1. Architecture and training

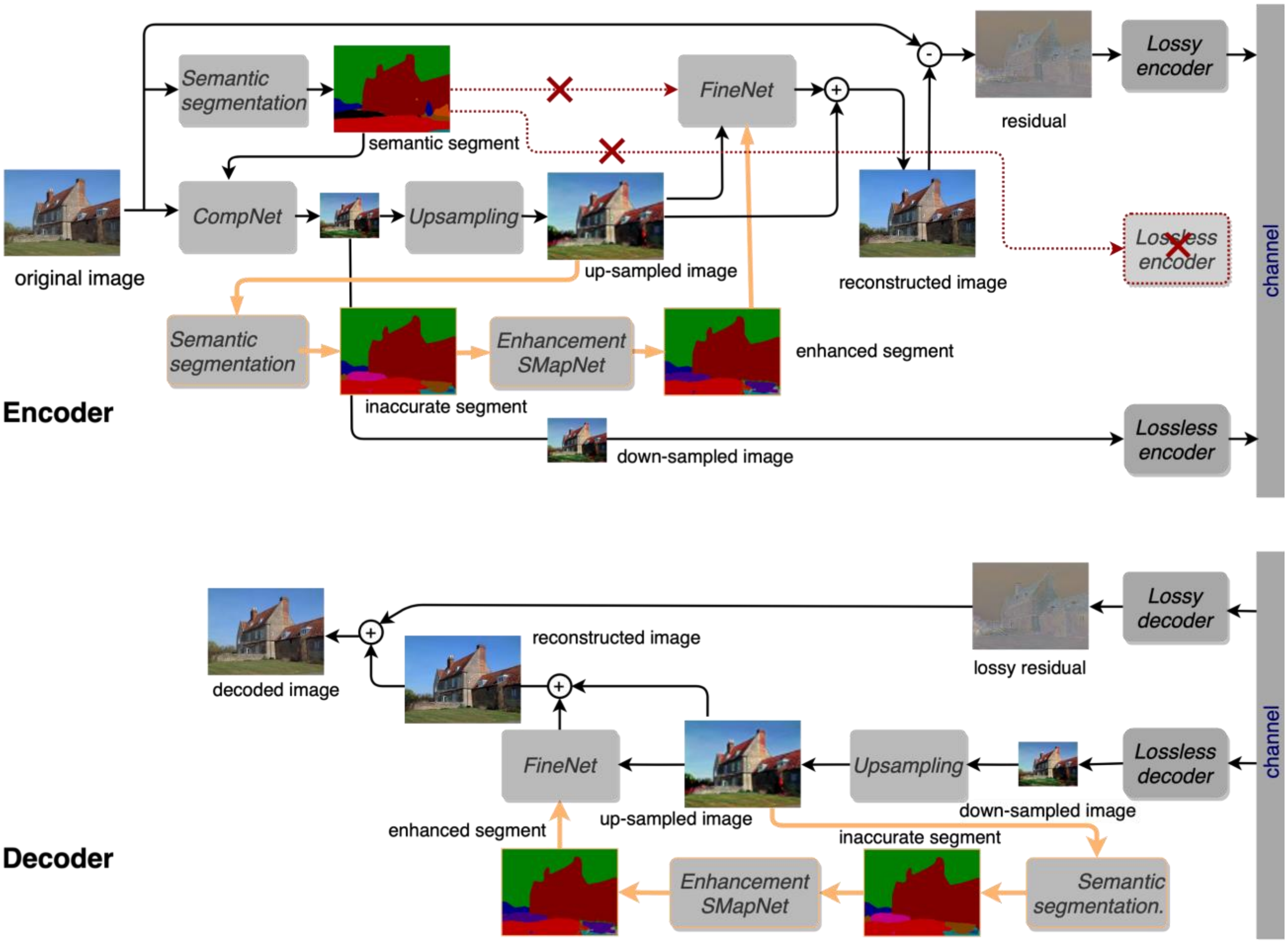

Figure. 1. Our proposed framework

1.1 Network architecture

As we mentioned in Section 2.4 of the main manuscript, our framework contains three main networks, below are the architectures of those networks, please refer the Figure 1 for our framework:

- CompNet: 7C64, 3C128, 3C256, 3C512, 7C3, tanh;
- FineNet: 7C64, 3C128, 3C256, 3C512, 9 x 3r512, 3u256, 3u128, 3u64, 7C3, tanh
- SMapNet: 3C64, 9 x 3V64, 3C3 – no ReLu

where

- sCk: (s x s) convolution layer with k filters and stride 1, followed by instance normalization and ReLU.
- srk: residual block with two (s x s) convolution layers and k filters, followed by instance normalization
- suk: (s x s) fractional-strided-convolution layer with k filters and stride ½, followed by instance normalization and ReLU.
- sVk: recursive residual block with two (s x s) convolution layers and k filters, followed by ReLU

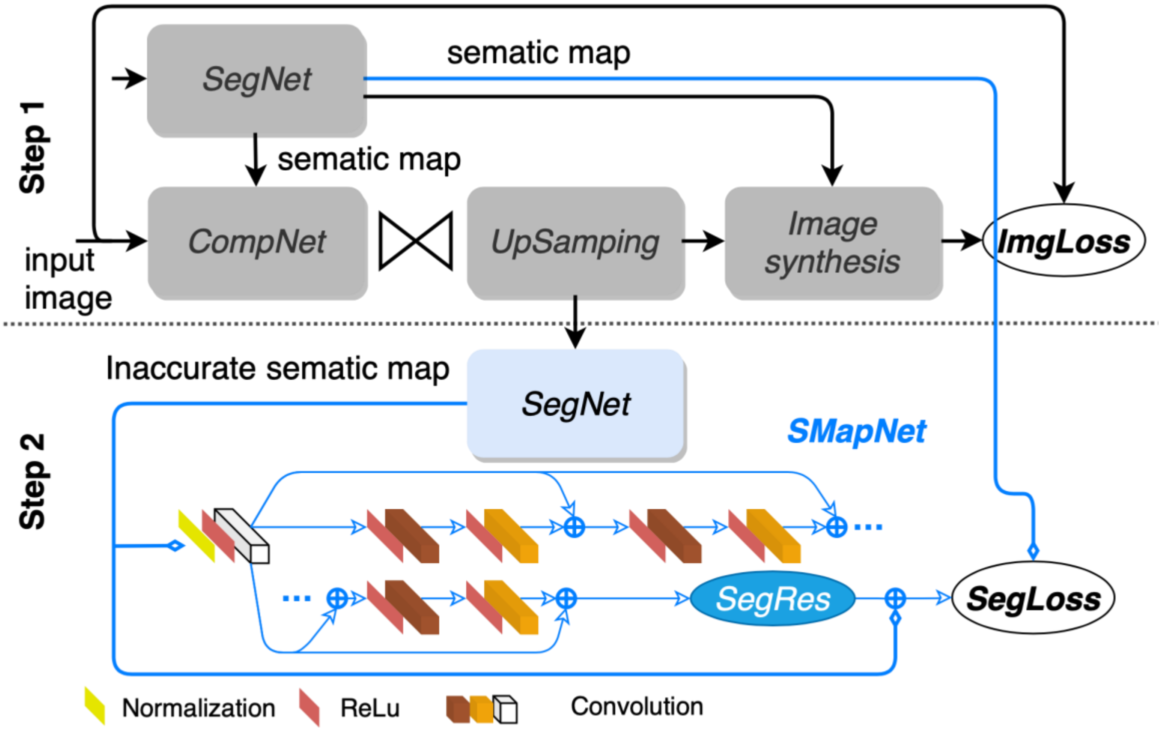

Figure. 2. The specific training procedure

1.2 Training procedure

As we mentioned in Section 2.2 of the main manuscript, to obtain a good enough semantic segment from the up-sampled image, we first need to perform the training process based on the original semantic segment. The ADE20K with 150 semantic labels are used for this stage of training. All the images are rescaled to 256x256 to have a fixed size of training. We set the down-sample factor of the CompNet. equal to 8 to get the compact representation of size 32x32x3. Then we perform the training process with five kinds of losses according to DSSLIC setting, which is demonstrated that got better results on both perceptual and PSNR quality (see Step 1 in Fig 2).

On the next step, based on the above pre-trained model, we use the PSP network to perform the semantic segmentation on the up-sampling images (256x256 from the ADE20K dataset) and use them as inputs for training SMapNet. The SMapNet is trained as a non-linear mapping operator between the extracted segment and its original version (see Step 2 in Fig 3). For SMapNet, we use a minibatch size of 32 and Adam optimizer, we start with a learning rate of 5e-04, for stable training, the final layer will have a learning rate equal to one tenth of other layers, then terminate training at 150 epoch.

2. More visual comparison on kodak dataset

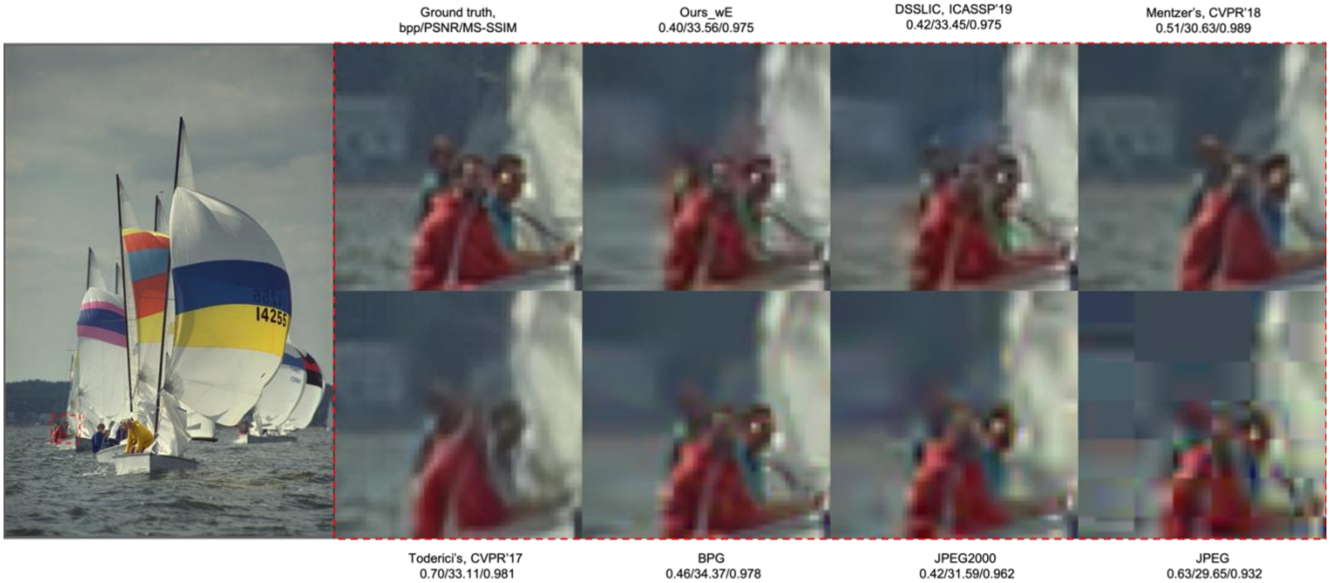

Figure. 3. Kodim09 test result. Note that our model can keep more details than others at the lowest bpp.

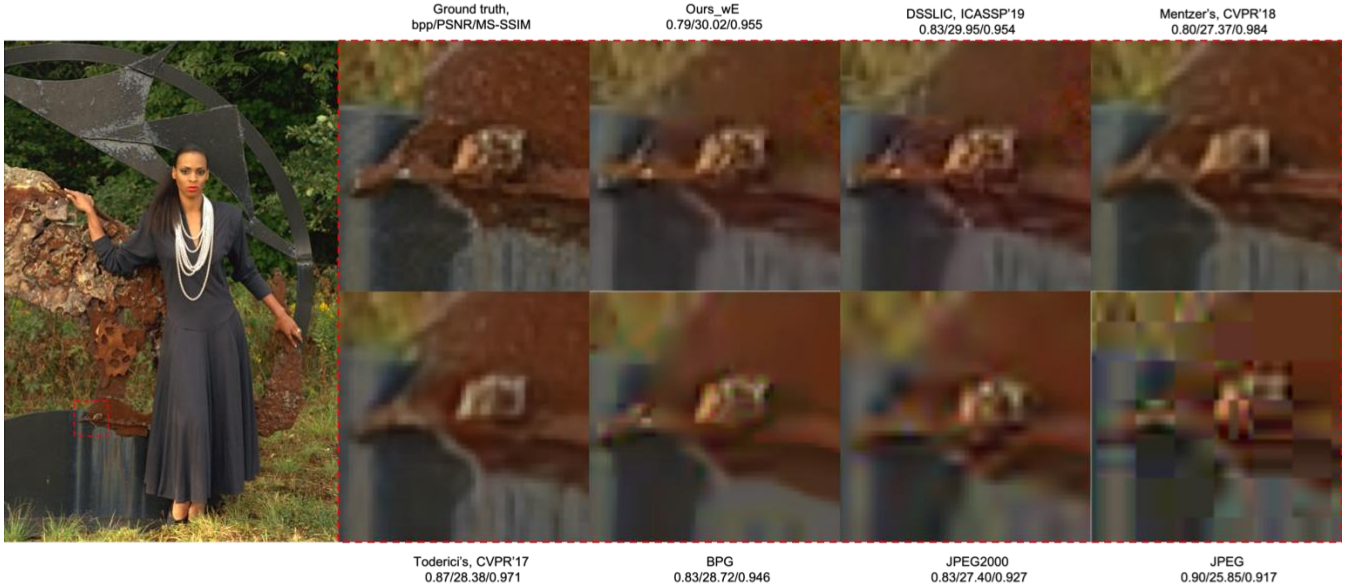

Figure. 4. Kodim18 test result. Our model could avoid more noise from lossy compression than DSSLIC[4].

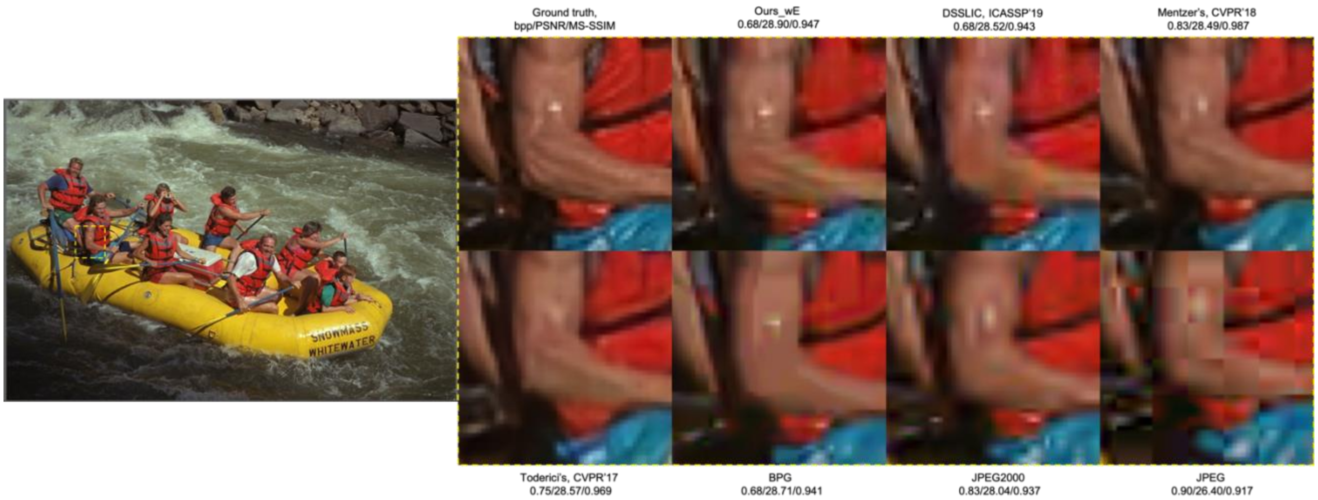

Figure. 5. Kodim14 test result. We can see the detail on the aim is kept at the lowest bpp of our model.

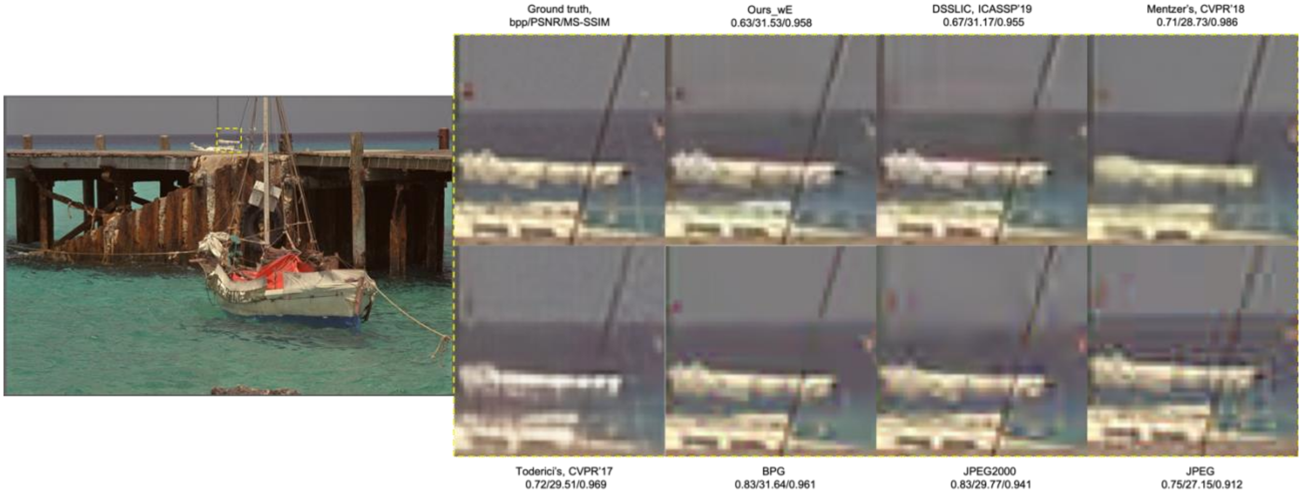

Figure. 6. Kodim11 test result. The rope is almost blurred in all other works while seen by our result.

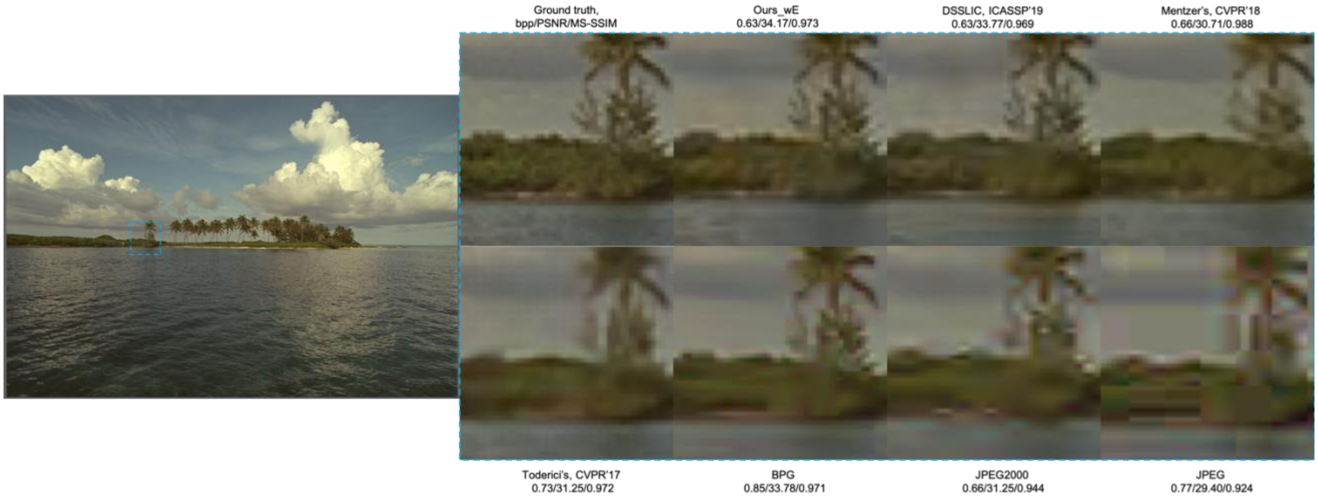

Figure. 7. Kodim16 test result. Only our result can keep the shape of the peak behind.

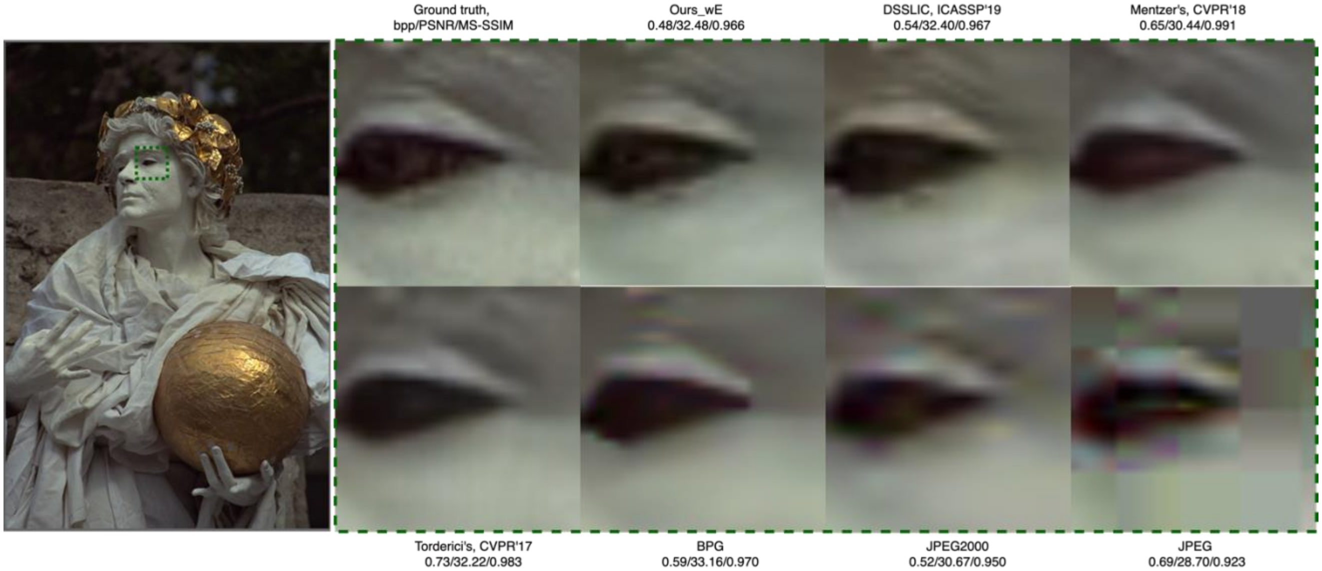

Figure. 8. Kodim17 test result. We only can see clearly what inside the eye with our result at the lowest bpp.