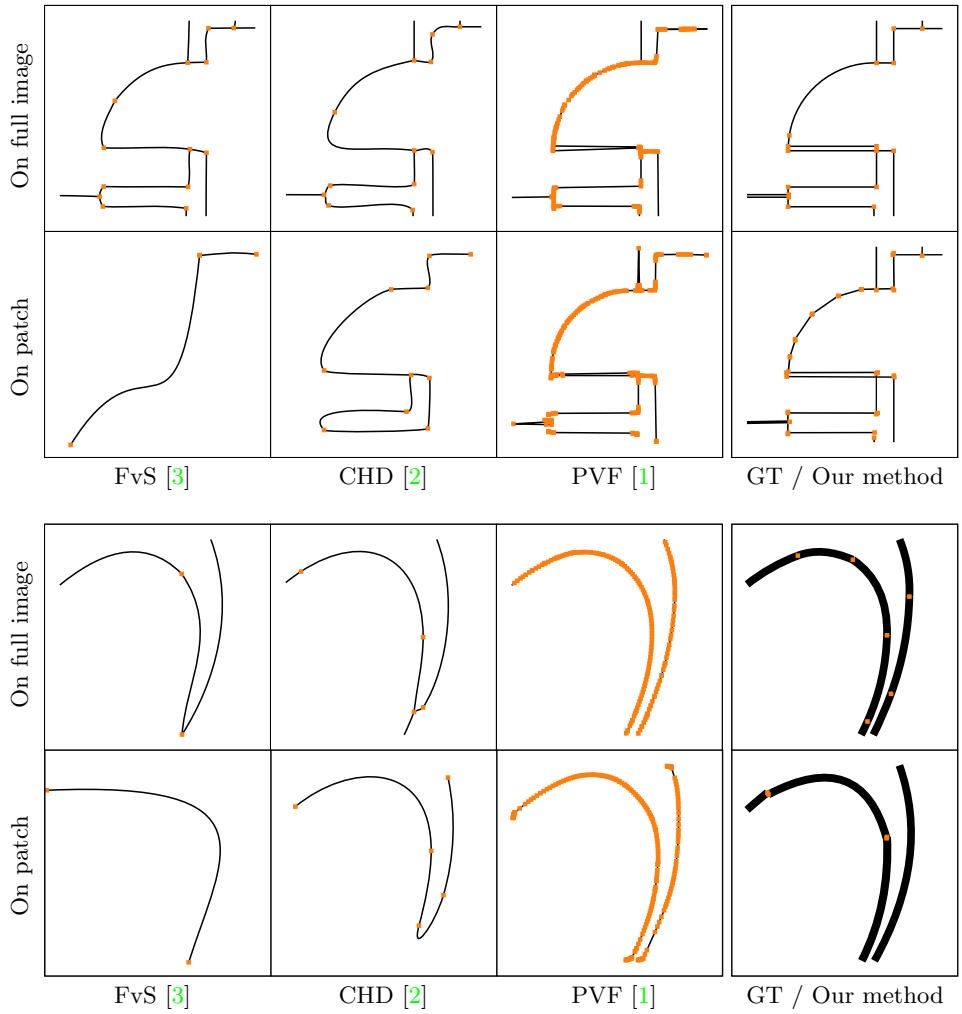

Fig. 6: Results of the prior work on small patches and the respective patches cut from the results on whole images. Endpoints of the primitives are shown in orange. The whole images are shown at the top of Figure 8 and in Figure 6 from the main text.

H Qualitative ablation study

In this section, we show qualitative results obtained using our system with the (a) full model without refinement and post-processing steps, (b) full model without post-processing, (c) full model. You can see this comparison on ABC dataset in Figure 11 and Figure 12.

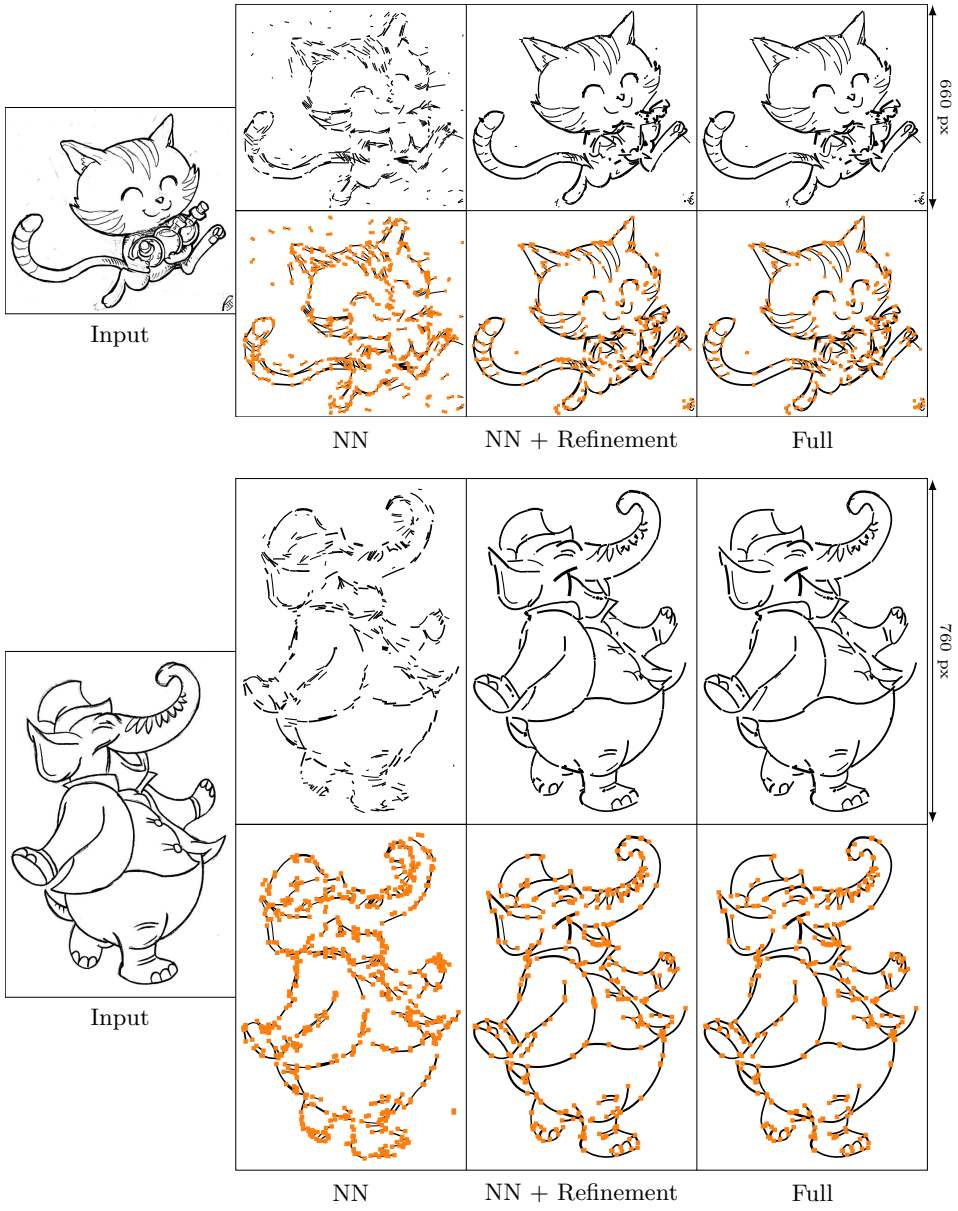

Fig. 7: Qualitative results of our system on clean cartoon drawings. Endpoints of primitives are shown in orange.

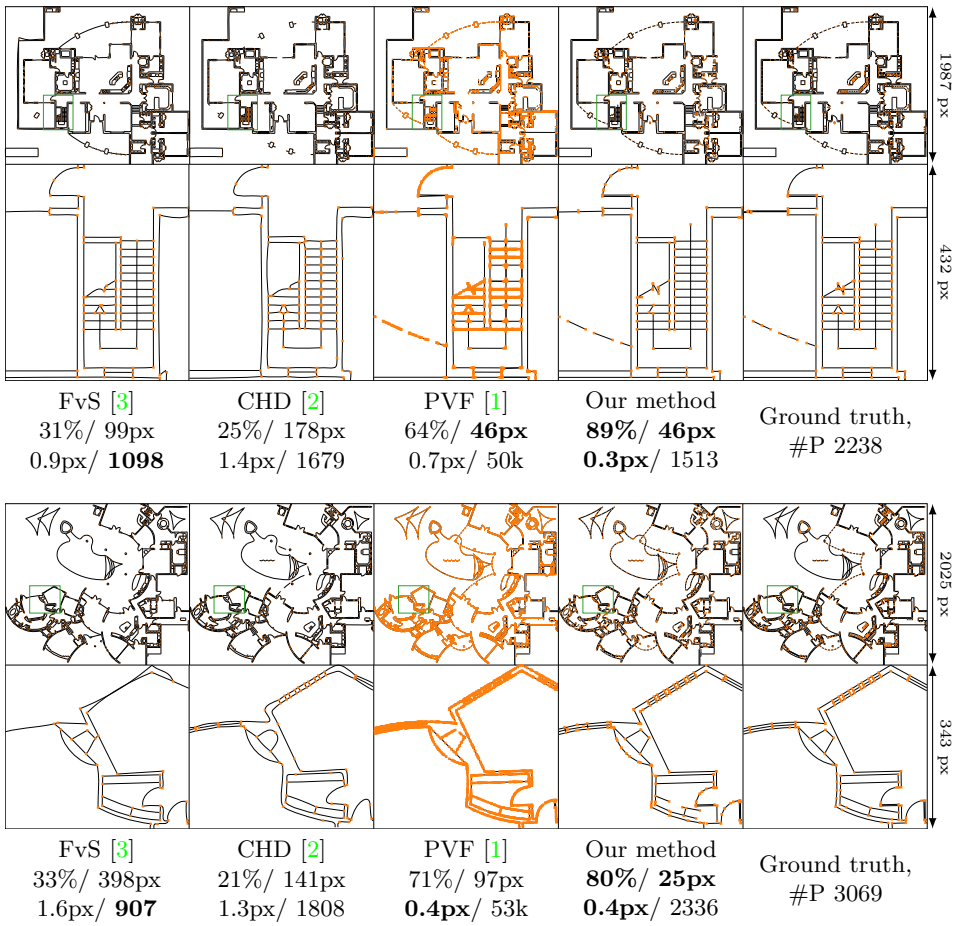

Fig. 8: Qualitative comparison on PFP images, and values of metrics IoU / d_H / d_M / #P with best in bold. Endpoints of the primitives are shown in orange.

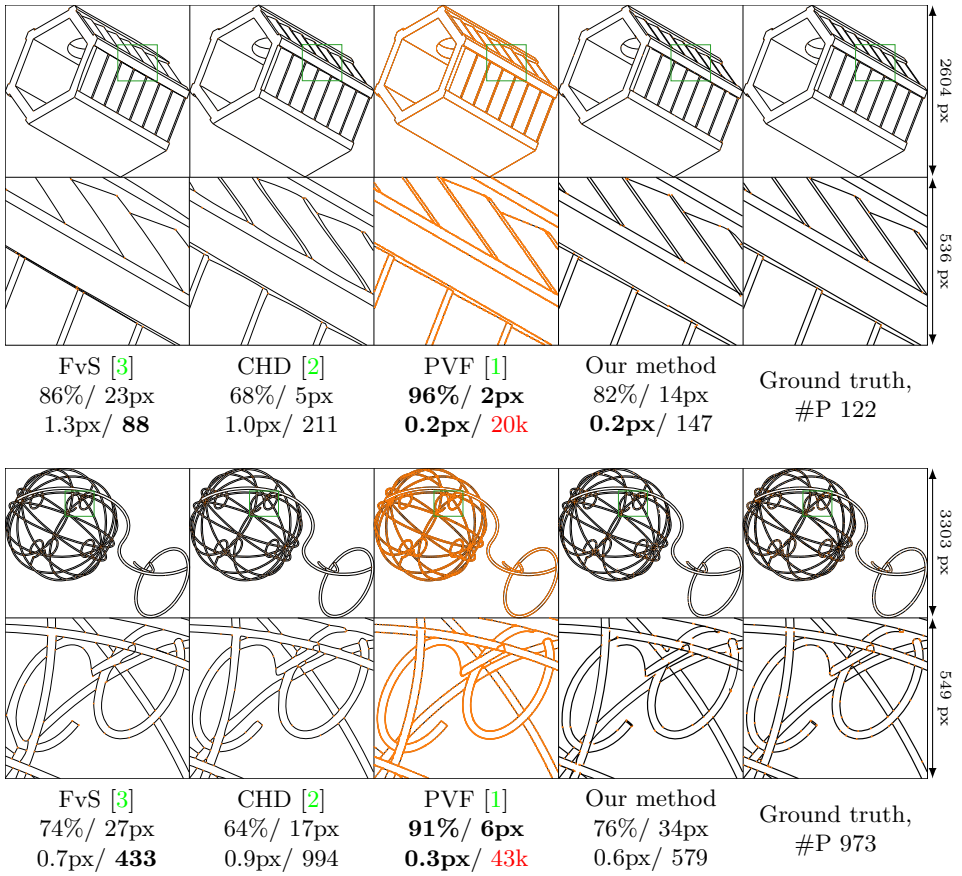

Fig. 9: Qualitative comparison on ABC images, and values of IoU / d_H / d_M / #P metrics, with the best result in boldface. The endpoints of primitives are shown in orange.

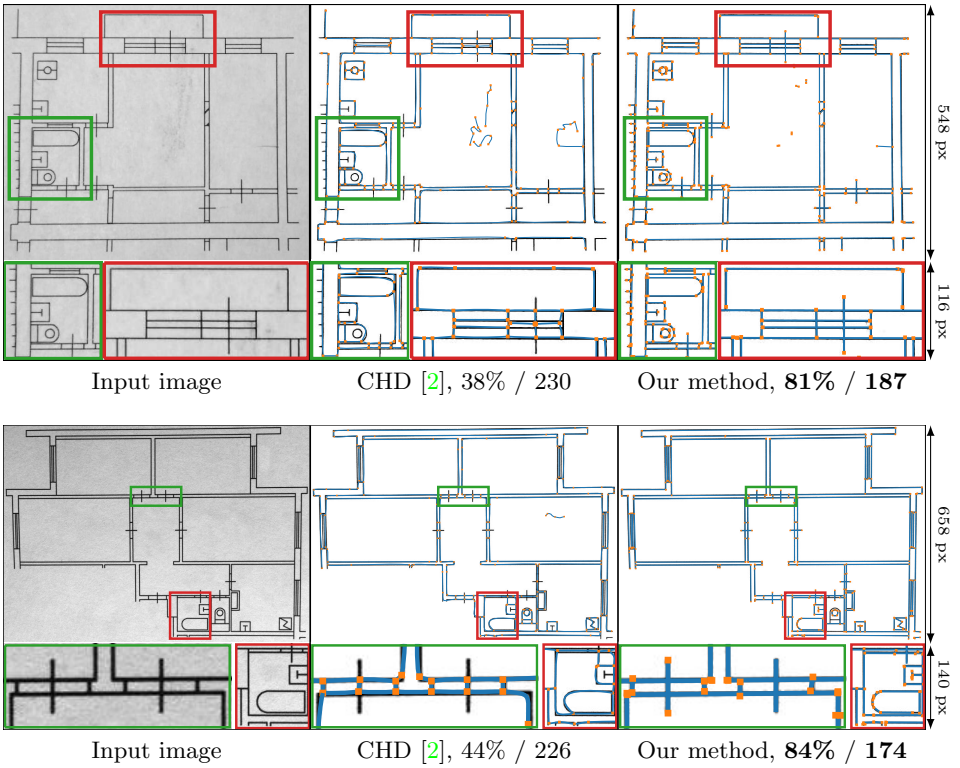

Fig. 10: Qualitative comparison on real noisy images, and values of metric IoU / #P with best in bold. Primitives are shown in blue with the endpoints in orange on top of the cleaned raster image.

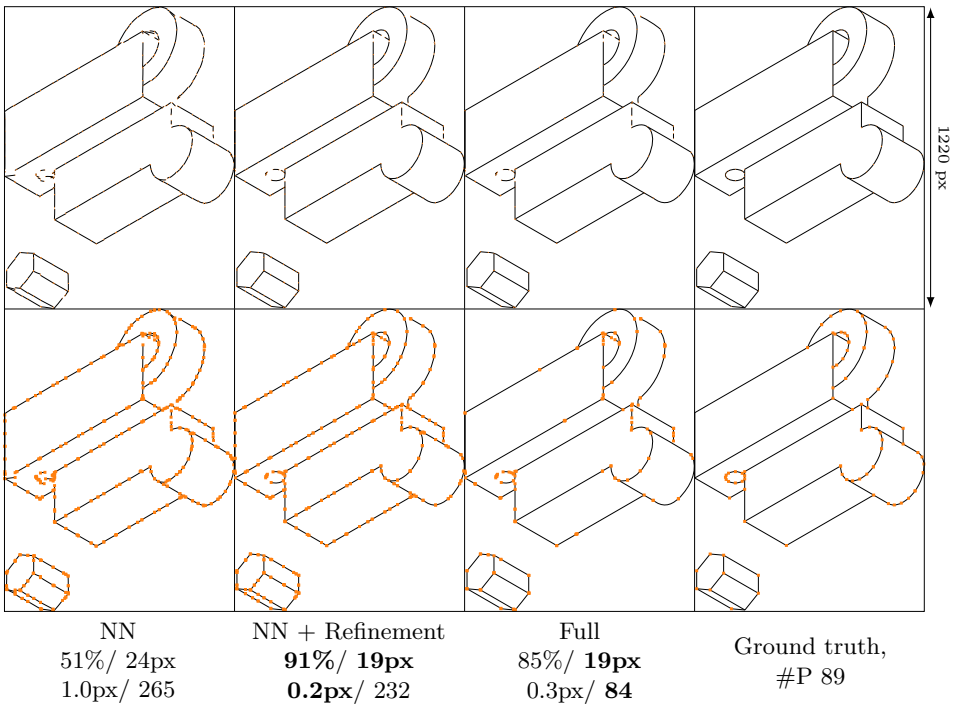

Fig. 11: Qualitative comparison on ABC images, and values of IoU / d_H / d_M / #P metrics, with the best results shown in boldface. The endpoints of primitives are shown in orange.

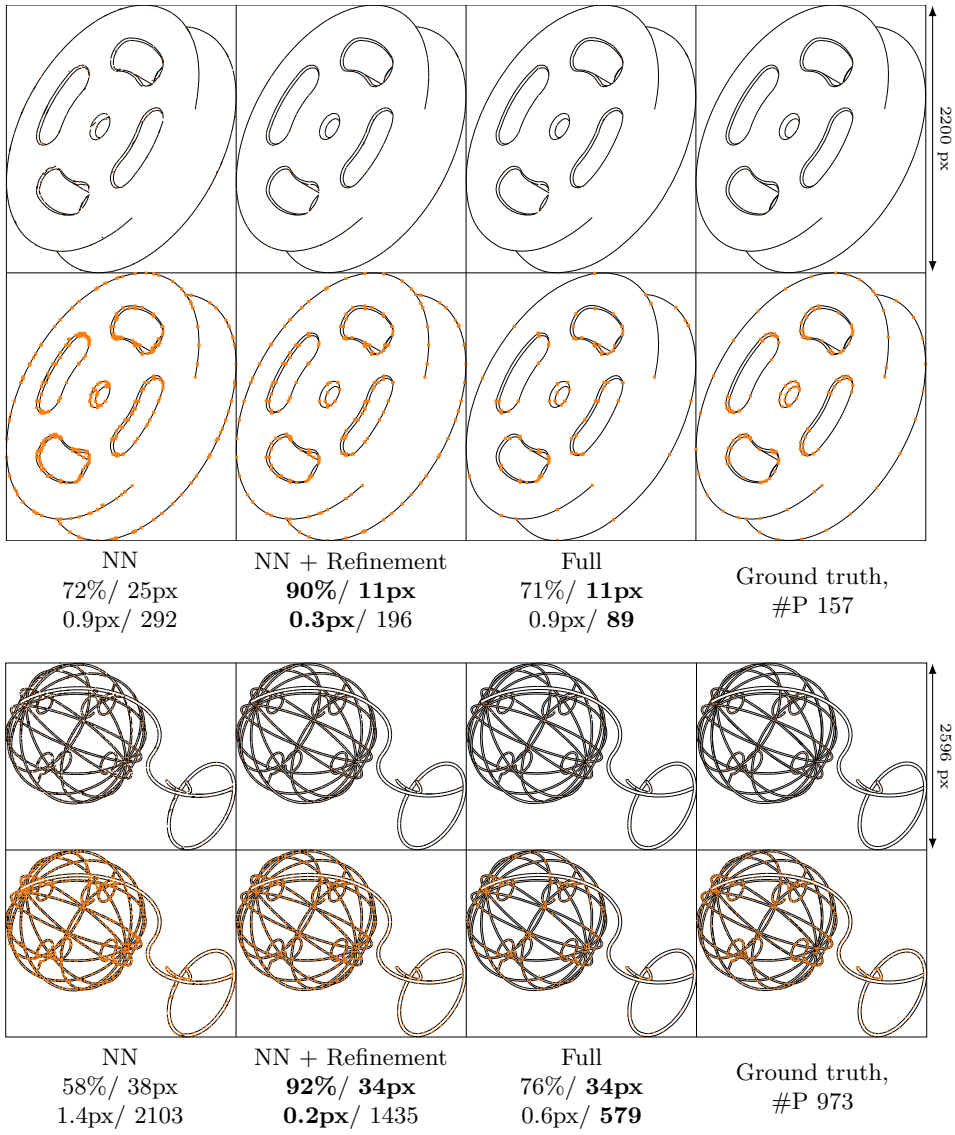

Fig. 12: Qualitative comparison on ABC images, and values of metrics IoU / d_H / d_M / #P with best in bold. Endpoints of the primitives are shown in orange.

I Details on refinement algorithm

I.1 Overall idea

The underlying idea in our approach is to use interaction potentials, qualitatively similar, *e.g.*, to electrostatic interaction, to construct our optimization functionals. Fixed charges are associated with filled pixels, and moving charges to the points on primitives. Primitives and filled pixels of the raster image are assigned charges of different signs: negative for pixels and positive for primitives. As a consequence, primitives and filled pixels and are attracted, and primitives repulse other primitives. Internal charges push primitives to expand, because their internal charges are repulsing each other. A number of modifications need to be made to this general approach to avoid undesirable minima.

The interaction energy of two charges at points $\boldsymbol{r}_1, \boldsymbol{r}_2$ is given by

$$q_1 q_2 \varphi\left(\|\boldsymbol{r}_1 - \boldsymbol{r}_2\|\right), \tag{3}$$

where q_1, q_2 are signed charges, and $\varphi(r)$ is the interaction potential of two charges at the distance r from each other. The standard 3D electrostatic potential is $\frac{1}{r}$; we replace it by an exponentially decaying potential as explained at the end of this section. The total energy is obtained by summation/integration over all charge pairs.

Energy. We split our energy into three parts: primitive-pixel interactions, interactions between distinct primitives and interaction between charges inside the same primitive. As the charges at pixels do not move, their interactions with each other can be ignored.

$$E = \sum_{k_{\text{prim}}, i_{\text{pix}}} E^{\text{prim,pix}}_{k_{\text{prim}}, i_{\text{pix}}} + \sum_{k_{\text{prim}} < j_{\text{prim}}} E^{\text{prim,prim}}_{k_{\text{prim}}, j_{\text{prim}}} + \sum_{k_{\text{prim}}} E^{\text{prim}}_{k_{\text{prim}}}. \tag{4}$$

Three parts of the energy have the following form:

$$E^{\text{prim,pix}}_{k_{\text{prim}}, i_{\text{pix}}} = -\hat{q}_{i_{\text{pix}}} \iint\limits_{\Omega_{k_{\text{prim}}}} \varphi\left(\|\boldsymbol{r} - \boldsymbol{r}_{i_{\text{pix}}}\|\right) dr^2, \tag{5}$$

where $\hat{q}_{i_{\text{pix}}}$ is the pixel intensity, $\boldsymbol{r}_{i_{\text{pix}}}$ and $\Omega_{k_{\text{prim}}}$ domain covered by the primitive;

$$E^{\text{prim,prim}}_{k_{\text{prim}}, j_{\text{prim}}} = \iint\limits_{\Omega_{k_{\text{prim}}}} \iint\limits_{\Omega_{j_{\text{prim}}}} \varphi\left(\|\boldsymbol{r}_1 - \boldsymbol{r}_2\|\right) dr_1^2 dr_2^2; \tag{6}$$

and

$$E^{\text{prim}}_{k_{\text{prim}}} = \frac{1}{2} E^{\text{prim,prim}}_{k_{\text{prim}}, k_{\text{prim}}} = \frac{1}{2} \iint\limits_{\Omega_{k_{\text{prim}}}} \iint\limits_{\Omega_{k_{\text{prim}}}} \varphi\left(\|\boldsymbol{r}_1 - \boldsymbol{r}_2\|\right) dr_1^2 dr_2^2. \tag{7}$$

Energy properties. Observe that pixel-primitive interaction is negative and decays (increases in magnitude) as primitive get close to a pixel, and also decreases as a primitive increase in size (more coverage is good). Primitive-primitive interaction energy is positive, decreases as the primitives move apart and also as the size of the primitives decreases. Finally, the self-interaction energy of a primitive is positive, does not depend on the primitive position and decreases if the primitive shrinks.

I.2 Mean-field-based optimization

For optimizing the energy efficiently, we use an approach based on a standard approach in the *mean-field* theory: the interactions between particles are viewed as individual interactions with a mean field, which is then updated using updated particle positions.

The basic gradient descent update of α^{th} parameter of k^{th} primitive is:

$$\theta_{k,\alpha} \leftarrow \theta_{k,\alpha} - \lambda \frac{\partial E}{\partial \theta_{k,\alpha}}. \tag{8}$$

We split the primitive parameters into size and position parameters and spell out the derivatives explicitly in each case, highlighting in blue the parts of the expressions that depend on the primitive.

$$\begin{gathered}\frac{\partial}{\partial \theta_{k,\alpha}^{\{\text{pos,size}\}}} \sum_{k_{\text{prim}}, i_{\text{pix}}} E_{k_{\text{prim}}, i_{\text{pix}}}^{\text{prim,pix}} = \partial \sum_{i_{\text{pix}}} E_{k, i_{\text{pix}}}^{\text{prim,pix}} = \\ - \sum_{i_{\text{pix}}} \hat{q}_{i_{\text{pix}}} \partial \iint\limits_{\Omega_k} \varphi \left(\| \boldsymbol{r} - \boldsymbol{r}_{i_{\text{pix}}} \| \right) dr_{,}^2\end{gathered} \tag{9}$$

$$\begin{gathered}\frac{\partial}{\partial \theta_{k,\alpha}^{\{\text{pos,size}\}}} \sum_{k_{\text{prim}} < j_{\text{prim}}} E_{k_{\text{prim}}, j_{\text{prim}}}^{\text{prim,prim}} = \partial \sum_{j_{\text{prim}} \neq k} E_{k, j_{\text{prim}}}^{\text{prim,prim}} = \\ \partial \iint\limits_{\Omega_k} \sum_{j_{\text{prim}} \neq k} \iint\limits_{\Omega_{j_{\text{prim}}}} \varphi \left(\| \boldsymbol{r}_1 - \boldsymbol{r}_2 \| \right) dr_1^2 dr_2^2,\end{gathered} \tag{10}$$

$$\frac{\partial}{\partial \theta_{k,\alpha}^{\text{pos}}} \sum_{k_{\text{prim}}} E_{k_{\text{prim}}}^{\text{prim}} = 0, \tag{11}$$

$$\begin{gathered}\frac{\partial}{\partial \theta_{k,\alpha}^{\text{size}}} \sum_{k_{\text{prim}}} E_{k_{\text{prim}}}^{\text{prim}} = \partial E_k^{\text{prim}} = \\ \frac{1}{2} \partial \iint\limits_{\Omega_k} \iint\limits_{\Omega_k} \varphi \left(\| \boldsymbol{r}_1 - \boldsymbol{r}_2 \| \right) dr_1^2 dr_2^2 + \frac{1}{2} \partial \iint\limits_{\Omega_k} \iint\limits_{\Omega_k} \varphi \left(\| \boldsymbol{r}_1 - \boldsymbol{r}_2 \| \right) dr_1^2 dr_2^2 = \\ \partial \iint\limits_{\Omega_k} \iint\limits_{\Omega_k} \varphi \left(\| \boldsymbol{r}_1 - \boldsymbol{r}_2 \| \right) dr_1^2 dr_2^2,\end{gathered} \tag{12}$$

The complete expressions for the energy derivatives with respect to positional parameters are:

$$\frac{\partial E}{\partial \theta^{\text{pos}}_{k,\alpha}} = \partial \sum_{i_{\text{pix}}} E^{\text{prim,pix}}_{k,i_{\text{pix}}} + \partial \sum_{j_{\text{prim}} \neq k} E^{\text{prim,prim}}_{k,j_{\text{prim}}} = \\ \partial \iint\limits_{\Omega_k} \left[\sum_{j_{\text{prim}} \neq k} \iint\limits_{\Omega_{j_{\text{prim}}}} \varphi\left(\|\boldsymbol{r} - \boldsymbol{r}_1\|\right) dr_1^2 - \sum_{i_{\text{pix}}} \hat{q}_{i_{\text{pix}}} \varphi\left(\|\boldsymbol{r} - \boldsymbol{r}_{i_{\text{pix}}}\|\right) \right] dr^2 \tag{13}$$

For size parameters, we obtain the following expression

$$\frac{\partial E}{\partial \theta^{\text{size}}_{k,\alpha}} = \partial \sum_{i_{\text{pix}}} E^{\text{prim,pix}}_{k,i_{\text{pix}}} + \partial \sum_{j_{\text{prim}}} E^{\text{prim,prim}}_{k,j_{\text{prim}}} = \\ \partial \iint\limits_{\Omega_k} \left[\sum_{j_{\text{prim}}} \iint\limits_{\Omega_{j_{\text{prim}}}} \varphi\left(\|\boldsymbol{r} - \boldsymbol{r}_1\|\right) dr_1^2 - \sum_{i_{\text{pix}}} \hat{q}_{i_{\text{pix}}} \varphi\left(\|\boldsymbol{r} - \boldsymbol{r}_{i_{\text{pix}}}\|\right) \right] dr^2 \tag{14}$$

where j_{prim} ranges over all primitives including k

We can interpret these derivatives as derivatives of a different function

$$E^* = \sum_k E^{\text{pos}}_k + E^{\text{size}}_k. \tag{15}$$

with terms defined below. Each term corresponds to particular parameters of one of the primitives, and can be viewed as the interaction energy of the primitive with a background charge distribution defined by all primitives at a given instance in time.

$$E_k\left(q\right) = \iint\limits_{S} q\left(\boldsymbol{r}_1\right) \iint\limits_{\Omega_k} \varphi\left(\|\boldsymbol{r}_1 - \boldsymbol{r}_2\|\right) dr_2^2 dr_1^2, \tag{16}$$

$$E^{\text{pos}}_k = E_k\left(q^{\text{pos}}_k\right)\big|_{\boldsymbol{\theta}^{\text{size}}_k = \text{const}}, \quad E^{\text{size}}_k = E_k\left(q^{\text{size}}_k\right)\big|_{\boldsymbol{\theta}^{\text{pos}}_k = \text{const}}, \tag{17}$$

$$q^{\text{pos}}_k\left(\boldsymbol{r}\right) = \sum_{j_{\text{prim}} \neq k} \mathbb{1}\left[\boldsymbol{r} \in \Omega_{j_{\text{prim}}}\right] - \sum_{i_{\text{pix}}} \hat{q}_{i_{\text{pix}}} \delta\left(\boldsymbol{r} - \boldsymbol{r}_{i_{\text{pix}}}\right), \tag{18}$$

$$q^{\text{size}}_k\left(\boldsymbol{r}\right) = \sum_{j_{\text{prim}}} \mathbb{1}\left[\boldsymbol{r} \in \Omega_{j_{\text{prim}}}\right] - \sum_{i_{\text{pix}}} \hat{q}_{i_{\text{pix}}} \delta\left(\boldsymbol{r} - \boldsymbol{r}_{i_{\text{pix}}}\right), \tag{19}$$

where $\mathbb{1}\left[\cdot\right]$ is the Iverson bracket, and δ is the delta-function.

Expressions (15)-(19) provide the physics-based foundation for our optimization: at every step, we use the new form of the energy terms to obtain the gradients using automatic differentiation; the "frozen" parts of each term are updated after parameter update at every step. In this initial form, the functional has a number of undesirable properties for our application; we make several modifications described in the next section.

I.3 Discretization and functional modifications

Discretization. While for simple primitives the integrals in (15)-(19) can be computed explicitly, we simplify the problem by using discrete charges instead of continuous distributions.

The expression (15) becomes equation (8) from the submission.

$$\iint\limits_{S} q\left(\boldsymbol{r}_1\right) \iint\limits_{\Omega_k} \varphi\left(\left\|\boldsymbol{r}_1-\boldsymbol{r}_2\right\|\right) dr_2^2 dr_1^2 \longrightarrow \sum_{i_{\text{pix}}} q_{i_{\text{pix}}} \iint\limits_{\Omega_k} \varphi\left(\left\|\boldsymbol{r}-\boldsymbol{r}_{i_{\text{pix}}}\right\|\right) dr^2 \tag{20}$$

Expressions (18) and (19) become

$$\iint\limits_{S} \mathbb{1}\left[\boldsymbol{r} \in \Omega_k\right] f\left(\boldsymbol{r}\right) dr^2 \longrightarrow \sum_{i_{\text{pix}}} q_{k,i_{\text{pix}}} f\left(\boldsymbol{r}_{i_{\text{pix}}}\right), \tag{21}$$

$$\iint\limits_{S} \sum_{i_{\text{pix}}} \hat{q}_{i_{\text{pix}}} \delta\left(\boldsymbol{r}-\boldsymbol{r}_{i_{\text{pix}}}\right) f\left(\boldsymbol{r}\right) dr^2 \longrightarrow \sum_{i_{\text{pix}}} \hat{q}_{i_{\text{pix}}} f\left(\boldsymbol{r}_{i_{\text{pix}}}\right), \tag{22}$$

$$q_k^{\text{pos}}\left(\boldsymbol{r}\right) \longrightarrow q_{k,i_{\text{pix}}}^{\text{pos}} = \sum_{j_{\text{prim}} \neq k} q_{j_{\text{prim}},i_{\text{pix}}} - \hat{q}_{i_{\text{pix}}}, \tag{23}$$

$$q_k^{\text{size}}\left(\boldsymbol{r}\right) \longrightarrow q_{k,i_{\text{pix}}}^{\text{size}} = \sum_{j_{\text{prim}}} q_{j_{\text{prim}},i_{\text{pix}}} - \hat{q}_{i_{\text{pix}}}, \tag{24}$$

where $\hat{q}_{i_{\text{pix}}}$ is the coverage of the $i_{\text{pix}}{}^{\text{th}}$ raster image pixel, and $q_{k_{\text{prim}},i_{\text{pix}}}$ is the coverage of the $k_{\text{prim}}{}^{\text{th}}$ primitive in $i_{\text{pix}}{}^{\text{th}}$ pixel.

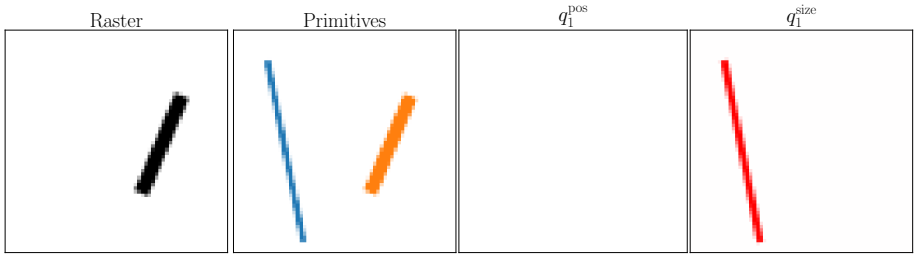

Fig. 13: Raster, primitives and charge grids of the first primitive. First primitive in blue, second in orange. In charge grids red represents excess charge.

Charge saturation. The charge distributions $\boldsymbol{q}_k^{\text{pos}} = \left\{q_{k,i_{\text{pix}}}^{\text{pos}}\right\}_{i_{\text{pix}}}$, $\boldsymbol{q}_k^{\text{size}} = \left\{q_{k,i_{\text{pix}}}^{\text{size}}\right\}_{i_{\text{pix}}}$ are excess or insufficient charges that need to be compensated by

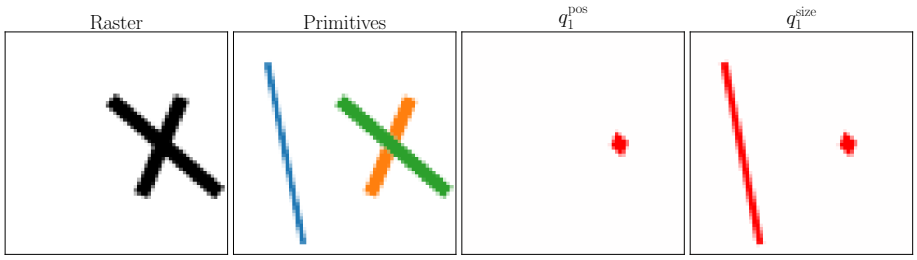

Fig. 14: Overlap affection on other primitives. First primitive in blue, second in orange. In charge grids red represents excess charge.

changing the k^{th} primitive. The energy terms corresponding to a primitive should not be affected by how many primitives cover a particular filled area. For example, in Figure 13, the second primitive covers the filled part of the raster perfectly, and this area does not affect the placement and size of the first primitive. Figure 14, the second and third primitives are covering the area equally well, but because of the overlap, the sum of their charges is higher than the negative charge of the raster image, and this creates a force acting on the first primitive.

To avoid the excess charge, we replace the sum of the charges with the maximum, leading to the following modification:

$$q^{\text{pos}}_{k,i_{\text{pix}}} = q_{-k,i_{\text{pix}}} - \hat{q}_{i_{\text{pix}}}, \tag{25}$$

$$q^{\text{size}}_{k,i_{\text{pix}}} = q_{i_{\text{pix}}} - \hat{q}_{i_{\text{pix}}}, \tag{26}$$

where $q_{i_{\text{pix}}}$ is the sum of coverages of ${i_{\text{pix}}}^{\text{th}}$ pixel for all primitives, and $q_{-k,i_{\text{pix}}}$ is the same sum with k^{th} primitive excluded.

Compared to (23), (24), modified charge distributions (25), (26) do not penalize overlaps.

Illustrative examples. Next, we consider several examples illustrating the behavior of the functional, which also help us to explain the modifications we make.

An isolated primitive.

For a single primitive (Figure 15) the position energy term is constant and does not depend on the parameters, so it would not move. The size energy term becomes lower with size: the primitive collapses to a point.

Primitive separated from the filled areas of the raster image. For a single primitive sufficiently far from the filled part of the image (Figure 16) the energy is decreasing if the primitive moves to the raster due to the second term in (25). If the primitive shrinks, the first term in (26) decreases and the second term increases. However, as we use fast-decaying potentials, we can neglect the interactions with distant charges, and overall the energy favors size reduction.

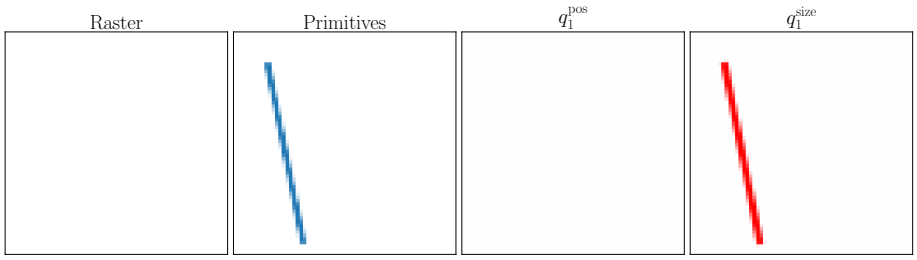

Fig. 15: Single primitive collapse. In charge grids red represents excess charge.

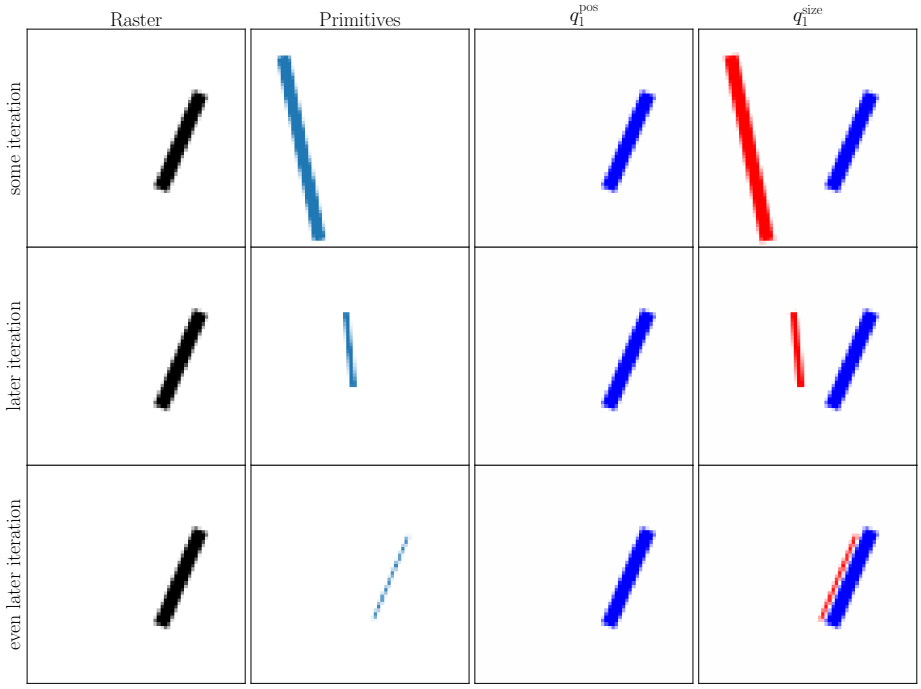

Fig. 16: Primitive interaction with far raster filled part. In charge grids red represents excess charge and blue represents uncovered raster.

A primitive close to a raster image. If a primitive is close to a filled part of the raster it will get aligned to the filled pixels, if these form a line and will increase in size until it covers the filled area (Figure 17) If we add additional filled pixels or other primitives at a distance, due to potential decay, they will have a minimal effect on the behavior.

Primitive aligned with a filled part of the primitive. In this case, the potentials from the raster image and the first primitive compensate each other, and the second primitive will not be affected by either, as in scenario of I.3.

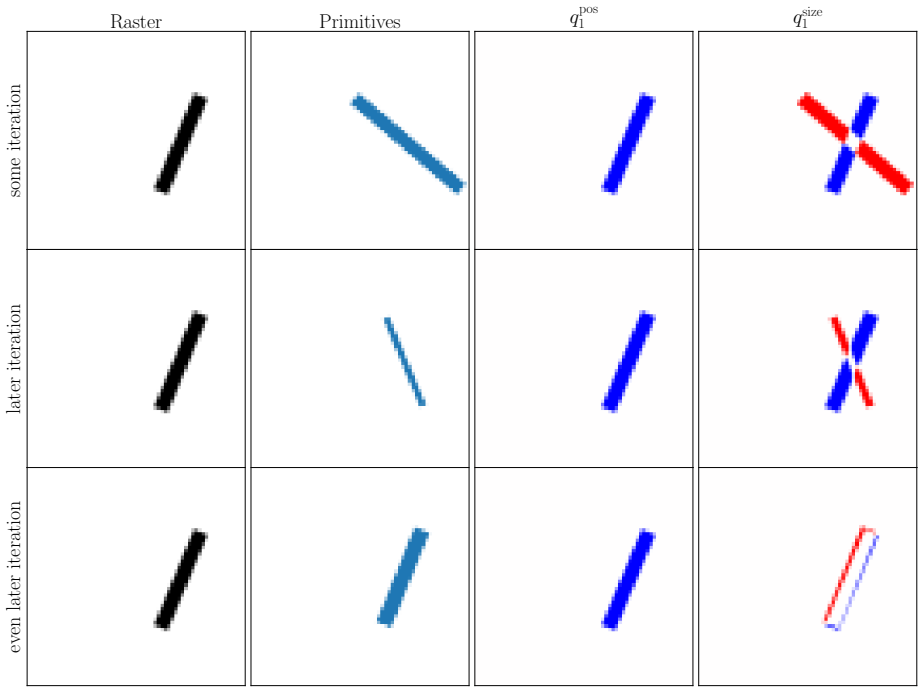

Fig. 17: Primitive interaction with close raster filled part. In charge grids red represents excess charge and blue represents uncovered raster.

Enabling primitive intersections. Consider the case in Figure 18. In this case,it would be desirable for the second primitive to cover the entire horizontal line, but this will not happen, as the second primitive, with the original energy formulation will remain close to its initial state. Primitive 1 instead of expanding will not change much either, as primitive 2 prevents its expansion.

To achieve the desired effect, i.e., expansion of the second primitive, we modify $\boldsymbol{q}_k^{\text{pos}}$ as follows:

$$q^{\text{pos}}_{k,i_{\text{pix}}} = q_{i_{\text{pix}}} - q_{k,i_{\text{pix}}} - \hat{q}_{i_{\text{pix}}}. \tag{27}$$

With this definition, the primitive interacts with pixels covered by other primitives, which allows it to expand across already filled areas, as in Figure 19

Penalty for overlapping collinear primitives $\boldsymbol{E_k^{\text{rdn}}}$. By itself, (27) allows not just transversal intersections, but also aligned primitives covering the same area (Figure 20). We add an additional penalty E_k^{rdn} to avoid this. This term is also based on the interaction of the primitive with a background charge excess/deficiency, in this case, created by nearby primitives with tangents close to its tangent at close points. We define this term for segments first, and then generalize to curves.

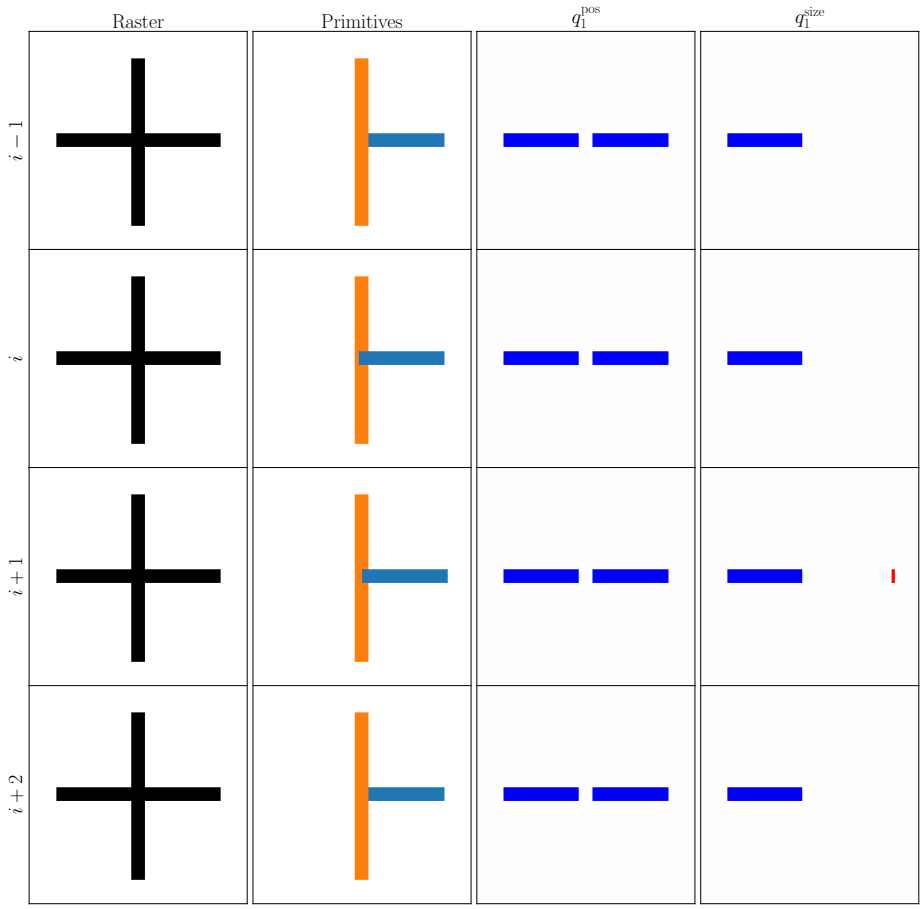

Fig. 18: Primitives interaction with disable primitive intersections. In charge grids red represents excess charge and blue represents uncovered raster.

Collinear penalty for line segments. For a system of two segments k and j we define it as

$$E_k^{\text{rdn}} = E_k\left(\boldsymbol{q}_k^{\text{rdn}}\right)\Big|_{\text{close }\varphi(r),\boldsymbol{\theta}_k^{\text{pos}}=\text{const}}, \tag{28}$$

$$q_{k,i_{\text{pix}}}^{\text{rdn}} = q_{j,i_{\text{pix}}} \exp\left(-\frac{\left(|\boldsymbol{l}_k \cdot \boldsymbol{l}_j| - 1\right)^2}{\left(|\cos\alpha_{\text{col}}| - 1\right)^2}\right), \tag{29}$$

where close $\varphi(r)$ means that we truncate the interaction at a fixed radius $\varphi(r|r > r^*) = 0$, as explained below, and charges $q_{k,i_{\text{pix}}}^{\text{rdn}}$ are defined by j^{th} primitive weighted by the cosine of the angle between segment directions, $\boldsymbol{l}_k, \boldsymbol{l}_j$, and α_{col} is a threshold angle for collinearity detection.

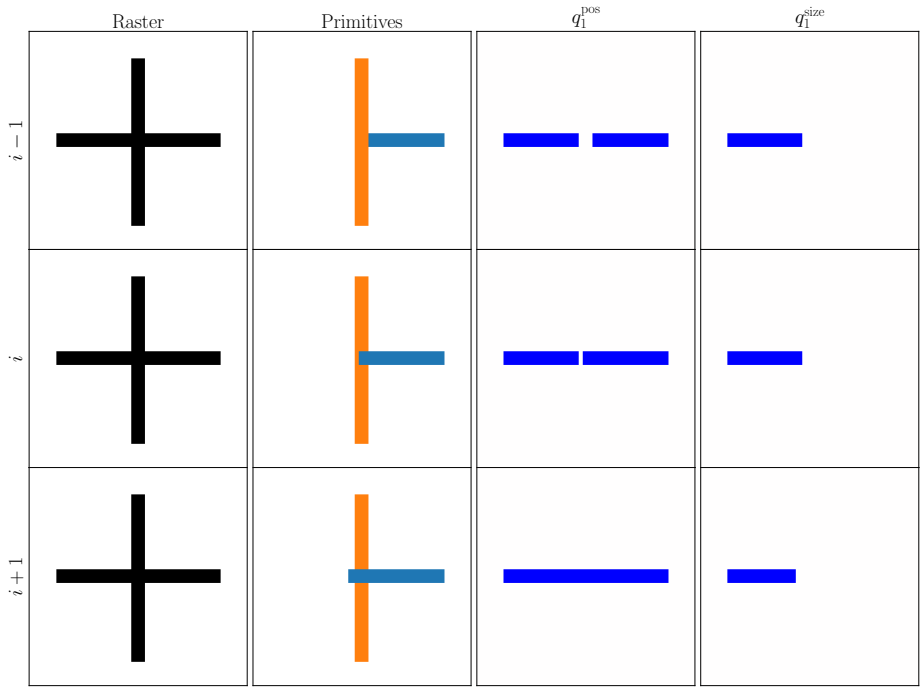

Fig. 19: Primitives interaction with enabled primitive intersections. In charge grids red represents excess charge and blue represents uncovered raster.

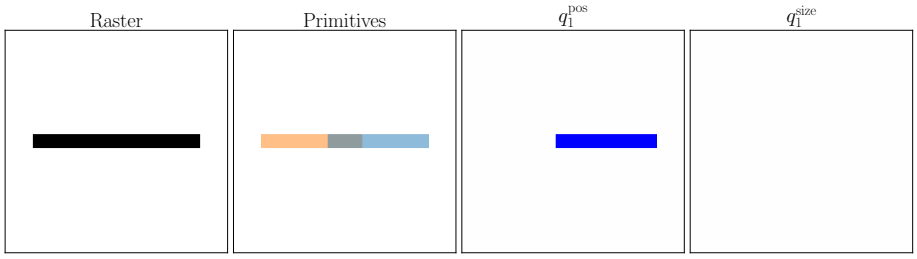

Fig. 20: Primitives covering the same area.

For many segments, we define the charges as

$$q^{\text{rdn}}_{k,i_{\text{pix}}} = \|\boldsymbol{m}_{k,i_{\text{pix}}}\| \exp\left(-\frac{\left(|\boldsymbol{l}_k \cdot \boldsymbol{m}_{k,i_{\text{pix}}}| - 1\right)^2}{\left(|\cos\alpha_{\text{col}}| - 1\right)^2}\right), \tag{30}$$

where $\boldsymbol{m}_{k,i_{\text{pix}}} = \sum_{j \neq k} \boldsymbol{l}_j q_{j,i_{\text{pix}}}$ is the sum of directions of all the other primitives weighted w.r.t. the mean direction of other primitives.

Collinearity penalization for curved segments. For curves, we use a similar idea, but need to use a different direction for every pixel, $\boldsymbol{l}_{k,i}$

$$q^{\text{rdn}}_{k,i_{\text{pix}}} = \|\boldsymbol{m}_{k,i_{\text{pix}}}\| \exp\left(-\frac{\left(|\boldsymbol{l}_{k,i_{\text{pix}}} \cdot \boldsymbol{m}_{k,i_{\text{pix}}}| - 1\right)^2}{\left(|\cos\alpha_{\text{col}}| - 1\right)^2}\right), \qquad \boldsymbol{m}_{k,i_{\text{pix}}} = \sum_{j \neq k} \boldsymbol{l}_{k,i_{\text{pix}}} q_{j,i_{\text{pix}}}. \tag{31}$$

This definition reduces to the definition for segments if the curve is straight.

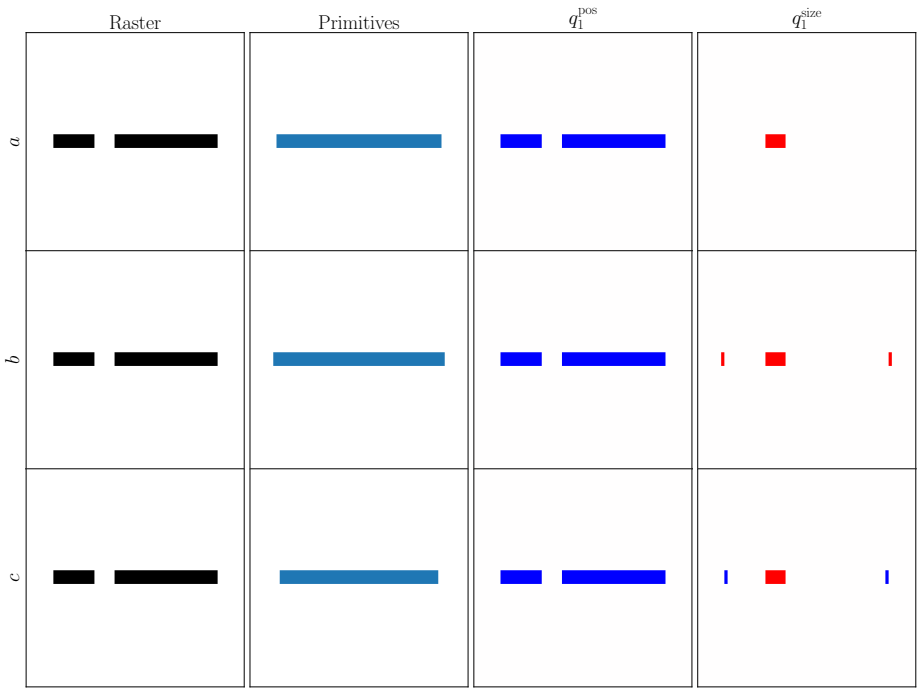

Fig. 21: Undesirable local minimum in which a primitive covers two disconnected raster parts.

Connected area mask. Consider the example in Figure 21 (a). Clearly the position and width of the primitive cannot be changed so that the energy decreases. Same is true for length: If the length increases or decreases (Figure 21 (b,c)), a counteracting force immediately appears because of the charge excess or deficit. We conclude that this is a local minimum, but clearly not a desirable solution. To reduce the chances of a primitive getting stuck in such a minimum, we limit the interaction of the primitive with the raster to the part that it can cover for

its current position and orientation, but arbitrary size. To be more precise, for line segments (Figure 22), for a fixed position and orientation, we find unfilled pixels along the line of the segment closest to its center, determining the length of the area. Once this is determined, then we find unfilled pixels closest to the line of the segment on two sides. This determines the rectangle for which the mask coefficient $c_{k,i_{\text{pix}}}$ is set to one, and for the rest of the image to zero.

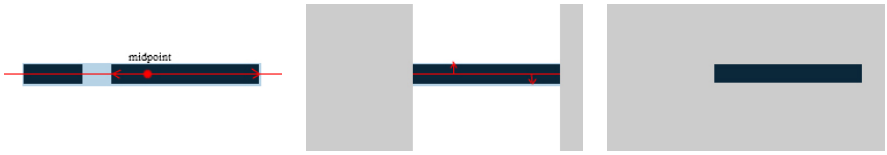

Fig. 22: Stages of $c_{k,i_{\text{pix}}}$ calculation.

For second order Bézier curves, we use a similar definition. Instead of a line we use a parabola containing the Bézier segment, and the distance along the parabola instead of the Euclidean distance.

The charge distribution for the size terms of the energy is redefined as follows

$$q^{\text{size}}_{k,i_{\text{pix}}} = \begin{cases} q_{i_{\text{pix}}} - \hat{q}_{i_{\text{pix}}} & \text{if } c_{k,i_{\text{pix}}} = 1, \\ q_{k,i_{\text{pix}}} & \text{if } c_{k,i_{\text{pix}}} = 0. \end{cases} \tag{32}$$

Connected area mask for positional terms. If we use the same masks for the charges used for the positional terms $q^{\text{pos}}_{k,i_{\text{pix}}}$ eliminating the influence of everything outside the area where $c_{k,i_{\text{pix}}}$ is positive, then the primitive will stay within the filled area they initially overlap, which may be undesirable if the initial position is inaccurate, or multiple primitives initially cluster in the same place. For this reason, we only amplify the charge in the masked area leaving it the same outside:

$$q^{\text{pos}}_{k,i_{\text{pix}}} = \begin{cases} \lambda_{\text{pos}} \left(q_{i_{\text{pix}}} - q_{k,i_{\text{pix}}} - \hat{q}_{i_{\text{pix}}} \right) & \text{if } c_{k,i_{\text{pix}}} = 1, \\ q_{i_{\text{pix}}} - q_{k,i_{\text{pix}}} - \hat{q}_{i_{\text{pix}}} & \text{if } c_{k,i_{\text{pix}}} = 0, \end{cases} \tag{33}$$

The amplification coefficient λ_{pos} is chosen empirically.

The choice of the potential function. The main property of the potential function $\varphi(r)$ used in our algorithm is rapid monotonic decrease with distance. We use the function

$$\varphi(r) = e^{-\frac{r^2}{R_{\text{c}}^2}} + \lambda_{\text{f}} e^{-\frac{r^2}{R_{\text{f}}^2}}, \tag{34}$$

which allows us to control interactions at close range $\sim R_{\text{c}}$ independently from interactions at far range $\sim R_{\text{f}}$. We choose these ranges and the weight experimentally $R_{\text{c}} = 1\,\text{px}$, $R_{\text{f}} = 32\,\text{px}$, $\lambda_{\text{f}} = 0.02$, and in E^{rdn}_k disable the far interactions by setting $\lambda_{\text{f}} = 0$.

References

1. Bessmeltsev, M., Solomon, J.: Vectorization of line drawings via polyvector fields. ACM Transactions on Graphics (TOG) **38**(1), 9 (2019) 1, 5, 6, 7, 9, 10
2. Donati, L., Cesano, S., Prati, A.: A complete hand-drawn sketch vectorization framework. Multimedia Tools and Applications **78**(14), 19083–19113 (2019) 1, 5, 6, 7, 9, 10, 11
3. 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) 1, 5, 6, 7, 9, 10
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016) 1
5. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014) 2
6. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019), http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf 2
7. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015) 1
8. Tange, O.: Gnu parallel - the command-line power tool. ;login: The USENIX Magazine **36**(1), 42–47 (Feb 2011), http://www.gnu.org/s/parallel 2
9. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems. pp. 5998–6008 (2017) 1, 3