Abstract

Annotating videos with object segmentation masks typically involves a two stage procedure of drawing polygons per object instance for all the frames and then linking them through time. While simple, this is a very tedious, time consuming and expensive process, making the creation of accurate annotations at scale only possible for well-funded labs. What if we were able to segment an object in the full video with only a single click? This will enable video segmentation at scale with a very low budget opening the door to many applications. Towards this goal, in this paper we propose a bottom up approach where given a single click for each object in a video, we obtain the segmentation masks of these objects in the full video. In particular, we construct a correlation volume that assigns each pixel in a target frame to either one of the objects in the reference frame or the background. We then refine this correlation volume via a recurrent attention module and decode the final segmentation.

To evaluate the performance, we label the popular and challenging Cityscapes dataset with video object segmentations. Results on this new CityscapesVideo dataset show that our approach outperforms all the baselines in this challenging setting.

1. Introduction

Video object segmentation aims to identify all countable objects in the video and produce a masklet, i.e. a sequence of masks, for each of them.

By detecting the pixels of unique objects in space and time, we can have a better understanding of the scene, learn a better representation, and even design embodied agents that are capable of interacting with the environment compliantly. Most recent approaches, however, rely on deep neural networks that are extremely data-hungry, necessitating large-scale datasets.

Unfortunately, annotating videos with object segmentation masks is very cumbersome. It requires annotators to painstakingly analyze each video frame, manually delineate the objects from the background, and then associate them across time.

One potential way to speed up the labeling process is to leverage existing video segmentation approaches to produce initial masklets and then pass it to humans for refinement. In order to track the objects across time, these methods usually assume each object mask is given in the first frame (manually annotated). While this assumption can be further relaxed to bounding boxes [11, 1] or extreme points [61, 83], it still requires the annotators to spent on average 7–35 seconds per object [68]. One thus ponders: can we further reduce the amount of human intervention?

In this paper we look at the problem of segmenting full videos by requiring only a single click for each object that appears in the video. The single click setting has several key advantages: (i) the click does not have to be an extreme point, it can be an arbitrary point so long as it lies within the object, significantly relieving the cognitive load for annotation. (ii) A click is remarkably easy and fast to annotate, allowing for video annotation at scale. At the same time, this is a very challenging task since we have to map a single point provided by the annotator in a video to the masklet of an object that can undergo various deformations in shape and appearance with possible occlusions in a potentially crowded scene.

With these intuitions in mind, we present VideoClick, a video object segmentation model that takes as input one sin-
single click per object and outputs the corresponding masklets for the full video. Our model is conceptually inspired by the state-of-the-art optical flow network RAFT [75]. Given a pair of consecutive images and a set of keypoints within the reference frame, we first construct a 3D correlation volume representing the similarities between the pixels within the target frame and the keypoints. We then refine the correlation volume with a novel recurrent attention module based on the extracted visual features. The mask can then be obtained by taking an arg max over the score of assigning each pixel in the target frame to one of the keypoints or the background. This approach is akin to clustering point features in the target frame to an instance prototype feature corresponding to annotator clicks. We apply this procedure iteratively to video frames by propagating the keypoints in an implicit fashion. This allows us to handle situations where the objects undergo drastic geometry changes over time (e.g., a vehicle may be split into multiple parts due to occlusion).

To validate the effectiveness of our approach, we label the popular and challenging Cityscapes dataset [16] with video object segmentations. We named this new dataset CityscapesVideo. Our dataset presents several unique challenges and has several key differences comparing to existing datasets: first, while most benchmarks annotate only a few unique objects per sequence (e.g., DAVIS [70]: ∼2.51 objects/sequence, YoutubeVOS [93]: ∼1.91 objects/sequence), we annotate all visible objects within the scene. This leads to on average 14.5 masklets per sequence, which is an order of magnitude larger than before. Second, previous datasets usually label objects with large size. However, in self-driving scenario, the camera can capture not only objects in vicinity but also extremely distant objects, such as those more than a hundred meters away. Furthermore, the sizes of different objects vary a lot. For instance, objects far from the ego-car are extremely small in the images. Finally, due to the (large) relative motion between the self-driving vehicle and the objects, the objects tend to undergo drastic appearance and geometry changes, which makes the task very difficult. In comparison to the recently introduced Kitti MOTS [80] self-driving dataset where only cars and pedestrians were annotated in a semi-automatic fashion, we annotate manually all the 8 categories of cityscapes leading to about 50× more masklets and 140× more sequences as described in Table 2. Moreover, while Kitti has captured data in only one city, CityscapesVideo benefits from the regional variation of the original Cityscapes dataset and as such provides a more diverse and challenging set of video sequences.

We evaluate our model and establish a baseline on this challenging dataset. In particular, we obtain a mIoU of 63.5 between our predicted masklets and the ground truth.

2. Related work

Instance segmentation: Modern instance segmentation methods can be categorized into two paradigms: top-down or bottom-up. For top-down approaches [17, 47, 28, 58, 8, 14, 41, 91, 13, 25, 37, 94, 50, 69, 10, 101, 22, 88], region proposals are predicted and a voting process is used to filter out confident proposals. Most recently, LevelSet R-CNN [33] combines the Mask R-CNN framework [28] with the classic variational level set method to segment the instances in a structured fashion. PolarMask [90] predicts the contour of an instance using polar coordinates. CondInst [76] employs dynamic instance-aware networks to eliminate the need for ROI cropping and feature alignment. Point-Set Anchors [86] uses a bounding box polygon as initialization from which they predict an offset for each point to bring it to the object boundary. SeaNet [12] proposes a supervised edge attention module in the mask head and a new branch to learn IoU scores for the bounding boxes.

For bottom-up approaches [52, 102, 103, 77, 7, 65, 23, 43, 4, 57, 40, 64], instances in a scene are segmented without an explicit object proposal. For example, PatchPerPix [59] predicts dense local shape descriptors and assembles them to predict the instance segmentation. [87] proposes a greedy algorithm for joint graph partitioning and labeling derived from the efficient Mutex Watershed partitioning algorithm. In [98], the authors adopt a variational loss to handle multiple instances in a permutation-invariant way.

Video segmentation: Video segmentation [99, 54, 81, 100, 21, 39, 35, 85, 34, 96, 67] has become a more and more popular research area. RVOS [79] proposes a recurrent model that is recurrent in both spatial and temporal domain. [80] introduces a new video instance segmentation dataset and extend Mask R-CNN to video by using 3D convolutions and an association embedding branch to track objects. In [53], the authors combine Mask R-CNN with a modified variational autoencoder to output video segmentation and tracking. MaskProp [6] adapts Mask R-CNN to video by adding a mask propagation branch that propagates instance masks from each video frame to one another. In [36], the authors propose a temporal aggregation network and a dynamic time evolving template matching mechanism. STEm-Seg [2] models a video clip as a single 3D spatio-temporal volume, then segments and tracks instances across space and time in a single stage. SipMask [9] uses a novel light weight spatial preservation module and a tracking branch to conduct single stage video instance segmentation.

Semi-automatic annotation: There has been significant efforts on speeding up the pixel-level image labeling process, such as incorporating the feedback from humans into
the model [26, 27, 72, 92, 48, 3, 38, 74, 18, 45, 30, 60, 101], or parameterizing the output in a way that is easy for annotators to adjust and refine [11, 1, 50, 56, 19, 84]. In [66, 63], annotators provide scribbles drawn on the image to iteratively refine each object mask across video frames. ScribbleBox [12] have annotators interactively correct both box tracks and segmentation masks for video object segmentation. [73] interactively annotate 3D object geometry using 2D scribbles. In [53], users provide an initial click to segment the main body of the object and then iteratively provide more points on mislabeled regions to refine the segmentation. Polygon-RNN series [11, 1, 56] take as input the ground truth bounding box and predict the object mask in the form of polygons so that the annotators or other models [50] can easily refine. DEXTR [62] adopts extreme points of instances as input to segment objects and show that this is much more efficient than leveraging bounding box. In [51], crosswalks, roads and lanes as polylines can be an efficient way to speed up annotation of such map elements.

**Point-based methods:** Extracting image features for a given set of points and directly operating on them has gradually drawn wider attention, due to its strong performance, as well as speed and memory advantage. PixelNet [5] extract hypercolumn features from sampled pixel coordinates to perform segmentation. PointRend [44] iteratively refines a coarse instance segmentation mask by making predictions on an adaptively selected set of finer points. CenterMask [82] performs instance segmentation by predicting a heat map of point locations from which a feature is extracted and processed to output a mask. Similarly, PointINS [71] use an instance aware convolution to perform instance segmentation using single point image features. Dense RepPoints [97] generates a set of initial points, refine them based on the sampled features, and then post-process to obtain the final mask. In [95], the authors tackle the task of multi-object tracking and segmentation by representing them as a set of randomly selected points.

### 3. Single Click Video Object Segmentation

We tackle the problem of single click video object segmentation. Let \( \{I_t \in \mathbb{R}^{3 \times H \times W}\}_{t=1}^{T} \) be a sequence of RGB images taken from a video. In our setting we assume the following workflow for the annotator. The annotator provides one and only one click per object in the first frame that it appears in the video snippet. Note that this click can be at any random point inside the object, preferably not close to its boundaries. We denote by \( P_t = \{p_{t,n}\}_{n=1}^{N_t} \) the click coordinates at time \( t \) and by \( N = \sum N_t \) the total number of clicks in the snippet. If the object disappears, say due to occlusion, the annotator clicks on it once again when it reappears in the video. Given these single clicks for the \( N \) objects, our goal is then to obtain their corresponding segmentation masks in the video, denoted by \( \{M_t \in \mathbb{N}^{H \times W}\}_{t=1}^{T} \). In particular, at a pixel position \( (i, j) \) at time \( t \) we have \( M_t(i, j) \in \{0, 1, \ldots, N\} \) where the label \( n \in \{1, \ldots, N\} \) corresponds to one of the \( N \) objects specified by the annotator and the label 0 represents the background.

We develop our model to operate on a pair of consecutive RGB images \( I_t \) and \( I_{t+1} \) and then later extend it to the full snippet \( \{I_t\}_{t=1}^{T} \). Specifically, suppose an annotator highlights with single point clicks \( P_t = \{p_{t,n}\}_{n=1}^{N_t} \) all the visible objects in \( I_t \). Given these point clicks, our goal is to obtain their corresponding mask \( M_{t+1} \) in \( I_{t+1} \). At a high level, our model has three steps as shown in Figure. 2: First we obtain deep spatio-temporal features on top of \( I_t \) and \( I_{t+1} \). Next, given the annotator point clicks, we sample instance features and fuse this with target image features to construct a correlation volume. This volume scores the compatibility of each pixel in \( I_{t+1} \) to the instances in \( I_t \) specified by the annotator point clicks or to the background. Finally, the correlation volume is refined for a number of iterations by a recurrent attention network. The correlation volume can be converted into a segmentation mask by taking a arg max
Figure 2. **VideoClick for single click video object segmentation:** Given a single click per object, our model samples instance features that are used to construct a correlation volume. This correlation volume assigns each feature pixel in the target frame to the corresponding instance feature in the reference frame or the background feature vector. Next, the cost volume is iteratively refined with the context feature using a recurrent attention mechanism to obtain the final segmentation.

over the score of these instances and the background assigned to each pixel in $I_{t+1}$. In the following, we describe each component of our model in detail.

**Deep Spatio-Temporal Features:** We input the images $I_t$ and $I_{t+1}$ to a siamese encoder-decoder network. The encoder is a 2D residual network [29], and the decoder is based on FPN [55] augmented with temporal connections to aggregate the features across time. We detail the exact decoder architecture in the supplemental material. At the end, we obtain down-sampled features $F_t$ and $F_{t+1} \in \mathbb{R}^{D \times H/4 \times W/4}$ corresponding to $I_t$ and $I_{t+1}$ respectively. These features will contain information about the object masks and also their association in time and will be used to construct a correlation volume between the pixels of $I_{t+1}$ and the keypoints of $N_t$ instances in $I_t$ or the background.

**Keypoint and Background Features:** Given the keypoints $P_t = \{p_{t,n}\}_{n=1}^{N_t}$ highlighted by the annotator, we extract using bilinear interpolation $N_t$ vectors of dimension $D$ from the feature map $F_t$. Each extracted feature vector captures the spatio-temporal semantic information about its corresponding instance specified by the annotator click. The goal is to match each pixel in $I_{t+1}$ to one of these $N_t$ instances. However, a pixel in $I_{t+1}$ could also correspond to the background which could include a new object appearing for the first time in $I_{t+1}$. As such, we simply define a global learnable $D$-dimensional feature vector corresponding to the background. Finally, we concatenate the $N_t$ object feature vectors and the background feature vector to create a matrix $E$ of dimensions $(N_t + 1) \times D$ which will be used in the correlation volume construction.

**Correlation Volume:** To obtain the segmentation masks in image $I_{t+1}$ corresponding to either the $N_t$ objects in $I_t$ or the background, we construct an initial correlation volume $C^0_{t+1} \in \mathbb{R}^{(N_t+1) \times H/4 \times W/4}$ where $C^0_{t+1}(n,i,j)$ scores the compatibility of feature pixel $(i,j)$ in $F_{t+1}$ with row $n$ of $E$ which is the feature vector of either the background or one of the $N_t$ instances in $I_t$. $C^0_{t+1}$ is constructed by taking the dot product between the rows of the matrix $E$ and each feature pixel in $F_{t+1}$ as follows:\footnote{We use a superscript to denote the initial cost volume and its refinement and a subscript for the video timesteps.}

$$C^0_{t+1}(n,i,j) = \sum_h E(n,h)F_{t+1}(h,i,j)$$

(1)

The correlation volume can then be converted into a segmentation mask by taking an $\arg \max$ along the first dimension of $C^0_{t+1}$:

$$M^0_{t+1}(i,j) = \arg \max_n C^0_{t+1}(n,i,j)$$

(2)
Note that if an object disappears in $I_{t+1}$, then its corresponding channel in $C_{i+1}$ would not have the highest score on any of the pixels and as such its label will not appear in the mask. In practice however, we could have spurious pixels in $I_{t+1}$ assigned to an occluded instance. As such, we consider an instance to be occluded if the area of its mask is less than a threshold obtained from the validation set.

\textbf{Correlation Volume Refinement:} Next, we proceed to refine the initial estimate $C_{i+1}$ through a recurrent attention module that depends on $C_{i+1}$ and the features extracted solely from $I_{t+1}$. In particular, we input the encoder features of $I_{t+1}$ to a new decoder to obtain features $F_{\text{context}} \in \mathbb{R}^{D \times H/4 \times W/4}$ which would have better localized information about the object boundaries in $I_{t+1}$ without being affected by the features of $I_t$. At a high level, at each refinement step, we find the feature channels from $F_{\text{context}}$ that are closest to a predicted instance mask captured in the correlation volume and use them to refine the correlation. Our refinement update rule is inspired by the self attention block of the transformers \cite{Lin2019} and proceeds as follows at each refinement timestep $t$:

First, we map the feature map $F_{\text{context}}$ to embedding tensors $F_{\text{key}}$ and $F_{\text{value}}$ using two residual blocks. Next, we convert the previous correlation volume $C_{i+1}^{t-1}$ to a probability tensor $S_{\tau}^{-1}$ by taking a softmax along its first dimension. Now, each channel of $S_{\tau}^{-1}$ represents the probability mask of the corresponding object or the background.

Then we compute an attention matrix $A_{\tau}^{-1} \in \mathbb{R}^{(N_i+1) \times D}$ between $S_{\tau}^{-1}$ and $F_{\text{key}}$:

$$A_{\tau}^{-1}(n,d) = \max_{dim=1} \sum_{i,j} S_{\tau}^{-1}(n,i,j) F_{\text{key}}(d,i,j)$$  \hspace{1cm} (3)

Here $A_{\tau}^{-1}(n,d)$ measures the compatibility of the probability mask of the $n$-th instance or the background and the $d$-th feature channel of $F_{\text{key}}$. Finally, we multiply $A_{\tau}^{-1}$ by $F_{\text{value}}$ to obtain a residual score that will be added to $C_{i+1}^{t-1}$.

$$C_{i+1}^{t} = C_{i+1}^{t-1} + A_{\tau}^{-1} F_{\text{value}}$$  \hspace{1cm} (4)

We repeat this update rule for a fixed number of timesteps.

\textbf{Extending to the full Snippet:} Next, we extend our model from operating on a pair of images to the full snippet. As before, denote the annotator clicks for new objects appearing at time $t$ by $P_t$. Also in contrast with the two frame case, we have in addition a correlation volume $C_t$ from the previous timestep. The next step is to obtain feature vectors corresponding to $P_t$ and $C_t$. For keypoints $P_t$, we use bilinear interpolation to extract keypoint features from feature map $F_t$. Next, we need to convert $C_t$ to a set of of keypoint features. First we obtain the mask of each instance from $C_t$. Then in order to remove potential mask outliers, we find the coordinates of the top 50\% highest scoring mask pixels for each instance. Finally, we average those points on $F_t$ to create a $D$-dimensional feature vector for each instance.

Now, we’ll explain the full snippet algorithm 1: Given previous correlation volume $C_t$ and new keypoints $P_t$, we apply the model on the same frame $I_t$ to obtain an updated correlation volume $C_{t+1}$. In particular, whereas $C_t$ gives the compatibility of each pixel in $I_t$ to instances appearing in previous frames $I_{t-1}$, the new correlation $C_{t+1}$ takes into account both $C_t$ and the new annotator clicks $P_t$. The segmentation mask $M_t$ is obtained from $C_t$. Finally, we supply the consecutive frames $I_t$ and $I_{t+1}$ and the keypoint features corresponding to $C_t$ to our model to obtain $C_{t+1}$ and the process continues.

\textbf{Learning:} We learn all the parameters of the model in an end-to-end fashion. In particular, from images $I_t$, $I_{t+1}$ and keypoints $P_t$ we obtain a sequence of refined correlation volumes $C_{t+1}$ corresponding to $I_{t+1}$. After applying softmax on the cost volumes, we can directly compare with the ground truth masks $M_{t+1}^{GT}$ using cross entropy. In practice, rather than directly comparing $C_{t+1}$ with $M_{t+1}^{GT}$, we sample using bilinear interpolation a fixed number of random points corresponding to the background, the object interior and around the object boundaries from $C_{t+1}$ and $M_{t+1}^{GT}$ and compute the cross entropy on this points. Our goal is to balance the the effect of background pixels as well as object of varying sizes. We also found that the auxiliary task of binary foreground/background prediction with an extra residual block from the features $F_t$ and $F_{t+1}$ improves the performance. Note that we do not use this branch at test time.
Table 3. Results on CityscapesVideo val and test sets: This table shows the multi object tracking and segmentation results given a single click. We report mean IOU for both the val and test set and also the mean IOU per class.

| Method                          | mIOU$_{val}$ | mIOU$_{test}$ | person | rider | car   | truck | bus  | train | mcycle | bicycle |
|---------------------------------|--------------|---------------|--------|-------|-------|-------|------|-------|--------|---------|
| Siamese Bounding Boxes          | 50.2         | 44.5          | 34.3   | 35.6  | 57.6  | 58.7  | 63.2 | 37.7  | 33.6   | 35.6    |
| Mask R-CNN + Key Point Matching | 46.6         | 44.5          | 43.1   | 41.8  | 59.5  | 50.2  | 59.7 | 33.5  | 35.3   | 32.9    |
| Mask R-CNN + Mask Matching      | 55.2         | 51.6          | 49.1   | 47.7  | 67.6  | 58.2  | 65.3 | 41.3  | 41.2   | 42.2    |
| Ours                            | 63.5         | 59.6          | 53.5   | 54.4  | 75.9  | 65.0  | 73.1 | 53.5  | 50.8   | 50.5    |

Table 4. Results on CityscapesVideo val and test full sequences: This table shows the multi object tracking and segmentation results given a single click reported on the full 30 frame sequence. We report mean IOU for both the val and test set and also the mean IOU per class. We show that our model can generalize on a different frame rate despite being trained only on 5 frame sequences.

| Method                          | mIOU$_{val}$ | mIOU$_{test}$ | person | rider | car   | truck | bus  | train | motorcycle | bicycle |
|---------------------------------|--------------|---------------|--------|-------|-------|-------|------|-------|-------------|---------|
| Ours Evaluate on the Full Snippet | 61.5         | 58.1          | 52.3   | 53.9  | 73.7  | 62.5  | 69.7 | 52.6  | 50.6        | 48.5    |
| Ours                            | 63.5         | 59.6          | 53.5   | 54.4  | 75.9  | 65.0  | 73.1 | 53.5  | 50.8        | 50.5    |

Table 5. We report our model results run on single images and compare with Mask R-CNN. Both models are given the annotator key points and have to predict a segmentation mask. We report the mIOU result across the entire dataset.

| Method                          | mIOU       |
|---------------------------------|------------|
| Mask R-CNN + Keypoint Mask Filter | 61.8       |
| Ours trained on Single Images   | 71.4       |

Table 4. Results on CityscapesVideo val and test full sequences: This table shows the multi object tracking and segmentation results given a single click reported on the full 30 frame sequence. We report mean IOU for both the val and test set and also the mean IOU per class. We show that our model can generalize on a different frame rate despite being trained only on 5 frame sequences.

4. Experiments

CityscapesVideo: In this paper, we introduce a new video instance segmentation dataset for autonomous driving built on top of the popular Cityscapes dataset. In particular, we annotate the 8 object categories (bicycle, bus, person, train, truck, motorcycle, car and rider) of Cityscapes instance segmentation task with instance segmentation tracklets for 3475 training and validation sequences of the dataset. In the original dataset, only the 20th frame of a 30 frame sequence is annotated. We re-annotate the 20th frame and annotate 4 more at regular intervals so that we obtain a sequence of 5 frames at 360 ms, specifically we annotate frames 1, 7, 13, 19 and 25 of each sequence. For this paper, we set aside 2 cities (weimar and zurich) in the training set as validation and consider the actual validation set to be the test set. This results in a total of 13530/1315/2500 frames for the train/val/test sets.

Implementation Details We train our model on the Cityscapes video dataset on pairs of images. Each pair could either be the same image or two consecutive images with the original 1024 × 2048 resolution. We also augment the training dataset by random horizontal flipping with equal probability. We train the model on 16 RTX5000 GPUs for 25 epochs with the Adam [42] optimizer with learning rate of 0.0001 and weight decay of 0.0001. We take the best model from evaluating on the validation set. For the backbone, we employ ResNet-101 [29] pretrained on Imagenet [46] and use our modified version of FPN [55] with random initialization. In order to fit our model operating on original image resolutions, we employ gradient check-pointing [15] during training.

Metrics For our metric, we compute the mIOU between the prediction mask volume \( \{ M_t \}_{t=1}^T \) and the corresponding ground truth mask volume. The volume has a size of \( N \times T \times H \times W \), where \( N \) is the total number of instances in the sequence and \( T \) is the number of time steps in the sequence. That is, each mask has its own 2D canvas that we plot it on. The benefit of such a metric is that it is a measure of both the segmentation and tracking quality of the model. Such a metric will also penalize both false negatives and false positives.

Baselines: We establish multiple competitive baselines for the task of single click video object segmentation. In particular, we consider the following baselines:

1. Inspired by [20, 24], we develop a top down baseline called Siamese Bounding Boxes that trained end to end for the leaderboard. We use the same siamese network to input two frames at a time. Given the annotator keypoint, the model extracts a point based feature and predicts a bounding box in the next frame from which features are extracted to predict a mask for each object. This center point of this mask is then used as key point for the next frame. We have a scoring head similar to the one in [28] to score whether the instance exists in the next image. That is, given the key point feature, we feed it to a mlp to predict a probability of existence in frame 2.

2. We train Mask R-CNN with the same backbone as ours on the new dataset. Note that since our model is class agnostic, we train Mask R-CNN in a class agnostic fashion. Next, we use the annotator provided instance keypoints in frame \( I_t \) to keep only the masks in bounding boxes with the highest confidence. Next, we use
the optical flow framework RAFT [75] to warp each instance mask from $I_t$ to $I_{t+1}$. We then average the warped mask coordinates to get new key points and repeat the above process to get the new masks. We call this baseline Mask R-CNN + Key Point Matching.

3. As another baseline with Mask R-CNN, after we warp the masks to the next frame, we greedily match the warped masks to the masks in frame $I_{t+1}$. We call this baseline Mask R-CNN + Mask Matching.

**Comparison with the Baselines:** Our results are given in Table. 3. We find that Mask R-CNN + Mask Matching to be a strong baseline as the RAFT flow is a strong estimation of the object movement across video frames. However, we observe that our model significantly outperforms the baselines; we outperform the Mask R-CNN + Mask Matching baseline by 8.3% and 8% in the validation and test sets.
Effect of the Refinement Branch: To understand the efficacy of the recurrent attention module for refining the initial correlation volume, we train a model without this branch. We obtain a mIOU of 61.6 on the validation set which is about 2 mIOU lower compared to the 63.5 mIOU of the model with the refinement branch.

Extra Decoder for the Refinement Branch: To obtain the context features that are passed to the refinement module, we have an extra decoder that is applied only on the second frame $I_{t+1}$ without any spatio-temporal context from $I_t$. Intuitively, these features would have better localized information about the object masks and boundaries in $I_{t+1}$. To test this, we train a model without the extra decoder where the recurrent refinement module takes features from the spatio-temporal features $F_{t+1}$ instead. We obtain a mIOU of 61.1 compared to the 63.5 of the full model on the validation set confirming the importance of this branch.

Annotator Point Selection In the single click video object segmentation setting, we require that the annotator click on any point inside the object. In this section, we evaluate our model when the annotator point is required to be the center of the object, more specifically, when the point is the one with the highest distance transform value to the boundaries. We observe the performance to be very similar at 63.4 mioU compared to 63.5 of our model. In other words, our model is robust to where the annotator clicks.

Frame Rate Generalization We trained and evaluated our model on the 5 frames of each Cityscapes sequence that have ground truth. These 5 frames are at 360 ms apart. Here we run the model on the full sequence of 30 frames which is at 60 ms apart to see how our model performs on longer sequences and at a different frame rate. Although we do not have ground truth for all the 30 frames, we can still apply the model to the full snippet and evaluate on the 5 frames. In Table 4 we see in fact the drop in performance to be minimal for both the validation and the test sets. This indicates that our model can generalize to a higher frame rate and also keep track of masklet for longer periods.

Our Model in Single Image Setting: Our model can also be used for single click image object segmentation. In particular, we train and evaluate our model on single images only. Here we have removed the complications of handling motion of the objects between two frames. We compare with Mask R-CNN with the same backbone where the automatically generated masks are filtered out based on the annotator provided keypoints. In Table 5 we observe that our model outperforms the baseline by about 10 mIOU points.

Qualitative Examples: In Figure 3 we showcase qualitative results of our model run on the validation set and compare it with the ground truth and the Mask R-CNN + Mask Flow + Matching baseline. Results are shown on the 5 frames which are 360ms apart. Compared to the baseline we see that our model results in higher resolution segmentation and better tracking.

Failure Mode: Our model sometimes showcases difficulty when an object occludes another which can cause the occluded object’s key point to switch onto the object causing the occlusion. We showcase an example of such a failure in Figure 4: In time step 2, the cyan car is about to pass over the green car in the background. In time step 3, it passes over but there is still a small area (top right) of the green car that is not fully occluded and we predict a green segmentation for that area. This prediction results in a new key point right on the border of both cars. At the next timestep, the model thinks that this keypoint on the border belongs to the cyan car as well causing its segmentation mask to be divided into two regions.

5. Conclusion

Annotating object instance masks in videos is an expensive process as it requires not only the object masks to be labeled but also instances must be tracked across time. To tackle this problem, we propose VideoClick, a bottom up multi object video segmentation method that takes in a single click for each object from an annotator and outputs the segmentation masks for an entire video. This is done by sampling point based features using the annotator key points and constructing a correlation volume that assigns each pixel in a target frame to one of these keypoints. This correlation volume is refined using a recurrent attention mechanism to output a more precise final segmentation. We demonstrate the effectiveness of our method on a new benchmark, CityscapesVideo, that we introduce in this
paper. We show competitive initial results with strong segmentation and tracking on this dataset.

References

[1] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In CVPR, 2018. 1, 3

[2] Ali Athar, Sabarinath Mahadevan, Aljosa Osep, Laura Leal-Taixé, and Bastian Leibe. Stem-seg: Spatio-temporal embeddings for instance segmentation in videos. In ECCV, 2020. 2

[3] J. Bai and X. Wu. Error-tolerant scribbles based interactive image segmentation. In CVPR, 2014. 3

[4] Min Bai and Raquel Urtasun. Deep watershed transform for instance segmentation. In CVPR, 2017. 2

[5] Aayush Bansal, Xinlei Chen, Bryan Russell, Abhinav Gupta, and Deva Ramanan. Pixelnet: Representation of the pixels, by the pixels, and for the pixels. arXiv, 2017. 3

[6] Gedas Bertasius and Lorenzo Torresani. Classifying, segmenting, and tracking object instances in video with mask propagation. In CVPR, 2020. 2

[7] Bert De Brabandere, Davy Neven, and Luc Van Gool. Semantic instance segmentation with a discriminative loss function. arXiv, 2017. 2

[8] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In CVPR, 2018. 2

[9] Jiale Cao, Rao Muhammad Anwer, Hisham Cholakkal, Fahad Shahbaz Khan, Yanwei Pang, and Ling Shao. Sipmask: Spatial information preservation for fast image and video instance segmentation. In ECCV, 2020. 2

[10] Jiale Cao, Hisham Cholakkal, Rao Muhammad Anwer, Fahad Shahbaz Khan, Yanwei Pang, and Ling Shao. D2det: Towards high quality object detection and instance segmentation. In CVPR, 2020. 2

[11] Lluis Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a polygon-rnn. In CVPR, 2017. 1, 3

[12] Bowen Chen, Huan Ling, Xiaohui Zeng, Jun Gao, Ziyue Du, and Rao Muhammad Anwer. Scribblebox: Interactive annotation framework for video object segmentation. In ECCV, 2020. 2, 3

[13] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. Hybrid task cascade for instance segmentation. In CVPR, 2019. 2

[14] Liang-Chieh Chen, Alexander Hermans, George Papandreou, Florian Schroff, Peng Wang, and Hartwig Adam. Masklab: Instance segmentation by refining object detection with semantic and direction features. 2018. 2

[15] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. CoRR, 2016. 6

[16] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In CVPR, 2016. 2

[17] Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. CVPR, 2015. 2

[18] Henghui Ding, Scott Cohen, Brian Price, and Xudong Jiang. Phraseclick: Toward achieving flexible interactive segmentation by phrase and click. In ECCV, 2020. 3

[19] Zihao Dong, Ruixun Zhang, and Xiuli Shao. Automatic annotation and segmentation of object instances with deep active curve network. IEEE Access, 2019. 3

[20] Kaimen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In ICCV, 2019. 6

[21] Kevin Duarte, Yogesh S. Rawat, and Mubarak Shah. 2

[22] Zhibo Fan, Jin-Gang Yu, Zhihao Liang, Jiarong Ou, Changxin Gao, Gui-Song Xia, and Yuanqing Li. Fgn: Fully guided network for few-shot instance segmentation. In CVPR, 2020. 2

[23] Alireza Fathi, Zbigniew Wojna, Vivek Rathod, Peng Wang, Hyun Oh Song, Sergio Guadarrama, and Kevin P. Murphy. Semantic instance segmentation via deep metric learning. arXiv, 2017. 2

[24] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In ICCV, 2017. 6

[25] Cheng-Yang Fu, Mykhailo Shvets, and Alexander C. Berg. Retinamask: Learning to predict masks improves state-of-the-art single-shot detection for free. arXiv, 2019. 2

[26] Varun Gulshan, Carsten Rother, Antonio Criminisi, Andrew Blake, and Andrew Zisserman. Semantic star convexity for interactive image segmentation. 2010. 3

[27] B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In ICCV, 2011. 3

[28] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In ICCV, 2017. 2, 6

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. CVPR, 2016. 4, 6, 12

[30] Yuk Heo, Yeong Jun Koh, and Chang-Su Kim. Interactive video object segmentation using global and local transfer modules. In ECCV, 2020. 2

[31] Namdar Homayounfar, Wei-Chiu Ma, Shrinidhi Kowshika Lakshmikanth, and Raquel Urtasun. Hierarchical recurrent attention networks for structured online maps. In CVPR, 2018. 3

[32] Namdar Homayounfar, Wei-Chiu Ma, Justin Liang, Xinyu Wu, Jack Fan, and Raquel Urtasun. Dagmapper: Learning to map by discovering lane topology. In ICCV, 2019. 3

[33] Namdar Homayounfar, Yuwen Xiong, Justin Liang, Wei-Chiu Ma, and Raquel Urtasun. Levelset r-cnn: A deep variational method for instance segmentation. In ECCV, 2020. 2

[34] Anthony Hu, Alex Kendall, and Roberto Cipolla. Learning a spatio-temporal embedding for video instance segmentation, 2019. 2
2017 davis challenge on video object segmentation. arXiv, 2017. 2

[71] Lu Qi, Xiangyu Zhang, Yingcong Chen, Yukang Chen, Jian Sun, and Jiaya Jia. Pointins: Point-based instance segmentation, 2020. 3

[72] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "grabcut" – interactive foreground extraction using iterated graph cuts. ACM TRANS. GRAPH, 2004. 3

[73] Tianchang Shen, Jun Gao, Amlan Kar, and Sanja Fidler. Interactive annotation of 3d object geometry using 2d scribbles. In ECCV, 2020. 3

[74] Konstantin Sofiuk, Ilia Petrov, Olga Barinova, and Anton Konushin. F-brs: Rethinking backpropagating refinement for interactive segmentation. In CVPR, 2020. 3

[75] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In ECCV, 2020. 2, 7

[76] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. In ECCV, 2020. 2

[77] Jonas Uhrig, Marius Cordts, Uwe Franke, and Thomas Brox. Pixel-level encoding and depth layering for instance-level semantic labeling. In GCPR, 2016. 2

[78] Ashish Vaswani, Noam Shazeer, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, L ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In NIPS, 2017. 5

[79] Carles Ventura, Miriam Bellver, Andreu Girbau, Amaia Salvador, Ferran Marques, and Xavier Giro-i Nieto. Rvos: End-to-end recurrent network for video object segmentation. In CVPR, 2019. 2

[80] Paul Voigtlaender, Michael Krause, Aljosa Osep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, and Bastian Leibe. 2, 3

[81] Wenguan Wang, Xiankai Lu, Jianbing Shen, David J. Cran- dall, and Ling Shao. Zero-shot video object segmentation via attentive graph neural networks. In ICCV, 2019. 2

[82] Yuqing Wang, Zhaoliang Xu, Hao Shen, Baoshan Cheng, and Bastian Leibe. Centermask: Single shot instance segmentation with point representation. In CVPR, 2020. 3

[83] Zhan Wang, David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Object instance annotation with deep extreme level set evolution. In CVPR, 2019. 1

[84] Zhan Wang, David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Object instance annotation with deep extreme level set evolution. In CVPR, 2019. 3

[85] Ziqin Wang, Jun Xu, Li Liu, Fan Zhu, and Ling Shao. Ranet: Ranking attention network for fast video object segmentation. In ICCV, 2019. 2

[86] Fangyun Wei, Xiaofei Xie, Hongyang Li, Jingdong Wang, and Stephen Lin. Point-set anchors for object detection, instance segmentation and pose estimation. In ECCV, 2020. 2

[87] Steffen Wolf, Yuyan Li, Constantin Pape, Alberto Bailoni, Anna Kreshuk, and Fred A. Hamprecht. Joint semantic instance segmentation on graphs with the semantic mutex watershed. In ECCV, 2020. 2

[88] Ruizheng Wu, Xin Tao, Yingcong Chen, Xiaoyong Shen, and Jiaya Jia. Particularity beyond commonality: Unpaired identity transfer with multiple references. In ECCV, 2020. 2

[89] Yuxin Wu and Kaiming He. Group normalization. In ECCV, 2018. 12

[90] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. Polarmask: Single shot instance segmentation with polar representation. arXiv, 2019. 2

[91] Yuwen Xiong, Renjie Liao, Hengshuang Zhao, Rui Hu, Min Bai, Ersin Yumer, and Raquel Urtasun. Upsnet: A unified panoptic segmentation network. CoRR, 2019. 2

[92] Ning Xu, Brian L. Price, Scott Cohen, Jimei Yang, and Thomas S. Huang. Deep interactive object selection. CoRR, 2016. 3

[93] Ning Xu, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. Youtube-vos: A large-scale video object segmentation benchmark, 2018. 2

[94] Wenqiang Xu, Haiyang Wang, Fubo Qi, and Cewu Lu. Explicit shape encoding for real-time instance segmentation. In ICCV, 2019. 2

[95] Zhenbo Xu, Wei Zhang, Xiao Tan, Wei Yang, Huan Huang, Shilei Wen, Errui Ding, and Huang. Segment as points for efficient online multi-object tracking and segmentation. In ECCV, 2020. 3

[96] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In ICCV, 2019. 2

[97] Ze Yang, Yinghao Xu, Han Xue, Zheng Zhang, Raquel Urtasun, Liwei Wang, Stephen Lin, and Han Hu. Dense rep-points: Representing visual objects with dense point sets. In ECCV, 2020. 3

[98] Jialin Yuan, Chao Chen, and Fuxin Li. Deep variational instance segmentation. In NeurIPS, 2020. 2

[99] Xiaohui Zeng, Renjie Liao, Li Gu, Yuwen Xiong, Sanja Fidler, and Raquel Urtasun. Dmm-net: Differentiable mask-matching network for video object segmentation. In ICCV, 2019. 2

[100] Lu Zhang, Zhe Lin, Jiaming Zhang, Huchuan Lu, and You He. Fast video object segmentation via dynamic targeting network. In ICCV, 2019. 2

[101] Rufeng Zhang, Zhi Tian, Chunhua Shen, Mingyu You, and Youliang Yan. Mask encoding for single shot instance segmentation. In CVPR, 2020. 2, 3

[102] Ziyu Zhang, Alexander G Schwing, Sanja Fidler, and Raquel Urtasun. Instance-level segmentation for autonomous driving with deep densely connected mrfs. In CVPR, 2016. 2

[103] Ziyu Zhang, Alexander G Schwing, Sanja Fidler, and Raquel Urtasun. Monocular object instance segmentation and depth ordering with cnns. In ICCV, 2015. 2
Appendix - VideoClick: Video Object Segmentation with a Single Click

In Section A of this appendix, we present the architectural details of the siamese network. In Section B, we showcase more qualitative examples.

Appendix A. Siamese Network Architecture

For the encoder of our siamese network, we use ResNet-101 [29], where we replace all the batch normalization layers with group normalization [89] with 32 groups each. For the decoder of the siamese network, we use a modified temporal version of FPN [55]. In particular, whereas the inner block of each FPN layer is just a simple $3 \times 3$ convolution, we use the architecture presented in Table. 6 instead before upsampling. Note that for temporal aggregation, we simply concatenate the features of two frames along the channel dimension and apply a $1 \times 1$ convolution. Also we use bilinear interpolation for upsampling instead of nearest neighbour. For the decoder of the refinement module, we only use the blocks until row 4 of Table. 6.

| Row | Type                  | # of Input Tensors | # Input Channels | # of Output Tensors | # Output Channels | Kernel Size |
|-----|-----------------------|--------------------|------------------|--------------------|-------------------|-------------|
| 1   | Conv2D + ReLU         | 2                  | Depends on resnet layer | 2                   | 256               | 3           |
| 2   | Conv2D + ReLU         | 2                  | 256              | 2                  | 256               | 3           |
| 3   | Conv2D + GN + ReLU    | 2                  | 256              | 2                  | 256               | 1           |
| 4   | Concat along channels | 2                  | 256              | 1                  | 512               | -           |
| 5   | Conv2D + ReLU         | 1                  | 512              | 1                  | 1024              | 1           |
| 6   | Conv2D + ReLU         | 1                  | 1024             | 1                  | 512               | 1           |
| 7   | Split along channels  | 1                  | 512              | 2                  | 256               | -           |
| 8   | Conv2D + GN + ReLU    | 2                  | 256              | 2                  | 256               | 1           |
| 9   | Add to row 2          | 2                  | 256              | 2                  | 256               | -           |

Table 6. Architecture of each decoder block. GN corresponds to group normalization with 32 groups.

Appendix B. Qualitative Examples

In the following figures, we showcase the inference results of our VideoClick model and the corresponding ground truth for various sequences in the validation set of the new CityscapesVideo dataset. Each sequence of Cityscapes is 1.8 s with 30 frames. We annotated frames 1, 7, 13, 19 and 25 with instance masklets. Here we are showing the results of our model being applied to this 5 frames only. In the accompanying video, we showcase the results of our model applied to the whole sequence.

For each sequence, the annotator point clicks are given by red points for new objects appearing in the frame. We observe that our model is able to infer accurately the masklets in the sequence from just a single click of the objects provided by the annotator.
Figure 5. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 6. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 7. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 8. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 9. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 10. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 11. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 12. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 13. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 14. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 15. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 16. We showcase qualitative segmentation results of our model on the Cityscapes Video validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 17. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 18. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 19. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 20. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 21. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 22. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 23. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.
Figure 24. We showcase qualitative segmentation results of our model on the CityscapesVideo validation set and compare it with the ground truth. Red points are the ground truth key point given by the annotator for the new objects.