Unsupervised 3D Shape Coverage Estimation with Applications to Colonoscopy Supplementary Material

Yochai Blau, Daniel Freedman, Valentin Dashinsky, Roman Goldenberg, Ehud Rivlin
Google Research
{yochaib, danielfreedman, valkad, rgoldenberg, ehud}@google.com

In this supplemental we provide additional information on the datasets and shape models used in the experiments, data pre-processing, the network architecture, and model training details.

A. Datasets and Data Preprocessing

A.1. Body Shapes Experiment (Section 4)

As mentioned in the main paper, we used a simplified body shape model, which is based on the SMPL's [5] PCA shape decomposition: Aq_i+b, where $b \in \mathbb{R}^{3n}$ is the average body shape mesh of $n = 6890$ vertices in a neutral (star-shaped) pose, and $A \in \mathbb{R}^{3n\times 10}$ is composed of the 10 first PCA vectors. We generated 2300 body shape meshes by randomly sampling the "intrinsic" shape parameters q_i from $q_i \sim \mathcal{N}(0,\, I)$.

For each body shape we generated a random camera path on a unit sphere around the body mesh. The path is generated by a random walk of $k = 5$ steps on the sphere, from a randomly selected initial position. At every step along the path the camera is oriented to look at the center of the sphere.

Depth images of 256×256 resolution are rendered for a virtual camera with 60-degree FOV, using a z-buffer algorithm [6].

The list of mesh triangles visible in a virtual camera is used to compute the ground-truth coverage rate as the ratio between the visible and the total surface area.

A.2. Synthetic Colonoscopy Experiment (Section 5.1)

Given the 3D colon model, we randomly generated 32 camera paths through the colon. The "base" path is a camera movement along the colon lumen, where the paths differ by: (a) smooth camera movements in directions perpendicular to the colon lumen, (b) smooth camera rotations, and (c) changing velocity of camera movement along the "base" path. Given the 3D model and camera poses along a path, $10K$ frames were rendered with Blender [7], where for each such frame we retain RGB-D data. The camera intrinsics were set to 256 pixels in each dimension and a view angle of 1.22 radians.

The model is trained on 10 second long segments. We randomly extract video segments by: (a) randomly drawing a video, and (b) randomly drawing a start frame and taking the following 300 frames. The segments are then uniformly temporally downsampled so we are left with data samples of $N = 20$ frames.

To compute the ground-truth coverage rates, we first identify the 3D colon mesh faces within the segment using the formulation in [2] with $\Delta_0 = \Delta_1 = 1.0$. Then, given the camera intrinsic matrix and camera pose in each frame, we perform a perspective projection of the 3D colon mesh onto the camera plane to identify which faces are in the field-of-view. The set of all faces in the camera field-of-view within the 300 segment frames are the seen faces. The coverage rate is then computed by taking the ratio of the seen faces area to area of all faces within the segment.

A.3. Real Colonoscopy Experiment (Section 5.2)

The model is trained on 200 frame segments. We randomly extract video segments by: (a) randomly drawing a video, and (b) randomly drawing a start frame and taking the following 200 frames. The segments are then uniformly temporally downsampled so we are left with data samples of $N = 50$ frames.

As we only have RGB frames at hand, we estimate the depth maps and camera poses using the unsupervised depth and camera egomotion estimation framework of [1] (available on GitHub[1]). We retrained this model on colonoscopy videos, using the default training and model parameters.

B. Shape Models

B.1. Body Shapes Experiment (Section 4)

As described in the main paper, for the body-shape experiment only, we used the same 10-vector PCA approximation both for the data generation, as well as the shape

[1] https://github.com/tensorflow/models/tree/archive/research/struct2depth

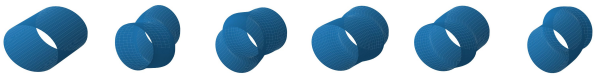

Figure S1. *Left:* "base" cylinder mesh of 1500 vertices. *Other:* samples of perturbed cylinders used to derive the shape model.

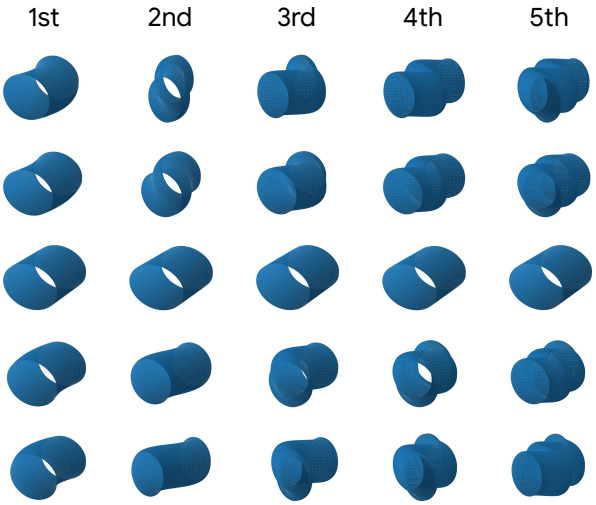

Figure S2. Visualization of the 5 shape model modes. In each column, the coefficient of the i'th mode (in eq. (12)) is interpolated between two values, while other coefficients are zeroed.

model. Having both the input and the estimated shapes parameterized using the same mesh allows for better visualization of the results and easy estimation of the coverage map accuracy (in addition to the coverage rate).

B.2. Colonoscopy Experiments (Section 5)

As described in the paper, we start with a cylinder mesh of $n = 1500$ vertices, see Figure S1. This cylinder mesh is formed by rings of 30 vertices, where 50 such equally spaced rings form the cylinder. We generate $m = 8000$ perturbed meshes, see samples of perturbed meshes in Figure S1. Each perturbed cylinder is created by translating a random ring of vertices in a direction perpendicular to the cylinder main axis, and also gradually translating adjacent rings so as to form a *smooth* perturbation. Each perturbed cylinder is represented by a vector $v \in \mathbb{R}^{3n}$ of the vertices' coordinates. All m vectors are stacked in a matrix $\mathbf{V} \in \mathbb{R}^{3n \times m}$, on which a PCA decomposition is performed and the first $k = 5$ components are extracted. See a visualization of these modes in Figure S2. Notice that the modes appear to represent sinusoidal patterns with increasing frequencies. The shape model is given by eq. (12), where $A \in \mathbb{R}^{3n \times k}$ is composed from the top 5 PCA components, and $b \in \mathbb{R}^{3n}$ is the mean shape.

C. Surface Estimator Architecture

Here, we will provide additional details regarding the surface estimator structure specified in Section 3.4. The network consists of three building blocks (see Figure 3 in the paper): (a) the depth map feature extractor, (b) the camera pose feature extractor, and (c) the surface parameter regression net.

The Depth map feature extractor is a MobileNet [3] after removing the top (classification) layer, with a depth multiplier $\alpha = 1.0$ and resolution multiplier $\rho = 1.0$. A global max-pooling is applied after the last convolutional layer. A dropout rate of 0.001 is used. The network weights are randomly initialized at the beginning of training.

The Camera pose feature extractor is an 8-layer fully-connected net, see the detailed structure in Table S1. Notice the skip connection from the input to layer #5. The Surface parameter regression net is an 8-layer 1D convolutional network, see the detailed structure in Table S2. The net output is a set of surface parameters: surface translation/rotation/scale parameters $\tilde{q}_t/\tilde{q}_r/\tilde{q}_s$, and the intrinsic shape parameters $\tilde{q}_i$.

Two additional operations are performed on the surface parameters. First, the final surface translation parameters are given by $q_t = \bar{t} + \tilde{q}_t$, where $\bar{t}$ is the mean camera translation[2]. That is, the net only needs to estimate the *deviation* of the surface translation from the mean camera pose. Second, the final intrinsic shape parameters are given by

$$q_i = c \times \tanh(\tilde{q}_i), \tag{S1}$$

where c is a "clipping hyperparameter" which controls the maximal absolute value of the intrinsic shape parameters. The other surface parameters are untouched, *i.e.* $q_r = \tilde{q}_r, q_s = \tilde{q}_s$, and the final model outputs are the set of surface parameters q_t, q_r, q_s, q_i which define the surface intrinsic shape parameters along with the translation, rotation and scale parameters.

D. Training Details

D.1. Body Shapes Experiment (Section 4)

Since the body-shape data was generated using a "metric" depth rendering, the surface estimation model doesn't need to estimate the scale parameters, as the scale is 1 in all dimensions by construction. Accordingly, for this experiment the estimation model was simplified by excluding the parts dealing with the scale estimation. The depth frames, originally rendered in 256×256 resolution, were subsampled to 32×32 before being input to the network. As mentioned in the main paper, the network was trained using the Adam optimizer [4] for 50 epochs, with a batch size of 4

[2] Concretely, $\bar{t} = \frac{1}{N}\sum_{i=1}^{N} t_i$ where t_i is the translation of the i'th camera pose in the input sequence.

#	Layer	Inputs
0	Input	-
1	Dense(256, ReLU)	0
2	Dense(256, ReLU)	1
3	Dense(256, ReLU)	2
4	Dense(256, ReLU)	3
5	Dense(256, ReLU)	0, 4
6	Dense(256, ReLU)	5
7	Dense(256, ReLU)	6
8	Dense(256, None)	7

Table S1. **Camera pose feature extractor architecture.** The *inputs* column specifies which layers are connected to the input, where if there are multiple such layers the tensors are concatenated. Dense(u, a) is a fully-connected layer with u units and activation type a.

#	Layer
0	Input
1	Conv1D(16, 1, 3, ReLU)
2	Conv1D(32, 2, 3, ReLU)
3	Conv1D(64, 2, 3, ReLU)
4	Conv1D(128, 2, 3, ReLU)
5	Conv1D(256, 2, 3, ReLU)
6	Conv1D(512, 2, 3, None)
7	Global-Max-Pool-1D
8	Dense($n_{\text{surf-params}}$, None)

Table S2. **Surface parameter regression net architecture.** Conv1D(f, s, k, a) is a 1D convolutional layer with f filters, stride of s, kernel size k, and an activation type a. Global-Max-Pool-1D is a 1D global max-pooling layer. Dense(u, a) is a fully-connected layer with u units and activation type a. $n_{\text{surf-params}}$ is the total number of parameters defining the surface (both for the global geometric transformation and the intrinsic shape parameters, see Section 3.5).

(each data point includes 5 input depth frames), and learning rate of $2 \cdot 10^{-4}$. The clipping hyperparameter in (S1) was set to $c = 5.0$.

D.2. Synthetic Colonoscopy Experiment (Section 5.1)

As mentioned in the paper, the model was trained with the Adam optimizer [4] for 10 epochs with a batch size of 8 and a learning rate of $2 \cdot 10^{-4}$. The depth maps were (spatially) downsampled to 32 pixels in each dimension before being input to the network. The clipping hyperparameter in (S1) was set to $c = 20.0$.

D.3. Real Colonoscopy Experiment (Section 5.2)

As mentioned in the paper, the model was trained with the Adam optimizer [4] for 10 epochs with a batch size of 2 and a learning rate of $2 \cdot 10^{-5}$. The depth maps were (spatially) downsampled to 32 pixels in each dimension before being input to the network. The clipping hyperparameter in (S1) was set to $c = 10.0$.

D.4. Point Cloud Reconstruction

As part of our training scheme, we reconstruct a point cloud from the input sequence of depth maps and camera poses (see the "reconstruction" block in the diagram of Figure 2). Each depth map is "unprojected" separately in a differentiable manner with the perspective transform functionality of the Tensorflow Graphics package[3]. Then, a point cloud is "assembled" by translating and rotating each set of "unprojected" 3D points according to the corresponding camera pose.

D.5. Loss parameters

In all experiments, the hyper-parameter σ in the loss function (11) is initialized to be the median distance between points in the shape model (for some canonical q), and further fine-tuned on a validation set.

References

[1] Vincent Casser, Soeren Pirk, Reza Mahjourian, and Anelia Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In *AAAI Conference on Artificial Intelligence*, volume 33, pages 8001–8008, 2019. 1

[2] Daniel Freedman, Yochai Blau, Liran Katzir, Amit Aides, Ilan Shimshoni, Danny Veikherman, Tomer Golany, Ariel Gordon, Greg Corrado, Yossi Matias, and Ehud Rivlin. Detecting deficient coverage in colonoscopies. *IEEE Transactions on Medical Imaging*, 2020. 1

[3] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 2

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2, 3

[5] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. SMPL: A skinned multi-person linear model. *ACM Transactions on Graphics*, 34(6):1–16, 2015. 1

[6] W StraBer. *Schnelle Kurven-und Flachendarstellung auf graphischen Sichtgeraten*. PhD thesis, 1974. 1

[7] Blender. https://www.blender.org/. 1

[3] https://www.tensorflow.org/graphics/api_docs/python/tfg/rendering/camera/perspective