This WACV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Lightweight Portrait Matting via Regional Attention and Refinement

Yatao Zhong
Microsoft
yazhong@microsoft.com

Ilya Zharkov
Microsoft
zharkov@microsoft.com

Abstract

We present a lightweight model for high resolution portrait matting. The model does not use any auxiliary inputs such as trimaps or background captures and achieves real time performance for HD videos and near real time for 4K. Our model is built upon a two-stage framework with a low resolution network for coarse alpha estimation followed by a refinement network for local region improvement. However, a naive implementation of the two-stage model suffers from poor matting quality if not utilizing any auxiliary inputs. We address the performance gap by leveraging the vision transformer (ViT) as the backbone of the low resolution network, motivated by the observation that the tokenization step of ViT can reduce spatial resolution while retain as much pixel information as possible. To inform local regions of the context, we propose a novel cross region attention (CRA) module in the refinement network to propagate the contextual information across the neighboring regions. We demonstrate that our method achieves superior results and outperforms other baselines on three benchmark datasets while only uses $1/20$ of the FLOPS compared to the existing state-of-the-art model.

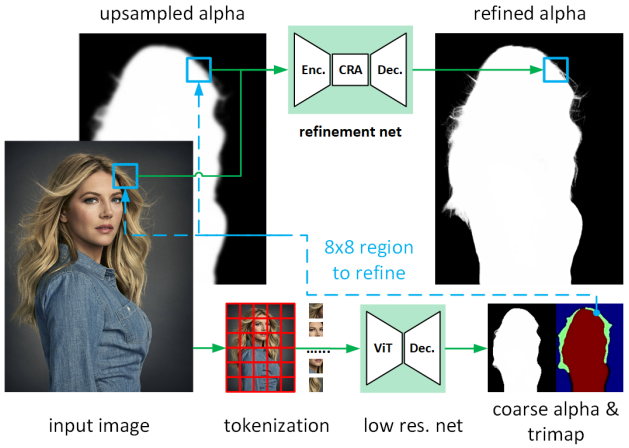

Figure 1. An overview of our method. 1) An input image is first tokenized before being fed to a low resolution network that consists of a ViT backbone and a decoder. 2) Coarse alpha is upsampled to full resolution and concatenated with the input image. 3) Regions of uncertainty are selected from the estimated trimap and cropped from the concatenated RGBA image. 4) Cropped regions run through a refinement network that features a cross region attention (CRA) module to obtained the refined alpha.

1. Introduction

Image matting is one of the most studied topics in computer vision. Formally a matting problem is formulated as

$$I = \alpha F + (1 - \alpha)B. \tag{1}$$

The goal is to solve for the alpha matte α, but the foreground F and background B are also unknown. Therefore, this is a highly under-constrained problem, which oftentimes requires some priors. One commonly used prior is a user provided trimap [1,3,4,6,11,14,18,19,25,29], where each pixel is categorized as "definite foreground", "definite background" or "unknown". However, trimaps require user interaction and are time-consuming to obtain, hence difficult to be deployed in a fully automated system. Another recently proposed prior is an additional background image [23]. However, capturing a second image under the same conditions (e.g., lighting and shadow) is not always possible and the background image is only useful if it is well aligned with the input image. There have also been efforts to remove all auxiliary inputs and predict the alpha mattes directly from input images [2, 12, 21, 30]. Approaches of this type are typically learning based and have been demonstrated to perform reasonably well even without any priors provided.

Nevertheless, all of these methods operate at full resolution, making them extremely compute-intensive and impossible to be deployed in real applications for high resolution portrait matting (e.g., HD and 4K). In this work, we aim at reducing the computation while also retaining the matting quality. We present a lightweight model that estimates the alpha matte directly from the image without any user interactions or auxiliary inputs such as trimaps or background

captures. Our model is built upon a prior observation that a portrait alpha matte is dominated by "definite foreground" ($\alpha = 1$) and "definite background" ($\alpha = 0$), which can be obtained by upsampling the estimated alpha matte from a low resolution model. Only a few uncertain regions around the boundaries ($0 < \alpha < 1$) need to be refined. Therefore, the proposed model consists of two stages: an initial stage for low resolution alpha estimation and a second stage for full resolution refinement. Fig. 1 gives an overview of our method.

However, we find naively adopting the two-stage framework leads to inferior results due to the missing auxiliary inputs which we intentionally eliminate. To address the performance gap, we leverage the vision transformer (ViT) as the backbone in the low resolution model. As opposed to image downsampling, image tokenization in the ViT is a better choice for reducing spatial resolution because it does not lose pixel information. Since the the refinement network operates on extracted local regions, to inform it of the context, a straightforward design choice would be to reuse the upsampled features from the low resolution network [15]. However, this adds to the compute budget by doing upsampling at high resolution. Therefore, we opt for an inverted process by first extracting local regions followed by gathering the context. To recover the contextual information, we propose a novel cross region attention (CRA) module, which propagates the information across the k nearest neighbors of each region through multi-head attention with a learnable lookup table for relative positional encoding. We demonstrate that, with all the aforementioned designs, our model outperforms other baselines by a large margin on the P3M and PPM datasets. We also show that our model is able to retain the matting quality using only $1/20$ of the FLOPS compared to the existing state-of-the-art model [12].

In summary, our work has the following contributions.

- We leverage the tokenization step of ViT to reduce spatial resolution while retain the full pixel information for coarse alpha estimation.
- We invert the order of computing contextual features and extracting local regions to avoid feature upsampling at high resolution to save computation.
- We propose a novel cross region attention (CRA) module to capture the contextual information across k nearest neighbors of each region.
- We conduct extensive experiments to demonstrate the effectiveness and efficiency of our model: achieving the state-of-the-art performance while using minimal FLOPS.

2. Related Work

Traditional matting. Traditional matting algorithms [1, 3, 4, 7, 11, 25] are derived from the matting equation Eq. 1. The goal is to solve for the alpha matte α, but at the same time one needs to also solve for the foreground F and background B. Since this is an ill-posed problem with only the observed image I being provided, a common practice is to use trimaps as constraints. [4] formulates the problem in a Bayesian framework and solves it using maximum a posteriori (MAP) estimation. [25] formulates matting as a problem of solving Poisson equations using matte gradient field. Both end up being an iterative solution. [11] proposes the first closed form solution, but their method is memory and compute intensive because the involvement of a large sparse linear system. [7] accelerates [11] by using large kernel Laplacian and adaptive kernel sizes obtained from KD-tree segmentation on trimaps. Other works [1, 3] improve [11] by removing the local color line model assumption. They use the global pixel affinities to propagate alpha values in trimaps from known regions to unknown regions.

Learning based matting. With the advance in deep learning, many recent approaches [14, 18, 19, 29] have shifted to a learning based paradigm, where a model takes the image and trimap as input and learns to predict the alpha matte. DIM [29] is the pioneer that leverages deep neural networks in the task of image matting. AlphaGAN [19] improves [29] by training the model with an adversarial loss. GCA [14] introduces a guided contextual attention module by computing the correlation between unknown regions. IndexNet [18] utilizes learned indices in the decoder for upsampling to guide the matte generation.

Trimap-free matting. There have also been efforts trying to eliminate the dependence on trimaps. [15, 23] propose to capture an additional background image as an auxiliary input for image matting. SHM [2] predicts the alpha matte by fusing a self-learned trimap and a raw alpha matte. LF [30] employs a similar fusion concept by estimating the foreground and background probability maps and blending them with self-learned weights. HATT [21] uses a spatial and channel-wise attention module to integrate low level and high level features. P3M-Net [12] adopts a multi-task framework by predicting trimaps and alpha mattes at multi-resolutions and uses a stack of integration modules to exchange feature information.

3. Method

The proposed two-stage framework (shown in Fig. 1) proceeds as follows. The low resolution network predicts a coarse alpha matte and a coarse trimap. Next we extract uncertain regions using the coarse trimap and crop the selected regions from the input image and upsampled coarse

alpha. The cropped patches then pass through the refinement network to obtain the refined alpha patches. Finally, we replace the refined alpha patches back in the upsampled coarse alpha to complete the full alpha. Below we illustrate each of the steps and explain our design choices. For brevity of writing, we use the notation $\mathfrak{R}_x$ to refer to a resolution at $\frac{1}{x}$ of the full resolution.

3.1. ViT as Low Resolution Backbone

Different from prior works that use additional inputs such as background captures [15, 23] or trimaps [6, 14, 18, 19,29], our models tackles portrait matting without any auxiliary inputs, which is significantly more challenging. In fact, as we will show in the experiments, simply adopting a CNN architecture results in poor matting quality if not given auxiliaries. We argue that, even for coarse alpha estimation, a higher input resolution could be beneficial to the overall better quality. Nonetheless, higher resolution inevitably adds more computation. This motivates us to think how we can improve it without resorting to increased compute budget. We therefore propose to transform an image $I \in \mathbb{R}^{H \times W \times C}$ to a grid of non-overlapping patches of $\mathbb{R}^{\frac{H}{P} \times \frac{W}{P} \times P^2C}$, where $\frac{H}{P} \times \frac{W}{P}$ corresponds to the grid resolution and P^2C is the channel dimension. This is often referred to as *pixel-unshuffle* or *space-to-depth*. As opposed to downsampling, pixel-unshuffle preserves the original pixel information when reducing the spatial resolution.

Table 1. Results of different downsampling and pixel-unshuffle strategies. d is the downsampling rate and p is the pixel-unshuffle patch size. We evaluate the models on the P3M-500 test data. For more details about the test data, please refer to Sec. 4.1.

			P3M-500-NP		P3M-500-P	
No.	Method	FLOPS	SAD	Grad	SAD	Grad
A	Resnet-50, d=4	24.7G	14.25	12.60	14.11	14.69
B	Resnet-50, d=2	59.9G	13.43	11.77	12.05	12.58
C	Resnet-50, p=4	28.2G	15.21	12.71	16.10	15.22
D	Swin-T, d=2, p=8	18.6G	10.89	10.72	10.39	12.73

To verify the advantage of pixel-unshuffle over downsampling, we test several models with different pixel-unshuffle and downsampling strategies and evaluate them on a benchmark dataset. We summarize their SAD (sum of absolute difference) and Grad (gradient difference) in Tab. 1. Resnet-50 [8, 9] is used as the low resolution backbone for models A, B and C. A sequence of upsampling and conv layers are used in the decoder to keep the low resolution output at $\mathfrak{R}_8$. All models share the same refinement stage, which we will discuss later in Sec. 3.2 and 3.3.

Comparing A and B, we can see that increasing the resolution of the low resolution network improves the accuracy. Nevertheless, this requires considerably more compute budget. To retain accuracy without adding more computation, we resort to pixel-unshuffle (C). However, we see C underperforms A. This seemingly contradicts the hypothetical strength of using pixel-unshuffle, but we argue that the performance drop, in fact, can be explained by the usage of large kernels. Many prior arts [8, 9, 24, 26, 31] have demonstrated the success of using small kernels because they help preserve the locality and translation invariance of CNNs. Large kernels break these nice properties, making CNNs suffer from poor generalization. In the case of C in Tab. 1, Resnet-50 already starts with a relatively large kernel (7×7). When used with pixel-unshuffle with a patch size of 4, the effective kernel size of the first layer becomes 28×28, which is obviously too large to be applied to a CNN.

Due to limitation of CNNs with large kernels, we opt for ViT as the low resolution backbone. ViT comes naturally a better choice for low resolution prediction because the first step of ViT — image tokenization — is equivalent to pixel-unshuffle, which can effectively reduce the spatial resolution while retain the full pixel information. Specifically, we choose the Swin-T [17] as the low resolution backbone. As shown in Tab. 1, model D uses a downsampling rate of 2 followed by pixel-unshuffle with a patch size of 8, which results in a ×16 reduction in resolution. With the proposed design principal, model D achieves a remarkable improvement in terms of both accuracy and FLOPS.

3.2. Refinement Stage

Modern neural network architectures for dense prediction tasks typically rely on a pyramid of upsampled features for global information. Similarly, for a refinement network to receive contextual information, the most straightforward idea would be to upsample the features to input resolution before extracting the refinement regions. This way the cropped regions are informed of their context. However, upsampling at high resolution (e.g. HD or 4K) is both memory and compute intensive. We propose an alternative to eliminate the heavy feature upsampling.

Our refinement stage avoids reusing any deep features from the low resolution network. The low resolution network only predicts a coarse alpha matte and a coarse trimap. Like a traditional trimap, the predicted trimap has three classes: “definite foreground”, “definite background” and “uncertain”. We encode it as a 3-channel softmax output. Since no ground truth trimaps are available at training time, we apply morphological operations with heuristics to create the target trimaps from the ground truth alpha mattes. At inference time, we select the pixels predicted as “uncertain” as the regions of interest to be refined.

In the refinement stage, we first upsample the coarse alpha matte to full resolution. This op is lightweight compared to the heavy feature upsampling. The upsampled alpha matte is concatenated with the input image to form a 4-channel RGBA image. With the selected “uncertain” pixels

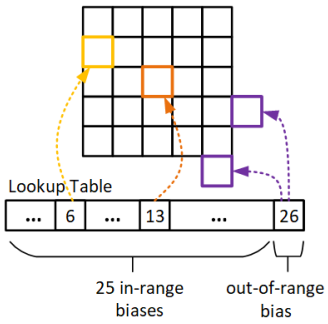

Figure 2. An illustration of relative positional bias encoding. This is an example with search range s=3, which ends up with a 5×5 search window and a lookup table of 25 biases for in-range positions and one extra bias for any out-of-range positions. The orange square denotes the center, which is encoded with the bias at the 13th slot. The yellow square denotes an in-range sample encoded with the 6th slot. The purple squares are two out-of-range samples, whose positional biases are given by the last table entry.

from the trimap, we locate the corresponding 8×8 regions in the RGBA image, which are cropped and fed to a tiny refinement network consisting of an encoder and a decoder. At last, we replace the respective 8×8 regions in the upsampled alpha with the refined crops to obtain the final alpha. Fig. 3a visualizes how the cropped regions run through the entire refinement stage.

3.3. Cross Region Attention

The aforementioned refinement stage only receives 8×8 local regions as input, inevitably losing the context. Thus a mechanism is needed to recover the context after region extraction. We therefore propose a novel cross region attention (CRA) module to capture the contextual information across the neighboring regions. CRA is inspired by the multi-head attention [5, 17, 27], but instead of consuming a sequence or regular grid of tokens, it operates on the k nearest neighbors (KNN) of a central token. Our proposed mechanism inverts the order of context collection and region extraction and has the advantage of eliminating the heavy feature upsampling as discussed in Sec. 3.2. Below we use "region" to refer to "token" since an extracted region is effectively a token.

KNN extraction. After identifying all "uncertain" regions from the trimap, for each region, we find the closest k regions as its KNNs (under the metric of Euclidean distance). We do pairwise comparisons at training time, but employ KD-tree as a faster search algorithm at inference time. The left part of Fig. 3a visualizes the locations of a region's KNNs.

Relative positional bias. The KNNs can potentially be scattered anywhere around the central region and distributed on a non-regular grid, so we need a way to encode their relative positions. We define a search range s on the image space. The relative positions on each axis are supposed to be in $[-s+1, s-1]$. This ends up with $(2s-1)^2$ possible relative positions within the search range. We encode the relative positions with a learnable lookup table $P \in \mathbb{R}^{(2s-1)^2+1}$. The first $(2s-1)^2$ entries encode all possible in-range positions. The last entry encodes any out-of-range positions. Each entry is used as the relative positional bias B in the attention formula:

$$\text{Attention}(Q, K, V) = \text{Softmax}(\frac{QK^T}{\sqrt{d}} + B)V, \qquad (2)$$

where d is the feature dimension; Q, K and V are the query, key and value respectively. Fig. 2 illustrates how the relative positional biases are encoded with a lookup table.

Cross region attention. After we obtain the features of all extracted regions from the refinement network's encoder, we locate the KNNs of each region and query their relative positional biases from the lookup table P. Let $f_i \in \mathbb{R}^d$ denote the feature of region i and $\{i_1, i_2, \cdots, i_k\}$ denote the k nearest neighbors of region i. For each region i, we feed the features $[f_i, f_{i_1}, f_{i_2}, \cdots, f_{i_k}] \in \mathbb{R}^{(k+1)\times d}$ of this region and its KNNs, along with their relative positional biases $B \in \mathbb{R}^{k+1}$, to two consecutive attention blocks, shown in Fig. 3b. Note that the second block does not need to do pairwise attention QK^T across all $k+1$ regions. Instead, it only computes the attention between the central region and its KNNs by Q_0K^T, where Q_0 is the query of the central region i. Finally, the output feature goes to the refinement network's decoder to obtain the refined alpha matte of region i.

3.4. Training

We train the low resolution network and refinement network end-to-end at the same time. During training, we apply various data augmentation strategies such as horizontal flipping, cropping and affine transformation as well as color adjustment in hue, saturation and brightness. Since the Swin-T backbone inherently uses an effective output stride of 32 and a window size of 7 for its window attention, with the original implementation [17] the input image size is expected to be multiples of $7 \times 32 = 224$. We make a modification to accommodate input images of arbitrary sizes (but no less than 224×224) by padding any intermediate feature maps with zeros if their spatial sizes are not already dividable by 7 and masking out the padded regions when computing the attention. At training time, all images are resized to 896×896, but at inference time, the model accepts images of arbitrary resolutions. We refer readers to the supplementary material for a full list of training losses.

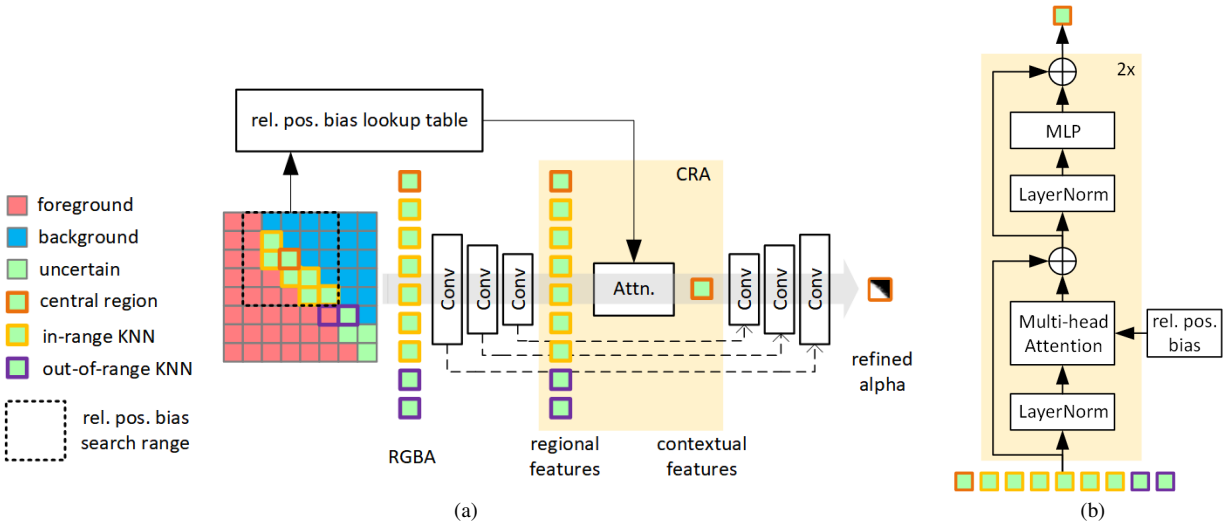

Figure 3. A visualization of the refinement stage. Each □ corresponds to a region of 8×8 pixels at full resolution. The example uses 8 nearest neighbors and a search range $s = 3$ for relative positional bias. (a) visualizes the KNNs of a central region and the identification of in-range and out-of-range neighbors. A central region obtains its contextual features by aggregating the features from its KNNs through the cross region attention (CRA) module. (b) shows the structure of the attention block in CRA.

Table 2. Quantitative results on the P3M-500 tet data. † indicates that a trimap is used.

			P3M-500-NP				P3M-500-P			
Method		GFLOPS	SAD	SAD-T	Grad	Conn	SAD	SAD-T	Grad	Conn
DIM† [29]		791.6	5.32	5.32	4.70	7.70	4.89	4.89	4.48	9.68
P3M-Net [12]		364.9	11.23	7.65	10.35	12.51	8.73	6.89	8.22	13.88
MODNet [10]	512x512 input	15.7	20.20	12.48	16.83	18.41	30.08	12.22	19.73	28.61
	fullres input	103.2	63.74	13.56	25.75	62.69	95.47	13.70	37.28	94.86
BGMv2 [15]	Resnet-50	26.5	16.72	7.55	13.00	15.39	15.70	7.23	15.54	14.71
	Resnet-101	33.9	15.66	7.72	12.42	14.65	13.90	7.23	14.69	13.13
Ours		19.0	10.60	6.83	10.78	9.77	10.04	6.44	12.65	9.41

4. Experiments

4.1. Experiment Setup

Datasets. We benchmark on two datasets: P3M-10k [12] and PPM-100 [10]. P3M-10k is by far the largest human portrait matting dataset and contains 10421 high-resolution in-the-wild images with annotated alpha mattes. For privacy issues, all faces in the images have been blurred. As shown in [12], training on images with blurred faces does not degrade the model performance. Instead, it may even help the model to generalize better. We use the provided 9421 images with blurred faces for training and 500 images with blurred faces for privacy-preserving test and rest 500 normal images (without face blurring) for non-privacy test. Following [12], we denote the two test subsets from P3M-10k as P3M-500-P (privacy-preserving) and P3M-500-NP (non-privacy). Compared to P3M-10k, PPM-100 is a smaller dataset curated specifically for evaluation purpose.

Baselines. We compare our model with the state-of-the-art trimap free method P3M-Net [12]. As a reference, we also include DIM [29], a commonly adopted trimap-based baseline, in the experiments. We also compare with MODNet [10] and BGMv2 [15], which are designated lightweight model for fast inference. The original BGMv2 relies on an additional image of background. To make it a fair comparison, we retrain a modified version by eliminating the background capture. Note that MODNet is designed for 512×512 input images while we are targeting at higher resolutions such as HD and 4K. Therefore, we use two strategies to accommodate MODNet in our test scenario – we either use 512×512 input and upsample the output to full resolution or run the model on full resolution directly.

Default model. Our default model uses Swin-T [17] as the low resolution network. To reduce the input image size for coarse alpha estimation, we apply a patch size of 16 for pixel-unshuffle, which is equivalent to a ×16 reduction in spatial resolution. Because the refinement network operates

on 8×8 regions, we let the low resolution network's decoder to produce a 4-channel output at $\mathfrak{R}_{16}$ and append a pixel-shuffle layer at the end to increase the resolution from $\mathfrak{R}_{16}$ to $\mathfrak{R}_8$. This way, a pixel at $\mathfrak{R}_8$ is equivalent to a 8×8 region at the full resolution. For CRA, we use 8 nearest neighbors and employ a search range of 4 for relative positional encoding.

Evaluation metrics. We follow previous works using the sum of absolute difference (SAD), the gradient difference (Grad) and the connectivity error (Conn) as the evaluation metrics. Conn is used as a way to measure the degree of connectivity, the intuition behind which is that unconnected components are more visually distracting when they are further away from the dominant connected components in the image [22]. We also report SAD within the transition area (a.k.a the "uncertain" region in a trimap), denoted as SAD-T. FLOPS is used as an indicator of compute budget. Since the FLOPS of BGMv2 [15] and our method depend on image content, we report the mean FLOPS over multiple inferences with an average of 1.63M pixels per inference.

Figure 4. Estimated coarse trimaps and refined alpha mattes. Trimaps are upsampled to full resolution for visualization.

4.2. Results

We visualize the intermediate coarse trimaps and the final alpha mattes in Fig. 4. One can see how accurately our model can adapt to the number of regions to be refined. For example, the low resolution network predicts more "uncertain" regions (shown in green) around fuzzy hair in the trimaps while refrains from doing so around the contour of body. More qualitative results are shown in Fig. 5.

Quantitative results on the P3M test data are presented in Tab. 2. Our model outperforms all baselines by a large margin on all metrics while uses the least amount of computation. Compared to the previous state-of-the-art method P3M-Net, our model achieves competitive results with nearly $1/20$ of the FLOPS. We are only slightly behind P3M-Net on Grad (for both P3M-500-NP and P3M-500-P) and SAD (for P3M-500-P) while obtains the state-of-the-art results on all other metrics. In the auxiliary-free setting, BGMv2 does not retain the good performance reported by [15] due to the lack of an additional background capture as input. Note that DIM has the best numbers for all metrics, but it is not directly comparable to other models because it takes trimap as an auxiliary input. We include it here merely for reference purpose as it is one of the mostly adopted methods for comparison.

It is worth noting that MODNet is originally designed and trained for 512x512 images. When running on high resolution input, not only does it incur degraded quality, but also increase its GFLOPS from 15.7 to 103.2. On the other hand, our model is super lightweight and can generate high quality full resolution mattes with only 19 FLOPS.

Numeric results on the PPM-100 test data are shown in Tab. 3. Since PPM-100 does not have training data, we use models trained on the P3M-10k data for evaluation. Our model is superior to others on all metrics except being slightly worse than P3M-Net on SAD-T. P3M-Net achieve competitive results on P3M-500-NP and P3M-500-P, but its performance drops significantly when evaluated on PPM-100. We believe this is because of the domain gap between the training and test sets. Some major differences we observe between the two datasets are image resolutions and imaging quality. Images in PPM-100 are higher resolutions but have worse imaging quality. This explains the overall performance degradation for all models on PPM-100. However, our model is more robust to this domain gap and achieves the best results on PPM-100.

Table 3. Quantitative results on the PPM-100 dataset.

Method		SAD	SAD-T	Grad	Conn
P3M-Net [12]		142.74	43.06	57.02	139.89
MODNet [10]	512x512	104.35	65.42	68.56	96.45
	fullres	324.07	68.97	77.42	319.70
BGMv2 [15]	Resnet-50	193.40	49.39	61.49	185.52
	Resnet-101	159.44	50.67	59.41	149.79
Ours		90.28	45.06	50.69	84.09

Table 4. FPS and GFLOPS for HD and 4K inputs. All models are evaluated with a single Nvidia Quadro RTX 6000 GPU. An empty entry means we fail to evaluate the model due to out-of-memory error.

		HD		4K	
Method		FPS	GFLOPS	FPS	GFLOPS
DIM [29]		5.0	1007.1	-	-
P3M-Net [12]		9.2	463.5	-	-
MODNet [10]		15.0	123.4	-	-
BGMv2 [15]	Resnet-50	57.4	32.7	23.7	128.6
	Resnet-101	45.8	42.2	17.8	166.8
Ours	w/ CRA	54.9	21.2	19.5	74.6
	w/o CRA	71.2	19.4	26.4	70.7

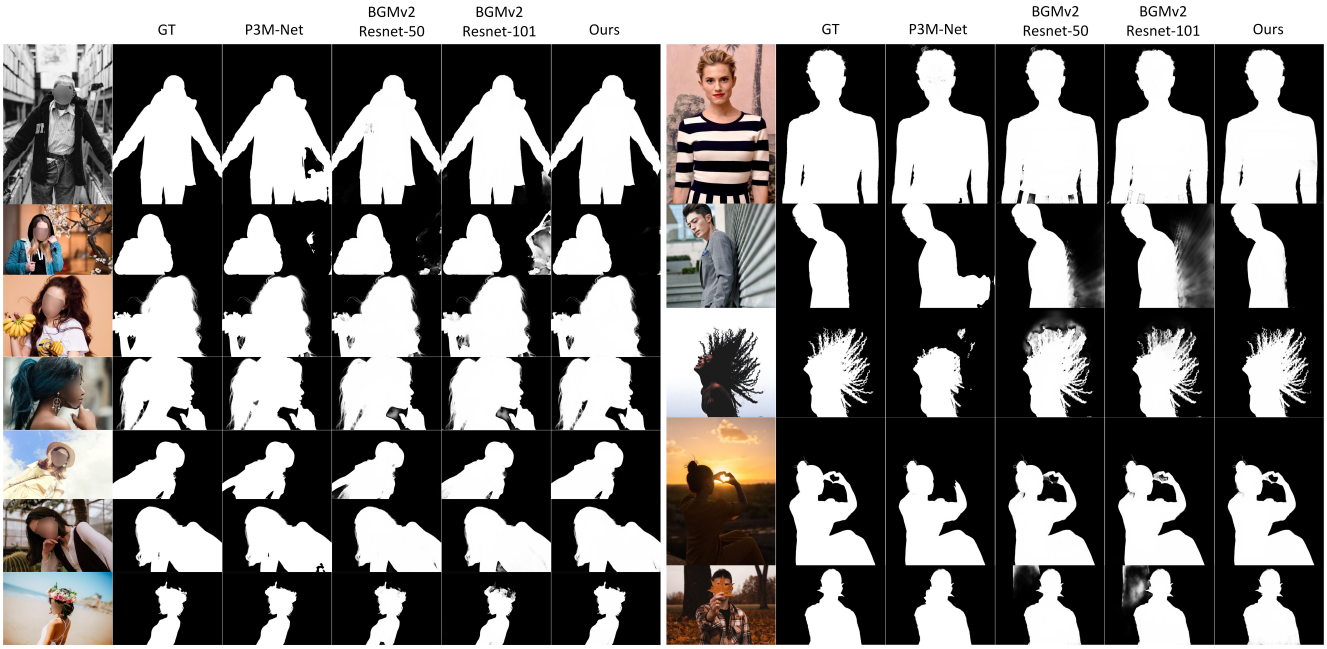

Figure 5. Qualitative results. On the left and right are respectively the results of P3M-500-P and P3M-500-NP. Zoom in for more details.

Table 5. Quantitative results for ablation study.

			P3M-500-NP				P3M-500-P			
No.	Refinement Method	CRA	SAD	SAD-T	Grad	Conn	SAD	SAD-T	Grad	Conn
E	[15]	NA	11.68	7.92	12.90	11.11	11.81	7.37	15.19	11.44
F	Ours	✗	11.71	7.28	11.72	10.88	10.69	6.90	13.74	10.06
G	Ours	✓	10.60	6.83	10.78	9.77	10.04	6.44	12.65	9.41
	Search Range	KNN								
I	2	8	11.58	6.82	11.04	10.76	10.52	6.48	13.06	9.87
H	3	8	10.74	6.85	10.78	9.91	10.41	6.46	12.76	9.77
G	4	8	10.60	6.83	10.78	9.77	10.04	6.44	12.65	9.41
J	8	8	10.60	6.79	10.87	9.79	10.46	6.56	13.08	9.88
K	4	4	11.11	6.96	11.20	10.28	10.40	6.54	13.04	9.81
L	4	16	10.77	6.80	10.93	9.95	9.40	6.33	12.59	8.84

4.3. Real Application Performance

We test on real-world HD videos from [15, 23] and show the qualitative results in Fig. 6. Please refer to the supplementary material for more video results.

We profile all models on HD and 4K inputs and compare their FPS and GFLOPS in Tab. 4. As shown in the table, our models use the least amount of computation and achieve competitive frame rate. For DIM, P3M-Net and MODNet running on full resolution, we fail to profile their performance on 4K input due to the massive memory footprint required. On the other hand, our models yield real time performance for HD input and near real time for 4K. It is also worth noting that our model with CRA, even with fewer GFLOPS, runs slightly slower than Resnet-50 backboned BGMV2. This is because most modern deep learning frameworks do not have well optimized transformer operators. As shown in [20], more than 2x speedup is possible with the optimized native CUDA kernels. Our current implementation of Swin-T and CRA utilizes generic pytorch functions to compute the multi-head attention. We believe a similar improvement at inference time is possible with the optimized implementation.

4.4. Ablation Study

In this section, we demonstrate the effectiveness of the proposed refinement stage, the CRA module and their associated hyper parameters.

Refinement stage. We compare the proposed refinement

Figure 6. Qualitative results of real-world HD videos. The red box shows some of the typical failure cases.

stage with that of BGMv2 [15] by fixing the low resolution backbone. As shown in Tab. 5, model G (our default) and model E differ only by the refinement stage. Model G outperforms model E on all metrics, demonstrating the effectiveness of the proposed refinement stage. We also observe similar results by comparing model A (from Tab. 1) with BGMv2 Resnet-50 (from Tab. 2). Both models use Resnet-50 as the low resolution network, but they differ by the refinement stage. Model A surpassing BGMv2 Resnet-50 on all evaluated metrics, again, demonstrates the advantage of the proposed refinement stage.

Cross region attention. The CRA lies at the core of our method. We show its individual impact by taking it out from our model, resulting in model F shown in Tab. 5. We can see that G achieves better results than F, demonstrating the effectiveness of CRA.

Search range. Search range is used in the CRA module to identify in-range neighbors and out-of-range neighbors for relative positional encoding (Sec. 3.3). Because all out-of-range neighbors share the same relative positional bias (Fig. 2), smaller search range enforces more out-of-range neighbors, making the model less discriminative against the neighboring regions. In Tab. 5, we list three models (G, H & I) with different search ranges. As the search range increases, we can see a trend [1] in improved performance.

However, we also observe from model J that a large search range of 8 does not boost the performance any further. We believe this is due to the search window being too large (15×15) [2]. At most 8 out of 225 positional biases are queried from the lookup table, leaving the rest of the positional biases untouched. Therefore, a small percentage of biases being queried and optimized during each gradient propagation results in sub-optimal bias lookup table, hence the degraded quality of the model.

[1] For the majority of the evaluated metrics.

[2] Recall that, given the search range s, the search window size is $(2s-1)\times(2s-1)$.

KNNs. We study the impact of KNNs by varying its number. As KNNs increase (K $\rightarrow$ G $\rightarrow$ L) in Tab. 5, we can see the overall [1] performance of the model improves. The reason is twofold. First, more KNNs means more contextual information, which helps the model do a better job at learning. Second, more KNNs benefit the training of bias lookup table. As we have just discussed, it improves the chances of positional biases being queried and optimized during training.

5. Failure cases and Future Work

When there is high contrast texture in the extracted regions, the refinement network finds it difficult to identify the correct foreground and background. As is shown in Fig. 6, the text in on the whiteboard is supposed to be background, but it is perfectly (and incorrectly) segmented as foreground. Also, the refinement network can only improvement the quality of local boundaries. Any false predictions in the original low resolution matte can not be recovered. For example, The chair in Fig. 6 has been false positive in the first stage, it is impossible to undo the false prediction by the refinement network.

Currently our model is trained only with limited amount of data (9421 images from the P3M data [12]), which is far from being robust in real-world applications. Because the refinement network only consumes an upsampled matte and does not rely on any intermediate features from the first stage, we believe it is possible to train the low resolution network and the refinement network separately to improvement the overall robustness of our method. The abundant low resolution segmentation data [13, 16, 32] can be leveraged to train the coarse model while the high resolution matting data plus an unlimited number of human synthetics [28] can used to train the refinement stage. We leave this as a future work.

6. Conclusion

We present a new lightweight two-stage method for high resolution portrait matting. At the heart of our method is a ViT backboned low resolution network for coarse alpha estimation and a novel cross region attention (CRA) module in the second stage for local refinement. We verify that using pixel-unshuffle rather than downsampling has the advantage of preserving original pixel information and that ViT comes naturally a good choice for that purpose. We demonstrate the effectiveness of the proposed low resolution network, refinement stage and CRA module and analyze the individual impact of several key hyper parameters. Through extensive experiments, we show the superiority of our method against the previous state-of-the-arts in terms of both accuracy, FPS and FLOPS.

References

[1] Yagiz Aksoy, Tunc Ozan Aydin, and Marc Pollefeys. Designing effective inter-pixel information flow for natural image matting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 29–37, 2017. 1, 2

[2] Quan Chen, Tiezheng Ge, Yanyu Xu, Zhiqiang Zhang, Xinxin Yang, and Kun Gai. Semantic human matting. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 618–626, 2018. 1, 2

[3] Qifeng Chen, Dingzeyu Li, and Chi-Keung Tang. Knn matting. *IEEE transactions on pattern analysis and machine intelligence*, 35(9):2175–2188, 2013. 1, 2

[4] Yung-Yu Chuang, Brian Curless, David H Salesin, and Richard Szeliski. A bayesian approach to digital matting. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 2, pages II–II. IEEE, 2001. 1, 2

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4

[6] Marco Forte and François Pitié. f, b, alpha matting. *arXiv preprint arXiv:2003.07711*, 2020. 1, 3

[7] Kaiming He, Jian Sun, and Xiaoou Tang. Fast matting using large kernel matting laplacian matrices. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2165–2172. IEEE, 2010. 2

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016. 3

[10] Zhanghan Ke, Jiayu Sun, Kaican Li, Qiong Yan, and Rynson WH Lau. Modnet: Real-time trimap-free portrait matting via objective decomposition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1140–1147, 2022. 5, 6

[11] Anat Levin, Dani Lischinski, and Yair Weiss. A closed-form solution to natural image matting. *IEEE transactions on pattern analysis and machine intelligence*, 30(2):228–242, 2007. 1, 2

[12] Jizhizi Li, Sihan Ma, Jing Zhang, and Dacheng Tao. Privacy-preserving portrait matting. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 3501–3509, 2021. 1, 2, 5, 6, 8

[13] Jianshu Li, Jian Zhao, Yunchao Wei, Congyan Lang, Yidong Li, Terence Sim, Shuicheng Yan, and Jiashi Feng. Multi-human parsing in the wild. *arXiv preprint arXiv:1705.07206*, 2017. 8

[14] Yaoyi Li and Hongtao Lu. Natural image matting via guided contextual attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11450–11457, 2020. 1, 2, 3

[15] Shanchuan Lin, Andrey Ryabtsev, Soumyadip Sengupta, Brian L Curless, Steven M Seitz, and Ira Kemelmacher-Shlizerman. Real-time high-resolution background matting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8762–8771, 2021. 2, 3, 5, 6, 7, 8

[16] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 8

[17] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 3, 4, 5

[18] Hao Lu, Yutong Dai, Chunhua Shen, and Songcen Xu. Indices matter: Learning to index for deep image matting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3266–3275, 2019. 1, 2, 3

[19] Sebastian Lutz, Konstantinos Amplianitis, and Aljosa Smolic. Alphagan: Generative adversarial networks for natural image matting. *arXiv preprint arXiv:1807.10088*, 2018. 1, 2, 3

[20] Scott Wolchok Rui Zhu Christian Puhrsch Michael Gschwind, Eric Han. A bettertransformer for fast transformer inference. `https://pytorch.org/blog/a-better-transformer-for-fast-transformer-encoder-inference/`. 7

[21] Yu Qiao, Yuhao Liu, Xin Yang, Dongsheng Zhou, Mingliang Xu, Qiang Zhang, and Xiaopeng Wei. Attention-guided hierarchical structure aggregation for image matting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13676–13685, 2020. 1, 2

[22] Christoph Rhemann, Carsten Rother, Jue Wang, Margrit Gelautz, Pushmeet Kohli, and Pamela Rott. A perceptually motivated online benchmark for image matting. In *2009 IEEE conference on computer vision and pattern recognition*, pages 1826–1833. IEEE, 2009. 6

[23] Soumyadip Sengupta, Vivek Jayaram, Brian Curless, Steven M Seitz, and Ira Kemelmacher-Shlizerman. Background matting: The world is your green screen. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2291–2300, 2020. 1, 2, 3, 7

[24] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3

[25] Jian Sun, Jiaya Jia, Chi-Keung Tang, and Heung-Yeung Shum. Poisson matting. In *ACM SIGGRAPH 2004 Papers*, pages 315–321. 2004. 1, 2

[26] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on*

computer vision and pattern recognition, pages 1–9, 2015. 3

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 4

[28] Erroll Wood, Tadas Baltrušaitis, Charlie Hewitt, Sebastian Dziadzio, Thomas J Cashman, and Jamie Shotton. Fake it till you make it: face analysis in the wild using synthetic data alone. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3681–3691, 2021. 8

[29] Ning Xu, Brian Price, Scott Cohen, and Thomas Huang. Deep image matting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2970–2979, 2017. 1, 2, 3, 5, 6

[30] Yunke Zhang, Lixue Gong, Lubin Fan, Peiran Ren, Qixing Huang, Hujun Bao, and Weiwei Xu. A late fusion cnn for digital matting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7469–7478, 2019. 1, 2

[31] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2472–2481, 2018. 3

[32] Jian Zhao, Jianshu Li, Yu Cheng, Li Zhou, Terence Sim, Shuicheng Yan, and Jiashi Feng. Understanding humans in crowded scenes: Deep nested adversarial learning and a new benchmark for multi-human parsing. *arXiv preprint arXiv:1804.03287*, 2018. 8