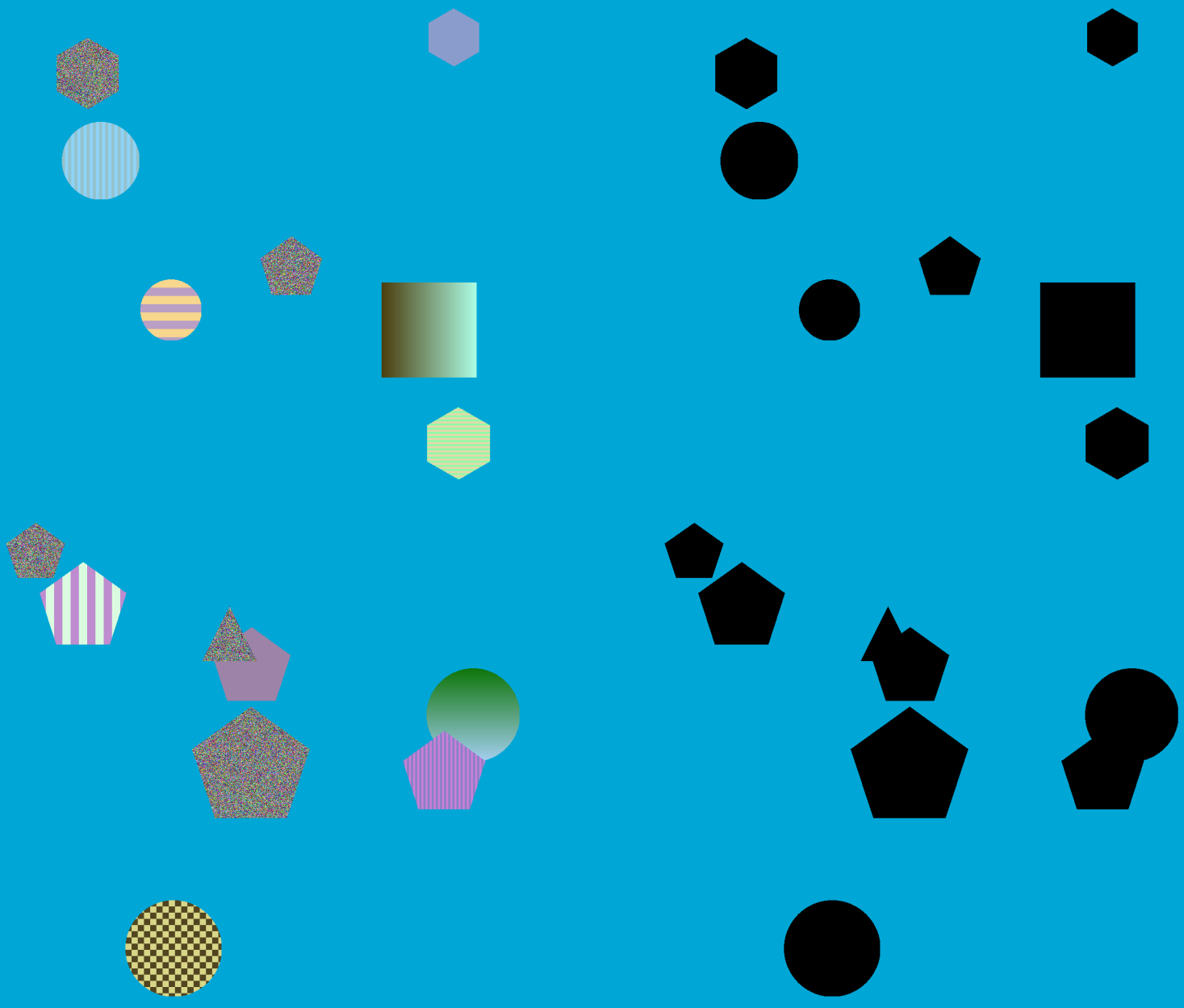
MoSIS: End-to-End Self-Supervised Instance Segmentation from Video



TU Delft

Nick Dubbeldam
October 23, 2025

MoSIS: End-to-End Self-Supervised Instance Segmentation from Video

by

Nick Dubbeldam

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday, October 30, 2025 at 14:00.

Student number:	5647703	
Project duration:	January 1, 2025 – October 23, 2025	
Academic supervisor:	Prof. Dr. D.M. Gavrila,	TU Delft
Local supervisor:	Ir. T. Lentsch,	TU Delft

TU Delft

MoSIS: End-to-End Self-Supervised Instance Segmentation from Video

Nick Dubbeldam
Department of Cognitive Robotics
Delft University of Technology

Abstract

*We present **MoSIS** (**Mo**tion-**S**upervised **I**nstance **S**egmentation), a self-supervised framework that learns instance masks from unlabelled video. Our method uses the movement in videos to define masks, which are then used to train a YOLO model to segment these instances. This makes the method end-to-end with minimal assumptions. Unlike methods that require motion at inference, MoSIS performs single-image instance segmentation, so it can detect objects that are currently static (e.g., vehicles waiting at a red light). To systematically evaluate our approach, we developed a controllable synthetic dataset with ground truth masks and motion fields. Overall, MoSIS shows that it can train an instance-segmentation model from unlabelled video while requiring only a single RGB frame at inference. While supervised training still attains higher BBox mAP, our label-free approach creates usable instance masks and points to a practical route for reducing annotation cost in perception systems.*

1. Introduction

The detection of separate objects in images (instance segmentation) is crucial for self-driving cars. This can be done with machine learning. This method requires a lot of labels, but we can train the machine by using the motion that is already provided in videos. Liu *et al*. [16] can already segment an object in a video in a self-supervised manner. This method does semantic segmentation, which makes it unsuitable for images with multiple moving objects. To make this work, we need instance segmentation. This means that every individual object gets its own detection. This is why we introduce our method **Mo**tion-**S**upervised **I**nstance **S**egmentation (**MoSIS**). This method uses two models. The first model is an appearance network which predicts instance masks. The second model is a motion network which estimates optical flow on image pairs. By combining the masks with the expected flow, we can warp the first image into a reconstruction of the second image. We supervise this reconstruction by comparing it directly with the original second image. This reconstruction error is then used to train both models. This creates an end-to-end self-supervised instance segmentation from video motion, which is fully trained from scratch. This makes the method a good option for diverse datasets, as it does not require handcrafted features. This high-level architecture is shown in Figure 1. We outline the broader societal relevance in section 7. Our main contributions are summarised below.

Contributions

1. **MoSIS**: An upgraded version of the AMD model from Liu *et al*. [16]. To go from semantic to instance segmentation. This makes MoSIS an end-to-end, self-supervised training method for instance segmentation from video data. We use segmentation and mask flow to reconstruct the next frame. We then train the model on the photometric loss between the reconstructed and original next frame. Segmentation is performed without motion at inference. This enables it to detect movable objects that stand still.
2. **Synthetic dataset**: A controllable video generator that returns images, instance masks, bounding boxes, and dense optical flow.

2. Related work

How can a model learn what an object is, without ever being shown one? Here are some methods that use object masks from motion cues, appearance regularities and self-supervision in video and images.

2.1. Motion-Guided Objectness from Video

What moves together belongs together [21] is the core idea in this chapter. If we know the movement of the pixels, we can create a mask of the object.

Yang *et al*. [30] introduces a semantic segmentation model that uses unlabelled video to learn object masks and optical flow jointly. It achieves this by utilising two distinct models. The segmentation model masks moving objects by examining the first frame. The motion model receives two consecutive frames to determine the motion. With this, the

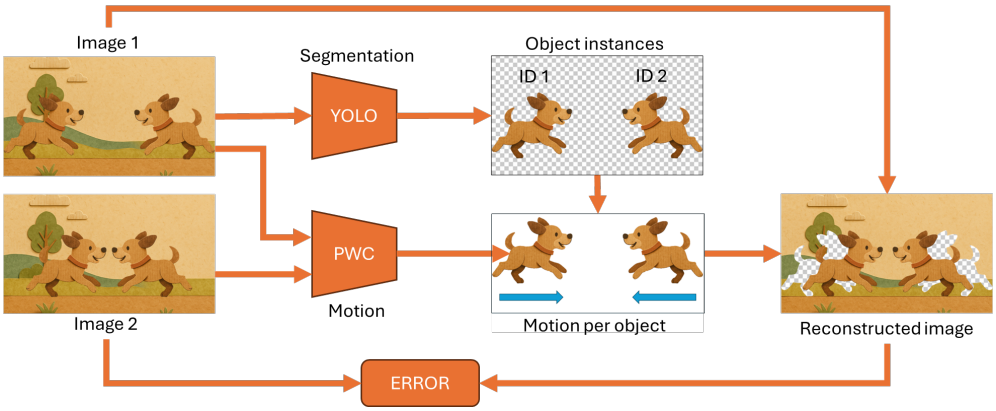


Figure 1. **MoSIS high level architecture**: During training two images are provided. YOLO predicts the segmentation, and PWC predicts the motion. By using this information, we can warp image 1 into a reconstruction of image 2. The error between image 2 and the reconstruction image will be the error to train both models. This idea is inspired from AMD [16][images generated with Sora [18]]

masked-out object is moved to the new position, recreating the second frame. The loss is computed by comparing the results of the reconstructed frame with the original second frame. Without any ground-truth masks, the appearance branch naturally discovers objectness, achieving strong zero-shot performance on video object segmentation benchmarks. This method detects objects only when they move, as it uses motion flow as part of its inference.

This is where **Liu *et al.*** [16] improves it with their AMD model. This method uses the motion network for training, but performs inference solely on the segmentation model. This way, it will also detect objects that can move when they are standing still. These methods only work when there is only one object in the training scene, as this is a semantic segmentation network. When more people are in the scene, it will give them all the same motion while moving in opposite directions, which will break the reconstruction. Motion-inductive Self-supervised Object Discovery in Videos [5] also uses the same idea. However, it is again semantic segmentation rather than instance segmentation.

Our work takes this idea further by upgrading the appearance side to *instance* predictions and explicitly coupling mask-wise motion to avoid such averaging.

2.2. Camera-based Segmentation

It is also possible to recognise objects without the use of motion. The most common way is to do supervised learning. **Mask DINO** [14] is a transformer-based extension of the DINO detector [32], which is a DETR variant [2]. DETR is a convolutional backbone with an encoder-decoder transformer, without the use of region proposals, non-maximum suppression (NMS) or anchor generation. DINO keeps DETR object queries and adds a small mask head. This design handles instance, panoptic, and semantic segmentation in one model and trains end-to-end. It is widely used as a supervised base.

It is also possible to recognise objects without labels. **TokenCut** [27] shows that modern self-supervised Vision Transformer (ViT) features with normalised cuts are enough to recognise objects. **CutLER** [25] makes it unsupervised with instance segmentation by adding a detector on pseudo masks and uses self-training. Another image-only method is **FreeSolo** [24], which learns class-agnostic instance segmentation without any annotations by building on SOLO with localisation-aware pretraining that discovers objects from appearance alone. **VideoCutLer** [26] shows that high-quality pseudo-masks and simple video synthesis are enough to train an unsupervised video instance segmentation that tracks multiple objects during the scene, without the use of motion supervision. Most recently **Hoang *et al.*** [9] uses the same image-only method but replaces the initial cut with a MultiCut over SSL features. In parallel, **STEGO** [8] (Self-supervised Transformer with Energy-based Graph Optimisation) focuses on unsupervised semantic segmentation. It shows that appearance correspondences alone can organise scenes without labels or motion.

These methods are good at segmenting and work on a single frame during inference. But they do not learn from actual scene dynamics.

A method that uses the flow of the objects is **SAVi++** [6]. An attention video model trained to predict depth. It creates per-object masks and tracks in real-world driving videos without mask labels. In contrast, our method uses photometric reconstruction only during training and performs single-image instance segmentation at inference, without the flow or depth information.

MoSIS uses the best properties from both sides while avoiding key compromises. MoSIS performs single-image inference and produces instance masks just like CutLER [26], FreeSOLO [24] and TokenCut [27]. But unlike

these methods, it is trained on real video dynamics. Meaning it can detect which regions move together even when they are currently static. Compared to AMD [16] and Motion grouping [30], MoSIS is an upgrade from semantic to instance segmentation. This prevents flow averaging across multiple objects. This makes the method work for various objects in the scene. The same is true for SAVI++ [6], which also does semantic segmentation. Unsupervised Instance Segmentation with Superpixels [9] uses only appearance (colour/texture/edges) for grouping. While MoSIS also uses motion, this motion would be better for adjacent objects with similar textures during training.

2.3. LiDAR-based Segmentation

It is also possible to discover objects without labels directly in 3D LiDAR with the use of geometry and temporal cues instead of manual annotations.

OYSTER [33] is an unsupervised 3D detector that learns to discover objects directly in unlabelled LiDAR point clouds. It clusters dense, close-range points with DBSCAN [7] to generate pseudo-boxes. It then makes the close-range objects sparser with the pseudo labels to learn to recognise objects far away. From here, it self-trains with consistency checks to suppress noise and improve detection quality. At inference, only the final 3D detector is used for zero-shot object detection in new scenes. **UNION** [13] fuses self-supervised scene flow with appearance-based clustering (from cameras) to obtain static and dynamic proposals and pseudo-classes, training a 3D detector in a single round. **SSF-PAN** [3] is another self-supervised method. This method also consists of a segmentation model with a scene flow model. But the difference with this model is that during inference, the model needs two consecutive frames instead of one. This is necessary because the scene flow network is not only used for training but also for inference. **CutS3D** [22] builds upon Cutler [25] and uses the depth information out of point clouds to go from semantic segmentation to instance segmentation.

MoSIS differs in 3 aspects from these methods. First, in the input and output. It uses only RGB images and predicts 2D instance masks, where LiDAR-centric methods output 3D bounding boxes [13, 33] or 3D masks [3]. CutS3D [22] uses LiDAR information instead of video to differentiate between instances. Secondly, MoSIS learns end-to-end from unlabelled video via photometric reconstruction with motion as a training signal. LiDAR work relies on geometric clustering or self-training [33], scene-flow with appearance pseudo classes [13]. Third, at test time, MoSIS only needs one RGB image as it does not use the motion flow. This is unlike SSF-PAN [3]. This results in being able to detect objects that are currently static.

2.4. Summary.

MoSIS combines the best of both worlds. Video-based end-to-end self-supervised training and single-frame instance segmentation during inference. All this without the help of depth information or motion at test time. This makes the solution work for videos with multiple moving objects, where some objects can remain stationary. To our knowledge, this is the first end-to-end self-supervised approach that learns instance segmentation from video motion, while only requiring one RGB image at test time. Table 1 summarises sensors, training signals, test inputs and outputs across methods.

3. Methodology

This section introduces our method, MoSIS. The goal is to learn instance segmentation without any manual mask annotations. This method uses motion cues in videos. We first formalise the task (subsection 3.1), then describe MoSIS (subsection 3.2) with its segmentation (subsection 3.3) and flow models (subsection 3.4), our segment-flow fusion (subsection 3.5), reconstruction (subsection 3.6), and losses (subsection 3.7), inference (subsection 3.8), and the synthetic dataset (subsection 3.9).

3.1. Self-supervised instance segmentation

Instance segmentation estimates the mask of each individual object in an image at the pixel level. Having no labels in the input and instead using motion cues between images makes it a self-supervised setting.

Input. An unlabelled RGB video $I_{1:N}$ of duration T at frame rate F is needed as the input. This video consists of the frames: $\{I_t\}_{t=1}^{N}$ where $N = T \cdot F$. With each frame set annotated as: $I_t, I_{t+1} \in \mathbb{R}^{H \times W \times 3}$

Output. The output consists of a set of classless detections which contain 2D bounding boxes and masks:

$$\mathcal{D}_t = \left\{(x_i, y_i, w_i, h_i, s_i, M_{t,i})\right\}_{i=1}^{K_t} \tag{1}$$

Where $(x_i \in [0, W], y_i \in [0, H])$ is the box centre position in pixels and (w_i, h_i) is the box width and height in pixels. $s_i \in [0, 1]$ is the objectness score and $M_{t,i} \in [0, 1]^{H \times W}$ an instance mask. No classes are predicted.

3.2. MoSIS

MoSIS couples an appearance head (YOLO-style) that predicts boxes and soft instance masks with a motion head (PWC-Lite-style) that estimates dense per-pixel motion. We pool V_t under each mask to get a per-segment feature, map it to a 2-D flow vector, reproject to a dense segment-flow, and warp I_t to reconstruct I_{t+1}. Both models are trained

Method	Sensor	Train signal	Test input	Output
MoSIS (ours)	RGB	Photometric reconstruction (video)	1 frame	**2D masks (Instance)**
AMD [16]	RGB	Photometric reconstruction (video)	1 frame	2D masks (Semantic)
Motion Grouping [30]	RGB	Self-sup on optical flow	Video/flow	2D moving masks (Semantic)
Mask DINO [14]	RGB	Supervised (det.+mask)	1 frame	2D masks (Instance, classed)
TokenCut [27]	RGB	SSL ViT feats + norm. cut	1 frame	2D mask (Semantic/salient)
CutLER [25]	RGB	Pseudo-masks + self-train	1 frame	Boxes & 2D masks (Instance, class-agn.)
OYSTER [33]	LiDAR	Clustering + temp. self-train	1 sweep	3D boxes (Instance)
UNION [13]	LiDAR+Cam	Scene flow + app. pseudo-classes	1 sweep	3D boxes (Instance)
SSF-PAN [3]	LiDAR	Self-sup scene flow + motion seg.	2 sweeps	3D moving masks (Semantic)
SAVi++ [6]	RGB+Depth	Depth-pred. target (video)	Video	2D masks & tracks (Instance)

Table 1. Comparison of methods by sensor, training, test input and output.

from photometric losses. In inference, only the appearance model gets used to produce instance masks from a single image. A simplified overview can be found in Figure 1. Look at section 8 for a detailed overview.

Assumption. We assume that the consecutive frame pairs (I_t, I_{t+1}) have a small movement between frames. In practice, we target a video rate of ≥ 12 Hz to ensure stable photometric warping. (see experiment in section 9.1).

Method. MoSIS consists of 2 models. The first model is a segmentation model that predicts bounding boxes in every frame. The second model is the flow model. This model predicts the motions between images.

Training. The model uses a segmentation model that creates masks for each frame, and a flow model that generates the motion between each frame. By moving the masks in the direction of the motion, we can create a new image. When done correctly, this image should look like the next frame in the video. This reconstructed frame can be compared with the ground-truth frame to construct a loss and adjust both models.

Inference. Only the segmentation model is used during inference: it produces instance masks in a single forward pass, without any flow computation.

3.3. Segmentation model

The segmentation model is a YOLOv3 [20] model with random weights. Each image I_t is passed through the network to create detections K.

$$\begin{aligned} \mathcal{SM}_t &= f_{\text{yolo}}(I_t), \\ \mathcal{SM}_t &= \left\{(s_i, x_i, y_i, w_i, h_i)\right\}_{i=1}^{K}. \end{aligned} \tag{2}$$

where (x_i, y_i) is the box centre (in pixels), (w_i, h_i) its width/height, and $s_i \in [0, 1]$ an objectness score.

YOLOv3 detections provide a set of bounding boxes. In this method we need masks, so we need to convert them.

Instance mask head. Our goal is to propose instance masks. We generate the masks from the image inside the bounding box. Given K proposals $\{b_i\}_{i=1}^{K}$ from the YOLO head, we extract a shared feature map at stride 4 and apply ROIAlign to obtain 28×28 crops per box. A lightweight mask head (four 3×3 conv + ReLU blocks and a 1×1 conv) produces a single-channel logit mask $\tilde{M}_i(x, y)$ per ROI, which is upsampled and pasted back into image coordinates.

Top-K selection Let s_i be the objectness score of proposal i. We keep the top K proposals by their objectness score s_i (we use K=32) and multiply the masks by their objectness score. This makes high-confidence detections contribute more strongly. The K is chosen as 32, as there are no images with more than 32 proposals. And making it bigger will ask for more compute power, which is not needed.

Background channel. A background channel M_0 is added as the complement of all foregrounds:

$$\widetilde{M}_0(x, y) \;=\; \prod_{i=1}^{K} \big(1 - \widetilde{M}_i(x, y)\big). \tag{3}$$

Then all the masks get stacked and normalised per pixel.

$$\begin{aligned} S_k(x, y) \;&=\; \frac{\widetilde{M}_k(x, y)}{\sum_{j=0}^{K} \widetilde{M}_j(x, y) + \epsilon}, \\ \sum_{k=0}^{K} S_k(x, y) &= 1, \qquad \epsilon = 1e^{-6}. \end{aligned} \tag{4}$$

3.4. Flow model

For a frame pair (I_t, I_{t+1}), a cost-volume network (PWC-Lite, a lightweight version of PWC-Net[23]) computes dense motion features:

$$V_t = f_{\text{mot}}(I_t, I_{t+1}) \in \mathbb{R}^{d\times H\times W}. \tag{5}$$

We do not predict per-pixel flow. Instead, we will pool V_t with the instance partition and map each pooled feature to a single 2-D flow vector per segment.

3.5. Segment-Flow Fusion

For each mask channel K (with $k = 0$ the background), we pool the motion features:

$$\bar{V}_t(k) \;=\; \frac{\sum_{x,y} S_k(x,y)\, V_t(:,x,y)}{\sum_{x,y} S_k(x,y) + \epsilon} \in \mathbb{R}^d. \tag{6}$$

Then a two-layer MLP maps $\bar{V}_t(k)$ to a 2-D flow vector per mask:

$$F_t^m(k) \;=\; \mathrm{MLP}\big(\bar{V}_t(k)\big) \in \mathbb{R}^2. \tag{7}$$

Pooling Per-Segment Flow. To get the motion per mask, we compute the average motion (the mask flow) for each mask channel k (per object):

$$\widetilde{F}_t(x,y) = \sum_{k=0}^{K} S_k(x,y)\, F_t^m(k) \in \mathbb{R}^2. \tag{8}$$

Thus, the final motion field is a 2D image with a flow vector assigned to each pixel. Every soft mask from the bounding box has a uniform vector. To be able to let each detected region move according to its estimated flow. This is required for the reconstruction.

3.6. Reconstruction

Given a frame pair (I_t, I_{t+1}), we first compute the segment-flow field $\widetilde{F}_t \in \mathbb{R}^{2\times H\times W}$ as described in the previous section. This flow encodes how each pixel in frame I_t is expected to move to match frame I_{t+1}.

We then warp the current frame I_t according to $\widetilde{F}_t$:

$$\widehat{I}_{t+1} = \mathrm{warp}\big(I_t,\; \widetilde{F}_{t\to t+1}\big). \tag{9}$$

where $\mathrm{warp}()$ is differentiable bilinear sampling.

Directional reconstruction. As the training is symmetric, we compute in both directions

$$\widehat{I}_t = \mathrm{warp}\big(I_{t+1},\; \widetilde{F}_{t+1\to t}\big). \tag{10}$$

3.7. Self-Supervised Loss

Just like AMD [16] we use the SSIM loss from Wang et al. [28]. With a 3×3 box window (implemented as 3×3 average pooling). We minimise the SSIM error as:

$$\mathcal{L}_{\mathrm{photo}}(A,B) = \tfrac{1}{2HW} \sum_{x,y} (1 - \mathrm{SSIM}(A,B)). \tag{11}$$

The total photometric loss is the sum of the photometric terms in both directions:

$$\mathcal{L}_{photo2} \;=\; \mathcal{L}_{\mathrm{photo}}\big(I_t, \widehat{I}_t\big) \;+\; \mathcal{L}_{\mathrm{photo}}\big(I_{t+1}, \widehat{I}_{t+1}\big). \tag{12}$$

This loss has one flaw. It will give bounding boxes without a mask a high score as they don't contribute to wrongly placed pixels. That is why we introduce the photometric targets for objectness. Let $\mathrm{SSIM}(\widehat{I}_{t\to t+1}, I_{t+1})$ produce a per-pixel dissimilarity map $E \in [0,1]^{H\times W}$. For proposal channels (excluding background), we compute the error per mask

$$E = 1 - \mathrm{SSIM}(\widehat{I}, I), \tag{13}$$

$$\tilde{t}_i \;=\; \frac{\sum_{x,y} S_i(x,y)\, E(x,y)}{\sum_{x,y} S_i(x,y) + \epsilon}\,. \tag{14}$$

and normalize $\tilde{t}$ per image, so the targets are $t_i \in [0,1]$. Let a_i be the objectness logits aligned to proposals; we add

$$\mathcal{L}_{\mathrm{obj}} \;=\; \mathrm{BCEWithLogits}(a,\; t)\,, \tag{15}$$

$$\mathcal{L}_{\mathrm{total}} \;=\; \mathcal{L}_{\mathrm{photo}}^{t\to t+1} + \mathcal{L}_{\mathrm{photo}}^{t+1\to t} + \lambda_{\mathrm{obj}}\, \mathcal{L}_{\mathrm{obj}}\,. \tag{16}$$

with λ_{obj}=0.2. The 0.2 makes sure the object loss and the photo loss contribute equally.

3.8. Inference

At test time, we use a single image I_t on only the segmentation model. We then apply a score threshold ($s_{\min}$ is 0.4) a box NMS (IoU is 0.1) and a mask threshold ($m_{\min}$ is 0.5) The output is a set of classless boxes with associated soft masks. We keep the top 32 proposals (K=32) by objectness.

Implementation details. We train end-to-end with Adam optimisation [12] (learning rate 1×10^{-4}), batch size 16, on 128×128 images.

3.9. Synthetic Dataset

Overview. To have complete control over the training environment of the model, we built a synthetic dataset generator. It creates a background with moving objects in front of it. Where there is complete control over the textures and movements. A significant advantage is that it can return the frames with their labels (bounding boxes, masks, and dense flow), while the dataset can be as simple or complex as it needs to be. Figure 2 shows a few frames which can be generated.

Background model. The background model can be a colour or image, which can be zoomed-in and moved to follow a specific movement function. When moving to the edge, it will resample itself to avoid black borders.

Object model. Each object is a geometric shape (circle, square, triangle, pentagon, hexagon, star) with a size, texture and programmable velocity.

Scene composition. Every frame gets created on the go. By getting the right background and adding the objects on the right location. the scene size and length are configurable.

The base dataset returns

$$\left\{ \begin{array}{l} \texttt{image} \in [0,1]^{3\times H\times W}, \\ \texttt{masks} \in \{0,1\}^{D\times H\times W}, \\ \texttt{bboxes} \in \mathbb{R}^{D\times 4} \end{array} \right\}.$$

Where D is the number of objects, the boxes are in shape: (x_1, y_1, x_2, y_2).

Different datasets. There are 3 wrappers on this dataset for different purposes:

- **ImagePairDataset**: returns consecutive pairs (I_t, I_{t+1}) for self-supervised setups.
- **FlowSupervisedDataset**: returns $(I_t, I_{t+1}, F_t^\star)$ with dense optical flow. The object flows are calculated by the difference between the centre positions and pasted onto the entire mask.
- **YOLOSupervisedDataset**: returns (I_t, boxes_t) for supervised detection.

4. Experiments

The most essential experiment is the comparison between supervised YOLO, AMD and MoSIS on the synthetic dataset. This can be seen in subsection 4.1. The MoSIS results on real videos can be found at subsection 4.2. Recreation of the AMD repository can be seen in subsection 9.1. The supervised flow experiment can be found at subsection 9.2. And the supervised instance segmentation can be found at subsection 9.3.

4.1. Comparison

The objective of this experiment is to compare supervised learning with self-supervised learning. We train YOLOv3 with a mask head using both methods and compare the resulting accuracies with AMD from the inspiration paper.

Dataset. We use the synthetic dataset in this experiment. Twenty-four random videos are generated with 120 frames per scene. Where every frame is 128 by 128 pixels. The videos feature a moving background image and randomly selected shapes, including squares, triangles, circles, and stars. Where the size, speed and starting positions are randomly bound. This dataset gets split into a train, validation, and test dataset [70%, 15% and 15%].

Models and training procedures. Three models were trained and evaluated using the same data interface:

- **YOLO-supervised [20]:** A YOLOv3 architecture extended with a mask head. This model is trained fully supervised using labels.
- **AMD [16]:** Our baseline model, which is the inspiration for MoSIS. AMD is a semantic segmentation model which will be trained without labels. It does need a human step to choose the correct output channel.
- **MoSIS:** This is our model, which will be trained without labels. This is an instance segmentation model which gets trained without labels. MoSIS also uses the same YOLO for the inference, but trains YOLO differently.

Implementation details. An Adam optimiser [12] is used with a learning rate of 1e-4. It has a batch size of 16, and it took 500 epochs. The same random seed and preprocessing (resize to 128 by 128 pixels, normalisation) were used for reproducibility. On the Tesla V100-SXM2-32GB GPU, YOLO-supervised trained in 43 min and AMD in 59 min. MoSIS required around 5 hours, while the current MoSIS implementation is not yet optimised.

Results. Table 2 summarises the quantitative results. AP stands for Average Precision in COCO [15] style. mAP is the mean average precision over IoU thresholds from 0.50 to 0.95 in steps of 0.05. The AP50 shows the average precision at an IoU of exactly 0.5, and the AP75 shows the average precision with an IoU of exactly 0.75. The qualitative results are presented in Figure 2.

The AMD model has five output channels. It masks the entire frame in one channel and does nothing in the other channels; this can be seen in Figure 3.

Analysis. YOLO-supervised clearly outperforms YOLO+MoSIS and AMD across most metrics. This is as expected, as the supervised detector directly optimises for box/mask quality with ground-truth labels. By contrast, MoSIS learns objectness from photometric consistency and motion cues and therefore tends to learn more slowly.

It can be observed that when multiple objects move in different directions, the AMD model stops working. This is because the model is designed for one object. With numerous objects, it will average the speeds of the detected objects and move both objects towards the average direction, instead of the individual directions. During training, it starts with detecting parts of a shape, but over time, the model collapses and detects nothing.

Overall, the qualitative results (Figure 2) indicate that the self-supervised pipeline is learning meaningful instances. While the supervised baseline sets a higher absolute AP today, MoSIS shows clear potential, and further optimisation is likely to narrow the gap.

Table 2. Comparison of Mask and BBox AP (higher is better).

	Labels	Mask AP			BBox AP		
		AP50	AP75	mAP	AP50	AP75	mAP
YOLO-supervised	✓	**0.875**	**0.274**	**0.391**	**0.866**	**0.551**	**0.491**
AMD	X	0.000	0.000	0.000	0.000	0.000	0.000
YOLO+MoSIS	X	0.556	0.005	0.149	0.611	0.104	0.229

Figure 2. **Qualitative results on the synthetic dataset.** The first two rows show successful cases where MoSIS predicts bounding boxes with masks for the objects. The bottom row shows a failure case.

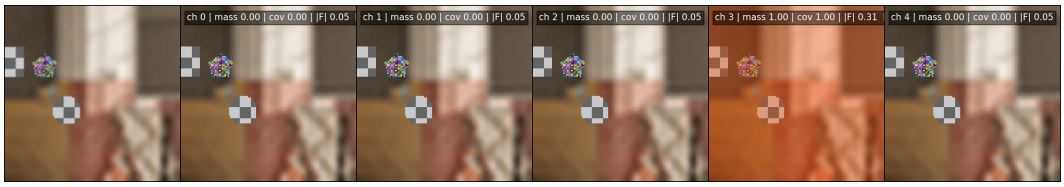


Figure 3. All the outputs from AMD. This shows that no channel shows an object. Just one channel, which marks everything as background.

4.2. Real images

Dataset. We used the YouTube-VIS 2021 dataset [31]. We took four videos (60, 63, 93 and 362). From this, we got 151 frames with 147 image pairs. These images are scaled down to 128 by 128 pixels.

Implementation details. An Adam optimiser [12] is used with a learning rate of 1e-4. It has a batch size of 16, and it took 500 epochs. The training took around 5 hours using a Tesla V100-SXM2-32GB GPU, while the current MoSIS implementation is not yet optimised.

Results. As seen in Figure 4, MoSIS detects the main moving object, the tennis player. But it splits the tennis player into multiple sections.

Analysis. It shows that the model can also detect objects in real-world images. But it sometimes splits it up into multiple smaller objects. This is the case, as some parts of the object move at another speed or deform. Adding rotation or deformation-aware motion to the model could improve its performance in such cases.

5. Potential improvements

To improve the score on the bounding box, we can post-process the size back into the outer mask size. This will directly increase the AP of the bounding boxes. Another improvement that can be made is to train for more epochs. MoSIS sometimes splits objects into multiple objects. This could be resolved by adding the right loss for it. Another improvement could be to add rotation and scale to the mask movement to make it more accurate.

The current dataset setup is created for 2D movements. Adding rotations will currently result in the wrong flow, as the flow in every part of the object is the same as the flow at the centre.

6. Conclusion

We introduced **MoSIS**, a self-supervised instance segmentation framework that learns instance masks from motion cues during training but requires only a single RGB frame at inference. Our pipeline combines a YOLO-style appearance head with a lightweight motion module and supervises objectness through photometric consistency, eliminating the need for human annotations.

On the synthetic dataset benchmark, YOLO-supervised remains the best (with a Mask mAP of 0.391 and a BBox mAP of 0.491). While the same YOLO model trained without labels by using MoSIS achieves a competitive performance with a more challenging task (with a Mask mAP of 0.149 and a BBox mAP of 0.229). The AMD baseline collapses with multiple objects in the scene (as it makes no detections, so all metrics are 0), which shows the benefit of MoSIS to produce instance masks without human annotations. But there is still some room to improve on MoSIS to make it as good as supervised methods.

On real videos (YouTube-VIS 2021 clips [31]), MoSIS detects the main moving object but can over-segment deforming parts. Adding rotation or deformation warping could improve the performance.

References

[1] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*, 2019. 1

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 2

[3] Yinqi Chen, Meiying Zhang, Qi Hao, and Guang Zhou. Ssf-pan: Semantic scene flow-based perception for autonomous navigation in traffic scenarios, 2025. 3, 4

[4] MMCV Contributors. MMCV: OpenMMLab computer vision foundation. https://github.com/open-mmlab/mmcv, 2018. 1

[5] Shuangrui Ding, Weidi Xie, Yabo Chen, Rui Qian, Xiaopeng Zhang, Hongkai Xiong, and Qi Tian. Motion-inductive self-supervised object discovery in videos. *arXiv preprint arXiv:2210.00221*, 2022. 2

[6] Gamaleldin Elsayed, Aravindh Mahendran, Sjoerd Van Steenkiste, Klaus Greff, Michael C Mozer, and Thomas Kipf. Savi++: Towards end-to-end object-centric learning from real-world videos. *Advances in Neural Information Processing Systems*, 35:28940–28954, 2022. 2, 3, 4

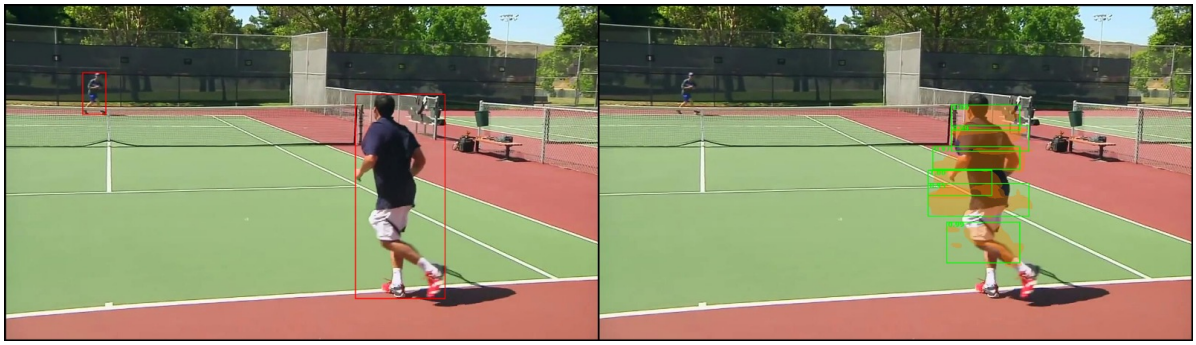


Figure 4. Results on real images. The left side shows the labelled object, and the right side shows the predicted object mask.

[7] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, pages 226–231, 1996. 3

[8] Mark Hamilton, Zhoutong Zhang, Bharath Hariharan, Noah Snavely, and William T Freeman. Unsupervised semantic segmentation by distilling feature correspondences. *arXiv preprint arXiv:2203.08414*, 2022. 2

[9] Cuong Manh Hoang. Unsupervised instance segmentation with superpixels. *arXiv preprint arXiv:2509.05352*, 2025. 2, 3

[10] Insurance Institute for Highway Safety. Front crash prevention slashes police-reported rear-end crashes. https://www.iihs.org/news/detail/front-crash-prevention-slashes-police-reported-rear-end-crashes, 2016. Accessed: 2025-08-26. 1

[11] Kili Technology. Data labeling services price [q3 2023 benchmark]. https://kili-technology.com/data-labeling/data-labeling-services-price-q3-2023-benchmark, 2023. Semantic segmentation: $0.10–$1.00 per mask. Accessed: 2025-07-24. 1

[12] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5, 6, 8, 1, 2

[13] Ted Lentsch, Holger Caesar, and Dariu M Gavrila. UNION: Unsupervised 3D object detection using object appearance-based pseudo-classes. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 3, 4

[14] Feng Li, Hao Zhang, Huaizhe Xu, Shilong Liu, Lei Zhang, Lionel M Ni, and Heung-Yeung Shum. Mask dino: Towards a unified transformer-based framework for object detection and segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3041–3050, 2023. 2, 4

[15] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Doll'a r, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 6

[16] Runtao Liu, Zhirong Wu, Stella Yu, and Stephen Lin. The emergence of objectness: Learning zero-shot segmentation from videos. *Advances in Neural Information Processing Systems*, 34:13137–13152, 2021. 1, 2, 3, 4, 5, 6

[17] Runtao Liu, Zhirong Wu, Stella Yu, and Stephen Lin. The emergence of objectness — official repository. https://github.com/rt219/The-Emergence-of-Objectness, 2022. GitHub repository, commit 33eca58. 1

[18] Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, Zhengqing Yuan, Yue Huang, Hanchi Sun, Jianfeng Gao, et al. Sora: A review on background, technology, limitations, and opportunities of large vision models. *arXiv preprint arXiv:2402.17177*, 2024. 2

[19] nuScenes Consortium. nuimages website. https://www.nuscenes.org/nuimages, 2023. Accessed: 2025-08-26. 1

[20] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 4, 6, 2

[21] Jenny Seidenschwarz, Aljosa Osep, Francesco Ferroni, Simon Lucey, and Laura Leal-Taixe. Semoli: What moves together belongs together. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14685–14694, 2024. 1

[22] Leon Sick, Dominik Engel, Sebastian Hartwig, Pedro Hermosilla, and Timo Ropinski. Cuts3d: Cutting semantics in 3d for 2d unsupervised instance segmentation. *arXiv preprint arXiv:2411.16319*, 2024. 3

[23] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018. 4

[24] Xinlong Wang, Zhiding Yu, Shalini De Mello, Jan Kautz, Anima Anandkumar, Chunhua Shen, and Jose M Alvarez. Freesolo: Learning to segment objects without annotations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14176–14186, 2022. 2

[25] Xudong Wang, Rohit Girdhar, Stella X Yu, and Ishan Misra. Cut and learn for unsupervised object detection and instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3124–3134, 2023. 2, 3, 4

[26] Xudong Wang, Ishan Misra, Ziyun Zeng, Rohit Girdhar, and Trevor Darrell. Videocutler: Surprisingly simple unsupervised video instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22755–22764, 2024. 2

[27] Yangtao Wang, Xi Shen, Yuan Yuan, Yuming Du, Mao-mao Li, and Shell Xu Hu. Tokencut: Segmenting objects in images and videos with self-supervised transformer and normalized cut. *arXiv: 2209.00383*, 2022. 2, 4

[28] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 5

[29] World Health Organization. Road traffic injuries. https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries, 2023. Accessed: 2025-08-26. 1

[30] Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. Self-supervised video object segmentation by motion grouping. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7177–7188, 2021. 1, 3, 4

[31] Linjie Yang, Yuchen Fan, Yang Fu, and Ning Xu. The 3rd large-scale video object segmentation challenge - video instance segmentation track, 2021. 8

[32] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022. 2

[33] Lunjun Zhang, Anqi Joyce Yang, Yuwen Xiong, Sergio Casas, Bin Yang, Mengye Ren, and Raquel Urtasun. Towards unsupervised object detection from lidar point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9317–9328, 2023. 3, 4

MoSIS: End-to-End Self-Supervised Instance Segmentation from Video

Supplementary Material

7. Societal Relevance

Road safety remains a big challenge: every year, 1.19 million people die as a result of road traffic crashes. This is the leading cause of death for children and young adults aged 5-29 years.[29]

Advanced driver-assistance systems (ADAS) and, ultimately, automated driving aim to reduce collisions by achieving faster response times and being able to see more objects around them. To be able to detect and respond accordingly, they must reliably perceive and understand their surroundings. For example, automatic emergency braking (AEB) is associated with roughly 50% fewer crashes.[10]

A critical aspect of perception is robust and accurate object detection systems. Upcoming systems are based on machine learning models which can do instance segmentation. Instance segmentation assigns a unique, pixel-level mask to every object instance in an image. Unlike object detection, which outputs bounding boxes, instance segmentation captures precise object shapes and handles overlapping instances, enabling finer reasoning about geometry and occlusion.

Supervised models need data, which includes those masks for training. This means that a person needs to label every pixel. This is a lot of work, which makes the data collection expensive. The price of labelling a mask can vary from $0.10 to $1.00 per mask. [11], while a dataset can contain 100k annotated masks [19].

To address these challenges, *self-supervised* machine learning models offer a promising approach for developing generalisable machine learning systems. Unlike traditional supervised learning, which relies on large amounts of manually labelled data, self-supervised learning reduces the need for expensive annotations by leveraging unlabelled data.

8. Detailed architecture

Figure 5 shows a detailed diagram of the MoSIS code.

9. Other experiments

This experiment section adds other important experiments that did not fit in the main paper. We start by recreating the original paper to find out the specifics of the method. Then we continue with training the flow network supervised to verify its design and trainability.

9.1. Recreation of AMD

The first experiment is to recreate the inspirational paper AMD (appearance–motion decomposition) [16]. As it used multiple outdated libraries (The original repo depends on MMCV v1 [4]). Which made the code hard to adapt.

Goals and scope We re-implemented the paper's code in pure PyTorch. To remove the dependency on MMCV [4]. The objective is to have the same results as the paper.

Dataset The original repository [17] has example data to check. It is a video of a moving swan.

Another dataset used is the nuScenes-mini [1] (camera only). We use consecutive front camera pairs. To match the model's limitations (section 9.1), Only scenes were used with a maximum of 1 moving object.

Implementation details

- The appearance network is a ResNet-50 backbone, initialised with random weights. It produces a specified amount of mask logits. These masks sum up to 1 by adding a softmax on it.
- The motion network is a PWC-Lite model.
- The flow readout is an MLP that maps the motion features into a 2D flow vector per segment.
- the reconstruction is done by averaging the vectors per mask and moving the masks in the direction of the vector.
- The L1 photometric loss is used between the second frame and the reconstructed frame.
- While training we used Adam optimisation [12] with a learning rate of 1e-4 with a batch size of 8. It is trained for 5,000 epochs

Results Since no ground-truth annotations were used during training, we primarily evaluate the model qualitatively. Figure 6 shows the masks following the shape of the swan. While the model has never seen any labels. Here, it can be seen that it also follows the moving water. This demonstrates that the model can align one of the channels with the moving object. Which is consistent with the original paper.

Limitations With the current setup, it only works reliably with one moving object in the scene. This is a problem for self-driving cars, as most scenes have multiple moving objects. The cause of this is that all the recognised objects get one average movement. This means that if two objects in the scene move towards each other, it will get averaged to no movement. This will result in an incorrect loss signal. This can be seen in Figure 7.

Another limitation is the frame rate. When using the nuScenes dataset (at 2 Hz) it did not learn correctly. But

after adding the sweep images (intermediate frames to get 12 Hz) it worked again. This means it works better when the images are more similar to each other.

9.2. Supervised flow network

Objective Before combining all the modules, we wanted to verify that the flow module is correctly made. This can be done by training it supervised.

Synthetic dataset We generate short videos of simple geometric shapes moving over a dynamic background. The background is a zoomed-in image which follows a sine/cosine path. On this background there is a square which moves with a constant velocity. For each adjacent frame pair (I_t, I_{t+1}) the ground-truth optical flow $F_t^\star \in \mathbb{R}^{2\times H\times W}$ is computed. Frames are 128×128 RGB; the dataset length is 100 frames.

Model We use a lightweight, PWCLite-style network (MotionFlow) with a search range of 4. This computes features for every pixel. These features are converted into a motion vector per pixel at full resolution. Correlation and convolutions use circular padding to avoid border artefacts.

Training setup The model is trained end-to-end with mean-squared error (L2) between the predicted flow $\hat{F}_t$ and ground truth $F_t^\star$:

$$\mathcal{L}_{\text{flow}} = \|\hat{F}_t - F_t^\star\|_2^2.$$

An Adam optimiser [12] is used with a learning rate of 1e-4. It has a batch size of 8, and it took 1,000 epochs.

Results Training converges smoothly from an initial loss of 6.30 to $\mathbf{0.013}$ by epoch 1000 (per-pixel MSE). Qualitatively, predicted flow closely matches the ground truth across both the moving foreground and the panning background; see Figure 8. This shows that the flow network is correctly implemented.

9.3. Supervised instance segmentation

Motivation. Before combining all the modules, we wanted to verify that the segmentation module can work supervised. We trained the segmentation model on the synthetic dataset. Where it receives the labels from the dataset. It is based on bounding boxes and not yet segmentation to create a solid base to work from.

Dataset. We use the same synthetic video generator as in the supervised flow experiment (section 9.2). For each frame, the dataset provides ground-truth boxes. All the moving objects will get the same class.

Model. The detector is a YOLOv3 head [20] with three scales at grid sizes $\{4, 8, 16\}$ for 128^2 inputs. Each scale predicts 3 anchors and outputs (objectness, x, y, w, h, class) per cell.

Anchors and targets. We use fixed anchors (pixel units, then normalised by image size):

$$\begin{aligned} \text{large}: &\ [(48,60),(64,80),(80,96)],\\ \text{medium}: &\ [(20,28),(28,36),(36,44)],\\ \text{small}: &\ [(8,10),(12,16),(16,24)]. \end{aligned}$$

Targets are built per scale by assigning each ground-truth box to its best-IoU anchor at the corresponding grid cell and writing $(\text{obj} = 1,\ t_x, t_y, w, h)$; all other locations remain negatives.

Loss. We used the standard YOLO losses:

1. no-objectness loss for negatives,
2. objectness MSE for positives with the target equal to IoU(pred, GT),
3. box regression MSE on (t_x, t_y, t_w, t_h) where $t_{x,y}$ are sigmoid offsets and $t_{w,h}$ are log-space sizes relative to anchors.
4. class loss is set to 0 (single class).

Implementation details. An Adam optimiser [12] is used with a learning rate of 1e-4. It has a batch size of 8 and it took 100 epochs.

Results. The YOLO loss decreases from 1.25 at epoch 1 to $\mathbf{0.0568}$ at epoch 50 and **0.0554** at epoch 100, indicating stable convergence. Qualitatively, the highest-confidence prediction shows a tight bounding box around the object (see Figure 9). The background motion does not trigger false positives, and the bounding box tracks the object consistently.

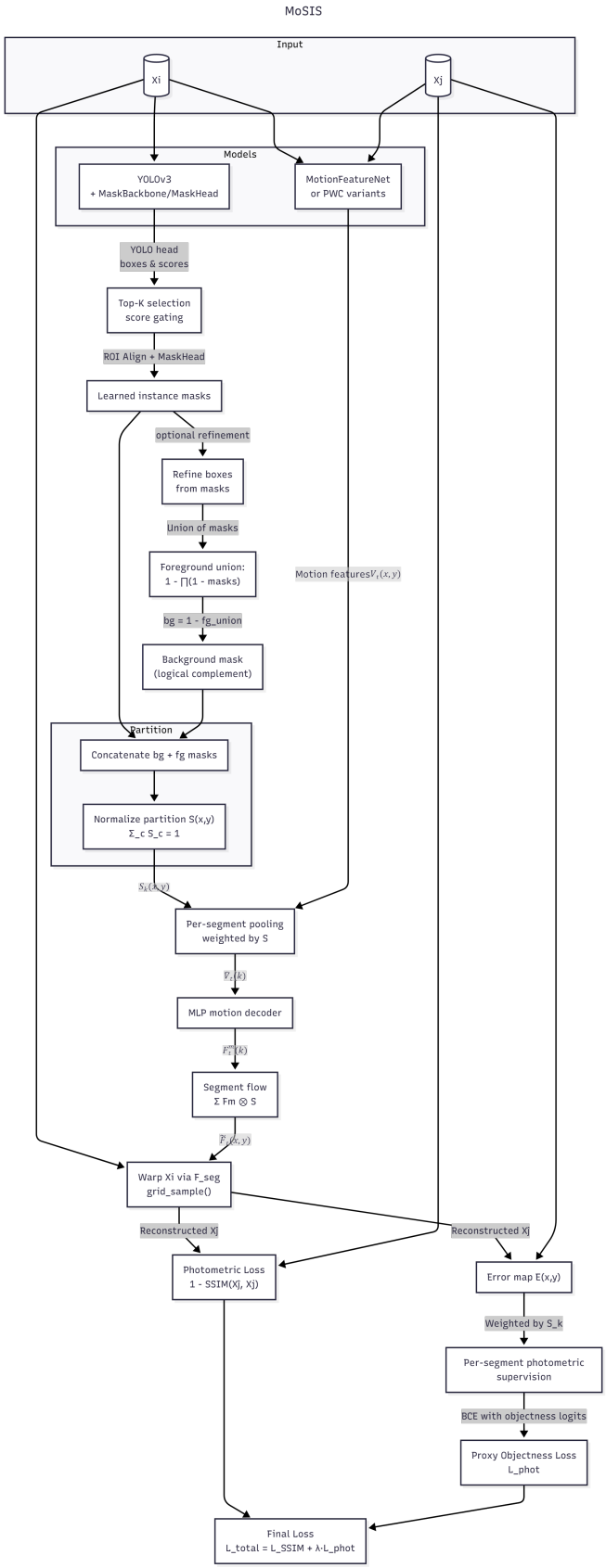


Figure 5. MoSIS detailed architecture

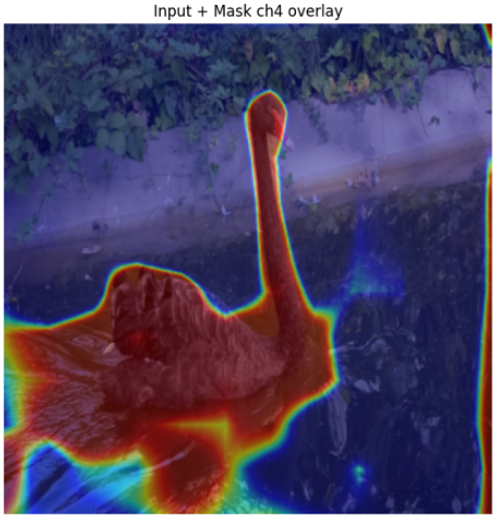


Figure 6. AMD result on the swan dataset. It nicely follows the outline of the swan. It also detects a bit of the water, as that also moves with it.

Figure 7. AMD result on the nuScenes dataset. The AMD model partially detects objects in different directions during training. But this stops when the training continues.

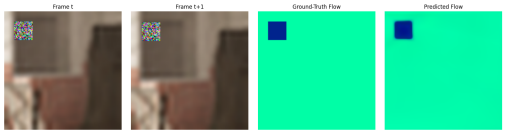


Figure 8. Supervised flow result. Every colour is a direction with the intensity as the speed. The flow shows exactly where the object is.

Figure 9. Supervised detection result. YOLO puts the red bounding boxes correctly on the square in the image.