FOUND BY NEMO: UNSUPERVISED OBJECT DETECTION FROM NEGATIVE EXAMPLES AND MOTION

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper introduces NEMO, an approach to unsupervised object detection that uses *motion*—instead of image labels—as a cue to learn object detection. To discriminate between motion of the target object and other changes in the image, it relies on *negative examples* that show the scene without the object. The required data can be collected very easily by recording two short videos, a positive one showing the object in motion and a negative one showing the scene without the object. Without any additional form of pretraining or supervision and despite of occlusions, distractions, camera motion, and adverse lighting, those videos are sufficient to learn object detectors that can be applied to new videos and even generalize to unseen scenes and camera angles. In a baseline comparison, unsupervised object detection outperforms off-the shelf template matching and tracking approaches that are given an initial bounding box of the object. The learned object representations are also shown to be accurate enough to capture the relevant information from manipulation task demonstrations, which makes them applicable to learning from demonstration in robotics. An example of object detection that was learned from 3 minutes of video can be found here [video link].

1 INTRODUCTION

Object-based representations are a powerful abstraction of our world. Since these representations remove large amounts of information—an image of size 120×160 for example has $120 \times 160 \times 3 = 57.600$ dimensions, while the coordinates of an object in that image only have 2 dimensions—object-based representations enable efficient generalization, simulation, planning, communication, etc. But grounding objects in sensory input currently relies on supervised learning, which requires a high number of labeled images, e.g. 500.000 manually annotated segments to learn 80 objects (Lin et al., 2014). This paper takes a step towards replacing this labor-intensive supervision by learning to detect objects from videos that can be gathered quickly with minimal supervision and by exploiting the physical properties of objects.

A physical object is a collection of matter that moves as a unit. *Motion*, in turn, can be a strong cue to learn object detection and replace the need for supervision in the form of labeled images. Given a video of a moving object, we can learn object-based representations by optimizing them to describe physically plausible motion (Jonschkowski et al., 2017). But this approach only works in the absence of visual distractions. With camera motion, other moving objects, or changes in the background, motion alone is not sufficient to learn such representations because many features in the image move in a physically plausible way.

This paper improves on previous approaches by learning to ignore visual distractions through *negative examples*, i.e., videos of the scene without the target object but with the distractions. These negative videos are easy to collect because they do not need to be in sync with the positive ones, i.e., they do not need to have the same sequence of camera movements or the same object motions. This paper also addresses the challenge

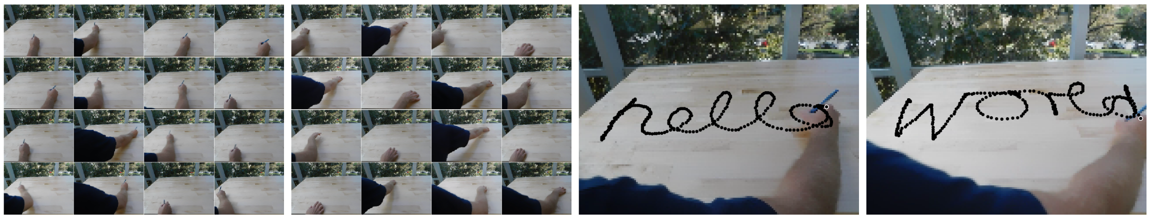


Figure 1: **Learning to detect an object from 3 min of video.** *Left to right:* training video of a pen in hand, negative video without pen, two test videos with per frame detections shown as black dots. [video link]

of changes between training and test videos, e.g. due to different lighting or changes in the background. Those changes can be harmful if an object representation is extracted using a standard pyramid-shaped convolutional network because every pixel directly affects the output, even if it is far from the object's location. Therefore, this paper uses a *spatial encoder* architecture that uses a spatial softmax output (Finn et al., 2016), which is only affected by the strongest local activations, making it invariant to many visual distractions.

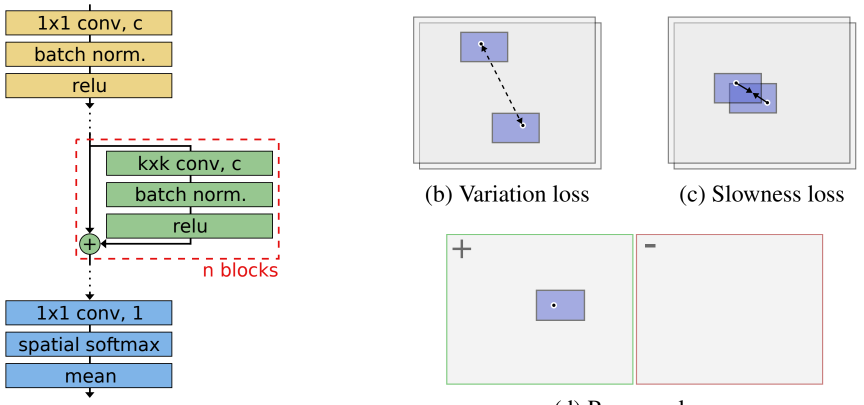
The contribution of this paper is to demonstrate unsupervised object detection based on data that is easy and fast to collect. This is achieved by formulating the use of negative examples for object detection as a loss function, combining it with motion-based learning objectives, and using these objectives to train a spatial encoder network using a combination of random search and gradient descent. The resulting method is called *learning from negative examples and motion* (NEMO). A glimpse of the results are shown in Figure 1.

Experimental results in Section 4 show that NEMO can learn new objects from only two short videos of a few minutes, without using pretrained models and without using supervision beyond marking these videos as positive and negative. The results also show that NEMO can learn object detection in the presence of frequent occlusions, distractions, camera motion, and changes in lighting and background. Even though it uses data that can be collected in seconds to minutes, the learned object detection generalizes to new scenes and camera angles and outperforms template matching and tracking baselines. The experiments also show how the learned object representations can be useful to demonstrate tasks such as writing or pick-and-place tasks, e.g. to make robot learning more data-efficient.

2 Related Work

This work is strongly related to *physics-based representation learning*, where a latent representation is learned by optimizing consistency with physics. Stewart & Ermon (2017) learn to map images to latent representations by optimizing consistency with a known dynamics model. Jonschkowski & Brock (2015) and Jonschkowski et al. (2017) make more general assumptions about physical interactions and define them as learning objectives. A number of approaches combine physical assumptions with image reconstruction to learn latent representations (Goroshin et al., 2015; Watter et al., 2015; Finn et al., 2016). Gao et al. (2016) learn to embed object regions in an image using spatio-temporal consistency. Jang et al. (2018) learn object embeddings from self-supervised interactions based on object persistence. (Jayaraman & Grauman, 2015) learn representations that are equivariant to known ego motion. Sermanet et al. (2017) learn latent representations from multiple synchronous videos of motion. While these approaches are similar to this paper in spirit, they learn image embeddings, while this paper learns to detect objects in the image coordinates. This more constrained object-based representation makes the presented approach particularly robust and efficient.

This paper is also connected to the idea of *active perception* (Bajcsy, 1988), where action is used to facilitate perception. Motion has been used for a long time to identify and track objects (Lipton et al., 1998), to segment them (Fitzpatrick, 2003), to understand their articulation (Katz & Brock, 2008), and so on. Recently, this idea has been combined with learning in order to generalize beyond the observed motion, e.g. to learn object segmentation from videos of moving objects (Pathak et al., 2017) and from videos generated by robot



(a) Spatial encoder network

(b) Variation loss

(c) Slowness loss

(d) Presence loss

Figure 2: **NEMO overview.** (a) Spatial encoder architecture with n residual blocks, c channels, and kernel size k. (b)-(d) Different losses for object detection; the target object is illustrated as a blue rectangle; the detected object location $\boldsymbol{z}$ is shown as a black dot. (b) The variation loss based on two frames at times t and $t+d$ enforces variation between detected object locations, resulting in a gradient that pushes $\boldsymbol{z}^{(t)}$ and $\boldsymbol{z}^{(t+d)}$ apart. (c) The slowness loss enforces object detections in frames t and $t+1$ to be close, which produces a gradient that pulls $\boldsymbol{z}^{(t)}$ and $\boldsymbol{z}^{(t+1)}$ together. (d) The presence loss enforces that the object is detected in the positive frame t rather than in the negative frame t^{-}, which creates a gradient that increases activations in the positive frame and decreases them in the negative frame.

interactions (Pathak et al., 2018). This paper goes into the same direction for learning object detection by introducing ideas from representation learning and by leveraging negative examples.

Labeling the training videos as positive and negative examples can also be viewed as *weakly supervised learning*, which deals with learning from labels that are only partially informative. Weakly supervised object detection relies on image-wide labels to learn to localize the corresponding objects in the image (Pandey & Lazebnik, 2011; Oquab et al., 2015). While these approaches use image-wide labels to replace object location labels, which are more difficult to obtain, this paper go a step further and only uses per-video labels and compensates this reduction of supervision by adding motion as a cue for learning object detection.

3 Unsupervised Learning from Negative Examples and Motion (NEMO)

The *key idea* of NEMO is to learn to detect an object from two videos, a *positive video* that shows the target object in motion and a *negative video* of the same scene without that object. These videos are used to optimize two objectives: 1) Learn to detect something that moves in a physically plausible way in the positive video, such that its location varies over time without having instantaneous jumps, which is defined below as a combination of a *variation loss* and a *slowness loss*. 2) Learn to detect something that is present in the positive video but not in the negative video, which is defined as a *presence loss*. These objectives are used to train a *spatial encoder* network, which produces an object detection based on the strongest activation after a stack of convolutions. Optimization is done by a combination of random search and gradient descent. We will now look in detail into each of these components.

3.1 Network Architecture: Spatial Encoder

NEMO's network architecture is based on the encoder part of deep spatial autoencoders (Finn et al., 2016) and therefore called a *spatial encoder*. The spatial encoder is a stack of convolutional layers (LeCun et al., 1998) without pooling or subsampling, which uses residual connections (He et al., 2016), batch normalization (Ioffe & Szegedy, 2015), and ReLU nonlinearities (Nair & Hinton, 2010). The spatial encoder has n residual blocks, each with c channels and kernel size k (see Figure 2a). The experiments in this paper used $c = 32$ channels, up to $n = 10$ layers, and kernel sizes $k \in \{3, 5, 7\}$. Since the parameters n and k control the receptive field of the object detector, they must roughly match the size of the target object. The experiments used $k = 7$ for learning to detect the Roomba and $k = 3$ for all other objects.

The output layer of the spatial encoder has a single channel, followed by a spatial softmax, which produces a probability distribution over the object's location in the image. The mean and mode of this distribution estimate the object's location. The mode can be used during inference because it is more robust to distractions but not during gradient-based training since it is not differentiable.

3.2 Losses: Cross-Entropy, Variation, and Slowness

The spatial encoder is trained by minimizing a combination of three losses—variation, slowness, and presence (see Figure 2b-d), which are defined here. Let us denote the input image at time t as $\boldsymbol{I}^{(t)} \in \mathbb{R}^{h \times w}$ where h and w are the height and width of the image. We will refer to the spatial encoder as f with parameters $\boldsymbol{\theta}$, and the output of f before the spatial softmax as $\boldsymbol{O}^{(t)} \in \mathbb{R}^{h \times w}$, such that $\boldsymbol{O}^{(t)} = f(\boldsymbol{I}^{(t)}; \boldsymbol{\theta})$. By applying the spatial softmax across image coordinates i and j, we get a probability image $\boldsymbol{P}^{(t)} \in \mathbb{R}^{h \times w}$ and its mean $\boldsymbol{z}^{(t)} \in \mathbb{R}^2$ normalized to $[-1, 1]$ as

$$P_{i,j}^{(t)} = \frac{e^{O_{i,j}^{(t)}}}{\sum_{i,j} e^{O_{i,j}^{(t)}}}, \quad \boldsymbol{z}^{(t)} = \begin{bmatrix} \sum_{i,j}(\frac{2i}{h} - 1)P_{i,j}^{(t)} \\ \sum_{i,j}(\frac{2j}{w} - 1)P_{i,j}^{(t)} \end{bmatrix}.$$

The first two losses, variation and slowness, operate on the mean $\boldsymbol{z}$ in positive frames. Together, they measure whether the detected object location $\boldsymbol{z}^{(t)}$ moves in a physically plausible way by comparing pairs of $\boldsymbol{z}^{(t)}$ for different t.

The *variation loss* encodes the assumption that the target object does not stay still by enforcing that $\boldsymbol{z}_{t+d}$ is different from $\boldsymbol{z}_t$ for d in some range $[d_{\text{min}}, d_{\text{max}}]$. The variation loss measures the proximity using $e^{-\text{distance}}$, which is 1 if $\boldsymbol{z}_t = \boldsymbol{z}_{t+d}$ and goes to 0 with increasing distance (Jonschkowski et al., 2017).

$$\mathcal{L}_{\text{variation}}(\boldsymbol{\theta}) = \mathbb{E}_{t, d \in [d_{\text{min}}, d_{\text{max}}]}[e^{-\beta ||\boldsymbol{z}_{t+d} - \boldsymbol{z}_t||}].$$

The hyperparameter β scales this distance and controls how far $\boldsymbol{z}_t$ and $\boldsymbol{z}_{t+d}$ need to be apart. The hyperparameters d_{min} and d_{max} define for which time differences variation is enforced; d_{min} should be large enough that the object has typically changed its location in the image after d_{min} frames; d_{max} should be small enough that slower changes in the background typically take longer than d_{max} frames. All experiments in this paper use the parameters $\beta = 10$, $d_{\text{min}} = 50$, and $d_{\text{max}} = 100$.

The *slowness loss* encodes the assumption that objects move with relatively low velocities, i.e., that their locations at time t and $t + 1$ are typically close to each other. Consequently, this loss measures the squared distance between $\boldsymbol{z}$ in consecutive time steps t and $t + 1$, which favors smooth over erratic object trajectories (Wiskott & Sejnowski, 2002; Jonschkowski & Brock, 2015).

$$\mathcal{L}_{\text{slowness}}(\boldsymbol{\theta}) = \mathbb{E}_t[||\boldsymbol{z}_{t+1} - \boldsymbol{z}_t||^2].$$

The *presence loss* encodes the assumption that the object is present in the positive video but not in the negative one. Taking a positive frame t and a negative frame t^-, we can compute the probability of the object being somewhere in the positive frame $q^{(t,t^-)}$ by computing the spatial softmax jointly over both frames and summing over all pixels. The loss is then defined as negative log probability.

$$\mathcal{L}_{\text{presence}}(\boldsymbol{\theta}) = \mathbb{E}_{t,t^-}[-\log(q^{(t,t^-)})], \text{where } \; q^{(t,t^-)} = \frac{\sum_{i,j} e^{O_{i,j}^{(t)}}}{\sum_{i,j} e^{O_{i,j}^{(t)}} + e^{O_{i,j}^{(t^-)}}}.$$

These three losses are combined in a weighted sum, $\mathcal{L}(\boldsymbol{\theta}) = w_{\text{variation}}\mathcal{L}_{\text{variation}}(\boldsymbol{\theta}) + w_{\text{slowness}}\mathcal{L}_{\text{slowness}}(\boldsymbol{\theta}) + w_{\text{presence}}\mathcal{L}_{\text{presence}}(\boldsymbol{\theta})$, where the weights were chosen such that all gradients have the same order of magnitude. All experiments in this paper use $w_{\text{variation}} = 2$, $w_{\text{slowness}} = 10$, and $w_{\text{presence}} = 1$.

3.3 Optimization: Random Search and Gradient Descent

The losses are optimized from minibatches of size b, such that every minibatch includes b samples of consecutive frames $\{(\boldsymbol{I}^{(t)}, \boldsymbol{I}^{(t+1)})\}_t$ and b samples of frames $d \in [d_{\text{min}}, d_{\text{max}}]$ steps apart $\{(\boldsymbol{I}^{(t)}, \boldsymbol{I}^{(t+d)})\}_{t,d}$, which are used to compute the variation and slowness losses. The presence loss uses all combinations of the positive frames in $\{(\boldsymbol{I}^{(t)}, \boldsymbol{I}^{(t+d)})\}_{t,d}$ with b negative frames $\{\boldsymbol{I}^{(t^-)}\}_{t^-}$ resulting in $2b^2$ pairs to average over. All experiments use $b = 10$. For a good initialization of the spatial softmax output, $\boldsymbol{O}^{(t)}$ is divided by a temperature $\alpha = 10$ before the softmax is applied. For numerical stability of the gradient computation, Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mu = 0, \sigma = 10^{-5})$ is added to $\boldsymbol{z}_t$ and $q^{(t,t^-)}$ is clipped at 10^{-3}.

When $\mathcal{L}(\boldsymbol{\theta})$ is optimized with an the adaptive gradient descent method Adam (Kingma & Ba, 2015), it either converges very quickly within a few hundred gradient descent steps or—for some objects—gets stuck in a local optimum. NEMO addresses this problem by optimizing the loss with a combination of random search and gradient descent. It initializes the spatial encoder $m = 10$ times, optimizes each by a small number, e.g. 200, gradient descent steps and then finetunes the best model in additional 1000 gradient steps. Stopping the training this early results in very fast training time of about 10 minutes on a single GPU and seemed to improve generalization compared to training to convergence, although this needs to be further investigated.

4 Experiments

The following experiments evaluate object detection for a number of different objects without using pretrained networks and without using any labeled images, instead relying on a few minutes of positive videos showing a moving object and negative videos of the same scene that are quick and easy to obtain (in total about 20 minutes of video for all experiments). All results show object detection on test videos not used for training. The results show per-frame detections in subsampled 120×160 or 90×160 videos without applying any tracking or filtering.

The experiments evaluate NEMO in three different settings, testing detection accuracy with static and moving cameras for single and multiple objects and generalization to new scenes and camera angles. The experiments also illustrate how learned object detection can enable learning from demonstration, and provide a comparison to template matching and tracking that shows a substantial advantage over these baselines.

The data and code based on TensorFlow (Abadi et al., 2015) and Keras (Chollet et al., 2015) to reproduce all experiments will be published with this paper.

4.1 Object Detection with a Static Camera

In the first experiment (see Figure 1), the algorithm learned to detect a pen in a hand from a positive video in which the pen was moved in random spirals across the table and a negative video of the hand moving without the pen. Testing the learned object detector on unseen videos of writing “hello” and “world” shows precise and stable per frame object detection from only 2.5 minutes of video. Note how the method learned to be invariant to distractions such as position of the arm.

4.2 Object Detection with a Moving Camera

Moving the camera causes motion in the entire video, which makes it more difficult to learn object detection based on motion. This experiment evaluates NEMO on such videos that were recorded by moving the camera in a way that keeps the object in video without constantly centering on it, because that would cancel any object motion in the image. For this set of experiments, the object detector uses the current frame and additionally a difference image from the previous frame as input, which produced more consistent results.

The results in Figure 3 show that NEMO works despite camera motion, that it can handle difficult lighting conditions as for the toy car, and that the learned object detection can even transfer to different scenes, as for the car detector that was trained in the balcony scene and tested in the hallway. The total training data used for learning all three objects is just under 12 minutes of video.

4.3 Detection of Multiple Objects

This experiment tests detection of multiple objects in a table top setting with five objects. The training data for this experiment are five videos of about 2 minutes, with one video per object in which the object is randomly moved on the table without any other object being present. Each video was used as positive and the remaining four as negative examples to train a different object detector. The results in Figure 4 show the output of these five object detectors in a test video in which objects are randomly rearranged on the table. These results show mostly correct detections, even under occlusions by the hand and other objects, and generalization to a different camera angle never seen during training.

The video linked in the caption reveals that the object detector is not perfect. It works most of time, but generates two kinds of errors: 1) It occasionally errorneously detects the hand as being the target object even when the object is visible—this is a clear error. 2) More often, it detects the object in a wrong location when it is actually occluded by the hand. This is because, in the current version, the spatial encoder cannot return nothing, it always returns the most likely object location. This could be resolved by estimating the detection uncertainty and only returning the object location when the certainty exceeds a threshold. Since the straight-forward approach of using the value of the pre-softmax activations as a certainty estimate did not produce satisfying results, this should be addressed in future work.

4.4 Enabling Learning from Demonstration

This experiment illustrates how object-based representations learned by NEMO can enable subsequent learning such as learning from demonstrations (see Figure 5). The experiment uses the object detectors learned in the last experiment and applies them to demonstrations for three pick-and-place tasks. Comparing the object locations in the first and last frame of the demonstration reveals which objects were moved to which locations. Simple statistics over the final object locations across all demonstrations of a task describe the task in a way that could be used as the goal for a planning algorithm or as reward for reinforcement learning.

4.5 Comparison to Baselines

Since there is no other approach for learning object detection from positive videos of moving objects and negative videos without those objects, there is no baseline to fairly compare against. To still put NEMO's performance in the context of existing work on object detection and tracking, this experiment compares it to established template matching and tracking approaches using their OpenCV implementations (Bradski, 2000): template matching with normalized cross correlation (Lewis, 1995), tracking with online boosting (OLB, Grabner et al., 2006), tracking with online multiple instance learning (MIL, Babenko et al., 2009), tracking with kernelized correlation filters (KCF, Henriques et al., 2015), and tracking-learning-detection (TLD, Kalal et al., 2012). Since all of these baselines need some form of supervision, they were given bounding

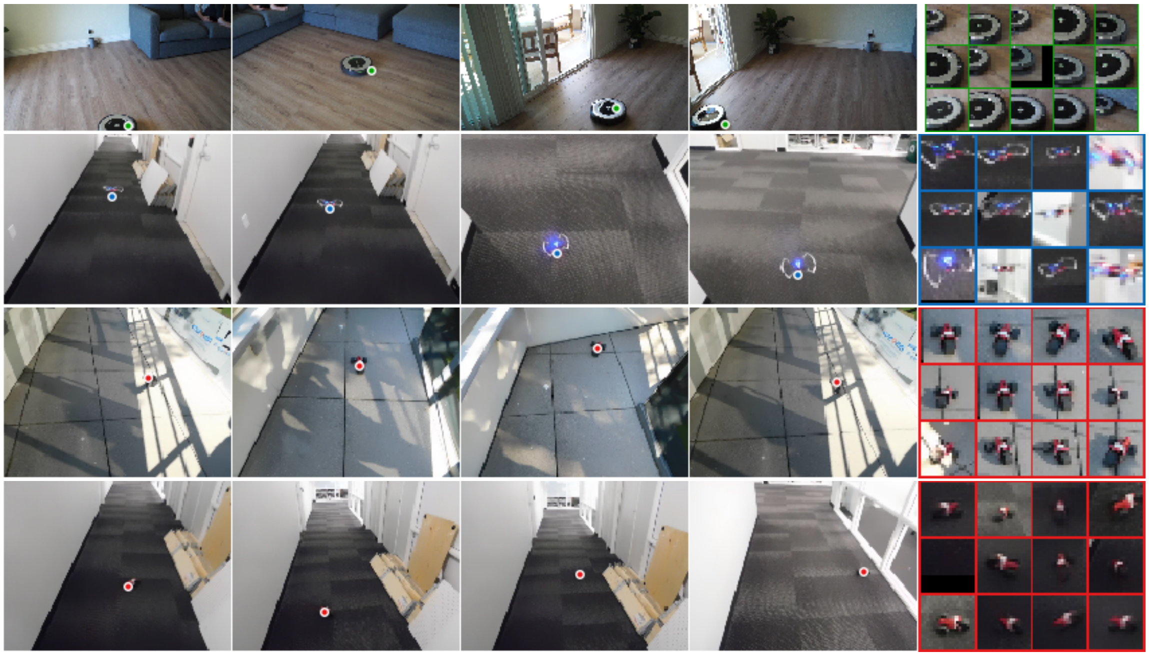


Figure 3: **Learning to detect a moving object with a moving camera.** *Left to Right:* Object detection in four random test frames; a set of image crops centered at the detected locations for random test frames. *Top to bottom:* Three separately trained object detectors for a Roomba in a living room, a drone in a hallway, and a toy car on a balcony. The last row shows toy car detector generalizing to a different scene. [video link]

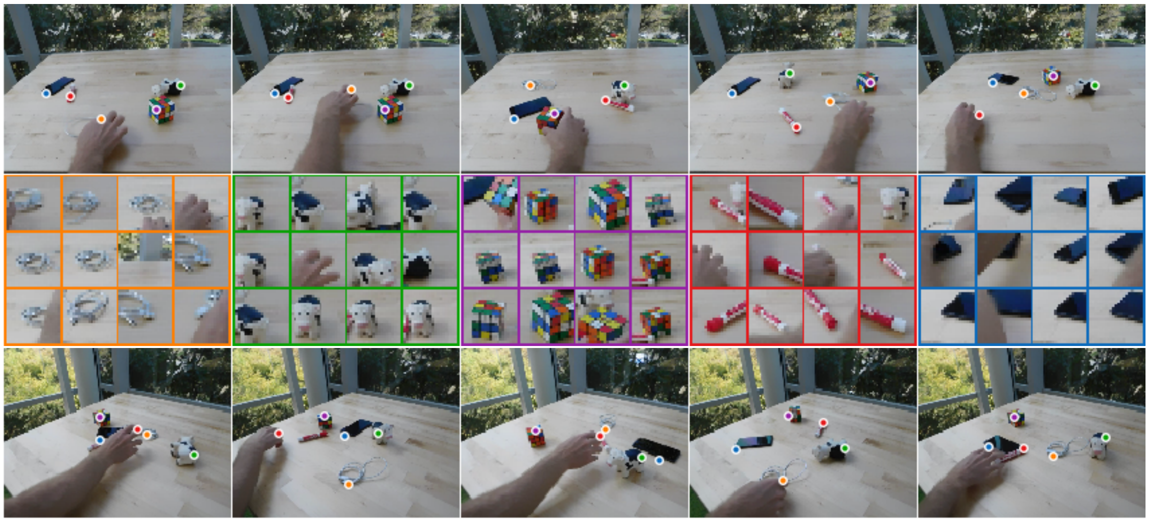


Figure 4: **Learning to detect multiple objects:** a USB cable (orange), a toy cow (green), a Rubik's Cube (purple), a whiteboard marker (red), and a phone (blue). *Top:* Random frames in a multi-object test video after training on single object videos. *Middle:* Image crops centered at the detected object locations in random test frames. *Bottom:* Generalization to a test video from a different viewpoint. [video link]

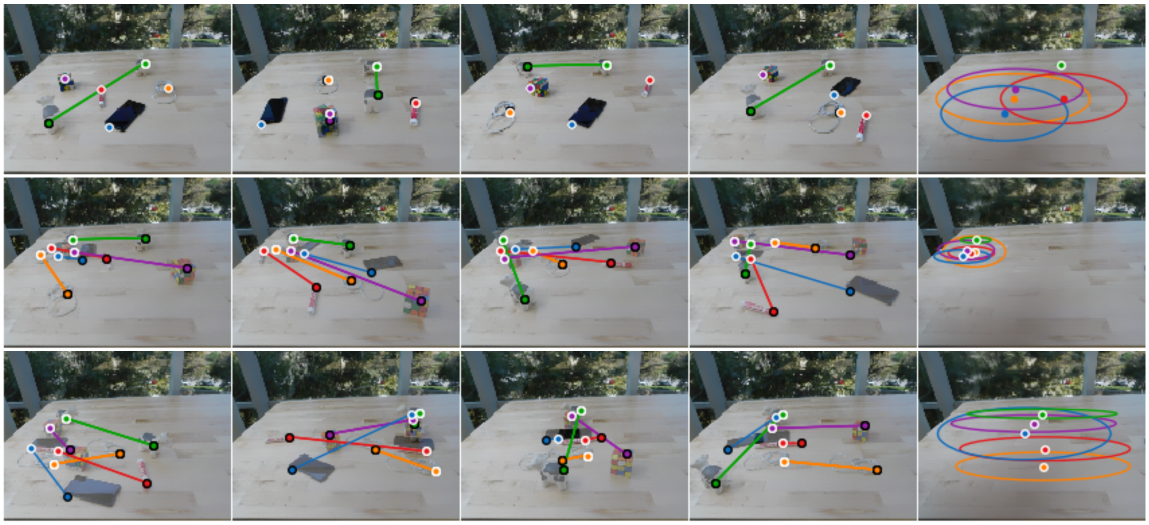
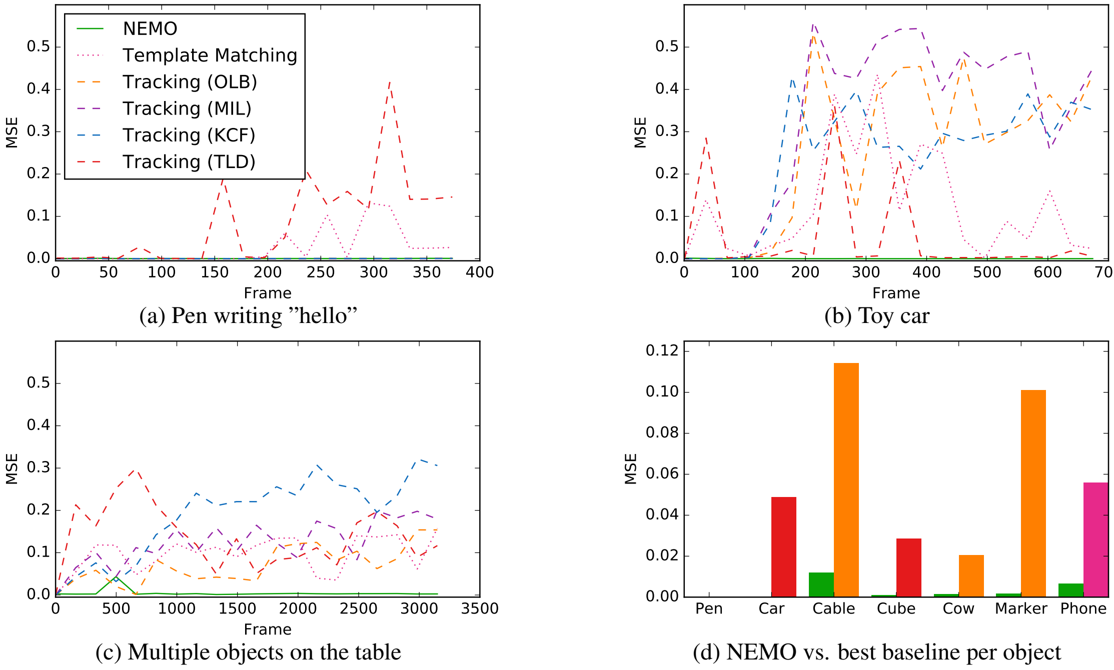


Figure 5: **Demonstrating manipulation tasks with learned object representations.** *Left:* Four demonstrations with overlayed first and the last frames. First frame object detections (black circles) are connected to last frame detections (white circles). *Right:* Mean and three standard deviations of last frame detections across all demonstrations reveal the goal of the tasks; white circle indicates task-relevance based on if average object motion is at least 10 pixels. *Top to bottom:* Three manipulation tasks: 1) Move the cow (green) to a goal location. 2) Move all objects to a goal location. 3) Arrange objects in a vertical line ordered by size.



(a) Pen writing ”hello”

(b) Toy car

(c) Multiple objects on the table

(d) NEMO vs. best baseline per object

Figure 6: **Baseline comparisons.** Mean squared error (MSE) is computed in image coordinates normalized to [0,1]. (a-c) MSE over time for three test videos. NEMO outperforms all baselines in (b) and (c). (d) MSE for a given object shows substantial advantage compared to best baselines; methods are colored as in (a).

boxes around the initial object locations to initialize tracking or extract an object template. The baselines were applied to the non-subsampled video with resolution 480×640.

Figure 6 compares NEMO to these baselines for three test videos from the experiments above, for which 20 equally spaced frames across the video were manually labeled. The results show that detecting or tracking the pen is relatively easy. NEMO, MIL, and KCF retain a very low error rate in this video. But for the other two videos, all tracking methods quickly diverge and template matching performs poorly as well, while NEMO solves these videos almost perfectly. The tracking methods fail in these videos due to substantial occlusions during object motion in the table top setting and difficult lighting conditions and fast motion in the toy car example. Template matching cannot handle the change in appearance of the objects based on lighting, scale, and orientation. The additional videos used by NEMO for learning the object detector, however, lead to robustness to these variations, which shows the advantage of unsupervised learning.

5 Conclusion

This paper presented NEMO, a novel approach to unsupervised object detection from short videos of moving objects and negative videos of scenes without those objects. By demonstrating data-efficient and robust object detection without the use of image labels, this paper opens up new research directions. There are a number of extensions that would improve the presented approach. Combining it with ensemble methods, for example, could provide an uncertainty estimate required to infer whether the object is visible in the current frame. Integrating the approach with tracking or filtering could exploit temporal consistency not only during training but also during inference. For learning multiple objects in the same scene, merging the different object detectors into a single network could improve performance by sharing intermediate features. And creating a large-scale data-set for this approach would be very valuable to develop it further.

Taking a broader view, the presented approach takes a step towards unsupervised learning of object-based representations. While this paper used manually recorded videos, the method can also be applied to data collected by a robot similar to Jang et al. (2018) and Pathak et al. (2018) to learn objects autonomously. Acquiring such object-based representations could build a bridge to geometric and symbolic reasoning and enable efficient learning, communication, prediction, and planning in object-based representations.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Visual tracking with online multiple instance learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 983–990, 2009.

Ruzena Bajcsy. Active perception. In *IEEE Proceedings*, volume 76, pp. 996–1006, 1988.

Gary Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

François Chollet et al. Keras. https://keras.io, 2015.

Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel. Deep spatial autoencoders for visuomotor learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 512–519, 2016.

Paul Fitzpatrick. First contact: an active vision approach to segmentation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pp. 2161–2166. IEEE, 2003.

Ruohan Gao, Dinesh Jayaraman, and Kristen Grauman. Object-centric representation learning from unlabeled videos. In *Asian Conference on Computer Vision (ACCV)*, November 2016.

Ross Goroshin, Michael F Mathieu, and Yann LeCun. Learning to linearize under uncertainty. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 1234–1242, 2015.

Helmut Grabner, Michael Grabner, and Horst Bischof. Real-time tracking via on-line boosting. In *British Machine Vision Conference (BMVC)*, volume 1, pp. 6, 2006.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, pp. 448–456, 2015.

Eric Jang, Coline Devin, Vincent Vanhoucke, and Sergey Levine. Grasp2vec: Learning object representations from self-supervised grasping. *Proceedings of Machine Learning Research*, 2018.

Dinesh Jayaraman and Kristen Grauman. Learning image representations tied to ego-motion. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 1413–1421, 2015.

Rico Jonschkowski and Oliver Brock. Learning state representations with robotic priors. *Autonomous Robots*, 39(3):407–428, 2015. ISSN 0929-5593.

Rico Jonschkowski, Roland Hafner, Jonathan Scholz, and Martin Riedmiller. Pves: Position-velocity encoders for unsupervised learning of structured state representations. In *New Frontiers for Deep Learning in Robotics Workshop at RSS*, 2017.

Zdenek Kalal, Krystian Mikolajczyk, Jiri Matas, et al. Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1409, 2012.

Dov Katz and Oliver Brock. Manipulating articulated objects with interactive perception. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 272–277. IEEE, 2008.

Diederik P Kingma and Jimmy Lei Ba. Adam: Amethod for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

John P Lewis. Fast template matching. In *Vision interface*, volume 95, pp. 15–19, 1995.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pp. 740–755. Springer, 2014.

Alan J Lipton, Hironobu Fujiyoshi, and Raju S Patil. Moving target classification and tracking from real-time video. In *Fourth IEEE Workshop on Applications of Computer Vision (WACV)*, pp. 8–14. IEEE, 1998.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning (ICML)*, pp. 807–814, 2010.

Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Is object localization for free?-weakly-supervised learning with convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 685–694, 2015.

Megha Pandey and Svetlana Lazebnik. Scene recognition and weakly supervised object localization with deformable part-based models. In *International Conference on Computer Vision (ICCV)*, pp. 1307–1314. IEEE Computer Society, 2011.

Deepak Pathak, Ross B Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pp. 7, 2017.

Deepak Pathak, Yide Shentu, Dian Chen, Pulkit Agrawal, Trevor Darrell, Sergey Levine, and Jitendra Malik. Learning instance segmentation by interaction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 2042–2045, 2018.

Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, and Sergey Levine. Time-contrastive networks: Self-supervised learning from video. *arXiv preprint arXiv:1704.06888*, 2017.

Russell Stewart and Stefano Ermon. Label-free supervision of neural networks with physics and domain knowledge. In *AAAI Conference on Artificial Intelligence*, volume 1, pp. 1–7, 2017.

Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 2746–2754, 2015.

Laurenz Wiskott and Terrence J Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, 2002.