Supplementary Material
Policy-based Foveated Imaging and Perception

A Method Details

A.1 Sensor Attention Parameters (Main Paper Sec. 3.1)

Our framework defines pixel bandwidth as a fraction of total pixels and is therefore resolution-agnostic. In our simulation results (Main Paper Sec. 4), we use SoccerNet, RoadText-1K, and ALOHA datasets at their native resolutions for reproducibility, and validate scalability to ultra-high-resolution captures using our 200 megapixel (MP) hardware prototype (Main Paper Sec. 5). The spatiotemporal tradeoffs analyzed in Main Paper Fig. 2 already emerge at these simulation resolutions.

In both the object tracking and text recognition tasks, we use 0.25× downsampled global images from the full-resolution images in the dataset as our policy input; Region-of-Interest (ROI) crops occupy 1/16 of the entire field of view and maintain aspect ratios, reducing the full-resolution bandwidth by more than 8×. In the robotic manipulation task, global images are downsampled by 0.17× (32× less bandwidth than full-resolution), and ROI crops occupy 1/16 of the entire field of view at a slight 0.7× downsampling (2× less bandwidth than full-resolution), reducing the full-resolution bandwidth by more than 16×. We do not apply temporal subsampling to the foveated acquisitions in downstream task evaluations across all simulated experiments.

A.2 Saliency Detection from Low-Resolution Context (Main Paper Sec. 3.2)

We use a YOLO-style detector across all simulated and real experiments for saliency detection. The YOLO-11 [Khanam and Hussain 2024] Nano model family is chosen for its favorable speed.

In the object tracking task, we randomly partition the SoccerNet Tracking Dataset [Cioppa et al. 2022] training split into 80% training and 20% validation. We finetune a YOLO-11 Nano model on the training split for 50 epochs with a batch size of 128. Other hyperparameters, including the input image size, are kept at their default values. We emulate low-resolution global frames by downsampling the training images with the same downsampling parameter (0.25×) for policy training. Bounding boxes from the tracking dataset are labeled as one of the three classes: player, referee, and ball. We select the best-performing model on the validation split and use it for saliency detection across all object tracking simulation experiments. We report precision and recall across all classes for the best-performing model on the validation split in Table 1. We use default hyperparameters at inference, including confidence thresholds and Non-Maximum Suppression (NMS) values.

Task	Precision	Recall
Object tracking	0.88	0.78
Scene text recognition	0.62	0.43

Table 1. **Saliency detection model evaluation**. The finetuned models demonstrate reasonable capability of detecting salient objects in unseen and blurry images.

In the text recognition task, we randomly partition the RoadText-1K dataset [Reddy et al. 2020] into 80% training and 20% validation, and finetune a YOLO-11 Nano-OBB model that predicts Oriented Bounding Boxes (OBB), as text is often not horizontal. Since RoadText-1K videos contain many frames with repetitive text detections, we further aggregate the training dataset using images and labels from TextOCR [Singh et al. 2021]. We finetune the model for 30 epochs with a batch size of 128, keeping the other hyperparameters at their default values. We emulate low-resolution global frames by downsampling the training images with the same downsampling parameter (0.25×) for policy training. Only one prediction class is used, as we are only interested in text predictions. We select the best-performing model on the validation split and use it for saliency detection in all scene text recognition simulation experiments. We report the precision and recall of the best-performing model on the validation split in Table 1. The same model is used for saliency detection in our real-world captures discussed in Main Paper Sec. 5.2. We use default hyperparameters at inference, including confidence thresholds and Non-Maximum Suppression (NMS) values.

For the robotic manipulation task, as no labeled saliency bounding boxes are available, we use the pre-trained YOLO-11 Nano model out-of-the-box and set the prediction confidence threshold to 0.02, the NMS value to 0.1, and enable class-agnostic NMS at inference.

Unlike downstream perception tasks (e.g. scene text recognition) that require fine-grained high-resolution details, the saliency detector only needs to *localize* candidate objects at the bounding-box level using coarse global features. Operating on low-resolution global frames is therefore sufficient for our saliency module, and our finetuned detectors achieve 0.88/0.78 precision/recall on tracking and 0.62/0.43 on scene text recognition at 0.25× downsampled input (Table 1). The robotic manipulation task further shows that the module generalizes to unseen environments: even though the ALOHA object classes do not match the pre-trained YOLO-11 Nano model's training labels, its detections are accurate enough for our policy to select task-relevant ROIs.

A.3 Motion Prediction (Main Paper Sec. 3.3)

We use the same hyperparameter settings for object association and motion prediction across all simulated and real experiments. For Hungarian matching, we use a minimum Intersection-over-Union (IoU) threshold of 0.1 and prioritize same-class detections.

SIGGRAPH Conference Papers '26, Los Angeles, CA, USA

ACM ISBN 979-8-4007-2554-8/2026/07
https://doi.org/10.1145/3799902.3811218

For motion prediction, we use a constant-velocity Kalman Filter with process noise set to 0.001 and measurement noise set to 0.01. When a tracklet is missing detections in the past T_o frames, we gap-fill the detections with a Kalman Filter using the same hyperparameters.

A.4 Scanpath Selection Policy (Main Paper Sec. 3.4)

We use the same Set Transformer [Lee et al. 2019] architecture for our scanpath selection policy across our simulated and real experiments.

High-resolution ROI features, global features, and detection and motion features are 64-dimensional each. High-resolution ROI object features are first extracted from the MobileNetV3-Small [Howard et al. 2019] visual encoder for objects contained in the previous foveated region, and then we use a 1D CNN to aggregate across the past T_o frames and output a 64-dimensional feature vector $\mathbf{r}_i^{(k)}$. Global features are directly extracted from the YOLO model's stride-8 layer using the RoIAlign method from Mask R-CNN [He et al. 2017] and temporally aggregated with a 1D CNN that uses different weights from the high-resolution feature's 1D CNN. Detection and motion bounding boxes are simply concatenated over time, and each stream is then projected to a 32-dimensional vector using a multi-layer perceptron (MLP).

Each object token $\mathbf{z}_i^{(k)}$ is a 192-dimensional feature vector. Each component (ROI features, global features, detection features, and motion features) is normalized independently using separate LayerNorm modules before we project the feature vector into a 128-dimensional embedding space. Object tokens are optionally concatenated with a conditioning token and pass through a standard 2-layer Transformer encoder with 4 attention heads and a 256-dimensional feedforward network. The Set Transformer allows permutation-invariant exchange of information about an object's spatial relationships, temporal dynamics, and task objective via the conditioning token. We use a maximum of 25 objects and attention masking across all simulated and real experiments. Only the object tokens are passed through the final MLP head, which produces selection logits over the next T_p frames. We apply softmax to obtain a categorical distribution over objects and sample from it for scanpath selection.

Training details. For all simulated experiments, during training and inference, we set $T_o = 4$ frames and $T_p = 16$ frames. During inference, we use receding-horizon control with a replanning interval of $T_a = 8$ frames. We use a batch size of 128 and train with the AdamW optimizer and a learning rate of 10^{-4} with cosine learning rate decay. For the object tracking task, we train for 3,000 iterations; for the scene text recognition task, 5,000 iterations; and for the robotic manipulation task, 10,000 iterations. 10% of the iterations are spent on learning rate warmup. For both simulated and real experiments, the total training time for all components of our foveated imaging pipeline is less than 12 hours on a single GPU with about 20 GB of VRAM.

A.5 Runtime Analysis (Main Paper Sec. 3.5)

Our policy is designed and optimized for real-time performance on low-end GPUs and CPUs. Table 2 summarizes runtime estimates for the different components of our foveated imaging framework on the laptop CPU used for our real-world captures. Most runtime is spent in the saliency detection module; despite this, our lightweight foveated imaging framework achieves real-time performance on CPUs with receding-horizon control.

Component	Parameters	GFLOPs / call	Runtime [ms]
YOLO (Saliency detection)	2.6M	6.5	55
MobileNetV3 (ROI feature)	2.5M	0.12	6.2
Kalman Filter (Motion prediction)	0	≈ 0	0.5
Set Transformer (Scanpath selection)	0.56M	0.01	3.0
Total	**5.66M**	**6.63**	**64.7**

Table 2. **Foveated imaging pipeline runtime evaluation**. Our full pipeline achieves about 2 frames of latency on a laptop CPU during 30 fps video capture.

The runtimes in Table 2 are measured on an Intel Core i7-7700HQ laptop CPU (16 GB of RAM) that also drives our 200 MP prototype via a Python API over USB 3.0. At 30 fps capture with receding-horizon control at $T_a = 8$ frames, three replans complete within each second, giving an aggregate compute of less than 80 GFLOPs/sec, dominated ($> 98\%$) by saliency detection. This compute requirement is well within the budgets of modern NPU-accelerated edge platforms such as smart glasses and drones.

B Experiment Details and Results

B.1 Downstream Models (Main Paper Sec. 4.2)

For object tracking, we use the MixFormerV2-Base model [Cui et al. 2023]. We set the search area scale to 5, the update interval to 10, and the online size to 1. We keep all hyperparameter settings consistent across all object tracking evaluations.

For scene text recognition, we use the DeepSolo model [Ye et al. 2023], finetuned on the ICDAR-15 Dataset [Zhou et al. 2015]. We intentionally disable automatic resizing and upsampling to a fixed resolution before the text detection and recognition pipeline for all experiments, as we compare the performance of our foveated imaging pipeline under both limited acquisition and processing bandwidth.

For robotic manipulation, we use the ALOHA Action Chunking Transformer (ACT) [Zhao et al. 2023]. We set the ACT's internal policy seed to 0 and report results across 100 random environment configurations in which the objects vary in position.

Combining FFoV and ROI streams. For object tracking and scene text recognition, the Full Field-of-View (FFoV) and ROI streams are fed *separately* into the downstream perception models. For robotic manipulation, the downstream ACT model is trained to consume a single image per view, so we upsample the FFoV frame and overlay the high-resolution ROI at the corresponding location before feeding the composite frame to the model. We hypothesize that providing sharp, task-relevant content at the ROI supplies consistent

visual attention cues that allow the frozen downstream model to maintain performance without retraining. Fine-tuning downstream perception models on mixed-resolution inputs remains future work.

B.2 Alternative Acquisition Baselines (Main Paper Sec. 4.3)

We compare against frame-skipping as a representative temporal downsampling baseline. An alternative temporal downsampling strategy that increases per-frame exposure yields substantially lower performance than frame-skipping (0.098, 0.071, 0.00 | 0.18 for the three Main Paper Table 1 tasks under the same bandwidth) due to motion blur.

Prior foveation methods discussed in Main Paper Sec. 2.1 all operate *post*-acquisition on already-captured full-resolution images or video. For example, Killick et al. [Killick et al. 2023] predict scanpaths on pre-captured images, and video-based methods such as AdaFocus [Wang et al. 2024] rely on global video features that are unavailable at acquisition time. Adapting such methods for predictive, acquisition-time foveated imaging would require significant architectural changes. Main Paper Table 1 compares against foveation baselines that are directly applicable at acquisition time, and the ablation studies (Main Paper Sec. 4.5) further analyze the design choices within our policy.

B.3 Additional Simulation Results (Main Paper Sec. 4.4)

For the object tracking task, the visual conditioning information consists of global visual features from the initial object location and the ground-truth (GT) bounding box. Varying this conditioning enables our scanpath selection policy to select different objects of interest. Table 3 summarizes quantitative results of our approach for different subjects in soccer tracking. We include qualitative visualizations of smooth-pursuit scanpaths for various tracked subjects in our supplemental video. Our method consistently outperforms relevant baselines at comparable bandwidth. Fig. 1 shows additional qualitative results across more diverse scenes for all three tasks. Across these scenes, our policy produces smooth-pursuit scanpaths for object tracking, rapid saccading scanpaths for scene text recognition, and a mixture of both for robotic manipulation.

Performance under different pixel budgets. To test how performance varies with bandwidth budget, we repeat the Main Paper Table 1 comparisons at higher and lower pixel budgets. Results are summarized in Table 4. Our approach maintains the best performance across all budgets for all tasks, with the largest advantage occurring when conventional downsampling discards the most critical spatiotemporal details.

C Hardware Prototype

C.1 200 MP Foveated Imaging Prototype (Main Paper Sec. 5)

Ultra-high-resolution sensors such as the 200 MP Samsung ISOCELL HP2 [Samsung Electronics Co., Ltd. 2025] are physically bandwidth-limited: the HP2 is capped at 15 fps at full resolution, with higher frame rates achievable only through pixel binning that reduces spatial resolution to 50 MP or 12.5 MP.

In our experimental setup, the prototype's hardware further restricts FFoV readout to 7.5 fps while the sensor captures at 30 fps. In addition, while the sensor is saving dual-stream raw Bayer frames at 30 fps, the sensor-captured frames are not accessible via the sensor's Python API. Therefore, we re-train both scanpath selection policies (for object tracking and scene text recognition) with $T_o = 10$ and $T_p = 200$ frames. Other hyperparameters are kept consistent with each policy's simulated-experiment training.

During real-world sensor captures, we capture up to 105 raw Bayer frames in dual-stream mode (both the FFoV and ROI streams have a 30 fps frame rate) while the sensor executes the ROI predictions from our foveated imaging pipeline. We use the replanning interval $T_a = 180$ in our real experiments. Without the sensor readout latency, our pipeline operates with a frame delay of about 12.8 frames in the setting of $T_o = 4$ frames and $T_p = 16$ frames used in our simulated experiments. In real-world captures, our policy operates with a frame delay of about 31.3 frames in the setting of $T_o = 10$ frames and $T_p = 200$ frames. Adding the sensor readout latency, the entire foveated imaging pipeline operates with approximately 60 frames delay in this setting, well below the receding control horizon of $T_a = 180$ frames.

C.2 Comparison to Samsung ISOCELL Zoom Anyplace

Samsung ISOCELL Zoom Anyplace [Samsung Electronics Co., Ltd. 2026] is a user-driven, single-object tracking feature supported on the same ISOCELL HP2 sensor we use for our prototype. In contrast, our framework is neither user-driven nor hard-coded to a single task: it autonomously learns sensor attention from downstream perception objectives and generalizes across object tracking, scene text recognition, and robotic manipulation. Our contribution is not an open-source reimplementation of proprietary firmware on specific hardware, but rather an intelligent sensing framework that is sensor-agnostic, task-general, and extensible to other high-bandwidth sensing modalities, multi-camera systems, or closed-loop robotic pipelines.

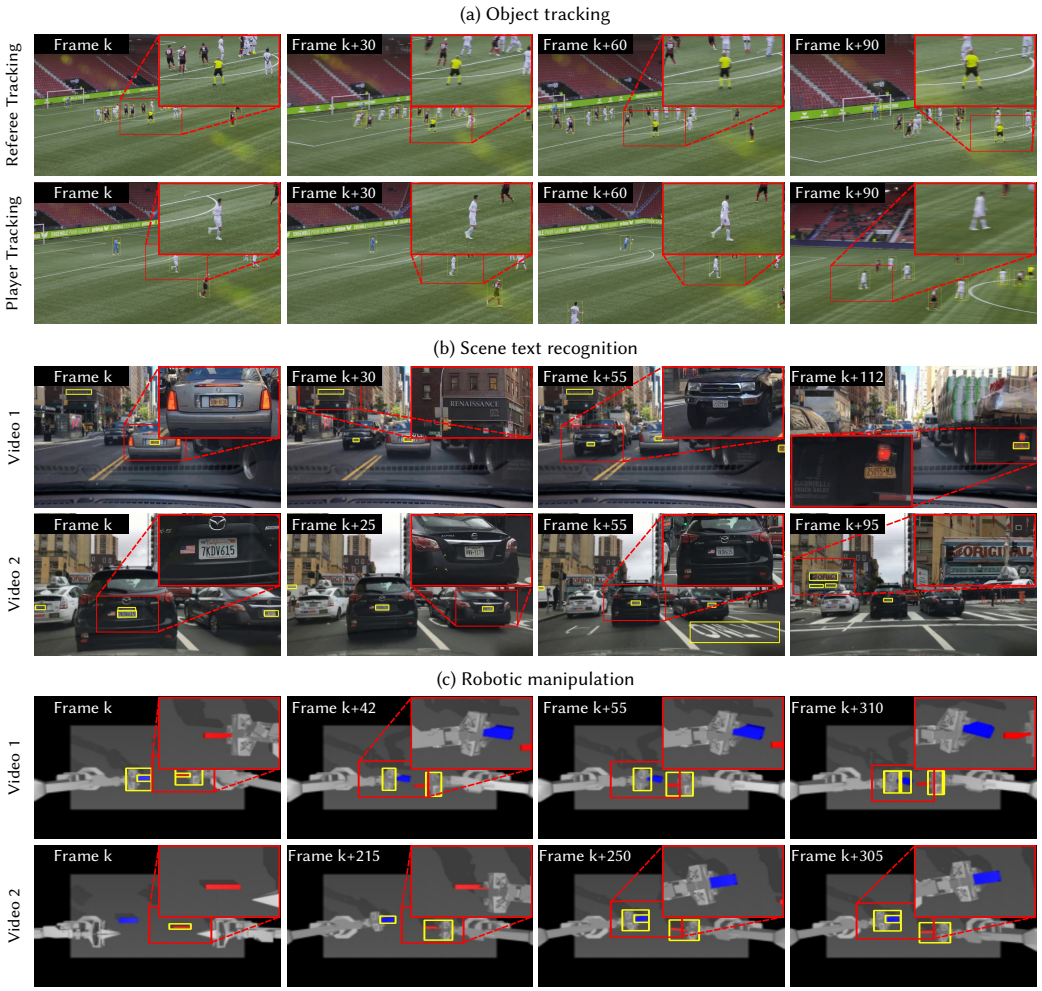

Fig. 1. **Additional qualitative results for simulated video tasks.** We show additional examples of our foveated imaging framework across diverse scenes for object tracking, scene text recognition, and robotic manipulation, demonstrating consistent performance improvements over task-agnostic baselines.

Tracking subject	Metric	Full-resolution (8× bandwidth)	GT Oracle	Spatial downsampling	Temporal downsampling	**Foveated**
Soccer ball	IoU ↑	0.281	0.405	0.122	0.148	**0.283**
Main referee	IoU ↑	0.494	0.602	0.474	0.456	**0.583**
Random player	IoU ↑	0.457	0.552	0.268	0.402	**0.513**

Table 3. **Additional results on policy-based foveated perception for object tracking.** We compare the downstream object tracking task performance for different classes of objects, including the soccer ball, the referees, and the players. Our approach performs best among relevant baselines at comparable bandwidth across all tracking scenarios.

Task	Pixel budget	Spatial downsampling	Temporal downsampling	**Ours**
Object Tracking	1/4	0.212	0.159	**0.287**
Object Tracking	1/8†	0.122	0.148	**0.283**
Object Tracking	1/16	0.048	0.144	**0.276**
Text Recognition	1/4	0.146	0.283	**0.290**
Text Recognition	1/8†	0.067	0.248	**0.264**
Text Recognition	1/16	0.023	0.173	**0.227**
Robotic Manipulation	1/8	0.12 | 0.54	0.08 | 0.38	**0.13** | **0.59**
Robotic Manipulation	1/16†	0.10 | 0.51	0.07 | 0.30	**0.12** | **0.57**
Robotic Manipulation	1/32	0.04 | 0.38	0.02 | 0.20	**0.08** | **0.40**

Table 4. **Pixel-budget sensitivity.** Our method outperforms spatial and temporal downsampling baselines at higher and lower pixel budgets across all three tasks. Rows marked with † correspond to the pixel budgets reported in Main Paper Table 1 and are reproduced here for direct comparison against the additional budgets evaluated in this supplement.

References

Anthony Cioppa, Silvio Giancola, Adrien Deliege, Le Kang, Xin Zhou, Zhiyu Cheng, Bernard Ghanem, and Marc Van Droogenbroeck. 2022. SoccerNet-Tracking: Multiple Object Tracking Dataset and Benchmark in Soccer Videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3491–3502.

Yutao Cui, Tianhui Song, Gangshan Wu, and Limin Wang. 2023. Mixformerv2: Efficient fully transformer tracking. *Advances in neural information processing systems* 36 (2023), 58736–58751.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*. 2961–2969.

Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. 2019. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*. 1314–1324.

Rahima Khanam and Muhammad Hussain. 2024. Yolov11: An overview of the key architectural enhancements. *arXiv preprint arXiv:2410.17725* (2024).

George Killick, Paul Henderson, Paul Siebert, and Gerardo Aragon-Camarasa. 2023. Foveation in the era of deep learning. *arXiv preprint arXiv:2312.01450* (2023).

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. 2019. Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*. PMLR, 3744–3753.

Sangeeth Reddy, Minesh Mathew, Lluis Gomez, Marçal Rusinol, Dimosthenis Karatzas, and CV Jawahar. 2020. Roadtext-1k: Text detection & recognition dataset for driving videos. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 11074–11080.

Samsung Electronics Co., Ltd. 2025. ISOCELL HP2 | Mobile Image Sensor. https://semiconductor.samsung.com/image-sensor/mobile-image-sensor/isocell-hp2/ Accessed 2025-11-11.

Samsung Electronics Co., Ltd. 2026. ISOCELL Zoom Anyplace. https://semiconductor.samsung.com/technologies/image-sensor/ultra-high-resolution/isocell-zoom-anyplace/ Accessed 2026-04-19.

Amanpreet Singh, Guan Pang, Mandy Toh, Jing Huang, Wojciech Galuba, and Tal Hassner. 2021. Textocr: Towards large-scale end-to-end reasoning for arbitrary-shaped scene text. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 8802–8812.

Yulin Wang, Haoji Zhang, Yang Yue, Shiji Song, Chao Deng, Junlan Feng, and Gao Huang. 2024. Uni-adafocus: spatial-temporal dynamic computation for video recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024).

Maoyuan Ye, Jing Zhang, Shanshan Zhao, Juhua Liu, Tongliang Liu, Bo Du, and Dacheng Tao. 2023. DeepSolo: Let Transformer Decoder with Explicit Points Solo for Text Spotting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 19348–19357.

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. 2023. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705* (2023).

Xinyu Zhou, Shuchang Zhou, Cong Yao, Zhimin Cao, and Qi Yin. 2015. Icdar 2015 text reading in the wild competition. *arXiv preprint arXiv:1506.03184* (2015).