# Unsupervised Part Representation by Flow Capsules

**Sara Sabour** [1 2] **Andrea Tagliasacchi** [1 2] **Soroosh Yazdani** [1] **Geoffrey E. Hinton** [1 2] **David J. Fleet** [1 2]

## Abstract

Capsule networks aim to parse images into a hierarchy of objects, parts and relations. While promising, they remain limited by an inability to learn effective low level part descriptions. To address this issue we propose a way to learn primary capsule encoders that detect atomic parts from a single image. During training we exploit motion as a powerful perceptual cue for part definition, with an expressive decoder for part generation within a layered image model with occlusion. Experiments demonstrate robust part discovery in the presence of multiple objects, cluttered backgrounds, and occlusion. The part decoder infers the underlying shape masks, effectively filling in occluded regions of the detected shapes. We evaluate FlowCapsules on unsupervised part segmentation and unsupervised image classification.

## 1. Introduction

Humans learn to perceive shapes in terms of parts and their spatial relationships (Singh & Hoffman, 2001). Studies show that infants form early object perception by dividing visual inputs into units that move rigidly and separately (Spelke, 1990), and they do so in a largely unsupervised way. Inspired by this and recent work on part discovery, we propose a self-supervised way to learn visual part descriptors for Capsule networks (Hinton et al., 2011).

Capsule networks represent objects in terms of primary part descriptors, in a local canonical frame, and coordinate transformations between parts and the whole. As a result of their architecture, they are robust to various challenges, including viewpoint changes and adversarial attacks. Stacked capsule network architectures (SCAE) (Kosiorek et al., 2019) have shown promising results on a number of simple image datasets. Nevertheless, because they are trained with an image reconstruction loss, foreground-background separation and part discovery in cluttered images remain challenging.

[1]Google Research, Brain Team. [2]Department of Computer Science, University of Toronto. Correspondence to: Sara Sabour <sasabour@google.com>.

*Proceedings of the $38^{th}$ International Conference on Machine Learning*, PMLR 139, 2021. 

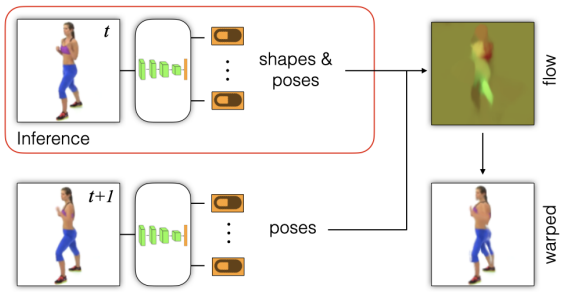


Figure 1: **Self-supervised training for learning primary capsules:** An image encoder is trained to decompose the scene into a collection of *primary capsules*. Learning is accomplished in an unsupervised manner, using flow estimation from capsule shapes and poses as a *proxy* task.

This paper introduces a way to learn encoders for object parts (aka., primary capsules) to address these challenges. The encoder takes as input a single image (see Fig. 1), but for training part discovery, it uses motion-based self-supervision (Bear et al., 2020; Mahendran et al., 2018). Like the classical literature on perceptual organization and common fate in Gestalt psychology (Spelke, 1990; Wagemans et al., 2012), we exploit the fact that regions of the image that move together often belong together. This is a strong perceptual cue that facilitates foreground-background segmentation and part discovery, and allows one to disentangle texture and other aspects of appearance from shape.

The proposed part encoder captures the underlying part shapes, their relative poses, and their relative depth ordering (see Fig. 2). The introduction of depth ordering is particularly useful in order to account for occlusion, as it is in layered motion models (Wang & Adelson, 1994). In this way, learning aggregates information about shape over many images, even though a given part may rarely be visible in its entirety in any single frame. In essence, the model prefers simple part-based descriptions, where many variations in appearance can be explained by a coordinate transform or by occlusion, rather than by changes in shape per se.

We demonstrate the FlowCapsules approach on several datasets showcasing challenges due to texture, occlusions, scale, and instance variation. We compare FlowCapsules to recent related work including PSD (Xu et al., 2019) and R-NEM (Van Steenkiste et al., 2018), where part masks and dynamics are learnt using motion. FlowCapsules provide

unsupervised shape segmentation, even in the face of texture and backgrounds, outperforming PSD (Xu et al., 2019). FlowCapsules also provide a depth ordering to account for occlusion, with the added benefit that part inference yields shape completion when parts are partially occluded.

We also report unsupervised classification of images using FlowCapsules part embeddings. We compare our results on several datasets with different challenges against SCAE (Kosiorek et al., 2019). Experiments show that FlowCapsules consistently outperform SCAE in unsupervised object classification, especially on images with textured backgrounds.

## 2. Related Work

Given the vast literature of part-based visual representations, we focus here only on the most closely related recent work.

Transforming autoencoders (Hinton et al., 2011) introduced capsule networks. Sabour et al. (2017) revisited the capsule concept and introduced capsule hierarchies for object classification, and subsequent work has produced improved routing algorithms (Hinton et al., 2018; Hahn et al., 2019; Ahmed & Torresani, 2019). Nevertheless, learning primary capsules from images has remained largely untouched. An analogy to text understanding would be a language with a well defined grammar and parser, but no good definition or representation of words. We introduce a technique for learning primary capsules to address this shortcoming.

Unsupervised capsule learning with an *image reconstruction* loss for part discovery has been explored by (Kosiorek et al., 2019) and (Rawlinson et al., 2018). Several works learn capsule autoencoders for 3D objects from point clouds (Srivastava et al., 2019; Zhao et al., 2019; Sun et al., 2020). But with the exception of capsule models trained with class labels (Hinton et al., 2018) or segmentation masks (LaLonde & Bagci, 2018; Duarte et al., 2018), previous methods struggle with natural images. Object-background discrimination with cluttered, textured scenes is challenging for an image reconstruction loss. With self-supervised training and visual motion, FlowCapsules achieve part discovery without ground truth labels or segmentation masks.

Recent approaches to object-centric learning, e.g., MONet (Burgess et al., 2019), IODINE (Greff et al., 2019), and Slot-attention (Locatello et al., 2020), focus on learning object representations via image reconstruction. Beyond the need to reconstruct image backgrounds, they require iterative refinement for symmetry breaking and forcing scenes into slots. In contrast, FlowCapsule learning relies on reconstruction of the flow rather than the image, and with motion as the primary cue, scenes are decomposed into parts without needing iterative refinement. Most recently, (Bear et al., 2020; Veerapaneni et al., 2020) extend such networks to incorporate motion, but still rely on iterative refinement. FlowCapsule encodings further disentangle shape and pose, enabling shape completion during partial occlusion.

FlowCapsules currently represent 2D objects, reminiscent of layered models but with a feedforward encoder. Classical layered models (Wang & Adelson, 1994; Jojic & Frey, 2001) used mixture models and assigned pixels to layers independently, often failing to capture the coherence or compactness of object occupancy. Some methods use MRFs to encourage spatial coherence (Weiss, 1997). Others enforce coherence via local parametric masks (Jepson et al., 2002).

Visual motion is well-known to be a strong cue for self-supervised learning. For example, (Vijayanarasimhan et al., 2017) learn to infer depth, segmentation, and relative 3D motion from consecutive frames using self-supervised learning with photometric constraints. These and related methods use *optical flow* or multiple frames as an input. FlowCapsules use video frame pairs during training, but the part encoder (see Fig. 2), takes as input a *single* frame. In essence, it learns to decompose images into *movable* objects.

S3CNNs (Mahendran et al., 2018) take a similar approach, but do not learn per-part shape encoders or coordinate frames. Rather, they learn to group pixels using patch-wise affine flow, rather than expressing flow in terms of coherent parts and their coordinate frames. A closely related method is PSD (Xu et al., 2019), which uses optical flow to learn hierarchical part-based models of shape and dynamics in a layered image model. It trains a VAE flow encoder and an image encoder to predict the next frame. Both PSD and S3CNNs require ground truth flow during training and lack an explicit canonical part descriptor like FlowCapsules. There also exist related methods applied to point could data; e.g., SE3Net (Byravan & Fox, 2017) uses a part-based representation, taking a point cloud and an action vector as input and predicts the point cloud in the next frame.

KeypointNet (Suwajanakorn et al., 2019) addresses the problem of keypoint discovery. One might view Flow-Capsules as a generalization from a sparse to a dense setting, and from a single object to multiple objects,

Our work is also related to generative shape models. Huang & Murphy (2016) learn parts in a layered model with depth order and occlusion. Given an image, *variational* inference is used to infer shape and foreground/background separation. FlowCapsule encoders, by comparison, are trained as autoencoders and are therefore easier to learn. Several recent papers learn generative models that *disentangle* shape and deformation (Skafte & Hauberg, 2019; Deng et al., 2021). FlowCapsules disentangle shape and transformations from canonical to image coordinates. In doing so they decompose shapes into multiple near-rigid parts with occlusions. FlowCapsules thereby disentangle shape at a finer granularity. Also, Skafte & Hauberg (2019) and Deng et al. (2021) use

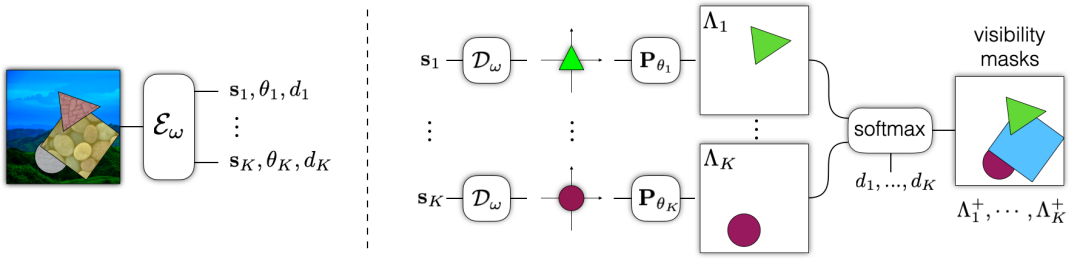


Figure 2: **Inference architecture.** (left) The encoder $\mathcal{E}_\omega$ parses an image into part capsules, each comprising a shape vector $s_k$, a pose $\theta_k$, and a scalar depth value $d_k$. (right) The shape decoder $\mathcal{D}_\omega$ is an implicit function. It takes as input a shape vector, $s_k$, and a location in canonical coordinates and returns the probability that the location is inside the shape. Shapes are mapped to image coordinates, using $\theta_k$, and layered according to the relative depths $d_k$, yielding visibility masks.

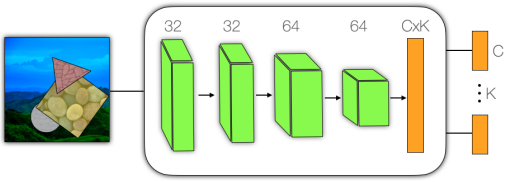


Figure 3: **Encoder architecture.** The encoder comprises convolution layers with ReLU activation, followed by downsampling via $2\times2$ AveragePooling. Following the last convolution layer is a `tanh` fully connected layer, and a fully connected layer grouped into $K$, $C$-dimensional capsules.

an image reconstruction loss, much like SCAE, while FlowCapsules only encode shape silhouettes, which simplifies training and the disentangled representation.

## 3. Model

Our goal is to learn an encoder that parses images of familiar shapes into parts. To facilitate training, and downstream tasks, we also learn a decoder capable of generating segment masks for the parts in the image. Below we describe the form of the proposed capsule encoder and the mask decoder. We then describe the objective and training procedure.

**Image encoder.** The capsule encoder $\mathcal{E}_\omega$, with parameters $\omega$, encodes a given image as a collection of $K$ primary capsules. The architecture we propose is depicted in Figure 3. Each capsule, $\mathbf{c}_k$, comprises a vector $\mathbf{s}_k$ that encodes the *shape* of the part, a *pose* vector $\boldsymbol{\theta}_k$, and a depth scalar $d_k$:

$$\mathcal{E}_\omega(I) = \{\mathbf{c}_0, \ldots, \mathbf{c}_k\}, \quad \mathbf{c}_k = (\mathbf{s}_k, \boldsymbol{\theta}_k, d_k)\,. \tag{1}$$

Capsule shapes are encoded in a canonical coordinate frame. The scalar $d_k$ specifies relative inverse depth (larger for foreground objects). The pose vector specifies a mapping from part-centric coordinates $\mathbf{v}$ to image coordinates $\mathbf{u}$ (or scene coordinates more generally), i.e., $\mathbf{u} = \mathbf{P}_{\boldsymbol{\theta}_k}\mathbf{v}$.

As we focus on planar layered models with depth $d$, we define $\mathbf{P}_{\boldsymbol{\theta}_k}$ to be a conformal map. Accordingly, let $\boldsymbol{\theta}_k \in \mathbb{R}^4$, where $[\boldsymbol{\theta}_k]_{0,1}$ represents the translation, $[\boldsymbol{\theta}_k]_2$ is the rotation angle, and $[\boldsymbol{\theta}_k]_3$ is the change in scale. More concretely (subscript $k$ is dropped for readability):

$$\mathbf{P}_{\boldsymbol{\theta}} = \begin{bmatrix} \boldsymbol{\theta}_3\cos(\boldsymbol{\theta}_2) & -\boldsymbol{\theta}_3\sin(\boldsymbol{\theta}_2) & \boldsymbol{\theta}_0 \\ \boldsymbol{\theta}_3\sin(\boldsymbol{\theta}_2) & \boldsymbol{\theta}_3\cos(\boldsymbol{\theta}_2) & \boldsymbol{\theta}_1 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Taken together, $\mathbf{c}_k \in \mathbb{R}^C$, where $\boldsymbol{\theta}_k \in \mathbb{R}^4$, $d_k \in \mathbb{R}$, and therefore $\mathbf{s}_k \in \mathbb{R}^{C-5}$.

**Mask decoder.** A mask decoder facilitates self-supervised learning of the encoder, as well as downstream segmentation tasks. It allows one to visualize parts and connect them to image observations. As depicted in Figure 2, the mask decoder $\mathcal{D}_\omega$ generates an object silhouette (or mask) in canonical coordinates, which is then mapped to image coordinates, incorporating occlusion and visibility.

Our current decoder architecture is depicted in Figure 5. The mask decoder, given the latent code $\mathbf{s}_k$, represents the part shape in a canonical coordinate frame, $\mathcal{D}_\omega(\mathbf{v}; \mathbf{s}_k)$. This is then mapped into image coordinates according to the pose vector $\boldsymbol{\theta}_k$, yielding the shape mask $\Lambda_k$ in the image frame:

$$\Lambda_k(\mathbf{u}) \;=\; \mathcal{D}_\omega(\mathbf{P}_{\boldsymbol{\theta}_k}^{-1}\mathbf{u}\,;\, \mathbf{s}_k)\,, \tag{3}$$

where the map $\mathbf{P}_{\boldsymbol{\theta}_k}$ has parameters $\boldsymbol{\theta}_k$. We also note that $\Lambda_k$ is a *function* of spatial position and a latent code (Chen & Zhang, 2019; Mescheder et al., 2019), but unlike previous work, our encoder disentangles individual part shapes and their poses with respect to canonical coordinates.

**Occlusion:** With opaque objects, parts will not always be visible in their entirety. To account for occlusion, part masks are layered according to their depth order, thereby determining the visible portion of each part in a given image. To ensure differentiable image formation, enabling gradient-based learning, we treat the scalar $d_k$ as a logit, and apply a softmax across the logits (depths) of all parts at every pixel to generate the visibility masks (Gadelha et al., 2019); see Fig. 2. The visible portion of the $k$-th part is given by

$$\Lambda_k^+(\mathbf{u}) = \frac{e^{d_k\Lambda_k(\mathbf{u})}}{\sum_{k'} e^{d_{k'}\Lambda_{k'}(\mathbf{u})}} \tag{4}$$

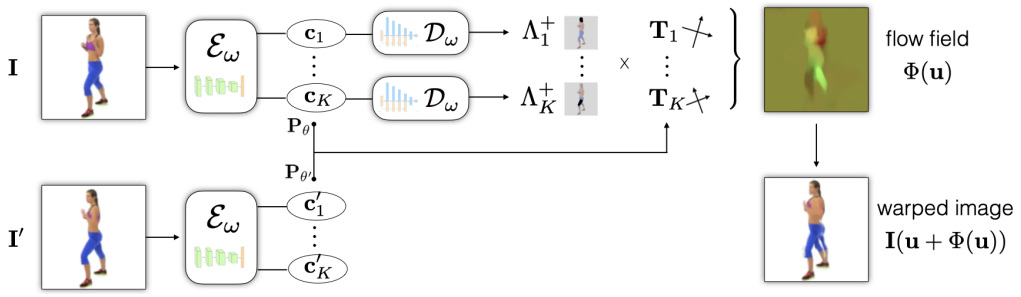


Figure 4: **Self-supervised training** – Training uses a *proxy* motion task in which the capsule encoder is applied to a pair of successive video frames, providing $K$ primary capsule encodings from each frame. Visible part masks, $\Lambda_k^+$, and their corresponding poses, $\mathbf{P}_{\boldsymbol{\theta}}$, determine a flow field $\Phi$ that is used to warp image $\mathbf{I}$ to predict $\mathbf{I}'$ in the loss $\mathcal{L}_{\text{render}}$ in (7).

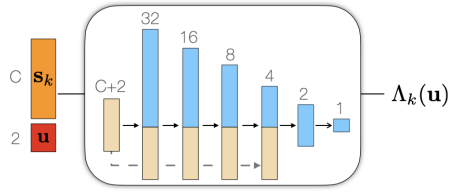


Figure 5: **Decoder architecture.** A neural implicit function (Chen & Zhang, 2019) is used to represent part masks. An MLP with SELU activations (Klambauer et al., 2017) takes as input a shape vector $s$ and a pixel position $\mathbf{u}$. Applied to a pixel grid, it produces a logit grid for the mask.

As the gap between the largest $d_k$ and other values grows, the softmax approaches the argmax, ideal for opaque layers.

A typical auto-encoder might reconstruct the image in terms of these masks, to formulate an image reconstruction loss. The problem with such an approach is that the encoder would also need to encode other properties of the images, such as texture, lighting and the background, with pixel level accuracy. To avoid this problem, here we aim only to learn an encoder for the part shapes, positions and depth layering. To this end we consider a form of self-supervised learning that relies on primarily on motion (optical flow) between consecutive frames in video. The use of flow provides a strong image cue for the segmentation of parts, *without* the need to model texture, lighting and other fine-grained properties tied to appearance.

## 4. Self-Supervised Learning

Training the capsule encoder exploits motion as a visual cue for separating objects and their parts from the immediate background. To that end, we assume that the training data comprises pairs of adjacent video frames. Given an image pair, the encoder provides an *ordered* set of capsules for each of the two images. The poses from corresponding capsules and their visibility masks then determine a deformation that is used to warp one frame to predict the other. This allows use of brightness constancy and other common objectives in optical flow estimation as a self-supervised training loss.

In more detail, let the two images of a training pair be denoted $\mathbf{I}$ and $\mathbf{I}'$. As shown in Figure 4, the capsule encoder extracts an ordered set of capsules from each image. The part capsules are denoted $\mathbf{c}_k{=}(\mathbf{s}_k, \boldsymbol{\theta}_k, d_k)$ and $\mathbf{c}'_k{=}(\mathbf{s}'_k, \boldsymbol{\theta}'_k, d'_k)$, for $k \in \{1, ..., K\}$. From corresponding part capsules we then compute the predicted flow $\Phi$ from the capsule poses, yielding a mapping $\mathbf{T}_k$ from one image to the next,

$$\mathbf{T}_k \;=\; \mathbf{P}_{\boldsymbol{\theta}'_k} \circ (\mathbf{P}_{\boldsymbol{\theta}_k})^{-1} \,. \tag{5}$$

This transform maps image locations in $\mathbf{I}$ to the canonical coordinate frame of part $k$, and then into the next frame $\mathbf{I}'$. When combined with the layered visibility masks, this provides the flow field:

$$\Phi(\mathbf{u} \,|\, \mathcal{E}_\omega(\mathbf{I}), \mathcal{E}_\omega(\mathbf{I}')) \;=\; \sum_{k=1}^{K} \underbrace{\Lambda_k^+(\mathbf{u})}_{\text{visibility}} \underbrace{[\mathbf{T}_k(\mathbf{u}) - \mathbf{u}]}_{\text{flow of } k\text{-th capsule}} \tag{6}$$

where $\mathbf{u} \in [-1, 1]^2$ denotes 2D normalized image coordinates. Note that the use of $[\mathbf{T}_k(\mathbf{u}) - \mathbf{u}]$ in (6) ensures that the generation of an *identity* flow is the easiest prediction for the network $\mathbf{T}_k(\mathbf{u})$ to make (like a residual connection).

Given the estimated flow between a given training pair, we warp the pixels of $\mathbf{I}$ according to $\Phi$, providing a prediction for $\mathbf{I}'$. Then we optimize an L2 brightness constancy loss on the residual errors between our warped version of the first frame and the second frame,

$$\mathcal{L}_{\text{render}} \;=\; \mathbb{E}_{\mathbf{u}\sim[0,1]^2} \,\|\, \mathbf{I}(\mathbf{u} + \Phi(\mathbf{u})) - \mathbf{I}'(\mathbf{u}) \,\|_2^2 \,, \tag{7}$$

where we abbreviated $\Phi(\mathbf{u} \,|\, \mathcal{E}_\omega(\mathbf{I}), \mathcal{E}_\omega(\mathbf{I}'))$ by $\Phi(\mathbf{u})$ for notational simplicity.

We also exploit two simple but effective regularizers on flow and the canonical shape representation. They are useful as we do not make use of ground truth segmentation masks or flow fields during training. The first regularizer, $\mathcal{L}_{\text{smooth}}$, is a smoothness term often used in optical flow estimation (Jason et al., 2016) to enhance gradient propagation through larger movements and regions with negligible image gradients:

$$\mathcal{L}_{\text{smooth}} \;=\; \left\| \frac{\partial \Phi}{\partial u_x}, \frac{\partial \Phi}{\partial u_y} \right\|_2^2 . \tag{8}$$

The second regularizer encourages part shapes to be centered at the origin in the canonical coordinate frame; i.e.,

$$\mathcal{L}_{\text{center}} = \frac{1}{K}\sum_{k=1}^{K}\frac{\sum_{\mathbf{v}}\|\mathbf{v}\,\Lambda_k(\mathbf{v})\|_2^2}{\sum_{\mathbf{v}'}\Lambda_k(\mathbf{v}')} \tag{9}$$

Keeping parts centered at $(0,0)$ improves the inference of rotations. For example, a part located far from the origin can easily be projected outside the image during training. Keeping it near the center tends to produce a smoother loss function. The final loss is a weighted sum of the render loss and the two regularizers.

## 5. Experiments

We evaluate FlowCapsules on images with different dynamics, shapes, backgrounds and textures.

**Geo.** For this synthetic dataset, we use the same code and setup as (Xu et al., 2019), generating 100k images for training, 1k for validation, and 10k for testing. Images have different background colors, with geometrical shapes (circle, triangle, square) of various colors, scales and positions. Objects in Geo undergo translation from frame to frame.

**Geo$^+$.** This variant of Geo incorporates natural image backgrounds (random images from ImageNet (Deng et al., 2009)), and textured foreground shapes. Textures are random samples from the Brodatz dataset (Picard et al., 1993).

**Exercise.** This dataset contains natural images of trainers demonstrating exercises, with articulated and out of plane motion (used by Xu et al. (2019)). It has 49356 pairs of images for training, extracted from 20 exercise demo videos. The test set has 30 images, for which Xu et al. (2019) provided ground truth segmentation masks.

**Experimental setup.** Models are trained using the Adam optimizer (Kingma & Ba, 2014) with a fixed learning rate of $1e{-}4$ for 150 epochs. We use $C{=}32$ and $K{=}8$ for Geo models and $C{=}16$ and $K{=}16$ for Exercise model. Regularization constants for $\mathcal{L}_{\text{center}}$ and $\mathcal{L}_{\text{smooth}}$ are $1e{-}2$ and $1e{-}4$. To calculate the intersection-over-union (IoU) performance measure on visibility masks, we normalize and then threshold the masks at $0.5$ to get a binary mask.

### 5.1. Estimated Part Motion (Figure 6 and Figure 7)

To verify that the model estimates flow effectively in an unsupervised manner we first inspect the quality of the flow inferred by FlowCapsules after training on each dataset.

Figure 6 shows estimated flow $\Phi$ alongside the ground truth $\Phi_{gt}$ for *training* image pairs from Geo and Geo$^+$. The flow is accurate for both datasets. Comparing the warped version of the first frame $I$ (last column) with the other frame $I'$ (second column), one can appreciate some of the challenges in unsupervised flow estimation. Because our prediction

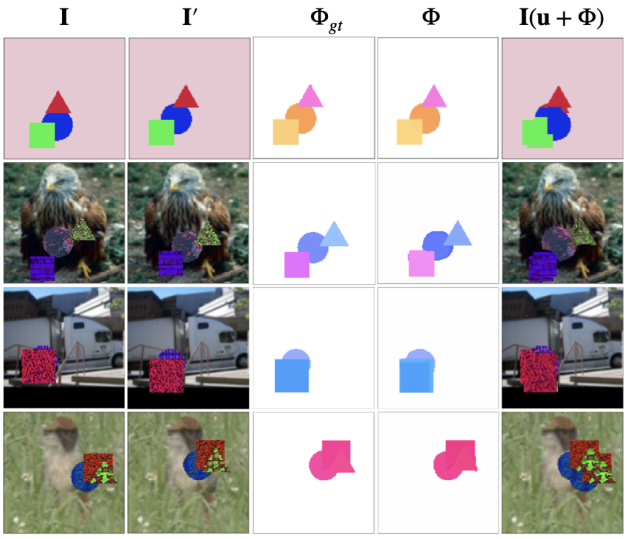


Figure 6: Estimated flows and predicted next frames on training data from Geo (first row) and Geo$^+$ (rows 2– 4).

of $\mathbf{I}'$ using $\Phi$ does not account for unoccluded pixels, $\mathcal{L}_{\text{render}}$ is not expected to reach $0$. We note that while the model uses conformal transformations from frame to frame, these datasets only have translation; for these data our model correctly estimates zero rotation and unit scale.

Figure 7 shows examples of model flow estimates for the Exercise dataset. The true flow here reflects the articulated motion of the people, and it is notable that the parts here are much smaller than those in Geo/Geo$^+$. Although the estimated flows are somewhat blurred, they still capture the movements of the different parts reasonably well, even though the model is limited to conformal deformations from one frame to the next.

### 5.2. Unsupervised Part Segmentation

One effective way to evaluate FlowCapsules is to see how well it learns to decompose a single image into its *movable* parts. We view this as an unsupervised part segmentation task and we note that, while trained on image pairs, inference is performed on a *single* test image, yielding part shapes and a coordinate transform for each part. Conversely, methods relying on optical flow only generate masks for parts *in motion*, as these models effectively build masks by *segmenting* the flow (Xu et al., 2019).

**Qualitative analysis on Geo (Figure 8).** Masks shown in Fig. 8 demonstrate that FlowCapsules learn to detect meaningful part shapes (e.g., a triangle or circle). Indeed, the model tends to explain a given image in terms of a small number of generic shapes and occlusion of overlapping parts, effectively performing *part completion* (Singh & Hoffman, 2001). This is particularly interesting because the model does not include an explicit regularizer that en-

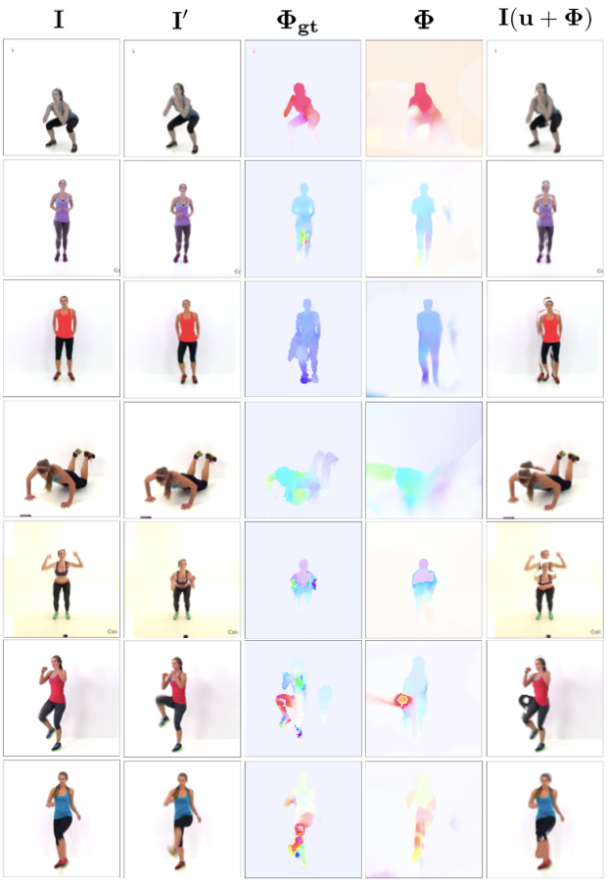


Figure 7: Estimated flows and predicted frames on randomly selected images from the `Exercise` *validation* set. Approximating articulated motion with conformal maps yields reasonable flow fields. The goal is not the best possible flow estimation, but rather, as long as different parts have different flow estimates, our encoder is be able to learn the correct part decomposition.

courages the model to learn a specific number of shapes, or sparsity in the space of shapes. One might not expect the model to learn to represent the entire shapes (e.g. an entire circle). For example, one might have expected the model to have learned a large number of different shapes from which the observed shapes are constructed, especially with occlusion where the entire shape is often not observed in a single image. Nevertheless, the model opts to explain the images with relatively few parts, and hence the capsule masks tend to cover all the pixels of a shape in the Geo dataset. This can be attributed to the architecture we use for mask decoders, and the inductive bias of MLPs in generating low-frequency functions (Tancik et al., 2020; Atzmon & Lipman, 2020; Basri et al., 2020; Rahaman et al., 2019).

Geo is synthetic, so correct masks for the full shapes are known. Since FlowCapsules provide both the part shapes, via $\Lambda_k$, and the associated visibility masks $\Lambda_k^+$ taking occlusion into account, we can compare $\Lambda_k$ to the full ground truth shapes. One can then quantify performance using the usual intersection over union (IoU) measure. FlowCapsules achieves segments with an IoU of **0.96** on all the shapes, circle, square, and triangle (see Table 1). This result indicates how well the model encodes the full shape, effectively filling in occluded portions of shapes in test images.

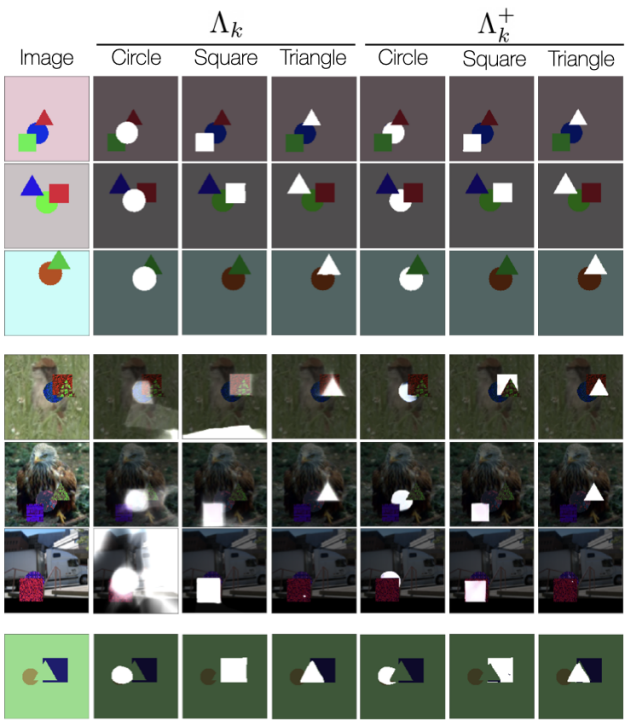


Figure 8: Inferred FlowCapsule shapes and corresponding visibility masks on Geo (rows 1–3), and Geo$^+$ (rows 4–6). The third row for each dataset shows an instance with only two objects present, so one mask is empty. The last row shows an interesting case in which the triangle is detected by the encoder even though it shares the color of the background, reminiscent of subjective contours (Kanizsa, 1976).

**Qualitative analysis on `Exercise` (Figure 11).** On the Exercise dataset, FlowCapsules learn to segment the body into *roughly* rigid parts. Fig. 11 illustrates the segmentation masks of some of the part capsules. The masks for individual capsules consistently capture the pixels associated with meaningful body parts, such as the head or right leg, regardless of the input image. As such, capsule identities are tied to semantic parts rather than spatial position. We also note that the capsules tend to delimit parts at joints, and separate the hips (lower torso) from the legs and from the upper torso, even though we do not use a kinematic prior.

**SCAE (Figure 10).** The most relevant prior work to FlowCapsules vis-a-vis part discovery is SCAE (Kosiorek et al., 2019). Figure 10 shows part templates and image recon-


Figure 9: The ground truth segment masks along with sample FlowCapsule masks $\Lambda_k^+$ on Exercise test data.

| | | R-NEM | PSD | Flow Capsules |
|---|---|---|---|---|
| Geo | Circle | 0.54 | 0.93 | **0.94** |
| | Square | 0.56 | 0.82 | **0.98** |
| | Triangle | 0.58 | 0.90 | **0.98** |
| | All | 0.56 | 0.88 | **0.95** |
| Exercise | Torso | 0.32 | 0.57 | **0.62** |
| | Left Leg | 0.29 | 0.37 | **0.59** |
| | Right Leg | 0.23 | 0.34 | **0.54** |
| | All | 0.28 | 0.43 | **0.58** |

Table 1: **Quantitative / Segmentation –** IoU of inferred segment masks w.r.t ground truth on Geo and Exercise data.

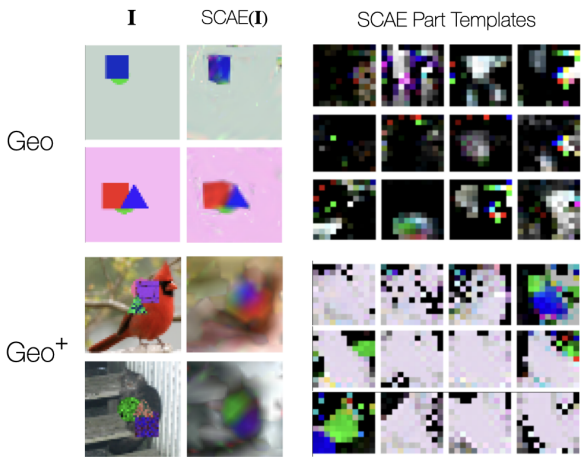


Figure 10: (left) SCAE reconstructions after training on Geo and Geo$^+$. (right) The learned part templates. SCAE approximately reconstructs the image but the part templates are not coherent parts. Comparing Geo$^+$ and Geo, the learned parts loose all shape information to enable reconstructing the color, texture and background in the images.

structions generated by SCAE. Even in simple cases without backgrounds or texture, SCAE fails to segment images into meaningful parts, unlike FlowCapsules, Fig. 8. This failure becomes markedly worse for Geo$^+$ when object textures and background are added. FlowCapsules are able to detect and focus on foreground objects with coherent part masks. But SCAE has to reconstruct the background, so the part shapes become blobs.

**PSD and R-NEM (Table 1).** We compare the IoU of our masks against PSD and R-NEM (Van Steenkiste et al., 2018). Although PSD additionally receives the *ground truth flow* during training, FlowCapsules consistently outperforms with equal or better IoUs during testing, on both the Geo and Exercise datasets (see Tab. 1). One difference between PSD and FlowCapsules stems from the way they generate shape masks. PSD generates segmentation masks directly using convolutional layers with no encoding of the shape per se. In contrast, FlowCapsules uses a low-dimensional shape code to explicitly model the shape, from which the decoder generates the mask. As such the FlowCapsules encoder disentangles meaningful shape and pose information.

On Geo$^+$, FlowCapsule IoU performance degrades approximately 10% to **0.85** (circle), **0.93** (square), **0.90** (triangle) and overall to **0.89**. But compared to results in Table 1, they remains as good or better than PSD on the simpler Geo data; we were not able to train PSD effectively on Geo$^+$.

### 5.3. Unsupervised Classification (Table 2)

To evaluate FlowCapsules in the broader context of capsule classification, we replace the *primary* capsule autoencoder (bottom of the stack) in SCAE (Kosiorek et al., 2019) with FlowCapsules. We call the new model *FlowSCAE*. We then train the top SCAE *object* capsules to reconstruct the pose of FlowCapsules, following the original SCAE paper. We

| | Geo | | Geo+ | |
|---|---|---|---|---|
| | N=4 | N=100 | N=4 | N=100 |
| SCAE | 0.48 | 0.59 | 0.49 | 0.51 |
| FlowCapsule | **0.79** | **0.99** | **0.52** | **0.74** |

Table 2: **Quantitative / Classification:** K-means clustering accuracy with 4 or 100 clusters for Geo and Geo$^+$. FlowCapsule part representations yields higher classification accuracy than those learned from SCAE.

| K | $\|s_k\|$ | Geo | Geo$^+$ |
|---|---|---|---|
| 4 | 11 | 0.94 | 0.77 |
| 8 | 11 | 0.93 | 0.83 |
| 16 | 11 | 0.94 | 0.88 |
| 8 | 3 | 0.91 | 0.86 |
| 8 | 27 | 0.96 | 0.89 |

| Depth | Decoder | Geo |
|---|---|---|
| No | 6-Layer | 0.54 |
| Yes | 2-Layer | 0.87 |
| Yes | 6-Layer | 0.96 |

Table 3: **IoU on Geo and Geo$^+$** for different number of capsules, encoding lengths, decoder depths, and depth ordering.

compare the results against SCAE trained on reconstructing images from Geo and Geo$^+$. SCAE training was modified slightly to produce coloured templates for the GEO dataset, and to produce textured templates in the primary capsules for Geo$^+$ (see supplementary material for details).

Table 2 reports unsupervised classification results using k-means clustering with $N$ clusters, for which the predicted label is set to the most common label in a given cluster. We report the accuracy with $N$=4 and $N$=100 clusters. Note that even though we trained the K-means of FlowSCAE with $N$=100 on the Geo data, the learnt representations contained only 28 clusters.

### 5.4. Ablation Studies

To better understand and analyze the significance of our design elements we perform ablations on various parameters.

**Number of capsules ($K$).** Results in Tab. 3 show that increasing the number of capsules tends to improve IoU performance. Given that our model *does not* have an explicit sparsity regularizer on the capsules, this result is intriguing. Even with large numbers of capsules available, FlowCapsules does not break shapes into smaller pieces. Rather, it learns one capsule per shape, relying more heavily on the layer occlusion to explain observed shape variation.

**Encoding length $|s_k|$.** The models are quite robust with Geo and Geo$^+$ data. As the encoding dimension decreases from 27 to 11, IofU performance changes by only 2%. Degradation occurs mainly with the circle class, where the circle boundary appears locally linear in places. The degradation becomes worse with $|s_k| = 3$, although even then, FlowCapsules still outperforms PSD.

**Number of hidden layers in $\mathcal{D}_\omega$.** One can hypothesize that deeper decoders can offset issues due to shorter shape encodings. Table 3 shows that increasing decoder depth from 2 to 6 improves IoU scores. With Geo, the deeper decoder produces smoother circles.

**Occlusion inductive bias.** Finally, we consider the effect of depth ordering in Eq. (4) for occlusion handling. Without depth ordering, Tab. 3 shows a significant drop in performance. In this case the masks become smoother and less certain in local regions, and the flow fields appear to be the result of mixing a larger number of capsules, which tend to fit the observations less well in most cases.

## 6. Conclusion

We introduce FlowCapsules, an unsupervised method for learning capsule part representations (i.e., primary capsules). The capsule encoder takes as input a single frame and estimates a set of primary capsules, each comprising a shape mask in canonical coordinates, a pose transformation from canonical to image coordinates, and a scalar representing relative depth. Training is done in a self-supervised manner from consecutive video frames. We use a Siamese architecture to estimate a parametric optical flow field between two frames, for which the flow is parameterized in terms of the poses of corresponding part capsules in the two frames. Given a single frame, our capsule encoder learns to detect and encode the movable parts in an image. This approach differs significantly from other approaches that essentially segment the flow field itself into *moving* parts (vs. *movable* parts in FlowCapsules).

Empirical results show that motion self-supervision in FlowCapsules is effective on real and synthetic data, learning meaningful representations, completing shapes when partially occluded. While formulated and tested within a specific capsule framework, our approach to self-supervised parts discovery is applicable to myriad encoder architectures, and to other approaches that currently use an image-reconstruction loss or rely on optical flow as input. Combining motion-based self-supervision with attention-based encoders (Locatello et al., 2020) would enhance compositionality, allowing scenes with different numbers of objects. Future work will also include scaling to larger video datasets and 3D parts. To that end it will be important to extend the approach to include camera motion, and to handle large motions of small objects for which more sophisticated losses for self-supervised learning will be necessary. Alternatively, the FlowCapsules framework should be directly applicable to 3D observations, like point cloud data (Zhao et al., 2019).

## Acknowledgements

We thank Luca Prasso and Deqing Sun for help preparing datasets, Dirk Weissenborn and Jakob Uszkoreit for helpful discussions in the initial stages of the project. Also Zhenjia Xu for help with the PSD experiment setup.

## References

Ahmed, K. and Torresani, L. Star-caps: Capsule networks with straight-through attentive routing. In *NeurIPS*, pp. 9098–9107, 2019.

Atzmon, M. and Lipman, Y. Sal: Sign agnostic learning of shapes from raw data. In *IEEE CVPR*, pp. 2565–2574, 2020.

Basri, R., Galun, M., Geifman, A., Jacobs, D., Kasten, Y., and Kritchman, S. Frequency bias in neural networks for input of non-uniform density. In *ICML*, pp. 685–694, 2020.

Bear, D., Fan, C., Mrowca, D., Li, Y., Alter, S., Nayebi, A., Schwartz, J., Fei-Fei, L. F., Wu, J., Tenenbaum, J., and Yamins, D. L. Learning physical graph representations from visual scenes. *NeurIPS*, 2020.

Burgess, C. P., Matthey, L., Watters, N., Kabra, R., Higgins, I., Botvinick, M., and Lerchner, A. Monet: Unsupervised scene decomposition and representation. *CoRR*, 2019.

Byravan, A. and Fox, D. Se3-nets: Learning rigid body motion using deep neural networks. *IEEE ICRA*, 2017.

Chen, Z. and Zhang, H. Learning Implicit Fields for Generative Shape Modeling. In *CVPR*, 2019.

Deng, F., Zhi, Z., Lee, D., and Ahn, S. Generative scene graph networks. In *ICLR*, 2021.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *IEEE CVPR*, pp. 248–255, 2009.

Duarte, K., Rawat, Y., and Shah, M. Videocapsulenet: A simplified network for action detection. In *NeurIPS*, 2018.

Gadelha, M., Wang, R., and Maji, S. Shape reconstruction using differentiable projections and deep priors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22–30, 2019.

Greff, K., Kaufmann, R. L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., and Lerchner, A. Multi-object representation learning with iterative variational inference. *CoRR*, 2019.

Hahn, T., Pyeon, M., and Kim, G. Self-routing capsule networks. In *NeurIPS*, 2019.

Hinton, G. E., Krizhevsky, A., and Wang, S. D. Transforming auto-encoders. In *International Conference on Artifical Neural Networks*, 2011.

Hinton, G. E., Sabour, S., and Frosst, N. Matrix capsules with EM routing. In *ICLR*, 2018.

Huang, J. and Murphy, K. Efficient inference in occlusion-aware generative models of images. *ICLR Workshop (arXiv:1511.06362)*, 2016.

Jason, J., Harley, A., and Derpanis, K. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothnes. *ECCV*, pp. 3–10, 2016.

Jepson, A., Fleet, D. J., and Black, M. J. A layered motion representation with occlusion and compact spatial support. *ECCV*, pp. 692–706, 2002.

Jojic, N. and Frey, B. Learning flexible sprites in video layers. *CVPR*, 2001.

Kanizsa, G. Subjective contours. *Scientific American*, 234 (4):48–53, 1976.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. *CoRR*, 2014.

Klambauer, G., Unterthiner, T., Mayr, A., and Hochreiter, S. Self-normalizing neural networks. *arXiv preprint arXiv:1706.02515*, 2017.

Kosiorek, A., Sabour, S., Teh, Y. W., and Hinton, G. E. Stacked capsule autoencoders. *NeurIPS*, pp. 15486–15496, 2019.

LaLonde, R. and Bagci, U. Capsules for object segmentation. *arXiv preprint arXiv:1804.04241*, 2018.

Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. Object-centric learning with slot attention. *arXiv preprint arXiv:2006.15055*, 2020.

Mahendran, A., Thewlis, J., and Vedaldi, A. Self-supervised segmentation by grouping optical-flow. *ECCV Workshop*, 2018.

Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., and Geiger, A. Occupancy networks: Learning 3d reconstruction in function space. *CVPR*, 2019.

Picard, R. W., Kabir, T., and Liu, F. Real-time recognition with the entire brodatz texture database. In *CVPR*, 1993.

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., and Courville, A. On the spectral bias of neural networks. In *ICML*, pp. 5301–5310. PMLR, 2019.

Rawlinson, D., Ahmed, A., and Kowadlo, G. Sparse unsupervised capsules generalize better. *CoRR*, 2018.

Sabour, S., Frosst, N., and Hinton, G. E. Dynamic routing between capsules. In *NeurIPS*, 2017.

Singh, M. and Hoffman, D. Part-based representations of visual shape and implications for visual cognition. In Kellman, P. and Shipley, T. (eds.), *From fragments to objects: Segmentation and grouping in vision*, chapter 9, pp. 401–459. Elsevier Science, 2001.

Skafte, N. and Hauberg, S. Explicit disentanglement of appearance and perspective in generative models. *NeurIPS*, pp. 1018–1028, 2019.

Spelke, E. S. Principles of object perception. *Cognitive science*, 14(1):29–56, 1990.

Srivastava, N., Goh, H., and Salakhutdinov, R. Geometric capsule autoencoders for 3d point clouds. *arXiv preprint arXiv:1912.03310*, 2019.

Sun, W., Tagliasacchi, A., Deng, B., Sabour, S., Yazdani, S., Hinton, G. E., and Yi, K. M. Canonical capsules: Unsupervised capsules in canonical pose. *arXiv preprint*, 2020.

Suwajanakorn, S., Snavely, N., Tompson, J., and Norouzi, M. Discovery of latent 3d keypoints via end-to-end geometric reasoning. *NeurIPS*, 2019.

Tancik, M., Srinivasan, P. P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J. T., and Ng, R. Fourier features let networks learn high frequency functions in low dimensional domains. *arXiv preprint arXiv:2006.10739*, 2020.

Van Steenkiste, S., Chang, M., Greff, K., and Schmidhuber, J. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. *arXiv preprint arXiv:1802.10353*, 2018.

Veerapaneni, R., Co-Reyes, J. D., Chang, M., Janner, M., Finn, C., Wu, J., Tenenbaum, J., and Levine, S. Entity abstraction in visual model-based reinforcement learning. In *Conference on Robot Learning*, pp. 1439–1456. PMLR, 2020.

Vijayanarasimhan, S., Ricco, S., Schmid, C., Sukthankar, R., and Fragkiadaki, K. Sfm-net: Learning of structure and motion from video. *arXiv preprint arXiv:1704.07804*, 2017.

Wagemans, J., Elder, J. H., Kubovy, M., E., P. S., A., P. M., M., S., and von der Heydt, R. A century of gestalt psychology in visual perception: I. perceptual grouping and figure–ground organization. *Psychological Bulletin*, 138 (6):1172–1217, 2012.

Wang, J. and Adelson, E. H. Representing moving images with layers. *IEEE Trans Image Processing*, 3(5):625–638, 1994.

Weiss, Y. Smoothness in layers: Motion segmentation using nonparametric mixture estima-tion. *IEEE CVPR*, 1997.

Xu, Z., Liu, Z., Sun, C., Murphy, K., Freeman, W. T., Tenenbaum, J. B., and Wu, J. Unsupervised discovery of parts, structure, and dynamics. In *ICLR*, 2019.

Zhao, Y., Birdal, T., Deng, H., and Tombari, F. 3d point capsule networks. *IEEE CVPR*, pp. 1009–1018, 2019.

# 7. Supplemantary Material

## 7.1. SCAE Training Details

While comparing FlowCapsules against SCAE, we updated SCAE training at various spots to make it more suitable for Geo and Geo$^+$ datasets. Here we detail these changes. First, We resized input images to $48 \times 48$ for memory reasons. Second, we added the option of inferring the background color as well as background image using a two level MLP. Similarly, we added the option of adding color or texture to each template. To enable colorization and texturization based on input image, the primary capsule features are passed to the template decoder. The color/texture is generated by a 2 layer MLP (32 dimensional hidden representation). The original fixed templates are used as masks and multiplied to the output of the color/texture MLP.

For generating a background template, we use the second to last hidden representation of the primary encoder as the image embedding. We pass the image embedding through a 2 layer MLP (32 dimensional hidden representation). We mix this background template with a presence probability of $0.5$.

All the other parameters, including training schedule is kept the same as the original SCAE.

## 7.2. Exercise masks

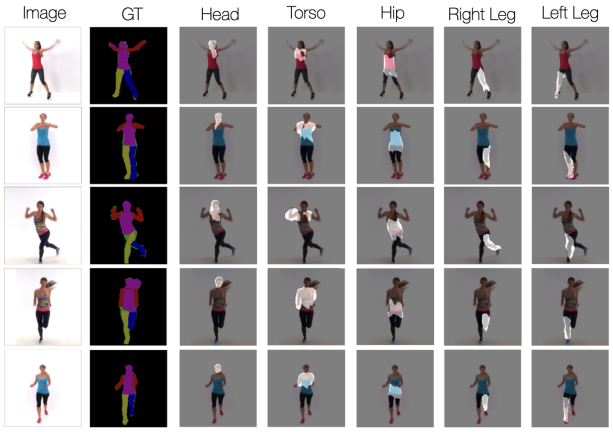


Figure 11: The ground truth segment masks along with sample FlowCapsule masks $\Lambda_k^+$ on Exercise test data.