Supplementary Materials for "Self-supervised Video Object Segmentation by Motion Grouping"

Charig Yang Hala Lamdouar Erika Lu Andrew Zisserman Weidi Xie

Visual Geometry Group, University of Oxford

{charig,lamdouar,erika,az,weidi}@robots.ox.ac.uk

https://charigyang.github.io/motiongroup/

1. Training Details

In this section, we include the details for reproducing our results, for example, architectures, pseudo-codes and hyper-parameters.

1.1. Encoder & Decoder

The backbone of network architecture for the model are shown in Table 1, we refer the readers to the pseudocode for iterative binding module.

	stage	operation	output sizes
	input	–	$3 \times 128 \times 224$
Encoder	conv1	$[5 \times 5, 64] \times 2$	$64 \times 128 \times 224$
	mp1	maxpool, stride = 2	$64 \times 64 \times 112$
	conv2	$[5 \times 5, 128] \times 2$	$128 \times 64 \times 112$
	mp2	maxpool, stride = 2	$128 \times 32 \times 56$
	conv3	$[5 \times 5, 256] \times 2$	$256 \times 32 \times 56$
Decoder	conv^T1	$5 \times 5, 64$, stride $= 2$	$64 \times 2 \times 32 \times 56$
	conv^T2	$5 \times 5, 64$, stride $= 2$	$64 \times 2 \times 64 \times 112$
	conv^T3	$5 \times 5, 64$, stride $= 2$	$64 \times 2 \times 128 \times 224$
	outconv	$5 \times 5, 64$, $5 \times 5, 4$	$4 \times 2 \times 128 \times 224$

Table 1. Network architecture. All convolutions have padding 2 to preserve spatial resolution, and are followed by instance normalization and ReLU activation, except the final layer. The details of the iterative binding module is in Figure 1.

1.2. Iterative Binding Module

The pseudocode for the iterative binding module is shown in Figure 1. The full code is available in the submitted source code, and will be made publicly available.

1.3. Hyperparameters

For all datasets, we train with batch size of 64, although note that this corresponds to 32 *pairs* of optical flow in order to train with consistency loss. Our initial learning rate is 5e-4, with first 200 steps being warmup, and decays by half every 8e4 iterations. During this decay, the scale for entropy and consistency loss is also increased by a factor of 5, gradually encouraging the predicted alpha channel to be binary. We train the algorithm for about 300k iterations.

Algorithm: Pseudo Code for Iterative Binding / Grouping

```
# inputs: feature maps + position encoding
slots = Embedding(D, 2)    # [bsz, 2, D]
K = project_K(inputs)      # [bsz, HW, D]
V = project_V(inputs)      # [bsz, HW, D]
for _ in range(T):         # T iterations
    slots = LayerNorm(slots) # [bsz, 2, D]
    Q = project_Q(slots)     # [bsz, 2, D]
    scores = dot(K, Q.t())   # [bsz, HW, 2]
    attn = softmax(scores / sqrt(D), axis=-1) # [bsz, HW, 2]
    updates = weighted_mean(attn.t(), V)      # [bsz, 2, D]
    slots = GRU(slots, updates)               # [bsz, 2, D]
    slots = slots + MLP(LayerNorm(slots))     # [bsz, 2, D]
```

Figure 1. Pseudocode for iterative binding. The linear projections for key (K) and value (V) have 256 dimensions. The MLP has two layers, both with 256 dimensions, with ReLU in between.

2. MoCA dataset curation

The MoCA dataset contains 141 high-definition video sequences, with an average duration of 11 seconds. These sequences were collected from YouTube with resolution 720×1280, and sampled at 24 fps, resulting in 37K frames depicting 67 kinds of camouflaged animals moving in natural scenes. Both temporal and spatial annotations are provided in the form of tight bounding boxes on every 5th frame. We use a modified version of this dataset in order to make it more suitable for segmentation tasks. We outline the process and rationale below.

- We crop away the channel logos and empty border spaces, and resize the low-resolution images to the

same resolution as the other images in the dataset (at 720×1280). We then adjust the ground-truth annotations accordingly.

- The original authors resampled all videos to 24 fps even when some original videos have less, causing some consecutive frames to be identical. To alleviate this, we sample every 3 frames from the original dataset, up to 100 frames per video.
- For annotations, we use linear interpolation to generate the missing frames' bounding boxes, resulting in a dense frame-wise annotation.
- The authors provided the motion labels for each annotated frame (locomotion, deformation, static), so we filter away videos with predominantly no locomotion.
- We also further discard videos that contain large amount of frames where the motion does not belong to the primary object.

This eventually results in 88 video sequences and 4803 frames, which we will release for fair comparison.

3. Results breakdown

The main evaluation metric used in this paper is the Jaccard score ($\mathcal{J}$), which is the intersection-over-union between the predicted and ground-truth masks. In line with previous works, we show the per-category results breakdown for our model on DAVIS2016, SegTrackv2, FBMS59, and MoCA in Tables 2-5. Note that, since we focus on both speed and accuracy, the predictions are only of 128×224 pixels, and we directly upsample this prediction to the original resolution and compare with the groundtruth.

4. Qualitative results

We show more qualitative results in Figures 2 and 3.

Sequence	$\mathcal{J}$(M)
bird of paradise	0.791
birdfall	0.300
bmx	0.609
cheetah	0.370
drift	0.797
frog	0.733
girl	0.746
hummingbird	0.506
monkey	0.751
monkeydog	0.133
parachute	0.914
penguin	0.697
soldier	0.741
worm	0.326
seq avg	0.601
frames avg	0.586

Table 2. Sequence-wise results on SegTrackv2.

Sequence	$\mathcal{J}$(M)	$\mathcal{J}$(R)	$\mathcal{J}$(D)	$\mathcal{F}$(M)	$\mathcal{F}$(R)	$\mathcal{F}$(D)
bear	0.766	1.000	-4.5	0.640	0.914	-1.9
blackswan	0.795	1.000	5.9	0.658	0.980	7.3
bmx-bumps	0.373	0.114	8.3	0.260	0.011	10.8
bmx-trees	0.744	1.000	14.5	0.666	0.909	13.9
boat	0.602	0.787	44.5	0.724	0.936	22.4
breakdance	0.792	0.990	-2.4	0.814	1.000	10.8
breakdance-flare	0.826	1.000	10.2	0.771	1.000	16.7
bus	0.391	0.000	-4.4	0.248	0.000	-5.0
camel	0.540	0.679	9.3	0.667	0.923	-9.8
car-roundabout	0.479	0.500	1.2	0.404	0.146	-10.6
car-shadow	0.879	1.000	-0.2	0.818	1.000	-3.3
car-turn	0.492	0.537	3.4	0.560	0.683	6.8
cows	0.690	0.898	-5.0	0.653	0.852	4.4
dance-jump	0.815	1.000	-4.8	0.683	1.000	5.9
dance-twirl	0.739	0.960	7.4	0.803	0.900	28.5
dog	0.701	0.789	2.9	0.477	0.526	9.3
dog-agility	0.810	1.000	7.7	0.858	1.000	4.0
drift-chicane	0.777	1.000	7.3	0.611	0.646	26.3
drift-straight	0.864	1.000	-5.2	0.685	1.000	-6.8
drift-turn	0.593	0.654	27.7	0.220	0.000	20.4
Average	0.683	0.795	6.2	0.611	0.721	7.5

Table 3. Full results on DAVIS2016. J refers to the Jaccard score, while the F-measure refers to the contour accuracy. M, R, and D refers to mean, recall and decay respectively.

Sequence	$\mathcal{J}$(M)
camel01	0.281
cars1	0.846
cars10	0.322
cars4	0.826
cars5	0.842
cats01	0.672
cats03	0.640
cats06	0.362
dogs01	0.629
dogs02	0.636
farm01	0.816
giraffes01	0.322
goats01	0.375
horses02	0.628
horses04	0.566
horses05	0.334
lion01	0.399
marple12	0.680
marple2	0.750
marple4	0.799
marple6	0.450
marple7	0.567
marple9	0.537
people03	0.598
people1	0.761
people2	0.842
rabbits02	0.415
rabbits03	0.319
rabbits04	0.400
tennis	0.561
seq avg	0.573
frames avg	0.531

Table 4. Sequence-wise results on FBMS59.

sequence	$\mathcal{J}$	$\tau_{0.5}$	$\tau_{0.6}$	$\tau_{0.7}$	$\tau_{0.8}$	$\tau_{0.9}$	avg
arabian_horn_viper	0.709	0.990	0.909	0.586	0.091	0.000	0.515
arctic_fox	0.381	0.404	0.383	0.298	0.191	0.000	0.255
arctic_fox_1	0.879	0.913	0.913	0.913	0.913	0.652	0.861
arctic_wolf_0	0.705	0.929	0.859	0.596	0.202	0.051	0.527
arctic_wolf_1	0.712	0.795	0.795	0.795	0.744	0.256	0.677
bear	0.508	0.611	0.453	0.221	0.074	0.000	0.272
black_cat_0	0.499	0.556	0.476	0.302	0.048	0.000	0.276
black_cat_1	0.086	0.030	0.020	0.010	0.000	0.000	0.012
crab	0.594	0.800	0.400	0.200	0.000	0.000	0.280
crab_1	0.288	0.309	0.200	0.073	0.000	0.000	0.116
cuttlefish_0	0.222	0.194	0.032	0.000	0.000	0.000	0.045
cuttlefish_1	0.034	0.043	0.000	0.000	0.000	0.000	0.009
cuttlefish_4	0.655	1.000	0.846	0.231	0.000	0.000	0.415
cuttlefish_5	0.724	0.870	0.870	0.783	0.304	0.043	0.574
dead_leaf_butterfly_1	0.784	0.913	0.783	0.739	0.696	0.304	0.687
desert_fox	0.470	0.362	0.234	0.191	0.149	0.106	0.209
devil_scorpionfish	0.938	1.000	1.000	1.000	0.913	0.826	0.948
devil_scorpionfish_1	0.913	1.000	1.000	1.000	1.000	0.565	0.913
devil_scorpionfish_2	0.857	0.968	0.903	0.871	0.806	0.548	0.819
egyptian_nightjar	0.765	0.905	0.842	0.789	0.611	0.158	0.661
elephant	0.728	0.783	0.652	0.609	0.565	0.348	0.591
flatfish_0	0.575	0.677	0.657	0.636	0.485	0.141	0.519
flatfish_1	0.682	0.848	0.835	0.722	0.354	0.051	0.562
flatfish_2	0.765	0.839	0.774	0.774	0.742	0.645	0.755
flatfish_4	0.697	0.958	0.895	0.568	0.126	0.000	0.509
flounder	0.896	1.000	1.000	0.986	0.986	0.437	0.882
flounder_3	0.505	0.429	0.286	0.143	0.143	0.000	0.200
flounder_4	0.767	0.949	0.897	0.769	0.487	0.128	0.646
flounder_5	0.681	0.797	0.747	0.696	0.468	0.165	0.575
flounder_6	0.683	0.768	0.677	0.616	0.465	0.263	0.558
flounder_7	0.719	0.930	0.845	0.662	0.254	0.028	0.544
flounder_8	0.707	0.925	0.774	0.645	0.409	0.000	0.551
flounder_9	0.636	0.821	0.718	0.436	0.231	0.026	0.446
fossa	0.280	0.143	0.000	0.000	0.000	0.000	0.029
goat_0	0.589	0.707	0.636	0.414	0.212	0.020	0.398
goat_1	0.744	0.930	0.930	0.704	0.380	0.085	0.606
groundhog	0.525	0.646	0.525	0.374	0.162	0.010	0.343
hedgehog_0	0.329	0.298	0.170	0.085	0.021	0.021	0.119
hedgehog_1	0.471	0.533	0.400	0.333	0.067	0.067	0.280
hedgehog_2	0.771	0.800	0.733	0.733	0.600	0.267	0.627
hedgehog_3	0.486	0.564	0.359	0.282	0.128	0.026	0.272
hermit_crab	0.674	0.806	0.806	0.677	0.419	0.065	0.555
ibex	0.390	0.513	0.359	0.128	0.000	0.000	0.200
jerboa	0.555	0.739	0.435	0.348	0.130	0.000	0.330

sequence	$\mathcal{J}$	$\tau_{0.5}$	$\tau_{0.6}$	$\tau_{0.7}$	$\tau_{0.8}$	$\tau_{0.9}$	avg
jerboa_1	0.450	0.452	0.323	0.226	0.097	0.000	0.219
lichen_katydid	0.502	0.455	0.323	0.162	0.020	0.010	0.194
lion_cub_0	0.728	0.972	0.873	0.549	0.282	0.127	0.561
lion_cub_1	0.571	0.687	0.556	0.354	0.141	0.010	0.349
lion_cub_3	0.179	0.182	0.141	0.081	0.051	0.000	0.091
lioness	0.423	0.419	0.129	0.000	0.000	0.000	0.110
marine_iguana	0.376	0.217	0.130	0.000	0.000	0.000	0.070
markhor	0.808	0.909	0.855	0.800	0.709	0.364	0.727
meerkat	0.778	0.871	0.774	0.742	0.710	0.323	0.684
mountain_goat	0.742	1.000	0.968	0.710	0.226	0.065	0.594
nile_monitor_1	0.524	0.616	0.434	0.283	0.121	0.000	0.291
octopus	0.637	0.838	0.687	0.273	0.131	0.020	0.390
octopus_1	0.411	0.242	0.091	0.061	0.061	0.010	0.093
peacock_flounder_0	0.884	0.931	0.931	0.931	0.931	0.701	0.885
peacock_flounder_1	0.812	0.970	0.929	0.859	0.667	0.222	0.729
peacock_flounder_2	0.873	1.000	1.000	0.989	0.926	0.368	0.857
polar_bear_0	0.622	0.845	0.676	0.352	0.141	0.000	0.403
polar_bear_1	0.487	0.603	0.556	0.476	0.175	0.016	0.365
polar_bear_2	0.792	0.949	0.846	0.744	0.641	0.308	0.697
pygmy_seahorse_2	0.478	0.582	0.364	0.255	0.036	0.000	0.247
pygmy_seahorse_4	0.654	0.935	0.839	0.516	0.000	0.000	0.458
rodent_x	0.738	0.870	0.826	0.696	0.435	0.174	0.600
scorpionfish_0	0.622	0.761	0.648	0.521	0.408	0.127	0.493
scorpionfish_1	0.616	0.681	0.574	0.426	0.319	0.128	0.426
scorpionfish_2	0.842	0.962	0.949	0.924	0.823	0.380	0.808
scorpionfish_3	0.804	0.975	0.861	0.785	0.595	0.354	0.714
scorpionfish_4	0.800	1.000	0.949	0.821	0.513	0.231	0.703
scorpionfish_5	0.899	1.000	1.000	1.000	0.957	0.478	0.887
seal_1	0.865	1.000	0.913	0.870	0.870	0.652	0.861
seal_2	0.676	0.800	0.655	0.455	0.291	0.145	0.469
seal_3	0.445	0.400	0.200	0.067	0.000	0.000	0.133
shrimp	0.772	0.933	0.867	0.733	0.667	0.133	0.667
snow_leopard_0	0.760	1.000	0.949	0.692	0.359	0.077	0.615
snow_leopard_1	0.772	0.826	0.696	0.652	0.652	0.522	0.670
snow_leopard_2	0.883	1.000	0.989	0.968	0.863	0.526	0.869
snow_leopard_3	0.608	0.778	0.556	0.417	0.333	0.083	0.433
snow_leopard_6	0.816	0.830	0.787	0.787	0.723	0.660	0.757
snow_leopard_7	0.679	0.871	0.806	0.581	0.258	0.000	0.503
snow_leopard_8	0.556	0.702	0.596	0.447	0.191	0.000	0.387
snowy_owl_0	0.564	0.532	0.340	0.298	0.128	0.000	0.260
spider_tailed_horned_viper_0	0.473	0.400	0.333	0.267	0.067	0.067	0.227
spider_tailed_horned_viper_1	0.558	0.620	0.423	0.310	0.254	0.056	0.332
spider_tailed_horned_viper_2	0.848	0.964	0.945	0.909	0.782	0.564	0.833
spider_tailed_horned_viper_3	0.860	1.000	1.000	1.000	0.677	0.387	0.813
seq avg	0.634	0.734	0.640	0.522	0.361	0.166	0.485
frames avg	0.634	0.742	0.654	0.524	0.351	0.147	0.484

Table 5. Results breakdown for MoCA.

Figure 2. Qualitative results on DAVIS2016 and MoCA, respectively.

Figure 3. Qualitative results on SegTrackv2 and FBMS59, respectively.