# Panoptic Segmentation Forecasting

Colin Graber[1]* Grace Tsai[2] Michael Firman[2] Gabriel Brostow[2,3] Alexander Schwing[1]
[1] University of Illinois at Urbana-Champaign [2] Niantic [3] University College London

## Abstract

*Our goal is to forecast the near future given a set of recent observations. We think this ability to forecast, i.e., to anticipate, is integral for the success of autonomous agents which need not only passively analyze an observation but also must react to it in real-time. Importantly, accurate forecasting hinges upon the chosen scene decomposition. We think that superior forecasting can be achieved by decomposing a dynamic scene into individual 'things' and background 'stuff'. Background 'stuff' largely moves because of camera motion, while foreground 'things' move because of both camera and individual object motion. Following this decomposition, we introduce panoptic segmentation forecasting. Panoptic segmentation forecasting opens up a middle-ground between existing extremes, which either forecast instance trajectories or predict the appearance of future image frames. To address this task we develop a two-component model: one component learns the dynamics of the background stuff by anticipating odometry, the other one anticipates the dynamics of detected things. We establish a leaderboard for this novel task, and validate a state-of-the-art model that outperforms available baselines.*

## 1. Introduction

An intelligent agent must *anticipate* the outcome of its movement in order to navigate safely [14, 41]. Said differently, successful autonomous agents need to understand the dynamics of their observations and forecast likely future scenarios in order to successfully operate in an evolving environment. However, contemporary work in computer vision largely *analyzes* observations, i.e., it studies the apparent. For instance, classical semantic segmentation [8, 42] aims to delineate the observed outline of objects. While understanding an observation is a first seminal step, it is only part of our job. Analyzing the currently observed frame means information is out of date by the time we know the outcome, regardless of the processing time. It is even more stale by the time an autonomous agent can perform an action. Successful agents therefore need to *anticipate* the future 'state' of the observed scene. An important question, however, remains open: what is a suitable 'state' representation for the future of an observed scene?

*Work done during an internship at Niantic.

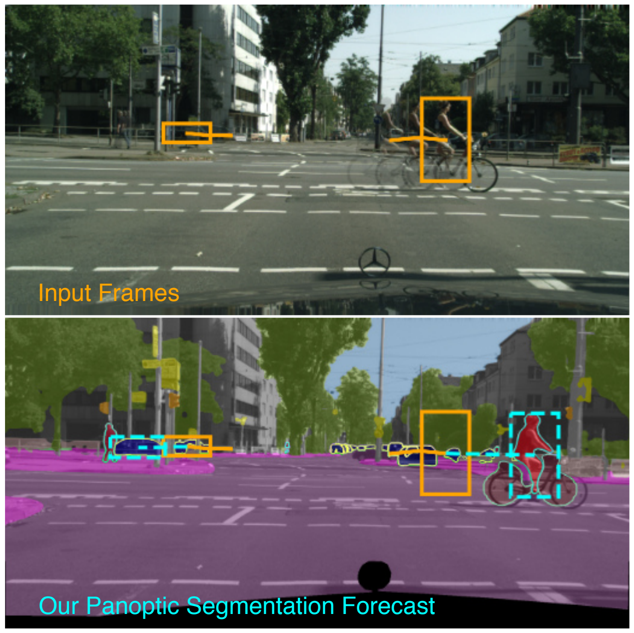


Figure 1. We study the novel task of 'panoptic segmentation forecasting' and propose a state-of-the-art method that models the motion of individual 'thing' instances separately while modeling 'stuff' as purely a function of estimated camera motion.

Panoptic segmentation recently emerged as a rich representation of a scene. Panoptic segmentation classifies each pixel as either belonging to a foreground instance, the union of which is referred to as 'things,' or as a background class, referred to as 'stuff' [23, 5]. This decomposition is useful for forecasting because we expect different dynamics for each component: 'stuff' moves because of the observer's motion, while 'things' move because of both observer and object motion. Use of panoptic segmentation is further underlined by the fact that it separates different instances of objects, each of which we expect to move individually.

Consequently, we propose to study the novel task of 'panoptic segmentation forecasting': given a set of observed frames, the goal is to forecast the panoptic segmentation for a set of unobserved frames (Fig. 1). We also propose a first approach to forecasting future panoptic segmentations. In contrast to typical semantic forecasting [44, 52], we model the motion of individual object instances and the background separately. This makes instance information persistent during forecasting, and allows us to understand the motion of each moving object.

To the best of our knowledge, we are the first to forecast panoptic segmentations for future, *unseen* frames in an image sequence. We establish a leaderboard for this task on the challenging Cityscapes dataset [12] and include a set of baseline algorithms. Our method for future panoptic segmentation relies on a number of innovations (Sec. 3.1), that we ablate to prove their value. Our method also results in state-of-the-art on previously established tasks of future semantic and instance segmentation. Code implementing models and experiments can be found at https://github.com/nianticlabs/panoptic-forecasting.

## 2. Related Work

We briefly review work which *analyzes* a single, given frame. We then discuss work which *anticipates* info about future, unseen frames. To reduce ambiguity, we avoid use of the word 'predict,' instead using analyze (looking at a current image) or anticipate (hypothesizing for a future frame).

### 2.1. Methods That Analyze

**Semantic segmentation:** Semantic segmentation has received a considerable amount of attention over decades. The task requires methods to delineate the outline of objects in a given image, either per instance or per object class [56, 54, 57]. Recently, deep-net-based methods report state-of-the-art results [42, 1, 40]. Many architecture improvements like dilated convolutions [72], skip-connections [51], *etc*., have been developed for semantic segmentation before they found use in other tasks. Our work differs as we care about *panoptic* segmentation, and we aim to anticipate the segmentation of future, unseen frames.

**Panoptic segmentation:** Recently, panoptic segmentation [32, 29] has emerged as a generalization of both semantic and instance segmentation. It requires methods to give both a per-pixel semantic segmentation of an input image while also grouping pixels corresponding to each object instance. This 'things' *vs*. 'stuff' view of the world [23] comes with its own set of metrics. Performing both tasks jointly has the benefit of reducing computation [32, 68] and enables both tasks to help each other [35, 37]. This is similar in spirit to multi-task learning [33, 55]. Other works have relaxed the high labeling demands of panoptic segmentation [36] or improved architectures [49, 9]. Panoptic segmentation has been extended to videos [30], but, again in contrast to our work only analyzing frames available at test time without anticipating future results.

### 2.2. Methods That Anticipate

Anticipating, or synonymously 'forecasting,' has received a considerable amount of attention in different communities [61]. Below, we briefly discuss work on forecasting non-semantic information such as object location before discussing forecasting of semantics and instances.

**Forecasting of non-semantic targets:** The most common forecasting techniques operate on trajectories. They track and anticipate the future position of individual objects, either in 2D or 3D [15, 46, 16, 71]. For instance, Hsieh *et al*. [26] disentangle position and pose of multiple moving objects – but only on synthetic data. Like ours, Kosiorek *et al*. [34] track instances to forecast their future, but only in limited experimental scenarios.

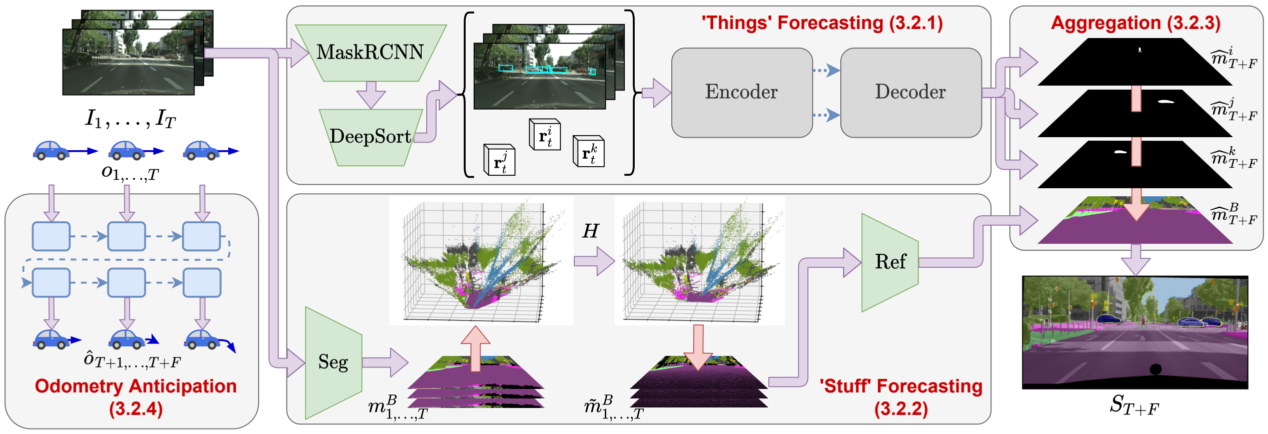
Several methods forecast future RGB frames [38, 17, 70]. Due to the high-dimensional space of the forecasts and due to ambiguity, results can be blurry, despite significant recent advances. Uncertainty over future frames can be modelled, e.g., using latent variables [63, 70]. Related to our approach, Wu *et al*. [66] treat foreground and background separately for RGB forecasting, but they do not model egomotion. All these methods differ from ours in output and architecture.

**Forecasting semantics:** Recently, various methods have been proposed to estimate semantics for future, unobserved frames. Luc *et al*. [44] use a conv net to estimate the future semantics given as input the current RGB and semantics, while Nabavi *et al*. [48] use recurrent models with semantic maps as input. Chiu *et al*. [10] further use a teacher net to provide the supervision signal during training, while Šarić *et al*. [52] use learnable deformations to help forecast future semantics from input RGB frames. However, these methods do not explicitly consider dynamics of the scene.

While Jin *et al*. [28] jointly predict flow and future semantics, some works explicitly warp deep features for future semantic segmentation [53]. Similarly, Terwilliger *et al*. [59] use an LSTM to estimate a flow field to warp the semantic output from an input frame. However, by warping in output space – rather than feature space – their model is limited in its ability to reason about occlusions and depth. While flow improves the modeling of the dynamic world, these methods only consider the dynamics at the pixel-level. Instead, we model dynamics at the object level.

Recent methods [50, 62, 69, 25] estimate future frames by reasoning about shape, egomotion, and foreground motion separately. However, none of these methods reason explicitly about individual instances, while our method yields a full future panoptic segmentation forecast.

**Forecasting future instances:** Recent approaches for forecasting instance segmentation use a conv net to regress the deep features corresponding to the future instance segmentation [43] or LSTMs [58]. Couprie *et al*. [13] use a conv net to forecast future *instance contours* together with an



'Things' Forecasting (3.2.1)
Aggregation (3.2.3)
MaskRCNN
DeepSort
Encoder
Decoder
$I_1, \ldots, I_T$
$o_{1,\ldots,T}$
$\hat{o}_{T+1,\ldots,T+F}$
Odometry Anticipation (3.2.4)
Seg
$H$
Ref
$m^B_{1,\ldots,T}$
$\tilde{m}^B_{1,\ldots,T}$
'Stuff' Forecasting (3.2.2)
$S_{T+F}$

Figure 2. **Method overview.** Given input frames $I_{1,\ldots,T}$, our method anticipates the panoptic segmentation $S_{T+F}$ of unseen frame $I_{T+F}$. Our method decomposes the scene into 'things' and 'stuff' forecasting. 'Things' are found via instance segmentation/tracking on the input frames, after which we forecast the segmentation mask and depth of each individual instance (Sec. 3.1.1). Next, 'Stuff' is modeled by warping input frame semantics to frame $T + F$ using a 3d rigid-body transformation and then passing the result through a refinement model (Sec. 3.1.2). Finally, we aggregate the forecasts from 'things' and 'stuff' into the final panoptic segmentation $S_{T+F}$ (Sec. 3.1.3). Various components require future odometry $\widehat{o}_{T+1,\ldots,T+F}$, which we anticipate using input odometry $o_{1,\ldots,T}$ (Sec. 3.1.4).

instance-wise semantic segmentation to estimate future instance segmentation. Their method only estimates foreground and not background semantics. Several works have focused on anticipating future pose and location of specific object types, often people [45, 20]. Ye *et al.* [70] forecast future RGB frames by modeling each foreground object separately. Unlike these works, we anticipate both instance segmentation masks for foreground objects and background semantics for future time steps.

## 3. Panoptic Segmentation Forecasting

We introduce *Panoptic Segmentation Forecasting*, a new task which requires to anticipate the panoptic segmentation for a future, unobserved scene. Different from classical panoptic segmentation which *analyzes* an observation, panoptic segmentation forecasting asks to *anticipate* what the panoptic segmentation looks like at a later time.

Formally, given a series of $T$ RGB images $I_1, \ldots, I_T$ of height $H$ and width $W$, the task is to anticipate the panoptic segmentation $S_{T+F}$ that corresponds to an unobserved future frame $I_{T+F}$ at a fixed number of timesteps $F$ from the last observation recorded at time $T$. Each pixel in $S_{T+F}$ is assigned a class $c \in \{1, \ldots, C\}$ and an instance ID.

### 3.1. Method

Anticipating the state of a future unobserved scene requires to understand the dynamics of its components. 'Things' like cars, pedestrians, *etc.* often traverse the world 'on their own.' Meanwhile, stationary 'stuff' changes position in the image due to movement of the observer camera. Because of this distinction, we expect the dynamics of 'things' and 'stuff' to differ. Therefore, we develop a model comprised of two components, one for the dynamics of detected 'things' and one for the rest of the 'stuff.'

In addition to RGB images, we assume access to camera poses $o_1, \ldots, o_T$ and depth maps $d_1, \ldots, d_T$ for input frames. Camera poses can come from odometry sensors or estimates of off-the-shelf visual SLAM methods [6]. We obtained our depth maps from input stereo pairs [21] (these could also be estimated from single frames [64]).

An overview of our panoptic segmentation forecasting is shown in Fig. 2. The method consists of four stages:

**1) 'Things' forecasting** (Sec. 3.1.1): For each instance $i$, we extract foreground instance tracks $l^i$ from the observed input images $I_1, \ldots, I_T$. We use these tracks in our model to anticipate a segmentation mask $\widehat{m}^i_{T+F}$ and depth $\widehat{d}^i_{T+F}$ for the unobserved future frame at time $T + F$.

**2) 'Stuff' forecasting** (Sec. 3.1.2): We predict the change in the background scene as a function of the anticipated camera motion, producing a background semantic output $\widehat{m}^B_{T+F}$ for the unobserved future frame $I_{T+F}$.

**3) Aggregation** (Sec. 3.1.3): We aggregate foreground 'things' instance forecasts $\widehat{m}^i_{T+F}$ and background scene forecast $\widehat{m}^B_{T+F}$, producing the final panoptic segmentation output $S_{T+F}$ for future frame $I_{T+F}$.

**4) Odometry anticipation** (Sec. 3.1.4): To better handle situations where we do not know future odometry, we train a model to forecast odometry from the input motion history.

#### 3.1.1 'Things' forecasting:

The foreground prediction model, sketched in Fig. 3, first locates the instance locations $l^i$ within the input sequence. These tracks are each then independently processed by an encoder which captures their motion and appearance history. Encoder outputs are then used to initialize the decoder, which predicts the appearance and location of instances for future frames, including depth $\widehat{d}^i_{T+F}$. These are processed using a mask prediction model

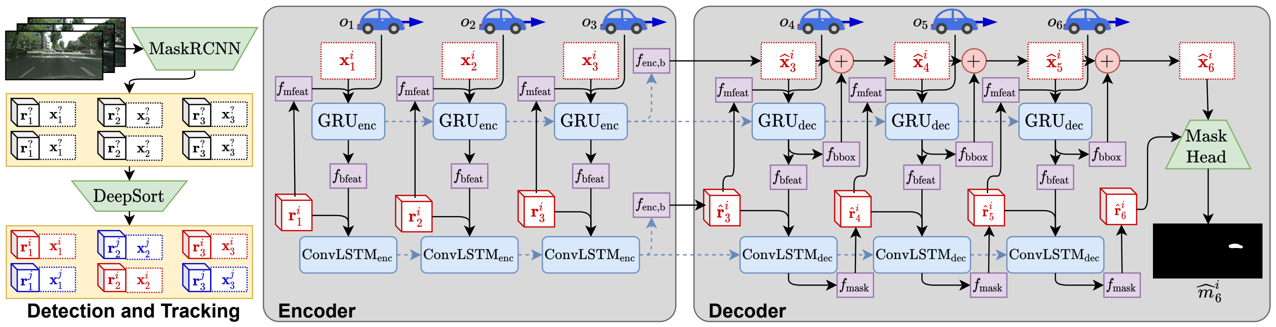


Figure 3. **The 'Things' forecasting model.** This produces instance masks $\widehat{m}^i_{T+F}$ for each instance $i$ at target frame $T+F$. These masks are obtained from input images $I_1, \ldots, I_T$ via the following procedure: images are used to produce bounding box feature $\mathbf{x}^i_t$ and mask features $\mathbf{r}^i_t$ using MaskR-CNN and DeepSort (left). These features are then input into an encoder to capture the instance motion history (middle). Encoder outputs are used to initialize a decoder, which predicts the features $\widehat{\mathbf{x}}^i_{T+F}$ and $\widehat{\mathbf{r}}^i_{T+F}$ for target frame $T+F$ (right). These features are passed through a mask prediction head to produce the final output $\widehat{m}^i_{T+F}$. Here, $T=3$ and $T+F=6$.

to produce the final instance mask $\widehat{m}^i_{T+F}$.

The output of the foreground prediction model is a set of estimated binary segmentation masks $\widehat{m}^i_{T+F} \in \{0,1\}^{H\times W}$ representing the per-pixel location for every detected instance $i$ at frame $T+F$. Formally we obtain the mask via

$$\tilde{m}^i_{T+F} = \text{MaskOut}\left(\widehat{\mathbf{r}}^i_{T+F}\right), \tag{1}$$

$$\widehat{m}^i_{T+F} = \text{Round}\left(\text{Resize}\left(\tilde{m}^i_{T+F}, \widehat{\mathbf{x}}^i_{T+F}\right)\right). \tag{2}$$

Here, in a first step, MaskOut uses a small convolutional network (with the same architecture as the mask decoder of [22]) to obtain fixed-size segmentation mask probabilities $\tilde{m}^i_{T+F} \in [0,1]^{28\times 28}$ from a mask feature tensor $\widehat{\mathbf{r}}^i_{T+F} \in \mathbb{R}^{256\times 14\times 14}$. In a second step, Resize scales this mask to the size of the predicted bounding box represented by the bounding box representation vector $\widehat{\mathbf{x}}^i_{T+F}$ using bilinear interpolation while filling all remaining locations with 0. The bounding box information vector $\widehat{\mathbf{x}}^i_{T+F} := [cx, cy, w, h, d, \Delta cx, \Delta cy, \Delta w, \Delta h, \Delta d]$ contains object center coordinates, width, and height, which are used in Resize, and also an estimate of the object's distance from the camera, and the changes of these quantities from the previous frame, which will be useful later. The output depth $\widehat{d}^i_{T+F}$ is also obtained from this vector.

**Decoder.** To anticipate the bounding box information vector $\widehat{\mathbf{x}}^i_{T+F}$ and its appearance $\widehat{\mathbf{r}}^i_{T+F}$, we use a **decoder**, as shown on the right-hand-side of Fig. 3. It is comprised primarily of two recurrent networks: a GRU [11] which models future bounding boxes and a ConvLSTM [67] which models the future mask features. Intuitively, the GRU and ConvLSTM update hidden states $h^i_{b,t}$ and $h^i_{m,t}$, representing the current location and appearance of instance $i$, as a function of the bounding box features $\widehat{\mathbf{x}}_{t-1}$ and mask features $\widehat{\mathbf{r}}^i_{t-1}$ from the previous time step. These states are used to predict location and appearance features for the current timestep, which are then autoregressively fed into the model to forecast into the future; this process continues for $F$ steps until reaching the target time step $T+F$. More formally,

$$h^i_{b,t} = \text{GRU}_{\text{dec}}([\widehat{\mathbf{x}}_{t-1}, o_t, f_{\text{mfeat}}(\widehat{\mathbf{r}}^i_{t-1})], h^i_{b,t-1}), \tag{3}$$

$$\widehat{\mathbf{x}}^i_t = \widehat{\mathbf{x}}^i_{t-1} + f_{\text{bbox}}(h^i_{b,t}), \tag{4}$$

$$h^i_{m,t} = \text{ConvLSTM}_{\text{dec}}([\widehat{\mathbf{r}}^i_{t-1}, f_{\text{bfeat}}(h^i_{b,t})], h^i_{m,t-1}), \tag{5}$$

$$\widehat{\mathbf{r}}^i_t = f_{\text{mask}}(h^i_{m,t}), \tag{6}$$

for $t \in \{T+1, \ldots, T+F\}$, where $o_t$ represents the odometry at time $t$, $f_{\text{bbox}}$ and $f_{\text{bfeat}}$ are multilayer perceptrons, and $f_{\text{mask}}$ and $f_{\text{mfeat}}$ are $1\times 1$ convolutional layers.

**Encoder.** The decoder uses bounding box hidden state $h^i_{b,T}$, appearance feature hidden state $h^i_{m,T}$, and estimates of the bounding box features $\widehat{\mathbf{x}}^i_T$ and mask appearance features $\widehat{\mathbf{r}}^i_T$ for the most recently observed frame $I_T$. We obtain these quantities from an **encoder** which processes the motion and appearance history of instance $i$. Provided with bounding box features $\mathbf{x}^i_t$, mask features $\mathbf{r}^i_t$, and odometry $o_t$ for input time steps $t \in \{1, \ldots, T\}$, the encoder computes the aforementioned quantities via

$$h^i_{b,t} = \text{GRU}_{\text{enc}}([\mathbf{x}^i_{t-1}, o_{t-1}, f_{\text{mfeat}}(\mathbf{r}^i_{t-1})], h^i_{b,t-1}), \tag{7}$$

$$h^i_{m,t} = \text{ConvLSTM}_{\text{enc}}([\mathbf{r}^i_{t-1}, f_{\text{bfeat}}(h^i_{b,t})], h^i_{m,t-1}). \tag{8}$$

Intuitively, the bounding box encoder is a GRU which processes input bounding box features $\mathbf{x}^i_t$, odometry $o_t$, and a transformation of mask features $\mathbf{r}^i_t$ to produce box state representation $h^i_{b,T}$. Additionally, the mask appearance encoder is a ConvLSTM which processes input mask features $\mathbf{r}^i_t$ and the representation of the input bounding box features $h^i_{b,t}$ produced by the bounding box encoder to obtain mask state representation $h^i_{m,T}$.

The estimated mask and bounding box feature estimates for the final input time step $T$ are computed by processing the final encoder hidden states via

$$\widehat{\mathbf{x}}^i_T = f_{\text{enc,b}}(h^i_{b,T}), \text{ and } \widehat{\mathbf{r}}^i_T = f_{\text{enc,m}}(h^i_{m,T}), \tag{9}$$

where $f_{\text{enc,b}}$ is a multilayer perceptron and $f_{\text{enc,m}}$ is a $1 \times 1$ convolution. These estimates are necessary because occlusions can prevent access to location and appearance for time step $T$ for some object instances. In those cases, use of Eq. (9) is able to fill the void.

**Tracking.** The encoder operates on estimated instance tracks/locations $l^i := \{c^i, (\mathbf{x}_t^i, \mathbf{r}_t^i)|_{t=1}^T\}$ which consist of object class $c^i$, bounding box features $\mathbf{x}_t^i$ and mask features $\mathbf{r}_t^i$ for all instances in the input video sequence $I_1, \ldots, I_T$. Obtaining these involves two steps: 1) we run MaskR-CNN [22] on every input frame to find the instances; 2) we link instances across time using DeepSort [65]. For a given tracked instance $i$, we use outputs provided by MaskR-CNN, including predicted class $c^i$, bounding boxes $\mathbf{x}_t^i$, and mask features $\mathbf{r}_t^i$ extracted after the ROIAlign stage. The distance $d$ within $\mathbf{x}_t^i$ refers to the median value of the input depth map $d_t$ at locations corresponding to the estimated instance segmentation mask found by MaskR-CNN for instance $i$ in input frame $t$. A given object instance may not be found in all input frames, either due to the presence of occlusions or because it has entered or left the scene. In these cases, we set the inputs to an all zeros tensor.

Note that it is possible for instances to be missed during the detection phase. We largely observe this to happen for static objects such as groups of bicycles parked on a sidewalk (for instance, the right side of our prediction in the fourth row of Fig. 4). One solution is to consider these instances as part of the background forecasting. However, in our experiments, we found that treating all the missed instances as background degraded our performance because some instances are actually dynamic. Thus, in this paper, we choose not to recover these instances.

**Losses.** To train the foreground model, we provide input location and appearance features, predict their future states, and regress against their pseudo-ground-truth future states. More specifically, the losses are computed using the estimated bounding boxes $\mathbf{x}_t^i$ and instance features $\mathbf{r}_t^i$ found by running instance detection and tracking on future frames. Note that losses are computed on intermediate predictions as well, which permits to properly model motion and appearance of instances across all future time steps. Our foreground model loss is a weighted sum of mean squared error and L1 losses. See appendix Sec. E.1 for full details.

#### 3.1.2 'Stuff' forecasting:

The background 'stuff' forecasting is tasked with predicting a semantic output $\widehat{m}_{T+F}^B \in \{1, \ldots, C_{\text{stuff}}\}^{H \times W}$ for every pixel in the target frame $T + F$. We assume they correspond to the static part of the scene, i.e., background changes in the images are caused solely by camera motion.

We predict the background changes by back-projecting 3D points from the background pixels in frame $t$ given depth $d_t$ and camera intrinsics, transforming with ego-motion $o_t$, and projecting to frame $T + F$. This process establishes pixel correspondences between input frame $t$ and target frame $T + F$. After running a pre-trained semantic segmentation model on frame $I_t$ to get semantic segmentation $m_t$, we use these correspondences to map the semantic labels from $m_t$, which correspond to "stuff" classes, to pixels in frame $T + F$ and maintain their projected depth at this frame. We denote the projected semantic map as $\tilde{m}_t^B$ and the projected depth as $\tilde{d}_t^B$. However, due to 1) sparsity of the point clouds, and 2) lack of information in regions which were previously occluded by foreground objects or were not previously in-frame, only a subset of pixels in $\tilde{m}_t^B$ are assigned a label. Therefore, we apply a refinement model that takes in $(\tilde{m}_t^B, \tilde{d}_t^B)$ from all input frames to complete the semantic segmentation map $\widehat{m}_{T+F}^B$.

**Losses.** To train the background refinement model, we use a cross-entropy loss applied at pixels which do not correspond to foreground objects in the target frame. This encourages the output of the refinement network to match the ground truth semantic segmentation at each pixel. We formalize this in appendix Sec. E.2.

#### 3.1.3 Aggregation:

This step combines foreground instance segmentations $\widehat{m}_{T+F}^i$, classes $c^i$, depths $d_{T+F}^i$ and background semantic prediction $\widehat{m}_{T+F}^B$ into the final future panoptic segmentation $S_{T+F}$. For simplicity, we assume that all foreground objects are located in front of all background components. We found this to be valid in most cases. Thus, to combine the foreground and background, we 'paste' foreground instances in order of decreasing predicted instance depth on top of the background. This approach is presented visually in Fig. 2, right, and described in more detail by Alg. 1 in the appendix.

#### 3.1.4 Egomotion estimation:

A large contributor to the observed motion of a scene is the movement of the recording camera. Properly modeling this movement is critical for accurate results. Here, we consider two scenarios: 1) an 'active' scenario where the model has access to the planned motion of an autonomous agent; 2) a 'passive' scenario in which the camera is controlled by an external agent and hence the model is not provided with the future motion.

In the active scenario, we use the speed and yaw rate of the camera from the dataset, which we process into the forms required by the foreground and background models. See appendix Sec. B for more details.

In the passive scenario, we use a GRU to predict the future camera motion as a function of its past movement. More formally and as sketched in Fig. 2(left),

$$h_{o,t+1} = \text{GRU}_{\text{cam}}\left(\widehat{o}_t, h_{o,t}\right) \text{ and } \widehat{o}_{t+1} = f_{\text{cam}}(h_{o,t+1}), \quad (10)$$

where $f_{\text{cam}}$ is a multilayer perceptron. For input time steps, i.e., $t \in \{1, \ldots, T\}$, we use known camera motion $o_t$ as model input. For future time steps, i.e., $t \in \{T+1, \ldots, T+F\}$, we use predicted camera motion $\widehat{o}_t$ as input.

| | Short term: $\Delta t = 3$ | | | | | | | | | Mid term: $\Delta t = 9$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | All | | | Things | | | Stuff | | | All | | | Things | | | Stuff | | |
| | PQ | SQ | RQ | PQ | SQ | RQ | PQ | SQ | RQ | PQ | SQ | RQ | PQ | SQ | RQ | PQ | SQ | RQ |
| Panoptic Deeplab (Oracle)† | 60.3 | 81.5 | 72.9 | 51.1 | 80.5 | 63.5 | 67.0 | 82.3 | 79.7 | 60.3 | 81.5 | 72.9 | 51.1 | 80.5 | 63.5 | 67.0 | 82.3 | 79.7 |
| Panoptic Deeplab (Last seen frame) | 32.7 | 71.3 | 42.7 | 22.1 | 68.4 | 30.8 | 40.4 | 73.3 | 51.4 | 22.4 | 68.5 | 30.4 | 10.7 | 65.1 | 16.0 | 31.0 | 71.0 | 40.9 |
| Flow | 41.4 | 73.4 | 53.4 | 30.6 | 70.6 | 42.0 | 49.3 | 75.4 | 61.8 | 25.9 | 69.5 | 34.6 | 13.4 | 67.1 | 19.3 | 35.0 | 71.3 | 45.7 |
| Hybrid [59] (bg) and [43] (fg) | 43.2 | 74.1 | 55.1 | 35.9 | 72.4 | 48.3 | 48.5 | 75.3 | 60.1 | 29.7 | 69.1 | 39.4 | 19.7 | 66.8 | 28.0 | 37.0 | 70.8 | 47.7 |
| **Ours** | **49.0** | **74.9** | **63.3** | **40.1** | **72.5** | **54.6** | **55.5** | **76.7** | **69.5** | **36.3** | **71.3** | **47.8** | **25.9** | **69.0** | **36.2** | **43.9** | **72.9** | **56.2** |

Table 1. **Panoptic segmentation forecasting evaluated on the Cityscapes validation set**. † has access to the RGB frame at $\Delta t$. Higher is better for all metrics.

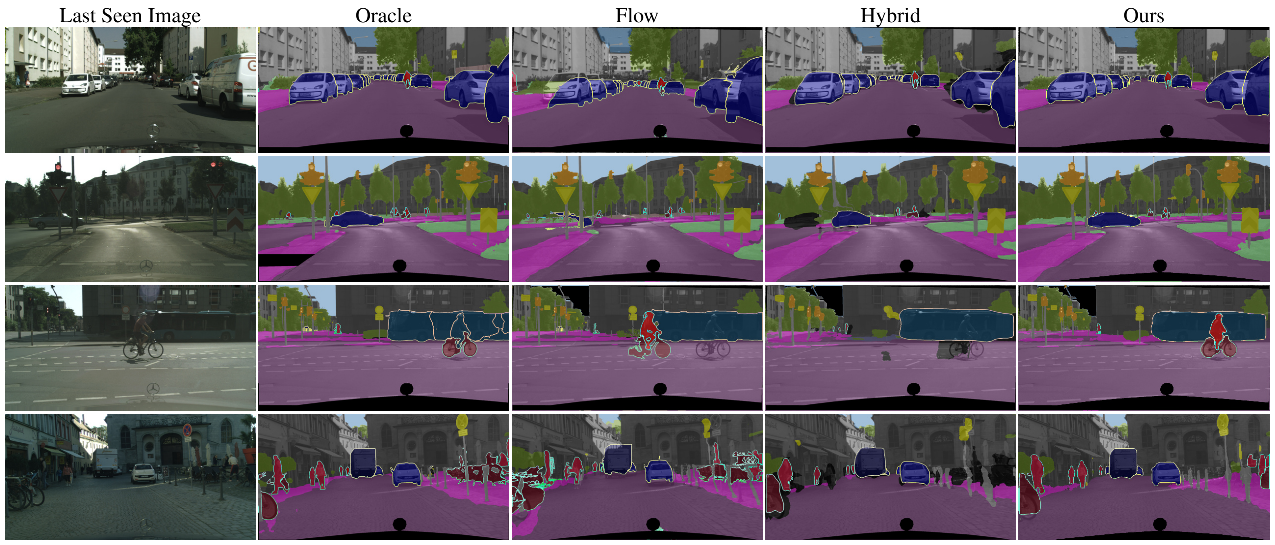


Figure 4. **Mid-term panoptic segmentation forecasts on Cityscapes.** Compared to *Hybrid*, our approach produces more well-defined silhouettes for instance classes (see the cars in the 1st row or the pedestrians in the 4th row), and handles instances with large motion much better *Hybrid* – the car in the 2nd row is not predicted to have moved sufficiently; the cyclist in the 3rd row is not predicted at all. Since *Flow* does not model instance-level trajectory, the 'things' are no longer intact in the forecasts.

## 4. Evaluation

We establish the first results for the task of panoptic segmentation forecasting by comparing our developed approach to several baselines. We also provide ablations to demonstrate the importance of our modeling decisions. We additionally evaluate on the tasks of semantic segmentation forecasting and instance segmentation forecasting to compare our method to prior work on established tasks.

**Data:** To evaluate panoptic segmentation forecasting, we need a dataset which contains both semantic and instance information as well as entire video sequences that lead up to the annotated frames. Cityscapes [12] fulfills these requirements and has been used in prior work for semantic and instance forecasting. This dataset consists of 5000 sequences of 30 frames each, spanning approximately 1.8 seconds. The data were recorded from a car driving in urban scenarios, and semantic and instance annotations are provided for the 20th frame in each sequence. Following standard practice for prior work in forecasting segmentations [44, 43, 59, 53], all experiments presented here are run on the validation data; a limited set of evaluations on test data are presented in appendix Sec. G.

To match prior work [44, 43, 53], we use every third frame as input and evaluate two different scenarios: short-term forecasting looks 3 frames ($\sim$0.18s) and medium-term forecasting looks 9 frames ($\sim$0.53s) into the future. All metrics are computed on the 20th frame of the sequence. We use an input length of $T = 3$. We hence use frames 11, 14, and 17 as input for short-term experiments and frames 5, 8, and 11 as input for medium-term experiments.

### 4.1. Panoptic Segmentation Forecasting

**Metrics.** We compare all approaches using metrics introduced in prior work [32] on panoptic segmentation. These metrics require to first compute matches between predicted segments and ground truth segments. A match between a predicted segment and a ground truth segment of the same class is a true positive if their intersection over union (IoU) is larger than 0.5. Using these matches, three metrics are considered: *segmentation quality* (SQ), which is the average IoU of true positive matched segments, *recognition quality* (RQ), which is the F1 score computed over matches, and *panoptic quality* (PQ), which is the product of SQ and RQ. All of these metrics are computed per class and then averaged to compute the final score.

**Baselines.** To compare our approach against baselines on the novel task of panoptic segmentation forecasting, we use:

|  | $\Delta t = 3$ |  |  | $\Delta t = 9$ |  |  |
|---|---|---|---|---|---|---|
|  | PQ | SQ | RQ | PQ | SQ | RQ |
| **Ours** | **49.0** | 74.9 | **63.3** | **36.3** | 71.3 | **47.8** |
| 1) w/ Hybrid bg [59] | 45.0 | 74.1 | 57.9 | 32.4 | 70.1 | 42.9 |
| 2) w/ Hybrid fg [43] | 47.3 | 74.8 | 60.7 | 33.4 | 70.4 | 43.9 |
| 3) w/ linear instance motion | 40.2 | 73.7 | 52.1 | 27.9 | 70.1 | 36.6 |
| 4) fg w/o odometry | 48.8 | 75.1 | 62.8 | 35.3 | 71.1 | 46.5 |
| 5) w/ ORB-SLAM odometry | 48.6 | 75.0 | 62.5 | 36.1 | 71.3 | 47.5 |
| 6) w/ SGM depth | 48.8 | **75.2** | 62.8 | 36.1 | **71.4** | 47.3 |
| 7) w/ monocular depth | 47.5 | 74.8 | 61.0 | 34.8 | 70.9 | 45.8 |
| w/ ground truth future odometry | 49.4 | 75.2 | 63.5 | 39.4 | 72.1 | 51.6 |

Table 2. **Validating our design choices** using Cityscapes. Higher is better for all metrics. All approaches use predicted future odometry unless otherwise specified.

*Panoptic Deeplab (Oracle)*: we apply the Panoptic Deeplab model [9] to *analyze* the target frame. This represents an upper bound on performance, as it has direct access to future information.

*Panoptic Deeplab (Last seen frame)*: we apply the same Panoptic Deeplab model to the most recently observed frame. This represents a model that assumes no camera or instance motion.

*Flow*: Warp the panoptic segmentation analyzed at the last observed frame using optical flow [27] computed from the last two observed frames.

*Hybrid Semantic/Instance Forecasting*: We fuse predictions made by a semantic segmentation forecasting model [59] and an instance segmentation forecasting model [43] to create a panoptic segmentation for the target frame.

**Results.** The results for all models on the panoptic segmentation forecasting task are presented in Tab. 1. We outperform all non-oracle approaches on the PQ, SQ, and RQ metrics for both short-term and mid-term settings. The improvements to PQ and RQ show that our model better captures the motion of all scene components, including static background 'stuff' regions and dynamic 'things.' In addition, the improvements to SQ imply that the per-pixel quality of true positive matches are not degraded. The *Flow* model performs worse than either *Hybrid* or our approach, which demonstrates that a simple linear extrapolation of per-pixel input motion is not sufficient to capture the scene and object movement. The fact that the gap between ours and *Hybrid* on 'things' PQ grows between the short- and mid-term settings shows the strength of our foreground model (Sec. 3.1.1) on anticipating object motion at longer time spans. Fig. 4 compares results to baselines. Our approach produces better defined object silhouettes and handles large motion better than the baselines.

**Ablations.** Tab. 2 shows results for ablation experiments which analyze the impact of our modeling choices: 1) *w/Hybrid bg* uses our foreground model, but replaces our background model with the one from [59]; 2) *w/Hybrid fg* uses our background model, but replaces our foreground model with the one from [43]; 3) *w/linear instance motion* replaces the foreground forecasting model with a simple model assuming linear instance motion and no mask appearance change; 4) *fg w/o odometry* does not use odometry as input to the foreground model; 5) *w/ ORB-SLAM odometry* uses input odometry obtained from [6]; 6) *w/ SGM depth* uses depths obtained from SGM [24] provided by [12] as input to the model; and 7) *w/ monocular depth* uses a monocular depth prediction model [19], finetuned on Cityscapes, to obtain input depth. Ablations 1) and 2) show that our improved model performance is due to the strength of both our foreground and background components. Ablation 3) shows that joint modeling of instance motion and appearance mask is key to success. 4) shows that odometry inputs help the model predict foreground locations better, and 5) demonstrates our method works well with odometry computed directly from input images. 6) and 7) suggest that our approach benefits from more accurate depth prediction, but it also works well with depth inputs obtained using single-frame methods.

| Accuracy (mIoU) | Short term: $\Delta t = 3$ All | MO | Mid term: $\Delta t = 9$ All | MO |
|---|---|---|---|---|
| Oracle | 80.6 | 81.7 | 80.6 | 81.7 |
| Copy last | 59.1 | 55.0 | 42.4 | 33.4 |
| 3Dconv-F2F [10] | 57.0 | / | 40.8 | / |
| Dil10-S2S [44] | 59.4 | 55.3 | 47.8 | 40.8 |
| LSTM S2S [48] | 60.1 | / | / | / |
| Bayesian S2S [4] | 65.1 | / | 51.2 | / |
| DeformF2F [52] | 65.5 | 63.8 | 53.6 | 49.9 |
| LSTM M2M [59] | 67.1 | 65.1 | 51.5 | 46.3 |
| F2MF [53] | **69.6** | **67.7** | 57.9 | **54.6** |
| **Ours** | 67.6 | 60.8 | **58.1** | 52.1 |

Table 3. **Semantic forecasting results on the Cityscapes validation dataset**. Baseline numbers, besides oracle and copy last, are from [53]. Higher is better for all metrics. Our model exploits stereo and odometry, which are provided by typical autonomous vehicle setups and are included in Cityscapes.

## 4.2. Semantic Segmentation Forecasting

For a comprehensive comparison, we also assess our approach on the task of semantic segmentation forecasting. This task asks to anticipate the correct semantic class per pixel for the target frame. Unlike the panoptic segmentation evaluation, this task doesn't care about instances, i.e., good performance only depends on the ability to anticipate the correct semantic class for each pixel. We obtain semantic segmentation outputs from our model by discarding instance information and only retaining the semantics.

**Metrics.** Future semantic segmentation is evaluated using intersection over union (IoU) of predictions compared to the ground truth, which are computed per class and averaged over classes. We additionally present an IoU score which is computed by averaging over 'things' classes only (MO).

**Baselines.** We compare to a number of recent works which forecast semantic segmentations. Many of these approaches

| | Short term: $\Delta t = 3$ | | Mid term: $\Delta t = 9$ | |
|---|---|---|---|---|
| | AP | AP50 | AP | AP50 |
| Oracle | 34.6 | 57.4 | 34.6 | 57.4 |
| Last seen frame | 8.9 | 21.3 | 1.7 | 6.6 |
| F2F [43] | **19.4** | **39.9** | 7.7 | 19.4 |
| Ours | 17.8 | 38.4 | **10.0** | **22.3** |

Table 4. **Instance segmentation forecasting on the Cityscapes validation dataset.** Higher is better for all metrics.

anticipate the features of a future scene [44, 48, 4, 52, 10, 53]. LSTM M2M [59] anticipates the optical flow between the most recent frame and the target with a warping function transforming input semantics. Different from these, we decompose the prediction into feature predictions for each individual instance as well as a transformation of background semantics before combining. Additionally, these approaches do not use depth inputs and all except Bayesian S2S [4] do not use egomotion as input.

**Results.** The results for this task are given in Tab. 3. We outperform most models on standard IoU as well as MO IoU. Unlike all other baselines, our model is able to produce instance-level predictions for moving object classes, which is a more challenging objective.

### 4.3. Instance Segmentation Forecasting

We also evaluate on instance segmentation forecasting, which only focuses on the 'things' classes within Cityscapes. Future instance segmentation can be obtained from our model by disregarding all pixels corresponding to 'stuff' classes from the panoptic forecasting output.

**Metrics.** Instance segmentation is evaluated using two metrics [12]: 1) Average Precision (AP) first averages over a number of overlapping thresholds required for matches to count as true positives and is then averaged across classes; 2) AP50 is the average precision computed using an overlap threshold of $0.5$ which is then averaged across classes.

**Baselines.** There is very little prior work on instance segmentation forecasting. We compare to Luc *et al.* [43], who train a model to predict the features of the entire future scene using a convolutional model and obtain final instances by running these predicted features through the prediction heads of MaskR-CNN. Instead, our approach predicts an individual set of features for each instance found in the scene.

**Results.** Tab. 4 presents the results. We outperform prior work in the mid-term setting. This indicates that modeling trajectory of individual instances has a higher potential on forecasting tasks. Since we use the same model created by Luc *et al.* [43] as the 'foreground' component of the *Hybrid* baseline (Sec. 4.1), Fig. 4 shows visual comparisons between these approaches. Again, our method gives higher-detailed instance contours and models objects with larger motion more accurately. Moreover, in some cases, F2F "deletes" some instances from the scene (such as the cyclist in row 3).

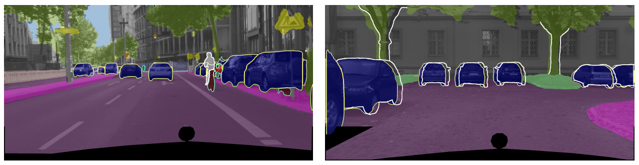


Figure 5. **Failure cases.** Left: the cyclist highlighted in white was missed by instance detection. Right: mispredicted odometry leads to misalignment between the forecast and the target image (the outlines of objects in the target image are shown in white).

### 4.4. Introspection

Why does our approach anticipate higher-fidelity instances than prior approaches? Many of these works attempt to predict future scenes by anticipating what a fixed-size feature tensor for the entire image will look like – this is true for both semantic segmentation forecasting [44, 10, 53] and instance segmentation forecasting [43]. Note, this conflates camera motion, which objects are present in a scene, how these objects move, and how the appearance of objects and background components change as a function of the scene motion. This increases the complexity of the prediction. Instead, our method decomposes these components into individual parts: the foreground model anticipates how each object moves and how its appearance changes as a function of this motion; the background model captures how static scene components appear when the camera moves; and the odometry model anticipates likely future motion based on past input. Modeling each of these separately simplifies individual prediction. Additionally, we predict separate features for every individual instance, so its size scales with the number of instances present in a scene, while past approaches [43] use a fixed size representation regardless of the complexity of the scene.

The performance of our approach is hampered in some cases by failures in instance detection and tracking (examples in Fig. 5). At the moment, our model cannot properly recover from situations where the input is noisy. That being said, our approach immediately benefits from improvements in the areas of instance detection and tracking, which are very active fields of research [2, 3].

## 5. Conclusions

We introduced the novel task 'panoptic segmentation forecasting.' It requires to anticipate a per-pixel instance-level segmentation of 'stuff' and 'things' for an unobserved future frame given as input a sequence of past frames. To solve this task, we developed a model which anticipates trajectory and appearance of 'things' and by reprojecting input semantics for 'stuff.' We demonstrated that the method outperforms compelling baselines on panoptic, semantic and instance segmentation forecasting.

**Acknowledgements:** This work is supported in part by NSF under Grant #1718221, 2008387, 2045586, MRI #1725729, and NIFA award 2020-67021-32799.

## References

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *PAMI*, 2017.

[2] P. Bergmann, T. Meinhardt, and L. Leal-Taixe. Tracking without bells and whistles. In *ICCV*, 2019.

[3] G. Bertasius and L. Torresani. Classifying, segmenting, and tracking object instances in video with mask propagation. In *CVPR*, 2020.

[4] A. Bhattacharyya, M. Fritz, and B. Schiele. Bayesian prediction of future street scenes using synthetic likelihoods. *ICLR*, 2019.

[5] H. Caesar, J. Uijlings, and V. Ferrari. Coco-stuff: Thing and stuff classes in context. In *CVPR*, 2018.

[6] C. Campos, R. Elvira, J. J. Gómez, J. M. M. Montiel, and J. D. Tardós. ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM. *arXiv preprint arXiv:2007.11898*, 2020.

[7] P. Chao, C.-Y. Kao, Y.-S. Ruan, C.-H. Huang, and Y.-L. Lin. Hardnet: A low memory traffic network. In *ICCV*, 2019.

[8] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. *arXiv:1412.7062*, 2014.

[9] B. Cheng, M. D. Collins, Y. Zhu, T. Liu, T. S. Huang, H. Adam, and L.-C. Chen. Panoptic-DeepLab: A simple, strong, and fast baseline for bottom-up panoptic segmentation. In *CVPR*, 2020.

[10] H.-K. Chiu, E. Adeli, and J. C. Niebles. Segmenting the future. *IEEE Robotics and Automation Letters*, 2020.

[11] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv:1406.1078*, 2014.

[12] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[13] C. Couprie, P. Luc, and J. Verbeek. Joint future semantic and instance segmentation prediction. In *ECCV workshops*, 2018.

[14] K. J. W. Craik. *The Nature of Explanation*. Cambridge University Press, 1943.

[15] Q. Dai, V. Patil, S. Hecker, D. Dai, L. Van Gool, and K. Schindler. Self-supervised object motion and depth estimation from video. In *CVPR Workshops*, 2020.

[16] S. Ehrhardt, O. Groth, A. Monszpart, M. Engelcke, I. Posner, N. Mitra, and A. Vedaldi. RELATE: Physically plausible multi-object scene synthesis using structured latent spaces. In *NeurIPS*, 2020.

[17] H. Gao, H. Xu, Q.-Z. Cai, R. Wang, F. Yu, and T. Darrell. Disentangling propagation and generation for video prediction. In *ICCV*, 2019.

[18] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *CVPR*, 2012.

[19] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow. Digging into self-supervised monocular depth estimation. In *ICCV*, 2019.

[20] C. Graber and A. Schwing. Dynamic neural relational inference. In *CVPR*, 2020.

[21] X. Gu, Z. Fan, S. Zhu, Z. Dai, F. Tan, and P. Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *CVPR*, 2020.

[22] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *ICCV*, 2017.

[23] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *ECCV*, 2008.

[24] H. Hirschmuller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *CVPR*, 2005.

[25] L. Hoyer, P. Kesper, A. Khoreva, and V. Fischer. Short-term prediction and multi-camera fusion on semantic grids. In *ICCV Workshops*, 2019.

[26] J.-T. Hsieh, B. Liu, D.-A. Huang, L. Fei-Fei, and J. C. Niebles. Learning to decompose and disentangle representations for video prediction. In *NeurIPS*, 2018.

[27] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, 2017.

[28] X. Jin, H. Xiao, X. Shen, J. Yang, Z. Lin, Y. Chen, Z. Jie, J. Feng, and S. Yan. Predicting scene parsing and motion dynamics in the future. In *NeurIPS*, 2017.

[29] A. Kendall, Y. Gal, and R. Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, 2018.

[30] D. Kim, S. Woo, J.-Y. Lee, and I. S. Kweon. Video panoptic segmentation. In *CVPR*, 2020.

[31] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

[32] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár. Panoptic segmentation. In *CVPR*, 2019.

[33] I. Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *CVPR*, 2017.

[34] A. Kosiorek, H. Kim, Y. W. Teh, and I. Posner. Sequential attend, infer, repeat: Generative modelling of moving objects. In *NeurIPS*, 2018.

[35] J. Li, A. Raventos, A. Bhargava, T. Tagawa, and A. Gaidon. Learning to fuse things and stuff. *arXiv:1812.01192*, 2018.

[36] Q. Li, A. Arnab, and P. H. Torr. Weakly-and semi-supervised panoptic segmentation. In *ECCV*, 2018.

[37] Y. Li, X. Chen, Z. Zhu, L. Xie, G. Huang, D. Du, and X. Wang. Attention-guided unified network for panoptic segmentation. In *CVPR*, 2019.

[38] X. Liang, L. Lee, W. Dai, and E. P. Xing. Dual motion GAN for future-flow embedded video prediction. In *ICCV*, 2017.

[39] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[40] C. Liu, L.-C. Chen, F. Schroff, H. Adam, W. Hua, A. L. Yuille, and L. Fei-Fei. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In *CVPR*, 2019.

[41] R. R. Llinás. *I of the vortex: from neurons to self*. MIT Press, 2001.

[42] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[43] P. Luc, C. Couprie, Y. LeCun, and J. Verbeek. Predicting future instance segmentation by forecasting convolutional features. In *ECCV*, 2018.

[44] P. Luc, N. Neverova, C. Couprie, J. Verbeek, and Y. LeCun. Predicting deeper into the future of semantic segmentation. In *ICCV*, 2017.

[45] K. Mangalam, E. Adeli, K.-H. Lee, A. Gaidon, and J. C. Niebles. Disentangling human dynamics for pedestrian locomotion forecasting with noisy supervision. In *WACV*, 2020.

[46] J. Martinez, M. J. Black, and J. Romero. On human motion prediction using recurrent neural networks. In *CVPR*, 2017.

[47] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016.

[48] S. S. Nabavi, M. Rochan, and Y. Wang. Future semantic segmentation with convolutional LSTM. In *BMVC*, 2018.

[49] L. Porzi, S. Rota Bulò, A. Colovic, and P. Kontschieder. Seamless scene segmentation. In *CVPR*, 2019.

[50] X. Qi, Z. Liu, Q. Chen, and J. Jia. 3D motion decomposition for RGBD future dynamic scene synthesis. In *CVPR*, 2019.

[51] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 2015.

[52] J. Šarić, M. Oršić, T. Antunović, S. Vražić, and S. Šegvić. Single level feature-to-feature forecasting with deformable convolutions. In *German Conference on Pattern Recognition*, 2019.

[53] J. Saric, M. Orsic, T. Antunovic, S. Vrazic, and S. Segvic. Warp to the future: Joint forecasting of features and feature motion. In *CVPR*, 2020.

[54] F. Schroff, A. Criminisi, and A. Zisserman. Object class segmentation using Random Forests. In *BMVC*, 2008.

[55] O. Sener and V. Koltun. Multi-task learning as multi-objective optimization. In *NeurIPS*, 2018.

[56] E. Sharon, A. Brandt, and R. Basri. Segmentation and boundary detection using multiscale intensity measurements. In *CVPR*, 2001.

[57] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *CVPR*, 2008.

[58] J. Sun, J. Xie, J.-F. Hu, Z. Lin, J. Lai, W. Zeng, and W.-s. Zheng. Predicting future instance segmentation with contextual pyramid convLSTMs. In *ACM International Conference on Multimedia*, 2019.

[59] A. Terwilliger, G. Brazil, and X. Liu. Recurrent flow-guided semantic forecasting. In *WACV*, 2019.

[60] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. MIT, 2005.

[61] E. Valassakis. Future object segmentation for complex correlated motions. Master's thesis, UCL, 2018.

[62] S. Vora, R. Mahjourian, S. Pirk, and A. Angelova. Future semantic segmentation using 3D structure. *arXiv:1811.11358*, 2018.

[63] J. Walker, C. Doersch, A. Gupta, and M. Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *ECCV*, 2016.

[64] J. Watson, M. Firman, G. Brostow, and D. Turmukhambetov. Self-supervised monocular depth hints. In *ICCV*, 2019.

[65] N. Wojke, A. Bewley, and D. Paulus. Simple online and realtime tracking with a deep association metric. In *ICIP*, 2017.

[66] Y. Wu, R. Gao, J. Park, and Q. Chen. Future video synthesis with object motion prediction. In *CVPR*, 2020.

[67] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NIPS*, 2015.

[68] Y. Xiong, R. Liao, H. Zhao, R. Hu, M. Bai, E. Yumer, and R. Urtasun. UPSNet: A unified panoptic segmentation network. In *CVPR*, 2019.

[69] J. Xu, B. Ni, Z. Li, S. Cheng, and X. Yang. Structure preserving video prediction. In *CVPR*, 2018.

[70] Y. Ye, M. Singh, A. Gupta, and S. Tulsiani. Compositional video prediction. In *ICCV*, 2019.

[71] R. Yeh, A. G. Schwing, J. Huang, and K. Murphy. Diverse Generation for Multi-agent Sports Games. In *Proc. CVPR*, 2019.

[72] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv:1511.07122*, 2015.

[73] Y. Zhu, K. Sapra, F. A. Reda, K. J. Shih, S. Newsam, A. Tao, and B. Catanzaro. Improving semantic segmentation via video propagation and label relaxation. In *CVPR*, 2019.