DeepStrip: High Resolution Boundary Refinement

Peng Zhou[1] Brian Price[2] Scott Cohen[2] Gregg Wilensky[2] Larry S. Davis[1]
[1]University of Maryland, College Park [2]Adobe Research, San Jose

Abstract

In this paper, we target refining the boundaries in high resolution images given low resolution masks. For memory and computation efficiency, we propose to convert the regions of interest into strip images and compute a boundary prediction in the strip domain. To detect the target boundary, we present a framework with two prediction layers. First, all potential boundaries are predicted as an initial prediction and then a selection layer is used to pick the target boundary and smooth the result. To encourage accurate prediction, a loss which measures the boundary distance in the strip domain is introduced. In addition, we enforce a matching consistency and C0 continuity regularization to the network to reduce false alarms. Extensive experiments on both public and a newly created high resolution dataset strongly validate our approach.

1. Introduction

Boundary detection is a well-studied problem and fundamental for human recognition [29, 9]. Recent decades have witnessed considerable effort to improve the boundary quality of an object that has been detected [42, 36, 33, 19, 38, 49, 16, 23, 43] or segmented [11, 35, 25, 7, 18]. Consequently, it is not difficult to separate object of interests from backgrounds with precise boundaries utilizing these methods. While current learning based boundary detection algorithms are usually computed on low resolution (LR) images (0.04-0.25 million pixels), most photos taken these days are much larger, ranging from cell phone size (8-16 million pixels) to professional camera size (16-400 million pixels). Most methods are not designed for images of this size and the excessive computation they require, and most machine learning based methods cannot process them due to memory constraints. Given a precise low resolution prediction, a workaround would be to directly apply upsampling to reach high resolution (HR). Nevertheless, this usually yields poor quality results because the semantic contents in the HR image are not considered. (See Figure 1.)

Most research in boundary detection focuses on improving the boundary quality in LR through introducing more semantic information [2, 46, 27] or human interaction [23, 26, 44, 25, 7]. While there has been some work on HR semantic segmentation [12, 48] and upsampling [41, 47], there is less focus on accurately capturing the boundary detail in HR. Instead of treating this problem as an upsampling problem, we treat it as boundary detection and harness the contents in HR images for prediction.

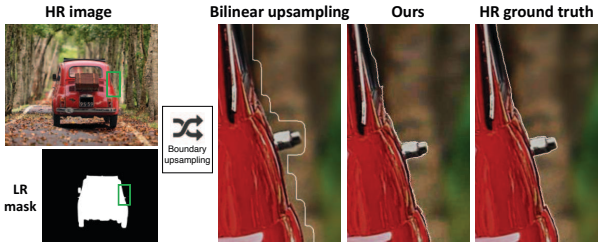

Figure 1. Concept overview. The example is from the newly created PixaHR dataset. Given low resolution mask and high resolution image on the left, a bilinear upsampling with scale factor $16\times$ would results in boundary misalignment in high resolution image, as is shown in the enlarged boundary region on the right. Also, the new details in high resolution would be missed.

To this end, we propose a novel approach to handle boundary refinement in HR images. (See Figure 2.) Our key idea is to allow the power of deep learning methods to be applied to HR images in a time and memory efficient manner by operating on narrow images made up of pixels near the boundary. Given an accurate LR mask, the boundary in HR is likely in proximity to the upsampled LR boundary. (See Figure 1.) Therefore, to save memory and computation, we propose to search for the target boundary in a strip region near the boundary of the upsampled mask. The strip image is formed by sampling pixels along and normal to the upsampled mask boundary. Since the normals may not be smooth due to inaccurate boundaries in the upsampled mask, we represent the LR boundary with a spline approximation and directly treat the orthogonal derivatives of the upsampled spline as the normal directions. Feeding as input the generated strip images, we train a network to firstly predict all potential boundaries. Based on the initial prediction, an additional selection layer is included to predict the target boundary more accurately. To encourage closer prediction and reduce false positives, we propose loss functions to minimize the boundary distance between the pre-

2575-7075/20/$31.00 ©2020 IEEE
DOI 10.1109/CVPR42600.2020.01057

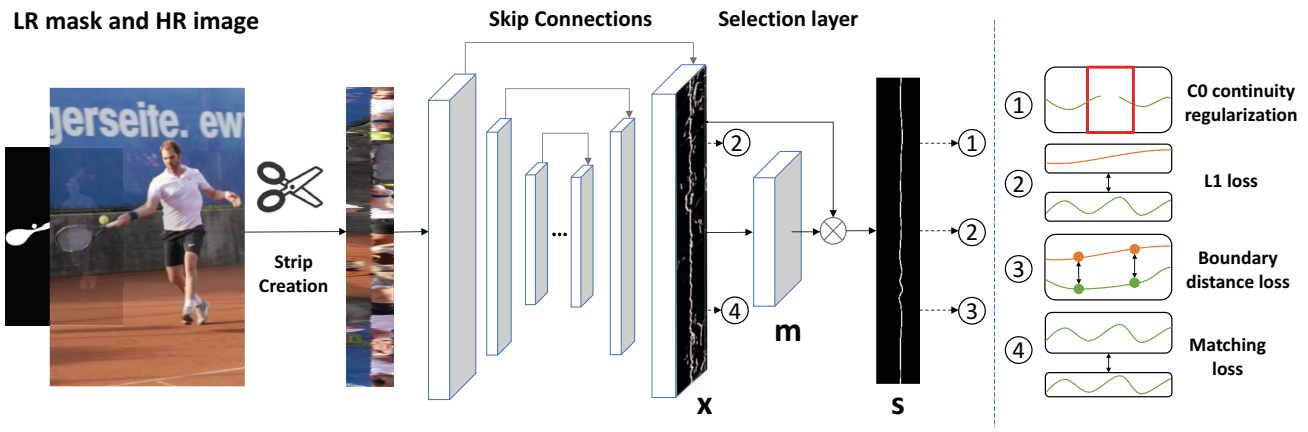

Figure 2. Framework. To save memory and computation, we predict the boundary in a strip image instead of the whole image. First, the strip image is extracted from the HR image and corresponding LR mask. Feeding the strip image as input, the network predicts all potential boundaries (denoted as "x") and passes the initial prediction to a selection layer (denoted as "m") to pursue more accurate prediction on the target boundary (denoted as "s"). The numbers are indicator to the losses displayed on the right. Orange and green curves denote the ground truth and prediction, respectively. Note that the strip image and prediction are rotated 90 degree for visualization.

diction and ground truth in the strip image and to encourage C0 continuity in the prediction. Lastly, we pursue consistent results through matching the prediction under different strip sizes to further boost the performance.

To validate our approach, we create a new PixaHR dataset (see Figure 1 for image example) consisting of 100 photos with average resolution $7k \times 7k$ and evaluate our approach up to scale factor $32\times$. Results on DAVIS 2016 and COCO coarse annotations also show our ability to refine coarse boundary annotations.

In a nutshell, our contribution is three-fold. 1) We propose an approach to predict the boundary in a strip image which converts potential boundary regions into a strip space. This approach allows us to apply neural networks in a computationally and memory efficient manner. 2) To improve performance and encourage closer prediction, we propose novel losses including boundary distance, matching and C0 continuity loss. 3) We create a high resolution dataset for evaluation. To the best of our knowledge, we are the first learning based approach to make HR dense boundary refinement with resolution up to $10k \times 10k$. Extensive experiments on both public and the new PixaHR dataset strongly highlight our effectiveness.

2. Related Work

Boundary Refinement. Multiple attempts have been made to improve boundary quality through extracting better features [42, 45, 27, 2, 14]. Xie *et al*. [42] utilize features from multiple layers and fuse both low and high level features to detect edges. Liu *et al*. [27] explore rich convolutional features to boost the performance. More related, attention has been taken to refine coarse boundary predictions or annotations [46, 2]. Conventional methods like dense Conditional Random Fields (CRF) [22], Graph Cuts [8] model the relationship between nearby pixels and thus can be applied to refine LR masks [24]. However, these are segmentation based and only low-level features have been utilized. With more supervision, Yu *et al*. [46] propose to simultaneously learn and align edges to refine misaligned boundaries directly. Acuna *et al*. [2] further improve the performance by introducing a thinning layer and active alignment strategy to obtain refined boundary. These methods mainly explore edge detection in LR images. In contrast, we tackle HR boundary refinement and apply detection only on regions around upsampled LR boundary splines and thus is more memory and computation efficient.

Active Contours. Active contour models like Snakes [19] have been introduced to refine boundaries from coarse ones. Various approaches have been explored to handle the limitation of Snakes through, *e.g*., better initialization, morphological operation [4] or user interaction [23]. Since our method also refines the curve upsampled from LR mask, we can benefit from these methods and refine the boundary further. Instead of taking the whole image as input, deep active contour [34] learns to predict the flow of boundary pixels in a patch by patch fashion. However, it cannot guarantee a continuous boundary prediction. Instead, our approach directly extracts a consecutive boundary region and thus contains more global information. Rather than predict the entire curve, other works have explored predicting control points [10, 3, 26] through recurrent neural networks or Graph Convolutional Networks (GCN) [20] and then fit a curve as the final prediction. However, boundary details are smoothed in the spline representation. In contrast, our approach predicts precise edge information directly. Another line of work implicitly represents boundary curves. For example, deep level set methods [30] evolve boundary curves by minimizing the level energy function. Other learning based approaches [28, 13, 39] have proposed to provide

useful features, including texture, color or shape, for better optimization. However, these learning based approaches suffer from computation and memory issues when the resolution increases because they process the entire image while our approach only focuses on the regions around upsampled LR boundaries, and thus requires less computation and memory overhead.

High Resolution Up-sampling. With the information of low resolution masks, researchers have focused on achieving high quality HR segmentation masks. Conventional methods [21, 6] reach HR by applying upsampling jointly with the LR mask reference. However, the fixed filter structures have difficulty capturing new HR boundary details. He *et al*. [17] propose guided filtering to smooth while preserving edge information when upsampling. Wu *et al*. [41] make the guided filter faster and learnable. For HR segmentation approaches, Zhao *et al*. [48] propose to aggregate LR features for HR segmentation and Chen *et al*. [12] align both global and local features to avoid heavy GPU consumption for HR segmentation. Even though these methods can be potentially adapted to boundary refinement, our method mainly focuses on boundary regions and is designed to detect boundaries in HR directly. Therefore, our approach learns new HR boundaries better, especially when LR boundaries are coarsely annotated.

3. Approach

Our goal lies in refining boundaries in HR images given LR precise masks. To achieve this purpose efficiently, we propose to predict on a strip image that captures the potential boundary region rather than the entire HR image. Figure 2 illustrates our framework. Our approach consists of strip image creation, which converts HR RGB image into strip image, strip boundary prediction, which refines the edges on the strip image using a network and strip reconstruction which reconstructs the prediction in the original image from the strip boundary prediction during testing.

3.1. Strip Image Creation

Figure 3 describes the procedure of strip image creation. Due to the interpolation introduced by upsampling, a directly upsampled boundary from the LR image is likely to be shifted from the ground truth boundary in HR. To localize the real HR boundary pixels, searching around the upsampled boundary is more necessary than searching the whole image. Therefore, we extract pixels near the upsampled boundary to create a strip image. To create the strip image, we step along the boundary and sample points along the normal direction at each point on the curve. To obtain smoothly varying normal directions along the coarse boundary, we represent the LR boundary by B-spline and upsample the LR spline to HR.

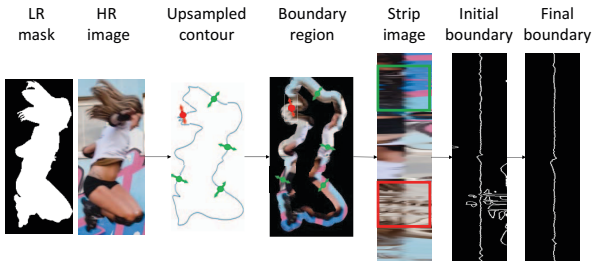

Figure 3. Strip image creation. To generate strip image, B-spline representation of the contour in the LR mask is upsampled to HR as a coarse boundary. The HR region along the normal direction (*e.g*., red and green arrows) of the contour is then extracted. Finally, the strip image and corresponding boundary ground truth is obtained by flattening the extracted region in both the HR image and mask. Note that the final boundary filters out noisy boundaries (*e.g*., the red box region) from the initial boundary. The strip image and boundaries are rotated 90 degree for visualization.

Given the HR image $I(p,q)$ and the upsampled spline representation $C = (p(k), q(k))$ of the boundary contour, where $(p(k), q(k))$ denotes the HR image coordinates parameterized by arclength k along the curve, the continuous strip image $J_{I,C}$ is defined by

$$J_{I,C}(k, t{+}H/2) = I(p(k){+}t{\times}n_p(k), q(k){+}t{\times}n_q(k)), \quad (1)$$

where t denotes the distance in the normal direction, H denotes the height of the strip image, and $(n_p(k), n_q(k))$ is the unit normal to the curve at arclength k. Accordingly, the strip image $J_{I,C}(j,i)$ with dimension $H \times W$ is obtained by sampling $k = j \times dk$, $t = i \times dt$, where tangential step size $dk = \lfloor |C|/W \rfloor$ and normal step size dt is set to 1 for simplicity. $|C|$ denotes the length of C, $j = 0, 1, ..., W$ and $i = -H/2, ..., 0, ..., H/2$. Also, bilinear interpolation is applied in the high resolution image to evaluate $I(p,q)$ for non-pixel coordinates (p,q).

The corresponding HR strip boundary ground truth is obtained similarly with two adaptations. First, for large sampling scale factors, the ground truth boundary is likely to be outside the range of the strip if the strip height is small, making the boundary in strip image not continuous. We add labels at the border of strip if no boundary pixel is included to maintain the C0 continuity of the boundary pixels in the strip image. Second, if the strip height is large, multiple boundary pixels might be included in each column in regions where the boundaries are closer than the strip height. In this case, we filter out the extraneous boundaries that are not connected to the current boundary. (See Figure 3.)

3.2. Strip Boundary Prediction

Provided the HR strip image as input, we train a network to predict the corresponding boundaries within the strip domain. For memory efficiency, we adapt light-weighted encoder-decoder based structure nested U-Net [32, 50] for boundary prediction. Given the fact that proper dimension

of strip image varies for different resolutions, we use instance normalization [37] during training so that the mean and variance are approximated per image.

As is shown in Figure 2, two prediction layers are proposed to learn the target boundary in strip image to account for the fact that multiple true boundaries may be present in a single column of the strip image. Firstly, we extract the last upsampling layer to predict all potential boundaries. This encourages the network to learn boundary features within the strip image. To predict the target boundary, we add a learnable selection layer to pick up the target boundary from potential boundaries. The input to the selection layer is the initial prediction, and we apply column-wise softmax to the output of the selection layer as a confidence score for the initial prediction. Finally, the target boundary is computed by the multiplication between the initial prediction and the selection score. The selection layer also smooths the initial prediction, analogous to the non-maximum suppression in Canny edge detection [9]. Formally,

$$s = x \odot m, \tag{2}$$

where $\odot$ denotes pixel-wise multiplication, s denotes the final prediction, x denotes the initial prediction which applies Sigmoid activation to the output of the last upsampling layer and m is the softmax activated output of the selection layer.

3.3. Loss Function

Our basic loss function for the initial and final boundary prediction is a weighted l_1 loss to differentiate the boundary from non-boundary pixels. Formally,

$$L_{\text{e}} = \beta \sum_{(i,j)\in Y_+} |y_{ij} - s_{ij}| + (1-\beta) \sum_{(i,j)\in Y_-} |y_{ij} - s_{ij}|, \tag{3}$$

where Y_+ and Y_- denote boundary and non-boundary pixels, respectively. $\beta = |Y_-|/|Y|$ denotes the weight to balance the label and $|Y|$ denotes the total number of pixels in strip mask. s_{ij} denotes the prediction and y_{ij} denotes the binary ground truth at position (i, j) in the strip image.

In addition, we adapt Dice loss [40] to boundary prediction to encourage intersection between prediction and ground truth:

$$L_{\text{dice}} = 1 - \frac{2\sum s_{ij} \times y_{ij} + \epsilon}{\sum s_{ij} + \sum y_{ij} + \epsilon}, \tag{4}$$

where ϵ denotes a small constant to avoid zero division. The loss aims to maximize the intersection over union between the prediction and ground truth.

3.3.1 Boundary Distance Loss

For boundary prediction, a closer prediction to the boundary ground truth is preferred. However, both weighted l_1 and dice loss are not sensitive to the distance from prediction to ground truth. Therefore, we introduce a boundary distance loss to measure the average distance between the predicted boundary and the ground truth to encourage closer prediction. Thanks to the strip domain which maps the regions along the normal direction in every column, the boundary distance can be calculated directly through the difference between the prediction and ground truth. Given the prior that only one boundary pixel exists in each column in the final strip mask, the boundary distance at every column can be measured by calculating the argmax difference at every column between the prediction and ground truth. Since argmax function is not differentiable, we approximate it through soft argmax before calculating the boundary distance and formulate the loss as

$$L_{\text{d}} = \frac{1}{W} \sum_{j=1}^{W} |\operatorname*{softarg}_i(s_{ij}) - \arg\max_i(y_{ij})|, \tag{5}$$

where W is the width of strip mask and the soft argmax in each column (normal direction) is computed as

$$\operatorname*{softarg}_i(s_{ij}) = \sum_{i=1}^{H} \left(\frac{|s_{ij}|}{||S_j||_1} \times i \right), \tag{6}$$

where $||S_j||_1$ is the l_1 normalization of s_{ij} at column j. Since the final prediction s_{ij} encourages a unimodal distribution according to Equation 2, this loss enforces the column-wise maximum activation of the final prediction to match with that in ground truth.

3.3.2 Matching Loss

Since the strip height is fixed during training, to introduce variance and avoid overfitting on specific strip height, we augment the data through cropping the strip height. Starting from a large height, we crop the strip to a shorter one and make a new prediction. For consistency, the overlapped regions between original and the cropped strip should have the same initial prediction since all potential boundaries are predicted. Formally, we take a l_1 loss between the cropped and original initial prediction to calculate the matching loss,

$$L_{\text{m}} = \frac{1}{|Y_{crop}|} \sum_{(i,j)\in Y_{crop}} |x'_{ij} - x_{ij}|, \tag{7}$$

where Y_{crop} is the cropped region of original mask Y and x'_{ij} is the new initial prediction for the cropped strip image. In addition, this loss also helps the network learn to ignore spurious edges detected near the border of the strip.

3.3.3 C0 Continuity Regularization

Additionally, we add a C0 continuity regularization to the final prediction to enforce a continuous prediction. Ideally,

at most one boundary pixel is allowed at every column in the final prediction, so the prediction is C0 continuous if the maximum activated position of every column is C0 continuous. Specifically, we compute the soft argmax of every column, calculate a marginal difference between nearby argmax columns and penalize the position within a window size where prediction becomes discontinuous. Formally,

$$L_{\text{C0}}=\frac{1}{W}\sum_{j=1}^{W}P(\max(0,|\operatorname*{softarg}_{i}(s_{ij})-\operatorname*{softarg}_{i}(s_{i,j+1})|-v)), \tag{8}$$

where v denotes the margin value and P denotes the max-pooling with a fixed kernel size so that all pixels within the range get penalized. s_{iW+1} is replicated by s_{i1} for calculation. This loss serves as a self regularization as no ground truth label is required.

The total loss function is therefore,

$$L_{\text{total}} = L_e + L_{dice} + \lambda_1 L_d + \lambda_2 L_m + \lambda_3 L_{C0}, \tag{9}$$

where $\lambda_1, \lambda_2, \lambda_3$ are hyper-parameters to adjust the weight of each loss. L_e is applied to both the initial and final prediction. L_m is only applied to the initial prediction and L_{dice}, L_d, L_{C0} are applied only to the final prediction. With the total loss function, a closer prediction is preferred and the network draws attention to the target boundaries.

3.4. Strip Reconstruction

To make a prediction on the HR image, a mapping between the predicted strip boundaries and the full HR mask is required at inference. For every pixel in the strip image, the corresponding coordinates in the HR image are recorded for reconstruction. Given the raw prediction, we optimize the path with a dynamic programming similar to seam carving [5] and find the path with minimum energy. We minimize the function

$$E_{ij} = -s_{ij} - \frac{|\partial I(i,j)|}{max(|\partial I|)}, \tag{10}$$

where $|\partial I(i,j)|$ denotes the magnitude of the image gradients at (i,j). The algorithm searches for the energy cost for neighborhood pixels and finds the path with a minimum energy cost, which indicates the boundary path with the highest probability. We then connect the original coordinates of the final path in the full mask to form the full prediction.

At inference, the flexible input dimension of our framework enables different strip sizes for different images. Benefitting from it, we determine the width of strip, which reflects the number of sampling points along the boundary, by multiplying the LR boundary length with the scale factor. We fix the height of strip with the assumption that all target boundaries are involved, and an adaptive height adjustment strategy is also discussed in Section 4.6. For objects containing multiple contours due to complex topology, the prediction is made on each contour separately.

3.5. Implementation Details

We generate the spline curve efficiently from the binary mask using the scipy function 'splprep' after extracting contours. To guarantee a consistent sign for the normals, we extract strip images from closed contours. The starting point of strip is not deterministic so that no bias is introduced in training. The final ground truth strip boundary mask is obtained by taking the gradient of the ground truth segmentation mask after removing any isolated noisy boundaries. Additionally, we randomly add small shifts to the spline representation to introduce position variation of the target boundary in strip image during training. Our framework is implemented in Pytorch. The encoder consists of 4 3×3 convolutional layers and the decoder consists of 4 upsampling layers. The selection layer consists of another convolutional layer with 3×3 kernel size. The activation function is ReLU [15] for all encoder and decoder layers. We use instance normalization for all normalization layers to enable flexible input size at inference. During training, the input strip dimension is fixed as 80×4096. We train the network for 70 epochs with batch size 6 on an NVIDIA GeForce TITAN P6000. We use Stochastic Gradient Descent (SGD) as optimizer and the initial learning rate is 0.1. The learning rate decays by a factor of 10 after every 20 epochs. The momentum is set to 0.9 and weight decay is set to 0.0005. λ_1, λ_2 and λ_3 are set to be 0.1, 20 and 1 empirically. We crop strip image by half to obtain Y_{crop} for matching loss and the maxpooling kernel size for C0 continuity regularization is 11. The margin in C0 continuity regularization is set to 1. Horizontal flipping is applied as data augmentation.

4. Experiments

We evaluate our approach on two HR datasets which provide both low and high resolution ground truth in Section 4.2, and then analyze the importance of each components in our framework in Section 4.3. We also provide memory and speed comparison in Section 4.4.

4.1. Datasets and Metrics

For our experiments, we need a dataset with highly accurate pixel-level HR annotation. Unfortunately, most current datasets are low resolution and many provide inaccurate polygon boundaries as ground truth annotations. We found DAVIS [31] to provide accurate enough results with a resolution that is usable for our needs. To better evaluate the results at large scaling factors, we introduce a new dataset—PixaHR. We describe these datasets below.

DAVIS 2016 [31]: A benchmark for video segmentation which consists of 50 classes with precise annotations in both 480P and 1080P. To enlarge the scale factor, we down sample the 480P mask by a factor of 2, train our approach on the 30-class 1080P training set with 240P LR masks and test on

Dataset	**DAVIS 2016** [31] 4×		**PixaHR** 8×		**PixaHR** 16×		**PixaHR** 32×	
Metrics	F(0 pix)	F(1 pix)	F(1 pix)	F(2 pix)	F(1 pix)	F(2 pix)	F(1 pix)	F(2 pix)
Bilinear Upsampling	0.171	0.521	0.116	0.194	0.15	0.187	0.07	0.106
Grabcut [33]	0.232	0.541	0.063	0.121	0.020	0.053	0.0	0.0
Dense CRF [22]	0.268	0.702	0.278	0.434	0.245	0.389	0.142	0.227
Bilateral Solver [6]	0.274	0.569	0.207	0.277	0.185	0.247	0.156	0.216
Curve-GCN [26]	0.076	0.160	0.021	0.033	0.018	0.028	0.012	0.028
DELSE [39]	0.271	0.531	0.096	0.133	0.086	0.132	0.080	0.130
STEAL [2]	0.171	0.348	0.282	0.457	0.151	0.255	0.09	0.144
JBU [21]	0.175	0.447	0.140	0.231	0.117	0.184	0.055	0.090
Guided Filtering [17]	0.129	0.349	0.121	0.195	0.092	0.145	0.060	0.097
Deep GF [41]	0.193	0.461	0.286	0.420	0.175	0.269	0.09	0.141
U-Net boundary	0.320	0.656	0.170	0.297	0.139	0.197	0.068	0.108
U-Net strip (baseline)	0.303	0.710	0.334	0.455	0.303	0.425	0.267	0.357
Ours	**0.423**	**0.788**	**0.416**	**0.508**	**0.396**	**0.498**	**0.330**	**0.447**

Table 1. Boundary-based F score comparison. The scale factor between low and high resolution image is 4 on DAVIS 2016 and 8, 16, 32 on PixaHR. For DAVIS 2016, the pixel dilation is 0 and 1 and for PixaHR is 1 and 2 instead.

20-class 1080P testing set. The scale factor is 4.5 for this experiment. The results are evaluated frame by frame.

PixaHR: To evaluate more realistic scenarios, we create a PixaHR dataset. It contains 100 images with average resolution $7k \times 7k$ (ranging from $5k \times 5k$ to $10k \times 10k$) collected from public photograph website Pixabay [1]. We manually annotate the object boundary in the HR images, downsample the HR mask by $8\times$, $16\times$ and $32\times$ and obtain binary LR mask for evaluation. The photos were uploaded by public users and have diverse contents. We apply our model that was trained on DAVIS to this dataset for evaluation.

Metrics: We use boundary-based F score introduced by Perazzi *et al*. [31] for evaluation, which is designed to evaluate the boundary quality of segmentation. As it allows changing pixel tolerance by dilation, we set 0 and 1 pixel dilation on DAVIS, and 1 and 2 pixel on PixaHR dataset to measure how close the prediction is to the ground truth.

4.2. Main Results

For upsampling based approaches, we compare our approach with **Bilinear Upsampling**, **Bilateral Solver [6]**, Joint Bilateral Upsampling [21] (**JBU**), **Guided Filtering [17]** and **Deep GF [41]**. The boundary is obtained by taking the gradient of the upsampled mask. For boundary refinement approaches, we compare with **Grabcut [33]**, **Dense CRF [22]** and **STEAL [2]** using upsampled mask as initialization. For active contour methods, the baselines are **Curve-GCN [26]** and **DELSE [39]**, and predictions on PixaHR are made in LR and upsampled to original resolution since the whole boundary region is required at inference. Learning based approaches are trained or fine-tuned on the training set of DAVIS and evaluated directly on all datasets. More details about baselines are provided in **supplementary material**. In addition, we also compare our own implemented baselines as below:

- **U-Net boundary**: We train U-Net directly on the full resolution images on DAVIS for boundary prediction. We concatenate both the full resolution image and upsampled masks as input so that the network learns to refine the coarse masks. The loss function is a weighted binary cross entropy following Xie *et al*. [42]. Similarly, we also add deep supervision and fuse all intermediate layers to obtain the final prediction. The prediction is made patch-by-patch with patch size 1920×1080 on PixaHR dataset.
- **U-Net strip (baseline)**: Our baseline method which learns to directly predict the target boundary on strip image. Only weighted $l1$ loss is used as loss function.
- **Ours**: Our full model which applies selection layer to predict the boundary in strip images with our boundary distance loss, matching loss and C0 continuity regularization.

Table 1 exhibits our advantage over the baselines. For the DAVIS dataset, a simple upsampling yields a boundary shift from the ground truth and thus performs poorly. Grabcut and dense CRF are segmentation based and thus yield worse performance than ours. Even though other methods including bilateral solver, JBU and Deep GF leverage the low resolution mask, they are designed for general upsampling instead of for boundary refinement and prediction. Curve-GCN fits the curve from the predicted control points which cannot generate as precise a boundary as ours. DELSE moves the contour along the gradient of its energy function, but is less robust than our approach which predicts the target boundary pixels. Additionally, our approach outperforms STEAL as the scale factor increases, indicating the active alignment in STEAL may not be accurate enough for pixel-level boundary prediction. Compared with U-Net boundary, predicting the boundary in strip image (U-Net strip) yields a slightly better performance, per-

Dataset	DAVIS 2016	PixaHR 16×
Metrics	F(0 pix)	F(1 pix)
U-Net strip	0.303	0.303
U-Net strip dice	0.323	0.320
U-Net strip dice+ selection	0.372	0.328
U-Net strip dice+selection+BD	0.390	0.342
Our w/o matching	0.405	0.365
Ours	0.423	0.396

Table 2. Ablation analysis on two datasets. Each entry is the boundary-based F score tested on individual dataset.

Methods	Memory (MB)	Speed (s/image)
Bilinear Upsampling	-	0.01/0.02
Grabcut [33]	-	5.17/320
Dense CRF [22]	-	3.22/310
Bilateral Solver [6]	-	4.18/158
JBU [21]	-	0.08/5.71
Guided filtering [17]	-	0.08/16.1
Deep GF [41]	-	0.07/3.95
STEAL [2]	7775/7959	43.1/4231
Curve-GCN [26]	17330/17330	0.93/75.2
DELSE [39]	17771/17771	1.02/20.4
U-net boundary	17000/17000	0.31/24.5
Ours	3300/3300	0.28/2.51

Table 3. Memory and speed comparison. Each entry is the memory or speed on DAVIS 2016/PixaHR dataset. We only compare the memory usage among learning-based approaches.

haps because the strip image narrows down the search space for target boundary. As expected, with our selection layer and proposed losses, we boost the performance further by better determining the target boundaries from other potential boundaries. A similar tendency is observed on PixaHR dataset. Note that in large scale factor 32, most of the methods fail to make close predictions to the ground truth while our method still has a relatively stable performance.

4.3. Ablation Analysis

We analyze the importance of each component in our framework as listed below:

- **U-Net strip dice**: Adding dice loss to the baseline.
- **U-Net strip dice + selection**: Adding dice loss and selection layer to the baseline.
- **U-Net strip dice + selection + BD**: Adding dice, boundary distance loss and selection layer to the baseline.
- **Ours w/o matching**: Adding additional C0 regularization. It is our full model without the matching loss.

Table 2 summarizes the comparison result. Starting from our baseline U-Net strip, adding dice loss encourages more intersection with the ground truth boundary and thus yields better performance. Comparing **U-Net strip + dice** with **U-Net strip + dice + selection**, the selection layer boosts the performance on DAVIS by a large margin, indicating it effectiveness in suppressing the noisy boundaries and

Dataset	PixaHR 32×
Metrics	F(1 pix)
Ours	0.330
Ours adaptive 1 segment	0.353
Ours adaptive 2 segments	0.365

Table 4. Strip height selection comparison on PixaHR 32×.

smoothing the final prediction. Also, with the boundary distance loss the network learns to have closer prediction. With C0 regularization (**Ours w/o matching**), the network filters out false positive boundaries by making a continuous prediction. Finally, the performance further improves with the matching loss because the network makes a consistent prediction over different strip heights to avoid overfitting.

4.4. Memory and Speed Comparison

Since we only extract a strip image for prediction, our approach is efficient in both memory and computation. Table 3 compares our memory overhead and speed performance with baselines. Over all, our computation and memory requirement is relatively small. Our memory requirement is smaller than other learning based approaches. Note that for U-Net boundary and STEAL, the prediction on PixaHR is made patch-by-patch due to the high resolution.

More specifically, the main computation in our approach lies in strip reconstruction. *e.g.*, for a 1920×1080 DAVIS image with around 3200 pixels along the boundary, our strip image creation takes 0.08s, prediction process takes 0.06s and the strip reconstruction takes 0.14s. A similar computation percentage is observed on PixaHR also.

4.5. Qualitative Results

We show visualization comparisons in Figure 4. It is clear that our approach produces more accurate boundariers than the other methods. To further show the effectiveness of our approach on refining the boundaries given LR or coarse masks, we provide qualitative results on COCO where only polygonal boundary ground truth is provided. We directly extract strip image using the coarse annotation on COCO, and visualize the prediction in Figure 5. Comparing with other approaches, our method provides more accurate boundaries, indicating the potential application of our approach to help refine the coarse boundaries. For more visualization results, please see **supplementary material**.

4.6. Strip Height Adaptation

We predict the target boundary in the strip image under the assumption that the target boundary exists within the pre-defined height range, however, it might not hold true especially for a large scale factor. While one solution is to predefine a larger height for strip image creation, we propose to progressively increase the height and regenerate strip image

Figure 4. Qualitative results on PixaHR 32×. Rows from top to down are the results of Dense CRF, STEAL, Ours and the Ground truth. We show the entire boundary (green color) result first and enlarge the blue bounding box region for comparison (boundaries are whitened).

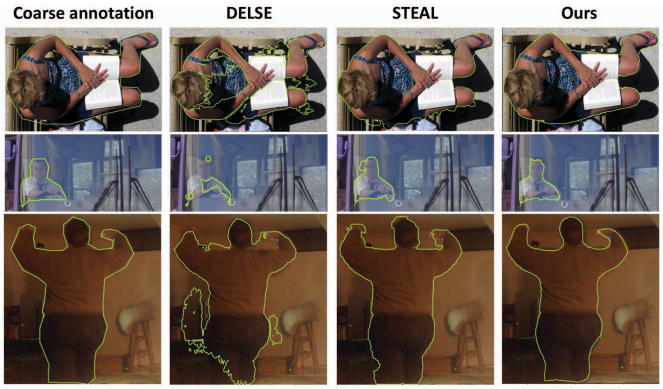

Figure 5. Qualitative results on COCO. Columns from left to right are coarse annotation, DELSE [39], STEAL [2] and Ours.

to make new predictions at inference. Specifically, we increase the height of strip image until the summation of the final prediction score decreases. Furthermore, height adjustment is more flexible by dividing the whole contour into several segments and adjusting them independently. The results are shown in Table 4. The comparison between **Ours** and **Ours adaptive 1 segment** indicates the effectiveness to have a flexible height. The performance increases further when dividing the whole contour into 2 segments which allows variable height for different regions.

5. Conclusion

In summary, this paper presents a novel strategy to handle HR boundary refinement computationally and memory efficiently given LR precise masks. To save memory, we propose to extract boundary regions along the upsampled boundary spline to form a strip image and make prediction within this strip image. To focus on the target boundaries in strip image, boundary distance, matching loss and C0 continuity regularization have been proposed. Extensive experiments on both public and our newly created dataset demonstrate the effectiveness of the proposed approach. However, the current approach still has difficulty predicting complicated topology and soft boundary regions. A smarter adaptive strip height adjustment for every pixel might be a potential solution, which is left for future research.

Acknowledgement

This work was partly funded by Adobe. The authors acknowledge the Maryland Advanced Research Computing Center (MARCC) for providing computing resources.

References

[1] Pixabay. https://pixabay.com. 6

[2] David Acuna, Amlan Kar, and Sanja Fidler. Devil is in the edges: Learning semantic boundaries from noisy annotations. In *CVPR*, 2019. 1, 2, 6, 7, 8

[3] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *CVPR*, 2018. 2

[4] Luis Álvarez, Luis Baumela, Pedro Henríquez, and Pablo Márquez-Neila. Morphological snakes. In *CVPR*, 2010. 2

[5] Shai Avidan and Ariel Shamir. Seam carving for content-aware image resizing. In *TOG*, 2007. 5

[6] Jonathan T Barron and Ben Poole. The fast bilateral solver. In *ECCV*, 2016. 3, 6, 7

[7] Rodrigo Benenson, Stefan Popov, and Vittorio Ferrari. Large-scale interactive object segmentation with human annotators. In *CVPR*, 2019. 1

[8] Yuri Y Boykov and M-P Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *ICCV*, 2001. 2

[9] John Canny. A computational approach to edge detection. *TPAMI*, 1986. 1, 4

[10] Lluis Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a polygon-rnn. In *CVPR*, 2017. 2

[11] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018. 1

[12] Wuyang Chen, Ziyu Jiang, Zhangyang Wang, Kexin Cui, and Xiaoning Qian. Collaborative global-local networks for memory-efcient segmentation of ultra-high resolution images. In *CVPR*, 2019. 1, 3

[13] Dominic Cheng, Renjie Liao, Sanja Fidler, and Raquel Urtasun. Darnet: Deep active ray network for building segmentation. In *CVPR*, 2019. 2

[14] Ruoxi Deng, Chunhua Shen, Shengjun Liu, Huibing Wang, and Xinru Liu. Learning to predict crisp boundaries. In *ECCV*, 2018. 2

[15] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *AISTATS*, 2011. 5

[16] Jianzhong He, Shiliang Zhang, Ming Yang, Yanhu Shan, and Tiejun Huang. Bi-directional cascade network for perceptual edge detection. In *CVPR*, 2019. 1

[17] Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. *TPAMI*, 2012. 3, 6, 7

[18] Hexiang Hu, Shiyi Lan, Yuning Jiang, Zhimin Cao, and Fei Sha. Fastmask: Segment multi-scale object candidates in one shot. In *CVPR*, 2017. 1

[19] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *IJCV*, 1988. 1, 2

[20] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017. 2

[21] Johannes Kopf, Michael F Cohen, Dani Lischinski, and Matt Uyttendaele. Joint bilateral upsampling. In *ToG*, 2007. 3, 6, 7

[22] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *NeurIPS*, 2011. 2, 6, 7

[23] Hoang Le, Long Mai, Brian Price, Scott Cohen, Hailin Jin, and Feng Liu. Interactive boundary prediction for object selection. In *ECCV*, 2018. 1, 2

[24] Yin Li, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. Lazy snapping. *ToG*, 2004. 2

[25] Zhuwen Li, Qifeng Chen, and Vladlen Koltun. Interactive image segmentation with latent diversity. In *CVPR*, 2018. 1

[26] Huan Ling, Jun Gao, Amlan Kar, Wenzheng Chen, and Sanja Fidler. Fast interactive object annotation with curve-gcn. In *CVPR*, 2019. 1, 2, 6, 7

[27] Yun Liu, Ming-Ming Cheng, Xiaowei Hu, Kai Wang, and Xiang Bai. Richer convolutional features for edge detection. In *CVPR*, 2017. 1, 2

[28] Diego Marcos, Devis Tuia, Benjamin Kellenberger, Lisa Zhang, Min Bai, Renjie Liao, and Raquel Urtasun. Learning deep structured active contours end-to-end. In *CVPR*, 2018. 2

[29] Andreas Opelt, Axel Pinz, and Andrew Zisserman. A boundary-fragment-model for object detection. In *ECCV*, 2006. 1

[30] Stanley Osher and James A Sethian. Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations. *JCP*, 1988. 2

[31] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016. 5, 6

[32] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 3

[33] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *TOG*, 2004. 1, 6, 7

[34] Christian Rupprecht, Elizabeth Huaroc, Maximilian Baust, and Nassir Navab. Deep active contours. *arXiv preprint arXiv:1607.05074*, 2016. 2

[35] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019. 1

[36] Meng Tang, Lena Gorelick, Olga Veksler, and Yuri Boykov. Grabcut in one cut. In *ICCV*, 2013. 1

[37] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *CoRR*, 2016. 4

[38] Tiantian Wang, Lihe Zhang, Shuo Wang, Huchuan Lu, Gang Yang, Xiang Ruan, and Ali Borji. Detect globally, refine locally: A novel approach to saliency detection. In *CVPR*, 2018. 1

[39] Zian Wang, David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Object instance annotation with deep extreme level set evolution. In *CVPR*, 2019. 2, 6, 7, 8

[40] Ken CL Wong, Mehdi Moradi, Hui Tang, and Tanveer Syeda-Mahmood. 3d segmentation with exponential logarithmic loss for highly unbalanced object sizes. In *MICCAI*, 2018. 4

[41] Huikai Wu, Shuai Zheng, Junge Zhang, and Kaiqi Huang. Fast end-to-end trainable guided filter. In *CVPR*, 2018. 1, 3, 6, 7

[42] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *CVPR*, 2015. 1, 2, 6

[43] Hongyu Xu, Xutao Lv, Xiaoyu Wang, Zhou Ren, Navaneeth Bodla, and Rama Chellappa. Deep regionlets for object detection. In *ECCV*, 2018. 1

[44] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas S Huang. Deep interactive object selection. In *CVPR*, 2016. 1

[45] Jimei Yang, Brian Price, Scott Cohen, Honglak Lee, and Ming-Hsuan Yang. Object contour detection with a fully convolutional encoder-decoder network. In *CVPR*, 2016. 2

[46] Zhiding Yu, Weiyang Liu, Yang Zou, Chen Feng, Srikumar Ramalingam, BVK Vijaya Kumar, and Jan Kautz. Simultaneous edge alignment and learning. In *ECCV*, 2018. 1, 2

[47] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *CVPR*, 2018. 1

[48] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. Icnet for real-time semantic segmentation on high-resolution images. In *ECCV*, 2018. 1, 3

[49] Ting Zhao and Xiangqian Wu. Pyramid feature attention network for saliency detection. In *CVPR*, 2019. 1

[50] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation. In *DLMIA*. 2018. 3