Volumetric Capture of Humans with a Single RGBD Camera via Semi-Parametric Learning Supplementary Material

Rohit Pandey, Anastasia Tkach, Shuoran Yang, Pavel Pidlypenskyi, Jonathan Taylor, Ricardo Martin-Brualla, Andrea Tagliasacchi, George Papandreou, Philip Davidson, Cem Keskin, Shahram Izadi, Sean Fanello
Google Inc.

In this supplementary material, we provide additional information regarding our method's implementation, more details and more ablation studies on important components of the proposed framework.

1. Framework Details

We detail here the choices of various parameters to aid in reproducing results.

1.1. Calibration Image Selector - Params

In Figure 1 we show some example outputs of the calibration selector module. Note how the module selects the calibration image that most closely matches the viewpoint the person is seen from, based on the target pose. For Eq. 5 we empirically select $\omega_{\text{head}} = 5$, $\omega_{\text{torso}} = 3$, and $\omega_{\text{sim}} = 1$ so as to weigh the head and torso components of the score highest, then factor in the transformation scores of the limbs.

1.2. Calibration Image Warper

Keypoint grouping: We detect 17 keypoints and group them into into 10 body parts. The body parts consist of 1) head 2) body 3) left upper arm 4) right upper arm 5) left lower arm 6) right lower arm 7) left upper leg 8) right upper leg 9) left lower leg, and 10) right lower leg. The head keypoints consist of the nose, left/right eyes, and left/right ears. The body keypoints consist of the left/right shoulder, and left/right hip keypoints. Each limb consists of two keypoints; the shoulder and elbow for the upper arms, the elbow and wrist for the lower arms, the hip and knee for the upper legs, and the knee and ankle for the lower legs.

Network architecture: Our U-Net architecture consists of 5 encoder blocks followed by 5 decoder blocks. Each encoder block downsamples the input by a factor of 2 and consists of 2 convolutional layers; the first with a kernel size of 3 and stride 1 and the second with a kernel size of 4 and stride 2. Each decoder block consists of a bilinear upsampling layer

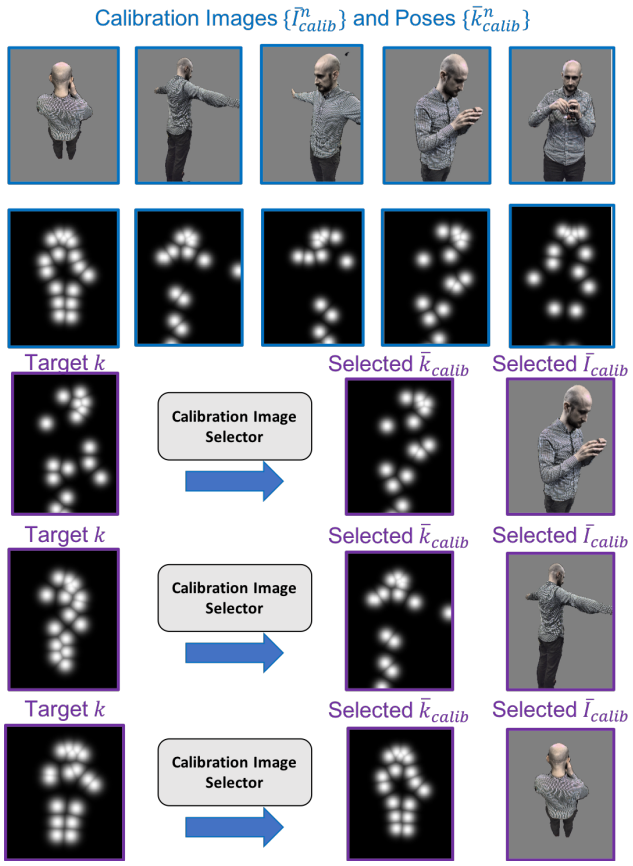

Figure 1. Examples of selected calibration images. The closest viewpoint to the target is selected.

that upsamples the input by a factor of 2, followed by a convolutional layer with kernel size 3 and stride 1. The encoder blocks use 64 filters for the first block followed by 128 filters for the remaining 4 blocks. The decoder blocks use 128 filters for the first 4 convolutional layers and 32 filters for the final block. Additionally, we add skip connections from the encoder to the decoder in the form of concatenation of feature maps of matching size. Leaky ReLu activations (with $alpha = 0.2$) are used for all convolutional layers.

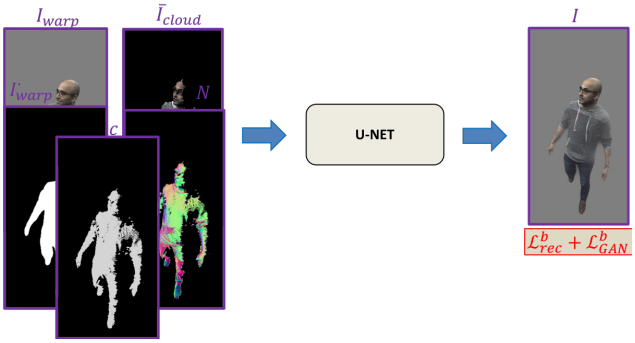

Figure 2. The final Neural Blender module takes the output RGB and mask of the calibration warper module, as well as the warped RGB, normals, and viewpoint confidence from the re-rendering module, and learns how to blend them into the final output RGB.

The calibration image warper module adds a final convolutional layer with 4 channels to produce the RGB output I_{warp}, and the mask $I^{\bullet}_{\text{warp}}$. Tanh activation is used for the RGB and sigmoid for the mask prediction.

1.3. Neural Blender

The neural blender module is shown in Figure 2. For simplicity we re-use the same U-Net architecture described in the calibration image warper module section above. However, the last convolutional layer now outputs only 3 channels for the final blended RGB.

1.4. Training Details

Our networks are implemented in Tensorflow and trained in parallel on 16 NVIDIA V100 GPUs each with 16 GB of memory. We use the Adam optimizer [1] with a learning rate of 1^{-5} for the generator and 1^{-6} for the discriminator. We perform light data augmentation during training with random cropping in a size range of 0.85 to 1. times of the input image size. Additionally, we add standard ℓ_2 loss regularization (with weight 1^{-5}) to the weights of the network. We found that our data augmentation and regularization, coupled with the variations introduced via using all possible combinations of source and target cameras in our training set, were sufficient to prevent over-fitting and make our network generalize to unseen poses, viewpoints and people.

2. Running time of the system

The proposed architecture has an end-to-end runtime of $104.3ms$ on a Titan V GPU. Note that, this is the unoptimized runtime and does not take advantage of float16 inference or tensor cores on the volta architectures. We leave achieving realtime inference using the proposed architecture for future work.

Figure 3. Examples of the warped foreground mask and refined foreground mask, compared to the ground truth mask.

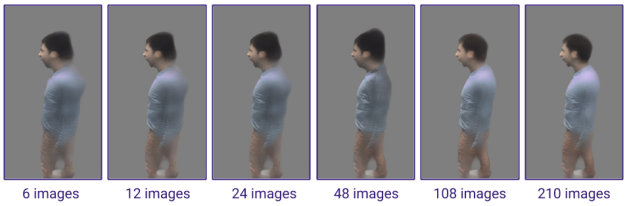

Figure 4. Effect of the number of calibration images in the pool on the output of the calibration image warper module.

3. Additional Evaluation

In Figure 3 we show some examples of the predicted warped part masks and refined masks compared to the ground truth foreground mask. Note that the warped part mask is limited in the accuracy of the silhouette it can produce due to the assumption of 2D similarity transformation between body parts, however, the predicted refined mask is able to overcome these limitations and produce a much cleaner silhouette.

In Figure 4 we show the effect of adding more images to the calibration image pool on the output of the calibration warping module. All calibration images are chosen at random from a held out sequence of the user. Notice how the quality of the output improves as the number of calibration images increases from 6 to 210. This is due to the higher probabil-

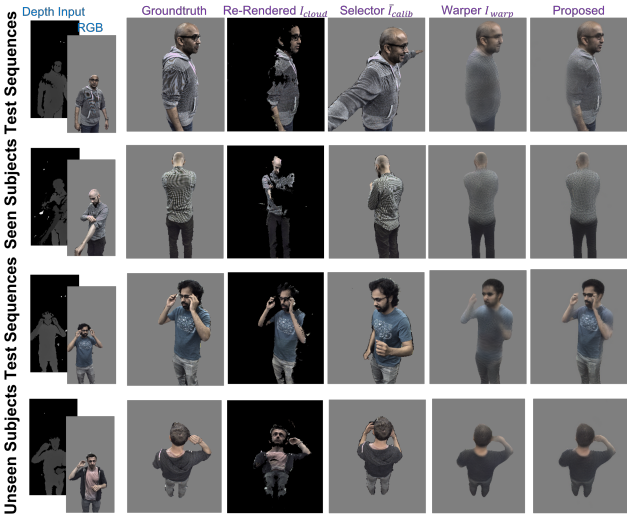

Figure 5. Additional results showing various stages of the pipeline.

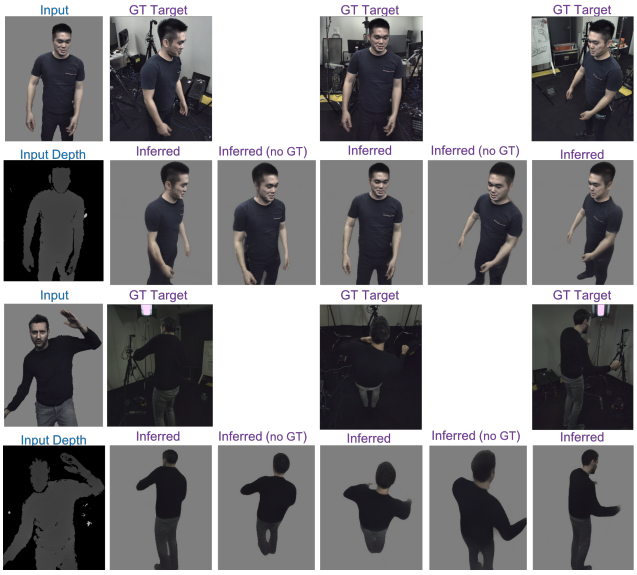

Figure 6. Additional results showing the viewpoint generalization of the proposed method.

ity of finding a calibration image with matching viewpoint as we increase the size of the image pool.

In Figure 5 we present additional results showing the output of various stages of the proposed pipeline on seen and unseen subjects.

In Figure 6 we present additional results showing the ability of the proposed method to generalize to viewpoints not in the training data.

4. Limitations and Future Work

One of the limitations of the proposed approach is that the calibration warper produces blurry results when the viewpoint of the selected calibration image is far from the target viewpoint. We notice this in results when the calibration image pool is small as shown in Figure 4. A larger calibration pool or a predefined calibration sequence where the user turns around the camera can help alleviate this issue. Another limitation of the system is that it struggles to produce reasonable outputs where keypoints are not present, for example hands. We hypothesize that adding additional finger keypoints like fingers and more facial keypoints can help both the calibration selection module, as well as the hallucination modules, to produce better results. Finally, the system shows some temporal flickering as can be seen in the supplementary video. This is especially evident when the selected calibration image changes, and likely can be alleviated via temporal architectures like RNNs or temporal coherency losses.

References

[1] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, 2014. 2