A. Additional Diagnostic Experiments

A.1. The Role of Explicit Backward Label Mapping

Related work either focus on tasks with labels invariant to warping like image classification or gaze estimation [6, 11] (discussed in Sec 3.1), or expect an implicit backward mapping to be learned through black-box end-to-end training [10] (discussed in Sec 2). In this section, we suggest that the implicit backward label mapping approach is not feasible for object detection. To this end, we train and test our KDE methods minus any bounding box unwarping. Specifically, we no longer unwarp bounding boxes when computing loss during training and when outputting final detections during testing. Instead, we expect the model to output detections in the original image space.

Due to instability, additional measures are taken to make it end-to-end trainable. First, we train with a decreased learning rate of 1e-4. Second, we train with and without adding ground truth bounding boxes to RoI proposals. The main KDE experiments do not add ground truth to RoI proposals, because there is no way of warping bounding boxes into the warped image space (the implementation of $\mathcal{T}$ does not exist). We additionally try setting this option here, because it would help the RoI head converge quicker, under the expectation that the RPN should output proposals in the original space. All other training settings are identical to the baseline setup (Sec 4.1.1).

Results are shown in Tab A. The overall AP is single-digit under all of these configurations, demonstrating the difficulty of implicitly learning the backward label mapping. This is likely due to the fact that our model is pretrained on COCO [9], so it has learned to localize objects based on their exact locations in the image, and finetuning on Argoverse-HD is not enough to "unlearn" this behavior and learn the backward label mapping. Another factor is that in the S_I and S_C cases, each image is warped differently, making the task of learning the backwards label mapping even more challenging. We suspect that training from scratch with a larger dataset like COCO and using the warp parameters (e.g. the saliency map) as input may produce better results. However, this only reinforces the appeal of our method due to ease of implementation and cross-warp generalizability (we can avoid having to train a new model for each warping mechanism).

A.2. Sensitivity to Quality of Previous-Frame Detections

Two of our methods, S_I and S_C are dependent on the accuracy of the previous-frame detections. In this section, we analyze the sensitivity of such a dependency through a soft upper bound on S_I and S_C, which is generated using the current frame's ground truth annotations in place of detections from the previous frame. This soft upper bound is a perfect saliency map, up to the amplitude and bandwidth hyperparameters. Note that this is only a change in the testing configuration.

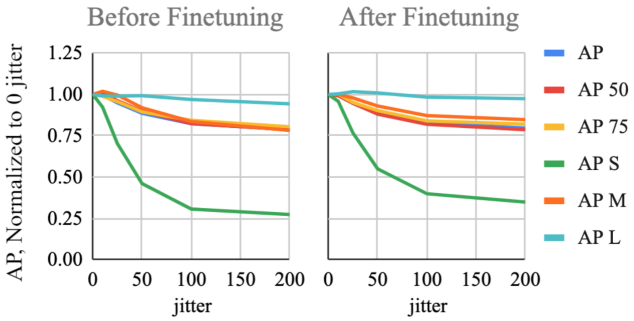

Figure A: Plots showing the effect of motion (jitter) on AP using the KDE S_I formulation. Results have been normalized according to the AP at 0 jitter. As is intuitive, motion affects AP_S the most and AP_L the least. After finetuning (with an artificial jitter of 50), we see that the model reacts less adversely to jitter, indicating that our regularization has helped.

We report results in Tab A. We see a significant boost in accuracy in all cases. Notably, the finetuned KDE S_I model at 0.5x scale achieves an AP of 29.6, outperforming the baseline's accuracy of 29.2 at 0.75x scale.

A.3. Sensitivity to Inter-Frame Motion

Having noted that the S_I and S_C formulations are sensitive to the accuracy of the previous-frame detections, in this section, we further test its robustness to motion between frames. We use ground truth bounding boxes (rather than detections) from the previous frame in order to isolate the effect of motion on accuracy. We introduce a jitter parameter j and translate each of the ground truth bounding boxes in the x and y directions by values sampled from $\mathcal{U}(-j, j)$. The translation values are in pixels in reference to the original image size of 1920×1200. As in Sec A.2, this is a purely testing-time change. Also note that the upper bound experiments in Sec A.2 follows by setting $j = 0$. We test only on S_I and report the full results in Tab A. We also plot summarized results and discuss observations in A.

B. FOVEA Beyond Faster R-CNN

In the main text and other sections of the appendix, we conduct our experiment based on Faster R-CNN. However, our proposed warping-for-detection framework is agnostic to specific detectors. To show this, we test our methods on RetinaNet [8], a popular single-stage object detector, and on YOLOF [3], a recent YOLO variant that avoids bells and whistles and long training schedules (up to 8x for ImageNet and 11x for COCO compared to standard schedules for YOLOv4 [1]).

Argoverse-HD before finetuning

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	person	mbike	tffclight	bike	bus	stop	car	truck
Main Results (copied from the main text for comparison)														
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
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
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
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**
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
Upp. Bound (1x)	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
Without an Explicit Backward Label Mapping (Sec A.1)														
KDE (S_D)	5.4	14.2	3.7	0.0	0.9	20.7	3.2	0.4	1.2	0.8	27.9	0.0	5.3	4.2
KDE (S_I)	6.1	15.6	4.0	0.2	0.8	20.3	2.3	0.6	0.7	1.8	30.8	0.0	7.0	5.4
KDE (S_C)	6.0	15.9	3.8	0.1	0.9	21.9	3.0	0.6	0.9	1.5	30.2	0.0	6.7	5.2
Upper Bound with Ground Truth Saliency (Sec A.2)														
KDE (S_I)	25.4	42.6	25.6	9.1	26.2	49.5	25.3	17.4	16.8	10.1	49.4	23.4	41.7	19.4
KDE (S_C)	24.5	41.7	24.6	7.5	26.8	48.8	23.6	14.5	15.2	9.7	49.7	22.6	41.3	19.8
Sensitivity to Inter-Frame Motion (Sec A.3)														
KDE (S_I), $j = 10$	25.3	42.9	25.3	8.4	26.7	49.1	25.0	16.4	16.2	10.1	48.8	25.0	41.8	19.5
KDE (S_I), $j = 25$	24.1	41.0	24.5	6.4	26.1	49.0	24.0	12.6	15.2	9.0	48.5	22.9	41.1	19.6
KDE (S_I), $j = 50$	22.5	38.3	22.9	4.2	24.1	49.1	21.9	9.9	14.4	8.2	48.4	18.5	39.0	19.7
KDE (S_I), $j = 100$	20.9	35.1	21.6	2.8	21.9	48.0	20.1	7.1	14.0	6.8	47.8	15.3	36.7	19.1
KDE (S_I), $j = 200$	20.0	33.5	20.6	2.5	20.5	46.7	19.2	6.0	13.4	6.2	46.7	14.3	35.5	18.5

Argoverse-HD after finetuning

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	person	mbike	tffclight	bike	bus	stop	car	truck
Main Results (copied from the main text for comparison)														
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
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
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
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
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
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**
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
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
Upp. Bound (1x)	31.6	51.4	33.5	14.5	33.5	54.1	31.8	15.2	37.4	9.0	43.9	35.3	50.2	30.2
Without an Explicit Backward Label Mapping (Sec A.1)														
KDE (S_D), no RoI GT	2.1	2.6	2.5	0.0	0.0	4.0	0.6	0.0	0.0	0.6	14.8	0.0	0.0	0.9
KDE (S_D)	1.8	2.7	1.9	0.0	0.0	3.2	0.6	0.0	0.0	0.0	13.3	0.0	0.1	0.6
KDE (S_I), no RoI GT	2.5	3.0	2.9	0.0	0.1	4.3	0.7	0.0	0.0	0.6	17.0	0.9	0.0	0.9
KDE (S_I)	2.0	2.8	2.4	0.0	0.0	3.7	0.6	0.0	0.0	0.0	14.8	0.0	0.3	0.5
Upper Bound with Ground Truth Saliency (Sec A.2)														
KDE (S_I)	29.6	48.7	30.7	12.0	32.8	54.4	28.3	16.3	27.7	9.9	43.9	30.6	50.9	28.8
KDE (S_C)	27.8	45.5	28.8	9.6	31.7	53.4	27.5	13.9	24.7	6.5	44.5	25.1	50.2	29.6
Sensitivity to Inter-Frame Motion (Sec A.3)														
KDE (S_I), $j = 10$	29.4	48.3	30.7	11.5	32.8	54.6	27.9	15.9	27.2	9.7	43.7	31.1	50.6	28.7
KDE (S_I), $j = 25$	28.0	46.1	29.2	9.2	32.1	55.3	26.4	13.9	25.9	9.3	43.9	26.8	49.2	28.7
KDE (S_I), $j = 50$	26.2	42.9	27.7	6.6	30.5	54.9	24.1	12.1	24.9	8.6	44.1	21.8	46.2	27.9
KDE (S_I), $j = 100$	24.5	39.9	25.8	4.8	28.6	53.5	22.3	10.2	23.5	7.6	43.5	17.7	43.9	27.1
KDE (S_I), $j = 200$	23.6	38.3	25.2	4.2	27.8	53.0	21.4	8.6	22.8	7.4	42.9	16.6	42.7	26.6

Table A: Results before and after finetuning on AVHD. Please refer to Sec A for a detailed discussion.

For both these detectors, we test baselines at 0.5x and 0.75x scales both before and after finetuning. We then compare these results against our KDE S_I method at 0.5x scale. We use a learning rate of 0.01 for the RetinaNet KDE S_I model and 0.005 for the RetinaNet baselines. All other training settings for RetinaNet are identical to the Faster-RCNN baseline. For YOLOF, we use a learning rate of 0.012 and keep all other settings true to the original paper.

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
RetinaNet, Before Finetuning on Argoverse-HD						
Baseline (0.5x)	18.5	29.7	18.6	1.3	17.2	48.8
KDE (S_I)	18.5	31.2	17.9	4.5	16.8	44.9
Upp. Bound (0.75x)	24.8	38.8	25.5	4.5	28.7	52.0
RetinaNet, After Finetuning on Argoverse-HD						
Baseline (0.5x)	22.6	38.9	21.4	4.0	22.0	53.1
KDE (S_I)	24.9	40.3	25.3	7.1	27.7	50.6
Upp. Bound (0.75x)	29.9	48.6	30.1	9.7	32.5	54.2
YOLOF, Before Finetuning on Argoverse-HD						
Baseline (0.5x)	15.0	25.4	14.3	0.6	11.0	46.0
KDE (S_I)	16.8	29.0	16.0	0.9	14.0	46.4
Upp. Bound (0.75x)	21.6	35.5	22.3	2.3	22.2	52.7
YOLOF, After Finetuning on Argoverse-HD						
Baseline (0.5x)	18.4	30.5	18.3	1.4	16.5	47.9
KDE (S_I)	21.3	36.7	20.2	3.5	21.8	49.7
Upp. Bound (0.75x)	25.1	41.3	25.3	4.7	27.6	54.1

Table B: Experiments with RetinaNet [8] and YOLOF [3]. We follow the same setup as the experiment with Faster R-CNN. The top quarter suggests that unlike Faster R-CNN, RetinaNet does not work off-the-shelf with our KDE warping. However, the second quarter suggests similar performance boosts as with Faster R-CNN can be gained after finetuning on Argoverse-HD. Interestingly, for YOLOF, our method boosts AP in all categories – small, medium, and large – even with off-the-shelf weights.

Results are presented in Tab B.

C. Comparison Against Additional Baselines

There are other approaches that make use of image warping or patch-wise zoom for visual understanding. The first noticeable work [11], explained extensively in the main text, warps the input image for tasks that have labels invariant to warping. The second noticeable work [5] employs reinforcement learning (RL) to decide which patches to zoom in for high-resolution processing. In this section, we attempt to compare our FOVEA with these two approaches.

Our method builds upon spatial transformer networks [6, 11] and we have already compared against [11] sporadically in the main text. Here provides a summary of all the differences (see Tab C). A naive approach might directly penalize the discrepancy between the output of the (warped) network and the unwarped ground-truth in an attempt to implicitly learn the inverse mapping, but this results in abysmal performance (dropping 28.1 to 2.5 AP, discussed in Sec A.1). To solve this issue, in Sec 3.1, we note that [6, 11] actually learn a backward map $\mathcal{T}^{-1}$ instead of a forward one $\mathcal{T}$. This allows us to add a backward-map layer that transforms bounding box coordinates back to the original space via $\mathcal{T}^{-1}$, dramatically improving accuracy. A second significant difference with [6, 11] is our focus on attention-for-efficiency. If the effort required to determine where to attend is more than the effort to run the raw detector, attentional processing can be inefficient (see the next paragraph). [11] introduces a lightweight saliency network to produce a heatmap for where to attend; however, this model does not extend to object detection, perhaps because it requires the larger capacity of a detection network (see Sec 4.1.1). Instead, we replace this feedforward network with an essentially *zero*-cost saliency map constructed via a simple but effective global spatial prior (computed offline) or temporal prior (computed from previous frame's detections). Next, we propose a technique to prevent cropping during warping (via reflection padding, as shown in Fig 5), which also boosts performance by a noticeable amount. Finally, as stated in the training formulation in Sec 3.2, it *doesn't even make sense* to train a standard RPN-based detector with warped input due to choice of delta encoding (which normally helps stabilize training). We must remove this standard encoding and use GIoU to compensate for the lost stability during training.

Method	AP
FOVEA (Ours full)	28.1
w/o Explicit backward mapping	2.5
w/o KDE saliency (using saliency net as in [11])	Doesn't train
w/o Anti-crop regularization	26.9
w/o direct RPN box encoding	N/A

Table C: Summary of key modifications in FOVEA.

Next, we attempt to compare against this RL-based zoom method [5] using our baseline detector (public implementation from mmdetection [2]) on their Caltech Pedestrian Dataset [4]. However, while their full-scale 800×600 Faster R-CNN detector reportedly takes 304ms, our implementation is *dramatically* faster (44ms), consistent with the literature for modern implementations and GPUs. This changes the conclusions of that work because full-scale processing is now faster than coarse plus zoomed-in processing (taking 28ms and 25ms respectively), even assuming a zero-runtime RL module (44ms $<$ 28ms + 25ms).

D. Additional Visualizations

Please refer to Fig B and C for additional qualitative results of our method.

E. Detection-Only Streaming Evaluation

In Sec 4.2 of the main text, we provide the full-stack evaluation for streaming detection. Here we provide the detection-only evaluation for completeness in Tab D. This setting only allows detection and scheduling, and thus isolating the contribution of tracking and forecasting. We observe similar trend as in the full-stack setting in Tab 2.

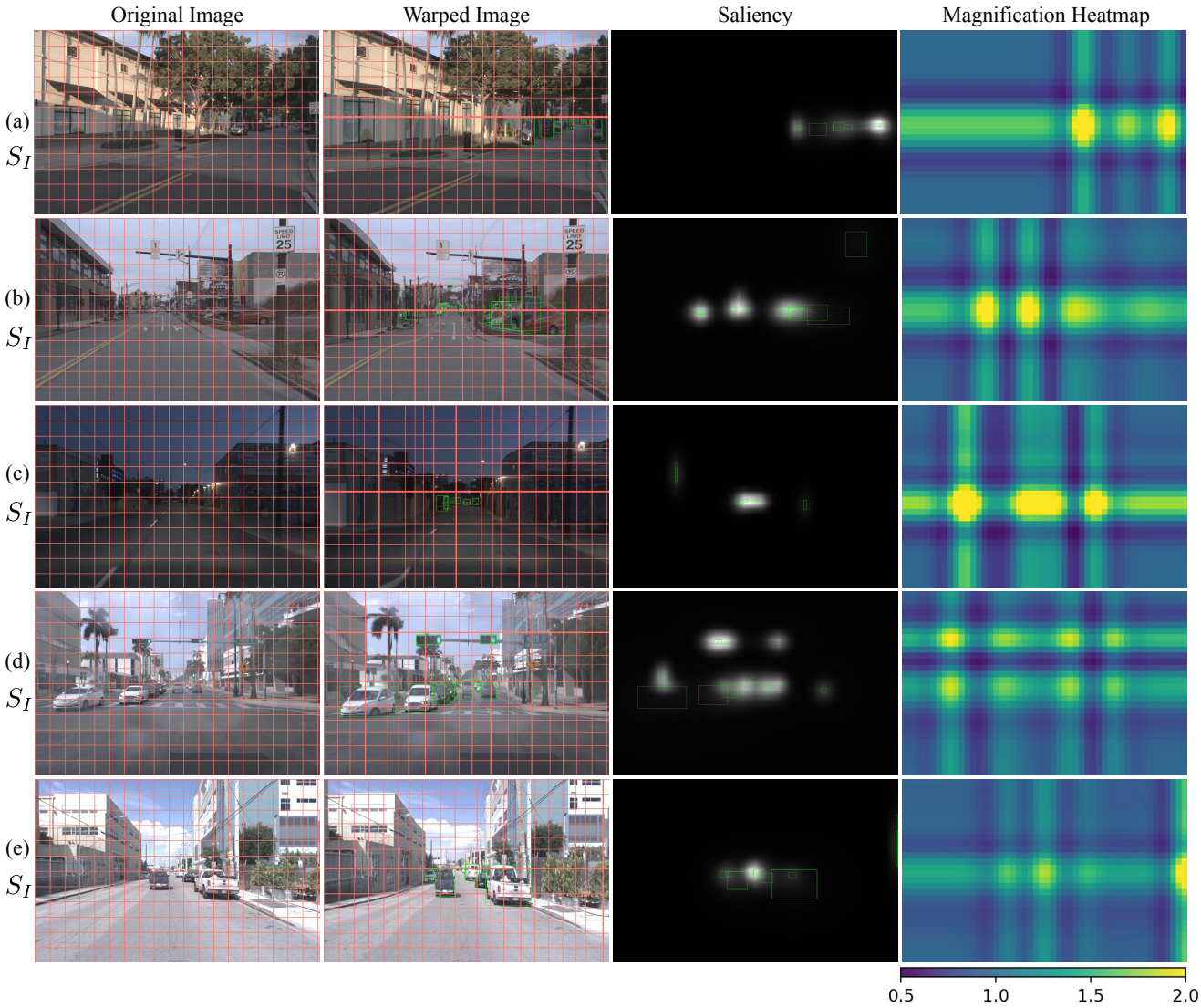

Figure B: Additional examples of the S_I KDE warping method. Bounding boxes on the saliency map denote previous frame detections, and bounding boxes on the warped image denote current frame detections. The magnification heatmap depicts the amount of magnification at different regions of the warped image. (a) is an example of S_I correctly adapting to an off-center horizon. (b) shows a multimodal saliency distribution, leading to a multimodal magnification in the x direction. (c) is another example of S_I correctly magnifying small objects in the horizon. (d) is a failure case in which duplicate detections of the traffic lights in the previous frame leads to more magnification than desired along that horizontal strip. One solution to this could be to weight our KDE kernels by the confidence of the detection. (e) is another failure case of S_I, in which a small clipped detection along the right edge leads to extreme magnification in that region. One general issue we observe is that the regions immediately adjacent to magnified regions are often contracted. This is visible in the magnification heatmaps as the blue shadows around magnified regions. This is a byproduct of the dropoff in attraction effect of the local attraction kernel. Perhaps using non-Gaussian kernels can mitigate this issue.

F. Additional Implementation Details

In this section, we provide additional details necessary to reproduce the results in the main text.

For the learned separable model from Sec 4.1.2, we use two arrays of length 31 to model saliency along the x and y dimensions, and during training, we blur the image with a 47×47 Gaussian filter in the first epoch, a trick introduced in [11] to force the model to zoom. For the learned nonseparable model, we use an 11×11 saliency grid, and we blur the image with a 31×31 filter in the first epoch. We use

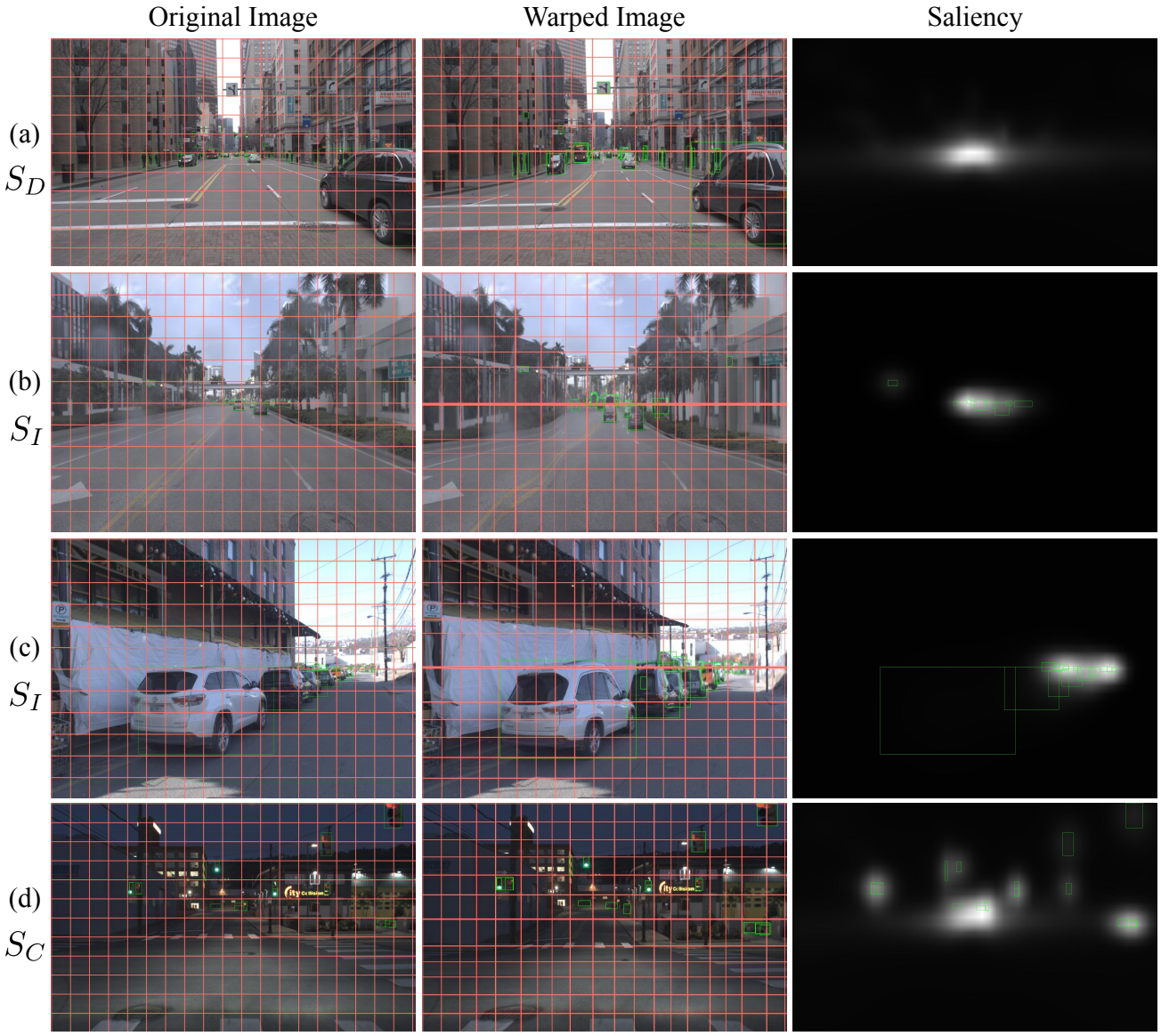

Figure C: Examples of KDE warp computed from bounding boxes, extracted from a training dataset (S_D) or the previous frame's detections (S_I, S_C). We visualize predicted bounding boxes in the warped image. Recall that large objects won't be visible in the saliency due to their large variance from Eq 8. (a) S_D magnifies the horizon (b) S_I magnifies the center of the image, similar to S_D (c) S_I adapts to magnify the mid-right region (d) S_C's saliency combines the temporal and spatial biases.

an attraction kernel k with a standard deviation of 5.5 for both versions. Additionally, we multiply the learning rate and weight decay of saliency parameters by 0.5 in the first epoch and 0.2 in the last two epochs, for stability. We find that we don't need anti-crop regularization here, because learning a fixed warp tends to behave nicely.

For each of our KDE methods, we use arrays of length 31 and 51 to model saliency in the vertical and horizontal directions, respectively. This is chosen to match the aspect ratio of the original input image and thereby preserve the vertical and horizontal "forces" exerted by the attraction kernel.

For the baseline detector, we adopt the Faster R-CNN implementation of mmdetection 2.7 [2]. All our experiments are conducted in an environment with PyTorch 1.6, CUDA 10.2 and cuDNN 7.6.5. For streaming evaluation, we mention a performance boost due to better implementa-

ID	Method	AP	AP_S	AP_M	AP_L
1	Prior art [7]	13.0	1.1	9.2	26.6
2	+ Better implementation	14.4	1.9	11.5	**27.9**
3	+ Train with pseudo GT	15.7	3.0	14.8	27.1
4	2 + Ours (S_I)	15.7	4.7	12.8	26.8
5	3 + Ours (S_I)	**17.1**	**5.5**	**15.1**	27.6

Table D: Streaming evaluation in the detection-only setting. First, we are able to improve over previous state-of-the-art through better implementation (row 2) and training with pseudo ground truth (row 3). Second, our proposed KDE warping further boosts the streaming accuracy (row 4-5).

tion in Tab D & Tab 2, and the changes are mainly adopting newer versions of mmdetection and cuDNN compared to the solution in [7] (switching from a smooth L1 loss to L1 loss for the regression part and code optimization).

References

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. 9

[2] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 11, 13

[3] Qiang Chen, Yingming Wang, Tong Yang, Xiangyu Zhang, Jian Cheng, and Jian Sun. You only look one-level feature. *CVPR*, 2021. 9, 11

[4] Piotr Dollár, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: An evaluation of the state of the art. *PAMI*, 34, 2012. 11

[5] 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. 11

[6] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. *NIPS*, 2015. 9, 11

[7] Mengtian Li, Yuxiong Wang, and Deva Ramanan. Towards streaming perception. In *ECCV*, 2020. 14

[8] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 9, 11

[9] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 9

[10] 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. 9

[11] 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. 9, 11, 12