FOVEA: Foveated Image Magnification for Autonomous Navigation

Chittesh Thavamani[1]* Mengtian Li[1]* Nicolas Cebron[2] Deva Ramanan[1,2]
[1]Carnegie Mellon University [2]Argo AI

Abstract

Efficient processing of high-resolution video streams is safety-critical for many robotics applications such as autonomous driving. Image downsampling is a commonly adopted technique to ensure the latency constraint is met. However, this naive approach greatly restricts an object detector's capability to identify small objects. In this paper, we propose an attentional approach that elastically magnifies certain regions while maintaining a small input canvas. The magnified regions are those that are believed to have a high probability of containing an object, whose signal can come from a dataset-wide prior or frame-level prior computed from recent object predictions. The magnification is implemented by a KDE-based mapping to transform the bounding boxes into warping parameters, which are then fed into an image sampler with anti-cropping regularization. The detector is then fed with the warped image and we apply a differentiable backward mapping to get bounding box outputs in the original space. Our regional magnification allows algorithms to make better use of high-resolution input without incurring the cost of high-resolution processing. On the autonomous driving datasets Argoverse-HD and BDD100K, we show our proposed method boosts the detection AP over standard Faster R-CNN, with and without finetuning. Additionally, building on top of the previous state-of-the-art in streaming detection, our method sets a new record for streaming AP on Argoverse-HD (from 17.8 to 23.0 on a GTX 1080 Ti GPU), suggesting that it has achieved a superior accuracy-latency tradeoff.

1. Introduction

Safety-critical robotic agents such as self-driving cars make use of an enormous suite of high-resolution perceptual sensors, with the goal of minimizing blind spots, maximizing perception range, and ensuring redundancy [5, 4, 37]. We argue that "over-sensed" perception platforms provide unique challenges for vision algorithms since those visual sensors must rapidly consume sensor streams while continuously reporting back the state of the world. While numerous techniques exist to make a particular model run fast, such as quantization [40], model compression [8], and inference optimization [30], at the end of the day, simple approaches that subsample sensor data (both spatially by frame downsampling and temporally by frame dropping) are still most effective for meeting latency constraints [19]. However, subsampling clearly throws away information, negating the goals of high-resolution sensing in the first place! This status quo calls for novel vision algorithms.

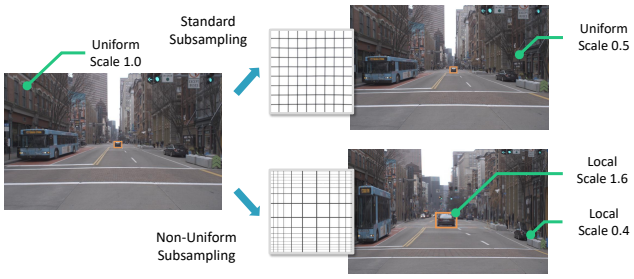


Figure 1: Standard image downsampling (top right) limits the capability of the object detector to find small objects. In this paper, we propose an attentional warping method (bottom right) that enlarges salient objects in the image while maintaining a small input resolution. Challenges arise when warping also alters the output labels (*e.g.*, bounding boxes).

To address this challenge, we take inspiration from the human visual system; biological vision makes fundamental use of *attentional* processing. While current sensing stacks make use of regular grid sampling, the human vision system in the periphery has a much lower resolution than in the center (fovea), due to the pooling of information from retinal receptors by retinal ganglion cells. Such variable resolution is commonly known as foveal vision [18].

In this paper, we propose FOVEAted image magnification (FOVEA) for object detection, which retains high resolution for objects of interest while maintaining a small canvas size. We exploit the sparsity in detection datasets – objects of interest usually only cover a portion of the image. *The key idea is to resample such that background pixels can make room for objects of interest.* The input images are downsampled and warped such that salient areas in the warped image have higher resolutions. While image warp-

* denotes equal contribution.

ing has been explored for image classification [16, 31] and regression [31], major challenges remain for object detection. First, processing the images in the warped space will produce bounding box outputs in the warped space. We make use of differentiable backward maps to unwarp the bounding box coordinates. Second, it's much more challenging to identify regions for magnification. Empirically, we find that end-to-end trained saliency networks that work well for image classification fail for object detection. However, unlike gaze estimation and fine-grained image classification, the tasks evaluated on by [31], we have an explicit signal for saliency with object detection – bounding box annotations or outputs. Specifically, we use dataset-wide priors and object locations from the previous frame (for video streams). We train a function that maps bounding box locations to warping parameters. Third, object detection has a much lower tolerance to cropping than image classification, since objects appear not only in the center but also near the edges of the image. We find that previous image warping methods are very susceptible to this issue, so we introduce an anti-cropping modification to the warping formulation.

We validate our approach on two self-driving datasets for 2D object detection: Argoverse-HD [19] and BDD100K [42]. First, we show that even without learning, our hand-coded bounding-box-guided magnification improves the average precision (AP) for off-the-shelf Faster R-CNN [33], suggesting that considerable sparsity exists in the input space for those datasets. Next, we finetune the detector with differentiable image warping and a backward label mapping, which further boosts AP. In both cases, the improvement for small objects is most significant. Finally, to show that such accuracy improvement is worth the latency cost, we evaluate our algorithm under the streaming perception framework [19], and we achieve state-of-the-art performance in terms of streaming AP.

2. Related Work

Object detection Object detection is one of the most fundamental problems in computer vision. Many methods have pushed the state-of-the-art in detection accuracy [11, 33, 23, 6, 29], and many others aim for improving the efficiency of the detectors [26, 32, 38, 3]. The introduction of fully convolution processing [36] and spatial pyramid pooling [13] have allowed us to process the input image in its original size and shape. However, it is still a common practice to downsample the input image for efficiency purposes. Efficiency becomes a more prominent issue when people move to the video domain. In video object detection, the focus has been on how to make use of temporal information to reduce the number of detectors invoked [44, 43, 27]. These methods work well on simple datasets like ImageNet VID [35], but might be unsuitable for the self-driving car senarios, where multiple new objects appear at almost every frame. Furthermore, those methods are usually designed to work in the offline fashion, *i.e.*, allowing access to future frames. Detection methods are the building blocks of our framework, and our proposed approach is largely agnostic to any particular detector.

Online/streaming perception In the online setting, the algorithm must work without future knowledge. [22] proposes the Temporal Shift Module that enables video understanding through channel shifting and in the online setting, the shifting is restricted to be uni-directional. [2] proposes a multi-object tracking method that takes input previous frame detection as addition proposals for the current frame. Our method also takes previous frame detection as input, but we use that to guide image warping. Streaming accuracy [19] is a recently proposed metric that evaluates the output of a perception algorithm at all time instants, forcing the algorithm to consider the amount of streaming data that must be ignored while computation is occuring. [19] demonstrates that streaming object detection accuracy can be significantly improved by tuning the input frame resolution and framerate. In this work, we demonstrate that adaptive attentional processing is an orthogonal dimension for improving streaming performance.

Adaptive visual attention Attentional processing has been well studied in the vision community, and it appears in different forms [9, 15, 17, 25, 21, 41]. Specially in this paper, we focus on dynamic resolutions. For image classification, [39] designs an algorithm to select high-resolution patches, assuming each patch is associated with a data acquisition cost. [28] applies non-uniform downsampling to semantic segmentation and relies on the network to learn both the forward and backward mapping, whose consistency is not guaranteed. For object detection, a dynamic zoom-in algorithm is proposed that processes high-resolution patches sequentially [10]. However, sequential execution might not meet latency requirements for real-time applications. Most similar to our work, [31] proposes an adaptive image sampling strategy that allocates more pixels for salient areas, allowing a better downstream task performance. But the method only works for image classification and regression, where the output is agnostic to the input transformation.

3. Approach

Assume we are given a training set of image-label pairs (I, L). We wish to learn a nonlinear deep predictor f that produces a low loss $\mathcal{L}(f(I), L)$. Inspired by past work [31, 16], we observe that certain labeling tasks can be performed more effectively by warping/resampling the input image. However, when the label L itself is spatially

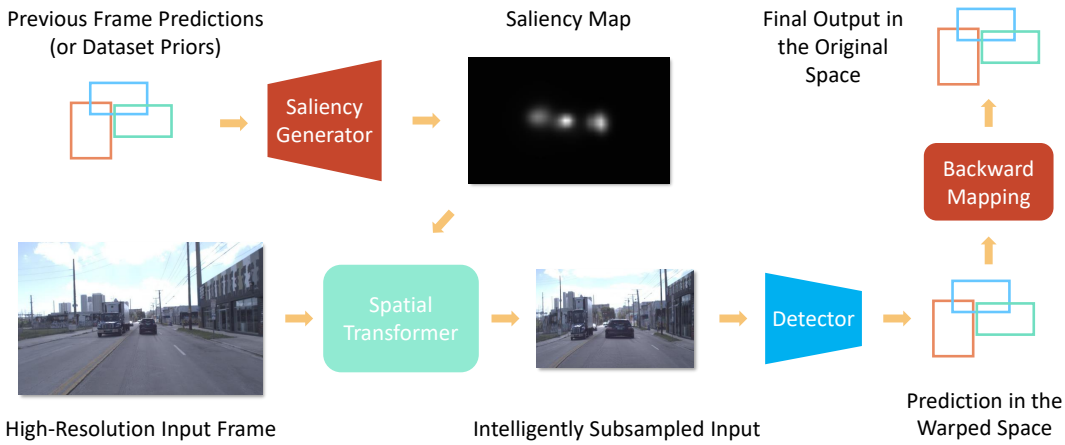


Figure 2: Our proposed method for object detection. Given bounding box predictions from the previous frame (if the input are videos) or a collection of all the ground truth bounding boxes in the training set, the saliency generator creates a saliency map and that is fed into the spatial transformer (adapted from [31, 16]) to downsample the high-resolution input frame while magnifying salient regions. Then we feed the downsampled input into a regular object detector, and it produces bounding-box output in the warped space, which is then converted back to the original image space as the final output.

defined (e.g., bounding box coordinates or semantic pixel labels), the label itself may need to be warped, or alternatively, the output of the deep predictor may need to be inverse-warped.

In this section, we first introduce the saliency-guided spatial transform from related work as the foundation of our method. Next, we introduce our solutions to address the challenges in image warping for object detection. An overview of FOVEA, our method, is shown in Fig 2.

3.1. Background: Saliency-Guided Spatial Transform

The seminal work of spatial transformer networks (STN) introduces a differentiable warping layer for input images and feature maps [16]. It was later extended to incorporate a saliency map to guide the warping [31]. Here we provide implementation details that are crucial to our method. Please refer to the original papers [16, 31] for more details.

A 2D transformation can be written as:

$$\mathcal{T} : (x, y) \rightarrow (x', y'), \tag{1}$$

where (x, y) and (x', y') are the input and output coordinates. Since image pixels are usually discrete, interpolation is required to sample values at non-integral coordinates. An image warp $\mathcal{W}_\mathcal{T}$ takes input an image I, samples the pixel values according to the given transformation $\mathcal{T}$, and outputs the warped image I':

$$I'(\mathcal{T}(x, y)) = I(x, y) \tag{2}$$

Naive forward warping of discrete pixel locations from input I can result in non-integral target pixel positions that need to be "splatted" onto the pixel grid of I, which can produce artifacts such as holes. Instead, image warps are routinely implemented via a *backward map* [1]: iterate over each output pixel grid location, compute its *inverse mapping* $\mathcal{T}^{-1}$ to find its corresponding input coordinates (which may be non-integral), and bilinearly interpolate its color from neighboring input pixel grid points:

$$I'(x, y) = I(\mathcal{T}^{-1}(x, y)) \tag{3}$$

In other words, the implementation of $\mathcal{W}_\mathcal{T}$ only requires the knowledge of the inverse transformation $\mathcal{T}^{-1}$. The pixel iteration can be replaced with a batch operation by using a grid generator and apply the transformation $\mathcal{T}^{-1}$ over the entire grid.

STN uses a differentiable formulation of $\mathcal{T}_\theta^{-1}$ (parameterized by θ) and an ensuing bilinear grid sampler, which is differentiable and parameter-free. [31] proposes a special form of $\mathcal{T}^{-1}$ parameterized by a saliency map S: $\mathcal{T}_\theta^{-1} = \mathcal{T}_S^{-1}$. This transform has a convolution form (therefore fast) using the intuition that each pixel in the input space (x, y) is attracting samples taken of the original image with a force $S(x, y)$, leading to more sampling at salient regions during the warp. *We point out that both [16] and [31] ignore the effect of warping on the corresponding label space and they skip the modeling of the forward transform $\mathcal{T}$, which is required for converting certain label types.*

3.2. Image Warping for Object Detection

In this section, we first explain our high-level inference formulation, then our specific form of the warping, and in the end some adjustments for training the task network.

Inference formulation We visually lay out the space of image and label warps in Fig 3, which shows that we have

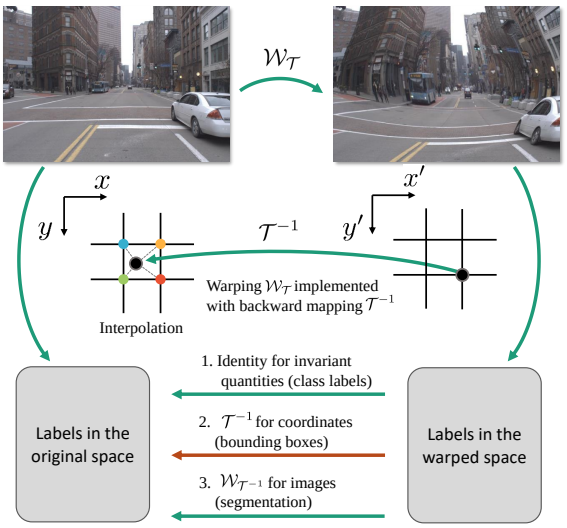


Figure 3: Image warps $\mathcal{W}_\mathcal{T}$ are commonly implemented via a backward map $\mathcal{T}^{-1}$ followed by (bilinear) interpolation of nearby source pixel grid values, since forward mapping $\mathcal{T}$ can result in target pixel positions that do not lie on the pixel grid (not shown). Though image warping is an extensively studied topic (notably by [16, 31] in the context of differentiable neural warps), its effect on labels is less explored because much prior art focuses on global labels invariant to warps (e.g. an image class label). We explore warping for spatial prediction tasks whose output must be transformed back into the original image space to generate consistent output. Interestingly, transforming pixel-level labels with warp $\mathcal{W}_{\mathcal{T}^{-1}}$ requires inverting $\mathcal{T}^{-1}$, which can be difficult depending on its parameterization [1]. In this paper, we focus on transforming pixel *coordinates* of bounding boxes, which requires only the already-computed backward map $\mathcal{T}^{-1}$ (the red arrow).

to transform the bounding box outputs accordingly after the warping. Quite conveniently, because standard image warping is implemented via the backward map $\mathcal{T}^{-1}$, the backward map is already computed in-network and so can be directly applied to the pixel coordinates of the predicted bounding box. The complete procedure for our approach $\hat{f}$ can be written as $\hat{f}(I, \mathcal{T}) = \mathcal{T}^{-1}(f(\mathcal{W}_\mathcal{T}(I)))$. where $f(\cdot)$ is the nonlinear function that returns bounding box coordinates of predicted detections. Importantly, this convenience doesn't exist when warping pixel-level *values*; *e.g.*, when warping a segmentation mask back to the original image input space (the third pathway in Fig 3). Here, one needs to invert $\mathcal{T}^{-1}$ to explicitly compute the forward warp $\mathcal{T}$.

Warping formulation We adopt the saliency-guided warping formulation from [31]:

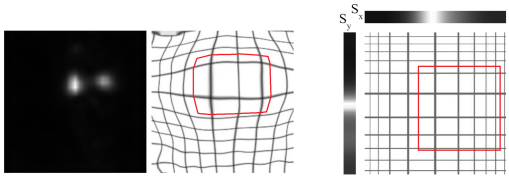


Figure 4: By restricting the general class of warps (**left**, figure adapted from [31]) to be separable (**right**), we ensure that bounding boxes in the warped image (examples outlined in red) remain axis-aligned. We demonstrate that such regularization (surprisingly) improves performance, even though doing so theoretically restricts the range of expressible warps (details in Sec 4.1.2).

$$\mathcal{T}_x^{-1}(x, y) = \frac{\int_{x',y'} S(x', y') k((x, y), (x', y')) x'}{\int_{x',y'} S(x', y') k((x, y), (x', y'))}, \quad (4)$$

$$\mathcal{T}_y^{-1}(x, y) = \frac{\int_{x',y'} S(x', y') k((x, y), (x', y')) y'}{\int_{x',y'} S(x', y') k((x, y), (x', y'))}, \quad (5)$$

where k is a distance kernel (we use a Gaussian kernel in our experiments). However, in this general form, axis-aligned bounding boxes might have different connotations in the original and warped space. To ensure axis-alignment is preserved during the mapping, we restrict the warping to be separable along the two dimensions, *i.e.*, $\mathcal{T}^{-1}(x, y) = (\mathcal{T}_x^{-1}(x), \mathcal{T}_y^{-1}(y))$. For each dimension, we adapt the previous formulation to 1D:

$$\mathcal{T}_x^{-1}(x) = \frac{\int_{x'} S_x(x') k(x', x) x'}{\int_{x'} S_x(x') k(x, x')}, \quad (6)$$

$$\mathcal{T}_y^{-1}(y) = \frac{\int_{y'} S_y(y') k(y', y) y'}{\int_{y'} S_y(y') k(y, y')}. \quad (7)$$

We call this formulation *separable* and the general form *nonseparable*. Note that the nonseparable formulation has a 2D saliency map parameter, whereas the separable formulation has two 1D saliency maps, one for each axis. Fig 4 shows an example of each type of warp.

One nice property of $\mathcal{T}^{-1}$ is that it is differentiable and thus can be trained with backpropagation. One limitation though is that its inverse $\mathcal{T}$ doesn't have a closed-form solution, nor does its derivative. The absence of $\mathcal{T}$ is not ideal, and we propose some workaround as shown in the following subsection.

Anti-Cropping Regularization We find the convolution form of saliency-guided spatial transform tends to crop the images, which might be acceptable for image classification where a large margin exists around the border. However, any cropping in object detection creates a chance to miss objects. We solve this by using reflect padding on the saliency

map while applying the attraction kernel in Eq 6. This introduces symmetries about each of the edges of the saliency map, eliminating all horizontal offsets along vertical image edges and vice versa. Thus cropping is impossible under this formulation. A 1D illustration is shown in Fig 5 to explain the problem and the solution.

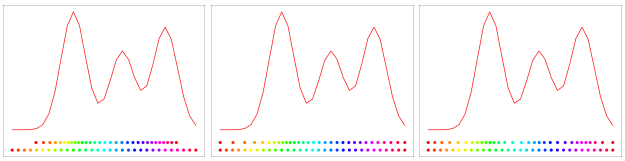
Training formulation Once we have the inference formulation, training is also straightforward as we require the loss $\mathcal{L}$ to be computed in the original space: $\mathcal{L}(\mathcal{Q}(f(\mathcal{W}_\mathcal{T}(I)), L)$, where $\mathcal{Q}$ is the label-type-specific backward mapping as shown in Fig 3, and in our case, $\mathcal{Q} = \mathcal{T}^{-1}$. Note that $\mathcal{W}_\mathcal{T}$, f and $\mathcal{T}^{-1}$ are all differentiable. While inference itself does not require the knowledge of $\mathcal{T}$, it is not the case for training detectors with region proposal networks (RPN) [33]. When training RPNs [33], the regression targets are the deltas between the anchors and the ground truth, and the deltas are later used in RoI Pooling/Align [13, 12]. The former should be computed in the original space (the ground truth is in the original space), while the latter is in the warped space (RoI Pooling/Align is over the warped image). This implies that the deltas need first to be learned in the original space, applied to the bounding box, and then mapped to the warped space using $\mathcal{T}$ for RoI Pooling/Align. But as discussed before, $\mathcal{T}$ cannot be easily computed. As a workaround, we omit the delta encoding and adopt Generalized IoU (GIoU) loss [34] to account for the lost stability. The main idea of GIoU is to better reflect the similarity of predicted and ground truth bounding boxes in cases of zero intersection; this has been shown to improve results.

3.3. KDE Saliency Generator

In prior arts [16, 31], saliency is supervised by the final task loss without intermediate supervision. We explore this for object detection in Sec 4.1.2, but find that saliency maps for object detection can be more complex due to the local structure of objects. In the streaming setting, predictions from the previous frame can serve as a strong intermediate signal for contextual priming. To learn a single attentional system that can generalize to frame-level or dataset-level priors, we describe an algorithmic approach for converting bounding boxes (be it from a dataset or the previous frame) to a saliency map.

To this end, we use kernel density estimation (KDE) with the bounding boxes as the data points. More precisely, given a set of bounding boxes B with centers c_i, heights h_i and widths w_i, we model the saliency map S_B as a sum of normal distributions:

$$S_B^{a,b} = \frac{1}{K^2} + a \sum_{(c_i,w_i,h_i)\in B} \mathcal{N}\left(c_i, b \begin{bmatrix} w_i & 0 \\ 0 & h_i \end{bmatrix}\right) \quad (8)$$



(a) Default, $\sigma \approx 5.5$ (b) Anti-crop, $\sigma \approx 5.5$ (c) Anti-crop, $\sigma \approx 1.7$

Figure 5: Saliency-guided transform illustrated in 1D. The red curve is a saliency map S. The bottom row of dots are the output points (at uniform intervals), and the top row of dots are the locations where we've sampled each output point from the original "image", as computed by applying $\mathcal{T}_S^{-1}$ to the output points. (a) The default transform can be understood as a weighted average over the output points and thus ignores points with near zero weights such as those at the boundaries. (b) Note the effects of introducing anti-crop reflect padding, and (c) how decreasing the std dev σ of the attraction kernel k results in more local warping around each peak (better for multimodal saliency distributions).

where a and b are hyperparameters for amplitude and bandwidth, respectively, and K is the size of the attraction kernel k in Eq 6. Adding the small constant is done to prevent extreme warps. We then normalize the 2D saliency map such that it sums to 1 and marginalize along the two axes if using the separable formulation[1]. As laid out in the previous section, this is then used to generate the image transformation $\mathcal{T}_S^{-1}$ according to Eq 6.

Once we have the saliency generator defined, one can either apply S_B to the previous frame prediction to obtain a frame-specific temporal prior (denoted as S_I), or to the set of all bounding boxes in the training set to obtain a dataset-wide prior (denoted as S_D). In the former case, we note that the KDE formulation effectively foveates the image at each of the previous frame's detections. For the first frame in each video sequence, this trivially defaults to the uniform saliency map. In the latter case, we note that for datasets like Argoverse-HD, the horizon tends to be in the center of the image, and thus objects are more likely to appear there. We also try combining these signals to capture both biases. The resulting saliency map is $S_C = \alpha \cdot S_I + (1 - \alpha) \cdot S_D$, where α is a hyperparameter for how much we trust the temporal bias. All of the above saliency generators are differentiable, so we can use the final task loss to learn our hyperparameters a and b.

4. Experiments

In this section, we first show on the autonomous driving dataset Argoverse-HD that FOVEA greatly improves

[1]When using the separable formulation, we could instead skip the intermediate 2D saliency map representation. However, we opt not to, because the intermediate 2D saliency map produces more interpretable visualizations, and the difference in runtime is negligible.

accuracy over naive downsampling. Next, we conduct cost-performance analysis under streaming perception showing that accuracy gain is worth the additional latency cost. Finally, we present results on BDD100K showing the generalization capability of our method. We include additional results, diagnostic experiments, and implementation details in the appendix.

4.1. Object Detection for Autonomous Navigation

Argoverse-HD [19], the primary dataset we evaluate our methods on, is an object detection dataset captured in an embodied autonomous vehicle setting. The data contain 30 FPS video sequences and dense 2D bounding box annotations. As a common practise for detection, we adopt AP as our primary evaluation metric. We also report *end-to-end* latency (including image preprocessing, network inference, and bounding box postprocessing) measured on a single GTX 1080 Ti GPU. The image resolution for this dataset is 1920×1200, much larger than COCO's, which is capped at 640. Since all models used in this paper are fully convolutional, we run them with different input scales, denoted by ratios to the native resolution, *e.g.*, 0.5x means an input resolution of 960×600.

4.1.1 Baseline and Setup

The baseline we compare to throughout our experiments is Faster RCNN [33] with a ResNet-50 backbone [14] plus FPN [23]. The default input scale for both the baseline and our method is 0.5x. For the baseline, however, we additionally train and test at 0.75x and 1x scales, to derive a sense of the latency-accuracy tradeoff using this model. Our contribution is orthogonal to the choice of the baseline detector and we obtain similar results with other detectors including RetinaNet [24] and YOLOF [7] (shown in Appendix B). Additionally, we compare against other zoom-based approaches [31, 10] in Appendix C.

Notably, Argoverse-HD's training set only contains *pseudo ground truth* (at the time of paper submission) generated by running high-performing detector HTC [6] in the offline setting. For all experiments, unless otherwise stated, we train on the train split with pseudo ground truth annotations, and evaluate on the val split with real annotations. Additional measures are taken to prevent overfitting to biased annotations. We finetune COCO pretrained models on Argoverse-HD for only 3 epochs (*i.e.*, early stopping). We use momentum SGD with a batch size of 8, a learning rate of 0.02, 0.9 momentum, 10^{-4} weight decay, and a step-wise linear learning rate decay for this short schedule [20]. Also, when training detectors with warped input, we apply our modifications to RPN and the loss function as discussed in Sec 3.2.

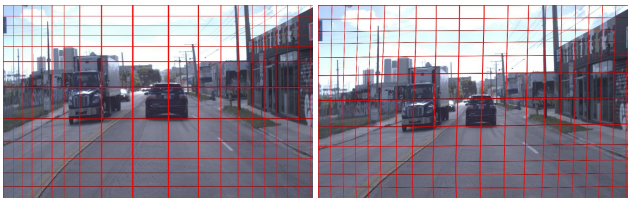


Figure 6: The learned direct separable (**left**) and nonseparable (**right**) dataset-wide warps. Despite the vastly greater flexibility of nonseparable warps, the learned warp is almost separable anyway.

4.1.2 Learned Saliency

Our first experiments directly try to learn saliency without using bounding box priors and under supervision from just the final task loss. This serves as a control for our later experiments using the KDE saliency generator formulation introduced in Sec 3.3.

In our first formulation, the saliency map is a parameter of our network and learned through backpropagation, yielding a learned dataset-wide fixed warp. We test both the separable and nonseparable versions of this and report results in Tab 1. Training configuration and implementation details are given in Appendix F.

We find that both separable and nonseparable methods significantly improve overall AP over the baseline, owing to the boosted performance on small objects. However, there is also a small decrease in AP on large objects. Interestingly, even though the nonseparable formulation is more flexible than the separable formulation, it performs worse, showing that the model struggles to learn in this parameter space. Also, as shown in Fig 6, the final learned nonseparable warp is actually uncannily close to being separable, suggesting that the separable class of warps might be preferred anyways. Therefore, going forward, we choose the separable warp formulation in our later experiments.

Following the lead of [31], we also try learning a "saliency network" that maps each input image to its saliency map via a ResNet-18 backbone [14]. In this sense, the learned saliency map would adapt to each image. However, we find that this approach doesn't translate well to object detection in that it makes training very unstable. From our experiments, even with a learning rate of 10^{-5} on the saliency network, the model learns a degeneracy in which an extreme warp leads to no proposals being matched with ground truth bounding boxes in the RoI bounding box head, leading to a regression loss of 0.

4.1.3 KDE Saliency Generator

In this section, we attempt to use bounding box detections to guide our saliency using the KDE formulation introduced in Sec 3.3.

We first try manually tuning the amplitude a and bandwidth b to obtain the desired magnifications. We find that an amplitude $a = 1$ and a bandwidth $b = 64$ works the best, paired with an attraction kernel of std. dev. of about 17.8% the image height, which allows for more local warps as illustrated in Fig 5. We finetune our models using the same configuration as the baseline, the only difference being the added bounding box and saliency-guided spatial transformation layer. To simplify training, we use jittered ground truth bounding boxes from the current frame rather than detections from the previous frame. We implement the S_I formulation, which uses just the bounding box detections from the previous frame, the S_D formulation, which uses all bounding boxes in the training set to generate a fixed dataset-wide prior, and the S_C formulation with $\alpha = 0.5$.

We then attempt to learn hyperparameters a and b through backpropagation, since our KDE formulation is differentiable. We initialize parameters a' and b' to 0, under the construction that $a = |1+a'|+0.1, b = 64 \cdot |1+b'|+0.1$. The learning rate of a' and b' is set to 10^{-4} with zero weight decay. Other than this, we train the learned KDE (LKDE) model with the same configuration as the baseline. We implement the S_I formulation.

All results are shown in Table 1. Even without finetuning our detector, using a simple fixed dataset-wide warp S_D, we find significant improvements in AP. As we switch to temporal bias and finetune, we see even more improvement. As in the learned saliency case, these improvements in overall AP are due to large boosts in AP_S, outweighing the small decreases in AP_L. Combining our saliency signals (S_C) doesn't help, because in our case, it seems that the temporal signal is strictly stronger than the dataset-wide signal. Perhaps if we had an alternate source of saliency like a map overlay, combining saliencies could help. Our best method overall is LKDE, which learned optimal values $a = 1.07, b = 71.6$. Learning a nonseparable saliency performs better than our hand-constructed dataset-wide warp S_D; however, they're both outperformed by S_I. Finally, we note that our increased performance comes at the cost of only about 2 ms in latency.

4.2. Streaming Accuracy for Cost-Performance Evaluation

Streaming accuracy is a metric that coherently integrates latency into standard accuracy evaluation and therefore is able to *quantitatively measure the accuracy-latency trade-off* for embodied perception [19]. Here we adopt their evaluation protocol for our cost-performance analysis. In our case of streaming object detection, the streaming accuracy refers to *streaming AP*. We use the same GPU (GTX 1080 Ti) and their public available codebase for a fair comparison with their proposed solution. Their proposed solution includes a scale-tuned detector (Faster R-CNN), dynamic scheduler (shrinking-tail) and Kalman Filter forecastor. Our experiments focus on improving the detector and we keep the scheduler and forecastor fixed.

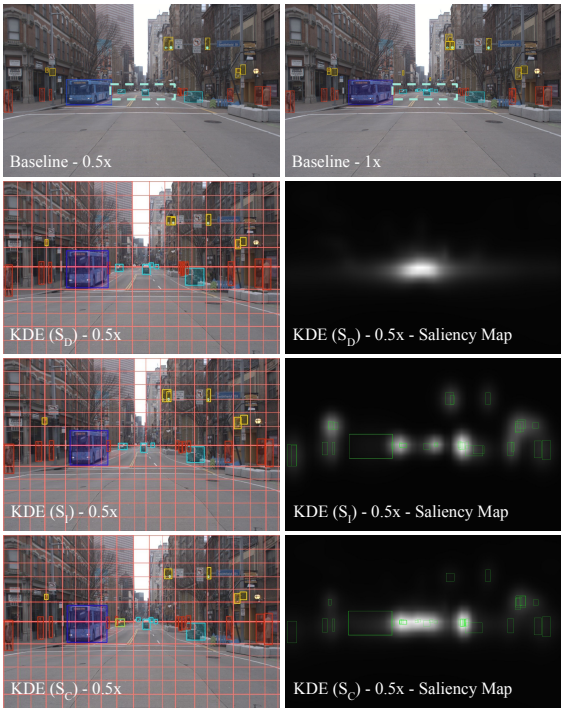


Figure 7: Qualitative results for our methods after finetuning on Argoverse-HD. The cars in the distance (in the dotted boxes), undetected at 0.5x scale, are detected at 1x scale, and partially detected by our methods. Different rows show the variations within our method based on the source of attention.

Tab 2 presents our evaluation under the full-stack setting (a table for the detection-only setting is included in Appendix E. We see that FOVEA greatly improves the previous state-of-the-art. The improvement first comes from a faster and slightly more accurate implementation of the baseline (please refer to Appendix F for the implementation details). Note that under streaming perception, a faster algorithm while maintaining the same offline accuracy translates to an algorithm with higher streaming accuracy. The second improvement is due to training on pseudo ground truth (discussed in Sec 4.1.1). Importantly, our KDE image warping further boosts the streaming accuracy significantly *on top of* these improvements. Overall, these results suggest that *image warping is a cost-efficient way to improve accuracy*.

4.3. Cross-Dataset Generalization

Our experiments so far are all conducted on the Argoverse-HD dataset. In this section, we cross-validate our proposed method on another autonomous driving dataset BDD100K [42]. Note that BDD100K and

Argoverse-HD before finetuning

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	person	mbike	tffclight	bike	bus	stop	car	truck	Latency (ms)
Baseline	21.5	35.8	22.3	2.8	22.4	**50.6**	20.8	9.1	13.9	7.1	48.0	16.1	37.2	20.2	49.4 ± 1.0
KDE (S_D)	23.3	40.0	22.9	5.4	25.5	48.9	20.9	13.7	12.2	9.3	**50.6**	20.1	40.0	19.5	52.0 ± 1.0
KDE (S_I)	**24.1**	**40.7**	**24.3**	**8.5**	24.5	48.3	**23.0**	**17.7**	**15.1**	**10.0**	49.5	17.5	**41.0**	19.4	51.2 ± 0.7
KDE (S_C)	24.0	40.5	**24.3**	7.4	**26.0**	48.2	22.5	14.9	14.0	9.5	49.7	**20.6**	**41.0**	**19.9**	52.0 ± 1.2
Upp. Bound (0.75x)	27.6	45.1	28.2	7.9	30.8	51.9	29.7	14.3	21.5	6.6	54.4	25.6	44.7	23.7	86.9 ± 1.6
Upp. Bound (1.0x)	32.7	51.9	34.3	14.4	35.6	51.8	33.7	21.1	33.1	5.7	57.2	36.7	49.5	24.6	133.9 ± 2.2

Argoverse-HD after finetuning

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	person	mbike	tffclight	bike	bus	stop	car	truck	Latency (ms)
Baseline	24.2	38.9	26.1	4.9	29.0	50.9	22.8	7.5	23.3	5.9	44.6	19.3	43.7	26.6	50.9 ± 0.9
Learned Sep.	27.2	44.8	28.3	**12.2**	29.1	46.6	24.2	14.0	22.6	7.7	39.5	**31.8**	50.0	27.8	51.5 ± 1.0
Learned Nonsep.	25.9	42.9	26.5	10.0	28.4	48.5	25.2	11.9	20.9	7.1	39.5	25.1	49.4	28.1	50.0 ± 0.8
KDE (S_D)	26.7	43.3	27.8	8.2	29.7	54.1	25.4	13.5	22.0	8.0	**45.9**	21.3	48.1	29.3	50.8 ± 1.2
KDE (S_I)	28.0	45.5	**29.2**	10.4	**31.0**	**54.5**	27.3	16.9	**24.3**	**9.0**	44.5	23.2	**50.5**	28.4	52.2 ± 0.9
KDE (S_C)	27.2	44.7	28.4	9.1	30.9	53.6	27.4	14.5	23.0	7.0	44.8	21.9	49.9	**29.5**	52.1 ± 0.9
LKDE (S_I)	**28.1**	**45.9**	28.9	10.3	30.9	54.1	**27.5**	**17.9**	23.6	8.1	45.4	23.1	50.2	28.7	50.5 ± 0.8
Upp. Bound (0.75x)	29.2	47.6	31.1	11.6	32.1	53.3	29.6	12.7	30.8	7.9	44.1	29.8	48.8	30.1	87.0 ± 1.4
Upper Bound (1.0x)	33.3	53.9	35.0	16.8	34.8	53.6	33.1	20.9	38.7	6.7	44.7	36.7	52.7	32.7	135.0 ± 1.6

Table 1: Results before and after finetuning on Argoverse-HD. Without retraining, processing warped images (KDE S_I, top table) improves overall AP by 2.6 points and triples AP_S. Even larger gains can be observed after finetuning, making our final solution (LKDE S_I) performing close to the 0.75x upper bound. Please refer to the text for a more detailed discussion.

ID	Method	AP	AP_S	AP_M	AP_L
1	Prior art [19]	17.8	3.2	16.3	33.3
2	+ Better implementation	19.3	4.1	18.3	34.9
3	+ Train with pseudo GT	21.2	3.7	23.9	43.8
4	2 + Ours (S_I)	19.3	5.2	18.5	39.0
5	3 + Ours (S_I)	**23.0**	**7.0**	**23.7**	**44.9**

Table 2: Streaming evaluation in the full-stack (with forecasting) setting on Argoverse-HD. We show that our proposed method significantly improves previous state-of-the-art by 5.2, in which 1.5 is from better implementation, 1.9 is from making use of pseudo ground truth and 1.8 is from our proposed KDE warping.

Argoverse-HD are collected in different cities. For simplicity, we only test out *off-the-shelf* generalization without any finetuning. We experiment on the validation split of the MOT2020 subset, which contains 200 videos with 2D bounding boxes annotated at 5 FPS (40K frames in total). Also, we only evaluate on common classes between BDD100K and Argoverse-HD: person, bicycle, car, motorcycle, bus, and truck. The results are summarized in Tab 3, which demonstrate the generalization capability of our proposed method.

5. Conclusion

We propose FOVEA, a highly efficient attentional model for object detection. Our model magnifies regions likely to contain objects, making use of top-down saliency priors learned from a dataset or from temporal context. To do so, we make use of differentiable image warping that ensures bounding box predictions can be mapped back to the original image space. The proposed approach significantly improves over the baselines on Argoverse-HD and BDD100K. For future work, it would be natural to make use of trajectory forecasting models to provide even more accurate saliency maps for online processing.

ID	Method	AP	AP_S	AP_M	AP_L
1	Baseline (0.5x)	15.1	1.0	10.6	**39.0**
2	Ours S_D (0.5x)	13.7	1.3	10.0	34.7
3	Ours S_I (0.5x)	**16.4**	**2.1**	**12.8**	38.6
4	Baseline (0.75x)	19.7	3.0	16.1	**44.2**
5	Ours S_D (0.75x)	18.2	3.4	15.4	40.0
6	Ours S_I (0.75x)	**20.1**	**5.2**	**17.0**	42.5
7	Upper bound (1.0x)	22.6	5.7	20.1	45.7

Table 3: Cross-dataset generalization to BDD100K [42]. Rows 2 & 5 are saliency computed on the Argoverse-HD training set, as expected, they fail to generalize to a novel dataset. Despite operating at a larger temporal stride (5 FPS vs 30 FPS), our proposed image-adaptive KDE warping generalizes to a novel dataset (row 3 & 6). Note that here the image native resolution is smaller at 1280×720.

Acknowledgements: This work was supported by the CMU Argo AI Center for Autonomous Vehicle Research.

References

[1] Thaddeus Beier and Shawn Neely. Feature-based image metamorphosis. *ACM SIGGRAPH computer graphics*, 26(2):35–42, 1992. 3, 4

[2] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixé. Tracking without bells and whistles. In *ICCV*, 2019. 2

[3] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. 2

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

[5] Ming-Fang Chang, John W Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3D tracking and forecasting with rich maps. In *CVPR*, 2019. 1

[6] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *CVPR*, pages 4974–4983, 2019. 2, 6

[7] Qiang Chen, Yingming Wang, Tong Yang, Xiangyu Zhang, Jian Cheng, and Jian Sun. You only look one-level feature. *CVPR*, 2021. 6

[8] Yu Cheng, D. Wang, Pan Zhou, and Tao Zhang. A survey of model compression and acceleration for deep neural networks. *ArXiv*, abs/1710.09282, 2017. 1

[9] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, pages 764–773, 2017. 2

[10] Mingfei Gao, Ruichi Yu, Ang Li, Vlad I Morariu, and Larry S Davis. Dynamic zoom-in network for fast object detection in large images. In *CVPR*, pages 6926–6935, 2018. 2, 6

[11] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014. 2

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *ICCV*, 2017. 5

[13] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *TPAMI*, 37:1904–1916, 2015. 2, 5

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 6

[15] Chen Huang, Simon Lucey, and Deva Ramanan. Learning policies for adaptive tracking with deep feature cascades. In *ICCV*, pages 105–114, 2017. 2

[16] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. *NIPS*, 2015. 2, 3, 4, 5

[17] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *CVPR*, pages 9799–9808, 2020. 2

[18] Adam M. Larson and Lester C. Loschky. The contributions of central versus peripheral vision to scene gist recognition. *Journal of vision*, 9 10:6.1–16, 2009. 1

[19] Mengtian Li, Yuxiong Wang, and Deva Ramanan. Towards streaming perception. In *ECCV*, 2020. 1, 2, 6, 7, 8

[20] Mengtian Li, Ersin Yumer, and Deva Ramanan. Budgeted training: Rethinking deep neural network training under resource constraints. In *ICLR*, 2020. 6

[21] Xiaoxiao Li, Ziwei Liu, Ping Luo, Chen Change Loy, and Xiaoou Tang. Not all pixels are equal: Difficulty-aware semantic segmentation via deep layer cascade. In *CVPR*, pages 3193–3202, 2017. 2

[22] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *ICCV*, pages 7083–7093, 2019. 2

[23] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017. 2, 6

[24] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 6

[25] Lanlan Liu and Jia Deng. Dynamic deep neural networks: Optimizing accuracy-efficiency trade-offs by selective execution. In *AAAI*, volume 32, 2018. 2

[26] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot multibox detector. In *ECCV*, 2016. 2

[27] Hao Luo, Wenxuan Xie, Xinggang Wang, and Wenjun Zeng. Detect or track: Towards cost-effective video object detection/tracking. In *AAAI*, volume 33, pages 8803–8810, 2019. 2

[28] Dmitrii Marin, Zijian He, Peter Vajda, Priyam Chatterjee, Sam Tsai, Fei Yang, and Yuri Boykov. Efficient segmentation: Learning downsampling near semantic boundaries. In *ICCV*, pages 2131–2141, 2019. 2

[29] Siyuan Qiao, Liang-Chieh Chen, and Alan Yuille. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. *arXiv preprint arXiv:2006.02334*, 2020. 2

[30] Jonathan Ragan-Kelley, Connelly Barnes, Andrew Adams, Sylvain Paris, F. Durand, and Saman P. Amarasinghe. Halide: a language and compiler for optimizing parallelism, locality, and recomputation in image processing pipelines. *Proceedings of the 34th ACM SIGPLAN Conference on Programming Language Design and Implementation*, 2013. 1

[31] Adria Recasens, Petr Kellnhofer, Simon Stent, Wojciech Matusik, and Antonio Torralba. Learning to zoom: a saliency-based sampling layer for neural networks. In *ECCV*, pages 51–66, 2018. 2, 3, 4, 5, 6

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

[33] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, pages 91–99, 2015. 2, 5, 6

[34] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *CVPR*, pages 658–666, 2019. 5

[35] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 115(3):211–252, 2015. 2

[36] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013. 2

[37] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset, 2019. 1

[38] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *CVPR*, pages 10781–10790, 2020. 2

[39] Burak Uzkent and Stefano Ermon. Learning when and where to zoom with deep reinforcement learning. In *CVPR*, pages 12345–12354, 2020. 2

[40] V. Vanhoucke, A. Senior, and M. Mao. Improving the speed of neural networks on cpus. In *Deep Learning and Unsupervised Feature Learning Workshop, NIPS*, 2011. 1

[41] Thomas Verelst and Tinne Tuytelaars. Dynamic convolutions: Exploiting spatial sparsity for faster inference. In *CVPR*, pages 2320–2329, 2020. 2

[42] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *CVPR*, June 2020. 2, 7, 8

[43] Xizhou Zhu, Jifeng Dai, Lu Yuan, and Yichen Wei. Towards high performance video object detection. In *CVPR*, pages 7210–7218, 2018. 2

[44] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. Flow-guided feature aggregation for video object detection. In *ICCV*, pages 408–417, 2017. 2