Detection-and-Tracking-Oriented Roadside Camera Placement via Observation-Augmented Bayesian Optimization

Masahiro Hirano[1], Hayato Fujii[2], Yuji Yamakawa[3]

***Abstract*— This paper proposes a roadside camera placement method that directly maximizes detection and tracking performance via Bayesian optimization. Conventional geometric approaches model the sensing space explicitly and optimize sensor layouts to maximize spatial coverage; however, they typically assume homogeneous sensors and environments, and therefore cannot reflect the performance of the actual perception algorithms to be deployed, nor site-specific factors that affect sensing (e.g., low-angle sunlight). To address this limitation, we design an evaluation metric for a roadside camera system that is computed in simulation using a practical object detection and tracking pipeline, and we directly maximize the metric via Bayesian optimization. Furthermore, to reduce the impact of the computational cost dominated by simulation in Bayesian optimization, we propose to place more cameras than the intended number to augment observations that can be evaluated per simulation run, thereby significantly reducing the overall optimization time. Experiments demonstrate that our method successfully derives a camera placement with performance superior to the best placement designed by multiple human participants, while achieving more than a 4 times speedup compared to the optimization without observation augmentation.**

I. Introduction

Infrastructure-cooperative automated driving has been an emerging trend in ITS, aiming to improve driving safety by transmitting object information detected by sensors installed in the traffic environment to automated vehicles. By complementing onboard perception with infrastructure sensing, such systems enable automated vehicles to recognize objects in occluded regions that are otherwise blind spots, thereby targeting a higher level of safety, which is typically required for public transportation. To deploy such systems in traffic environments, it is crucial to determine where to install roadside sensors so that traffic participants in the environment can be detected and tracked, reliably and comprehensively.

Roadside sensors are typically installed by engineers who are familiar with the sensor characteristics. However, a sensor itself has various parameters such as installation location, orientation, and field of view, and the sensing space often exhibits a complex three-dimensional structure with occluders such as trees. In addition, many site-specific conditions, such as low-angle sunlight, can strongly affect perception, making it difficult to manually derive effective sensor layouts while accounting for all relevant factors. Moreover, to scale such systems across multiple regions, establishing an automated methodology for sensor placement is highly expected. In this work, we specifically focus on roadside cameras and propose a method to derive optimal camera placements (Fig. 1).

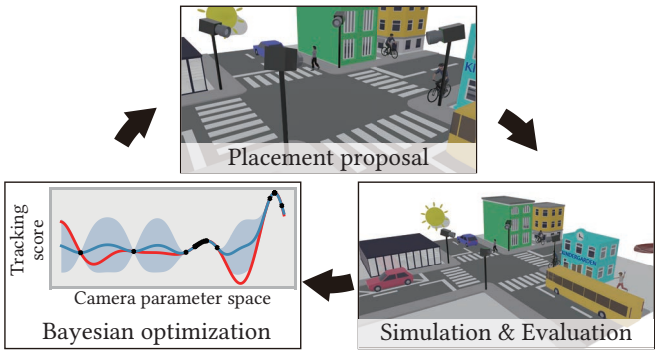

Fig. 1. Roadside camera placement is obtained via Bayesian optimization with realistic traffic simulation and multi-camera tracking evaluation.

As rooted in classical computational geometry, a wide range of studies has addressed sensor placement optimization, where many methods geometrically model the sensor field of view and aim to maximize coverage of the road space. Such geometric methods are often formulated as mathematical optimization problems for relatively simple structures such as highways, and can be difficult to apply to complex 3D scenes such as intersections. In addition, these methods typically regard an object as "detected" if it lies within the camera frustum, which fails to account for real-world factors that significantly affect vision-based detection performance, including illumination conditions. As a result, they do not necessarily reflect the actual detection and tracking performance of the deployed sensing system under realistic conditions.

In contrast, we propose an optimization framework that places a camera system on a digital twin reproducing the target environment, introduces a performance metric based on the Detection and Tracking algorithms used for recognition, and directly maximizes the metric. Specifically, we run traffic simulation on a digital twin that captures the properties of the target space, obtain image sequences rendered from the deployed cameras, and compute a performance score using an object detection/tracking pipeline that includes image processing. Because the resulting objective function implicitly involves complex detection and tracking procedures, it cannot be written in a closed form and is difficult to optimize in an analytical way. Therefore, we adopt Bayesian optimization, an approach for optimizing black-box objective functions that are difficult to model explicitly. This

[1]Institute of Industrial Science, The University of Tokyo, Tokyo, Japan. mhirano@iis.u-tokyo.ac.jp

approach can naturally incorporate complex 3D structures through a digital twin, and can also reflect hard-to-model environmental effects (e.g., low-angle sunlight) by adjusting illumination conditions in simulation.

The Bayesian optimization loop requires repeated execution of time-consuming processes, namely simulation and object detection/tracking, which raises a challenge in computational efficiency. To address this issue, we propose a method that substantially improves efficiency by augmenting the observations used in Bayesian optimization within a single simulation run, which we call *Observation Augmentation*. By placing more cameras than the planned number at a cycle, we can gain a combinatorial number of camera subsets that can be evaluated, thereby increasing the number of observations available for Bayesian learning per cycle.

To evaluate the proposed method, we optimized camera placement on a digital twin and compared it against placements designed by human participants. We show that the proposed method realized a camera placement with object detection/tracking performance superior to human-designed placements. Moreover, with observation augmentation, we achieved more than a 4 times efficiency improvement in terms of computation time compared to evaluating only one camera placement per cycle.

II. Related Work

Sensor placement optimization has long drawn attention from the research community. It dates back to the Art Gallery Problem [1], which seeks the minimum number of cameras (guards) required to cover the entire interior of a given polygon (gallery). This problem can be viewed as minimizing the number of cameras under coverage constraints, and has been widely studied in computational geometry.

In the 2000s, sensor placement optimization began to incorporate camera properties such as field of view and resolution. For example, studies have derived indoor surveillance camera layouts that maximize coverage using genetic algorithms under physical camera models [2]. Malhotra *et al.* proposed a combinatorial optimization method to cover a 3D space with fewer cameras [3]. However, these approaches are limited to optimization within closed indoor spaces. More recently, sensor placement optimization methods grounded in practical industrial scenarios have been proposed. Kim *et al.* [4] optimize coverage more practically by weighting important areas such as hazardous zones at construction sites. In the autonomous driving domain, heuristic algorithms have also been used to optimize configurations of onboard cameras [5].

Many studies have also investigated roadside sensor placement for infrastructure-cooperative automated driving systems. Examples include camera placement that minimizes the number of cameras under coverage constraints in highway merging areas [6], coverage optimization at intersections using roadside LiDAR based on integer programming [7]. More recently, approaches have leveraged 3D digital maps, including height information, to capture objects in 3D and optimize infrastructure sensor placement using genetic algorithms [8]. There exist studies that evaluate object detection algorithms using simulators; however, their optimization solves an integer programming problem that aims to maximize coverage [9].

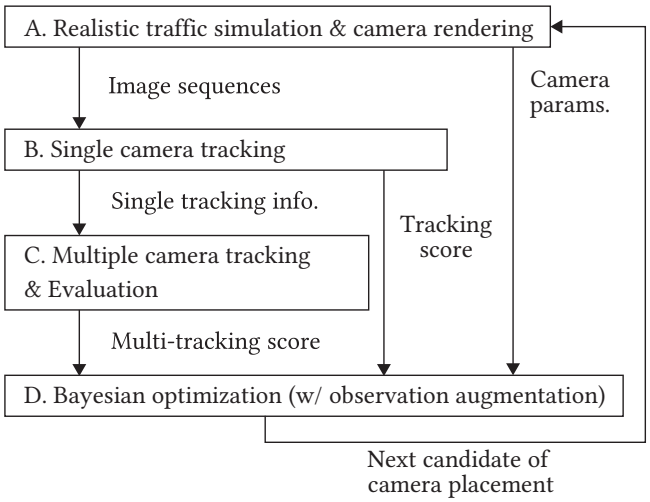

Fig. 2. Overview of the proposed method.

However, many methods are limited to relatively simple measurement spaces, and the optimization objective is often restricted to spatial coverage, which does not account for the actual detection performance of the sensors, posing practical limitations as addressed in [10] in the context of sensor fusion. To address these issues, our method reproduces complex 3D urban environments via a digital-twin-based simulator and proposes a roadside camera placement method that directly maximizes a performance metric reflecting detection and tracking accuracy achieved by the sensors. By optimizing an objective function defined by a metric reflecting the system's performance using realistic sensors and detection/tracking pipeline—rather than geometric coverage—the proposed method can account for dynamic and diverse traffic environments, including weather, illumination, and various traffic flow patterns, as well as region-specific external conditions reproduced in simulation.

III. Optimal Road-side Camera Placement via Bayesian Optimization

A. Overview

To derive a camera placement that maximizes object perception performance, we propose a method that maximizes a newly proposed roadside camera system metric based on a detection and tracking pipeline via Bayesian optimization. The overview of the proposed method is shown in Fig. 2. Given a 3D digital twin reproducing the target space, (A) we place cameras with parameters to be estimated (e.g., position and focal length), run traffic simulation, and obtain image sequences for each camera. Next, (B) we perform object detection and tracking on each camera stream, and (C) integrate per-camera tracking results to compute a multi-camera tracking score. Then, (D) we run Bayesian optimization using

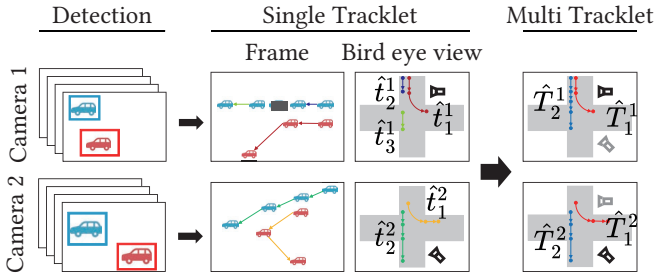

Fig. 3. Procedure of Multi-Camera Multi-Object Tracking. Single tracklets for each camera are represented in different colors, while multi-tracklets, obtained by associating them across cameras, are represented in the same colors.

the computed tracking score and camera parameters, and propose camera parameters to be evaluated in the next cycle. By iterating this cycle, we obtain camera placements with higher performance. The following subsections describe each step in detail.

B. Realistic Traffic Simulation

To reproduce real-world traffic environments, we use a realistic traffic simulator based on a 3D city digital twin. We simulate real traffic conditions, such as vehicle flows, and obtain image sequences from multiple cameras placed roadside. In this paper, for n cameras C_k $(k = 1, 2, \ldots, n)$ to be optimized, we parameterize each camera by its 3D position (x_k, y_k, z_k), pitch angle θ_k, yaw angle ϕ_k, and focal length f_k, and define the parameter set to be optimized as

$$X = \{x_k, y_k, z_k, \theta_k, \phi_k, f_k\}_{k=1,2,\ldots,n}. \tag{1}$$

We optimize focal length because it is a key parameter that determines the field of view. Considering practical installation settings, we fix the roll angle corresponding to rotation around the optical axis to 0.

C. Tracking Score

1) Tracklet: This subsection describes how to design a metric to evaluate a camera placement reproduced in traffic simulation. To measure how comprehensively multiple roadside cameras can track many vehicles in simulation, we adopt an evaluation metric that quantifies the degree of vehicle tracking. We first perform Multi-Camera Multi-Object Tracking (MCMOT), which continuously detects and tracks multiple objects that move across multiple cameras. By running MCMOT, we obtain object trajectories referred to as *tracklets*, and later compare them against ground truth obtained from the simulator to compute a quantitative metric.

In MCMOT, objects are first detected in each camera stream, tracked within each camera, and finally associated across cameras to achieve multi-camera tracking as depicted in Fig. 3. Specifically, after performing object detection per frame using a method such as YOLO [11], we obtain predicted *Single Tracklets* in each camera based on the detection locations and appearance features. Let the set of predicted Single Tracklets in camera k over all detected vehicle IDs p be denoted by $\hat{\mathbf{t}}^k$:

$$\hat{\mathbf{t}}^k = \{\hat{t}_1^k, \hat{t}_2^k, \ldots, \hat{t}_p^k, \ldots, \hat{t}_{\hat{N}_s^k}^k\}, \tag{2}$$

where $\hat{N}_s^k$ is the total number of predicted Single Tracklets in camera k. (We denote predicted variables with a hat $\hat{}$ to distinguish them from ground-truth variables.)

The predicted Single Tracklets from each camera are then associated across cameras based on appearance features and merged by assigning consistent IDs. The resulting trajectories are referred to as predicted *Multi Tracklets*. Let the set of predicted Multi Tracklets in camera k over all vehicles be denoted by $\hat{\mathbf{T}}^k$:

$$\hat{\mathbf{T}}^k = \{\hat{T}_1^k, \hat{T}_2^k, \ldots, \hat{T}_p^k, \ldots, \hat{T}_{\hat{N}_m^k}^k\}, \tag{3}$$

where $\hat{N}_m^k$ is the total number of predicted Multi Tracklets in camera k. Note that the Multi Tracklets are defined per camera.

2) Tracking status: To compute an evaluation metric for camera placement, we design a metric based on the ratio of time during which vehicles are correctly tracked within the target region, relative to the total dwell time of vehicles in that region. Here, for vehicles within the target region, if a vehicle is tracked by at least one camera at a given frame, we regard the system as successfully tracking that vehicle at that frame.

We first determine, for each camera, the per-frame *Tracking status*, which indicates whether each vehicle appearing in the target region during simulation is tracked or not. To determine whether a vehicle is tracked, we match the predicted Multi Tracklet set $\hat{\mathbf{T}}^k$ in Eq. (3) against the ground truth obtained by the simulator.

Let $\mathbf{T}^k$ be the ground-truth Multi Tracklet set. With N_m^k denoting the ground-truth total number of vehicle IDs, we define

$$\mathbf{T}^k = \{T_1^k, T_2^k, \ldots, T_q^k, \ldots, T_{N_m^k}^k\}. \tag{4}$$

To determine whether a predicted Multi Tracklet $\hat{T}_p^k$ $(p = 1, 2, \ldots, \hat{N}_m^k)$ correctly tracks an object, we need to identify the corresponding ground-truth Multi Tracklet T_q^k $(q = 1, 2, \ldots, N_m^k)$. We define an assignment cost using the Intersection over Union (IoU) between the ground-truth and predicted bounding boxes, and solve the assignment by minimizing the total cost using the Hungarian algorithm [12].

After solving this assignment, if a predicted Multi Tracklet $\hat{T}_p^k$ exists for a ground-truth Multi Tracklet T_q^k and the two tracklets share a common frame index, then the vehicle is considered correctly tracked at that frame, i.e., marked as *Tracked*. Otherwise (no corresponding tracklet, or no shared frame index), it is marked as *Untracked*. Formally, for camera k, we define a variable $\Theta_k(i, q)$ representing the tracking state of the q-th ground-truth Multi Tracklet at frame i as

$$\Theta_k(i, q) = \begin{cases} 1 & \text{if } T_q^k \text{ is } \textit{Tracked} \text{ at frame } i \\ 0 & \text{if } T_q^k \text{ is } \textit{Untracked} \text{ at frame } i. \end{cases} \tag{5}$$

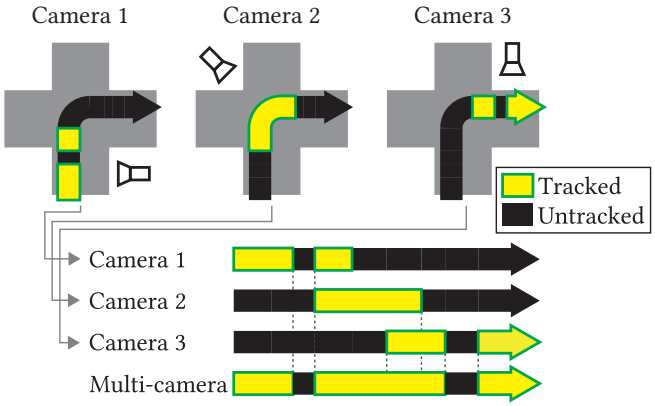

Fig. 4. Multi-camera tracking status.

3) Tracking score: We then integrate the Tracked status across cameras and compute a multi-camera tracking score. Fig. 4 illustrates the computation with three cameras. Since tracking a vehicle requires that it be in the *Tracked* state in at least one camera, we introduce a variable $\Phi(i,q)$ indicating whether the q-th vehicle is *Tracked* at frame i in at least one camera:

$$\Phi(i,q) = \begin{cases} 1 & \text{if there exists a } k \text{ s.t. } \Theta_k(i,q) = 1 \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

We also define a variable $\Psi(i,q)$ indicating whether the ground-truth Multi Tracklet T_q^k exists within the target region $\mathcal{S}$ at frame i:

$$\Psi(i,q) = \begin{cases} 1 & \text{if } T_q^k \text{ at frame } i \text{ exists in } \mathcal{S} \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

Finally, we define the quantitative evaluation metric for camera placement. We define the multi-tracking score as the ratio, aggregated over all vehicles, between the number of frames where the vehicle is *Tracked* by at least one camera and the number of frames where the vehicle exists in the target region:

$$y_{multi} = \frac{\sum_{q=1}^{N}\sum_{i=1}^{I}\Phi(i,q)}{\sum_{q=1}^{N}\sum_{i=1}^{I}\Psi(i,q)}. \tag{8}$$

Here, N is the total number of vehicles that appear in the target space during the simulation, and I is the total number of frames.

We similarly define a per-camera tracking score y_k using the predicted Single Tracklets $\hat{\mathbf{t}}^k$ and its corresponding ground-truth $\mathbf{T}^k$. Introducing a variable $\theta_k(i,q)$ representing the tracking state of the q-th vehicle at frame i in camera k, the single-camera tracking score y_k is defined as

$$y_k = \frac{\sum_{q=1}^{N}\sum_{i=1}^{I}\theta_k(i,q)}{\sum_{q=1}^{N}\sum_{i=1}^{I}\Psi(i,q)}. \tag{9}$$

D. Bayesian Optimization

To find a camera placement that maximizes the tracking scores described above, we use Bayesian optimization to learn the relationship between camera placement parameters and tracking scores, and to identify promising placements that are likely to yield high tracking performance. By iteratively using the tracking score obtained from simulation with the acquired camera placement as additional training data, we aim to obtain camera placements with higher tracking scores.

To reflect the contribution of each camera more efficiently during learning, we incorporate not only the multi-camera tracking score but also the per-camera tracking scores into the objective. That is, we formulate the problem as multi-objective optimization with $Y = \{y_{multi}, y_1, \ldots, y_n\}$.

In Bayesian optimization, we fit a Gaussian process model $Y = \mathcal{GP}(X)$ with a Matérn kernel to a set of known pairs $\{(X_1, Y_1), \ldots, (X_N, Y_N)\}$ of camera parameters X_k and tracking scores Y_k. Given the fitted model, we obtain the next camera parameters to evaluate, X_{N+1}, by maximizing an acquisition function called EHVI (Expected Hypervolume Improvement). Since feasible camera installation locations are limited, we perform Bayesian optimization under positional constraints; details of the constraints used in experiments are described in the Evaluation section.

E. Observation Augmentation for Efficient Optimization

When optimizing camera placement by repeating steps A–D in Fig. 2, the computation time for (A) simulation and (B) detection/tracking in each camera is substantially larger than that for (C) multi-camera tracking and (D) Bayesian optimization. This yields an efficiency bottleneck for the overall cycle. To mitigate the computational cost dominated by simulation, we propose to efficiently search the parameter space by augmenting the number of observations obtained per cycle. Specifically, we increase the number of cameras placed in a single cycle beyond the number used for evaluation, and compute the multi-tracking score for multiple camera subsets, thereby acquiring multiple observations in the parameter space simultaneously.

Concretely, to evaluate a placement of n cameras, we place l $(n \leq l)$ cameras in simulation. Because the acquisition function in Bayesian optimization encourages exploration, it may occasionally place cameras with extremely low tracking scores. We exclude the m $(0 \leq m \leq l-n)$ cameras with low single-camera tracking scores and compute the evaluation metric for all combinations of choosing n cameras from the remaining $l-m$ cameras. As a result, we can acquire $\binom{l-m}{n}$ observations from a single cycle[1]. We refer to this as *multi-point observation*, in contrast to *single-point observation*, where only one observation is evaluated per cycle.

Fig. 5 compares the time required to learn $\binom{l-m}{n}$ observations under single-point and multi-point observation. In each cycle, let the time for traffic simulation (A) be t_a, the total time for per-camera tracking processing (B) be t_b, the

[1] $\binom{l-m}{n}$ denotes the number of ways to choose n samples from $l-m$ candidates.

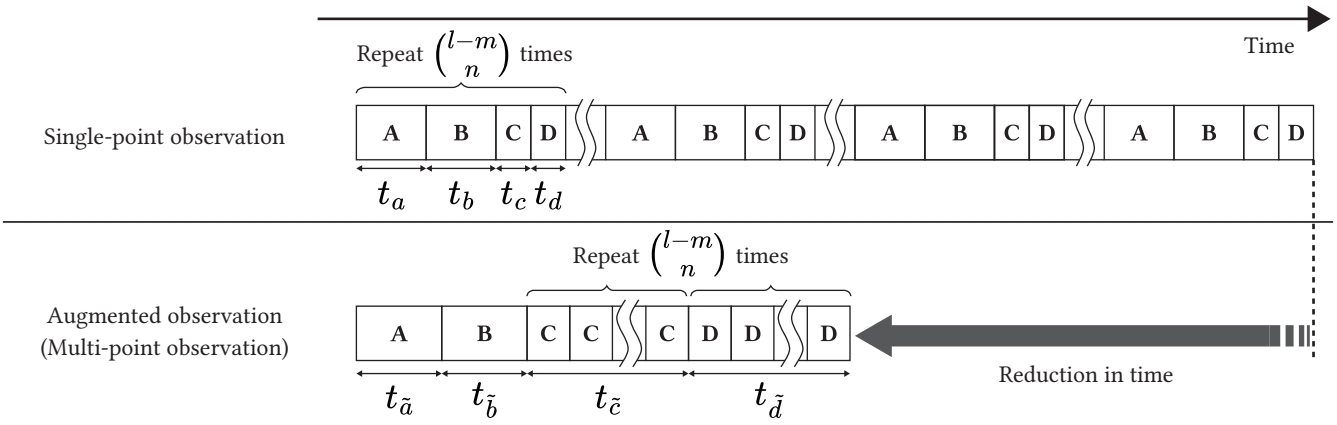

Fig. 5. Time chart of augmented observation for efficient optimization. With multiple-point observation, the time required to learn the same number of observations is significantly reduced compared to single-point observation.

time for integration and multi-tracking score computation (C) be t_c, and the time for Bayesian optimization (D) be t_d. In single-point observation, since one simulation yields only one observation, the time T required to learn $\binom{l-m}{n}$ observations is

$$T = \binom{l-m}{n} \cdot (t_a + t_b + t_c + t_d). \tag{10}$$

In multi-point observation, let the time for each phase be $t_{\tilde{a}}, t_{\tilde{b}}, t_{\tilde{c}}, t_{\tilde{d}}$, respectively. Then, the time $\tilde{T}$ required to learn $\binom{l-m}{n}$ observations is

$$\tilde{T} = t_{\tilde{a}} + t_{\tilde{b}} + t_{\tilde{c}} + t_{\tilde{d}}. \tag{11}$$

The simulation time is comparable to the single-point case in terms of the core traffic simulation, but increases due to additional rendering and file I/O; thus $t_{\tilde{a}} < \frac{l}{n} \cdot t_a$. The per-camera tracking time $t_{\tilde{b}}$ scales approximately with the number of cameras, so $t_{\tilde{b}} \simeq \frac{l}{n} \cdot t_b$ (we ignore parallelization effects and variations due to whether objects are detected for brevity). The integration and score computation time $t_{\tilde{c}}$ must be executed $\binom{l-m}{n}$ times, so $t_{\tilde{c}} \simeq \binom{l-m}{n} \cdot t_c$, so does the Bayesian optimization, yielding $t_{\tilde{d}} \simeq \binom{l-m}{n} \cdot t_d$.

Therefore, the time $\tilde{T}$ required by multi-point observation satisfies

$$\begin{aligned}\tilde{T} &= t_{\tilde{a}} + t_{\tilde{b}} + t_{\tilde{c}} + t_{\tilde{d}} \\ &< \frac{l}{n} \cdot t_a + \frac{l}{n} \cdot t_b + \binom{l-m}{n} \cdot t_c + \binom{l-m}{n} \cdot t_d. \end{aligned} \tag{12}$$

As shown in Table II, $t_a + t_b \gg t_c + t_d$. Furthermore, when $n > 2$, $\binom{l-m}{n} \gg \frac{l}{n}$ (e.g., $n = 4, l = 8, m = 2$ yields $\binom{l-m}{n} = 15$ and $\frac{l}{n} = 2$). Thus, multi-point observation is expected to significantly reduce the time needed to learn $\binom{l-m}{n}$ observations compared to single-point observation.

IV. Evaluation

A. Experimental Setup

We used an open-source driving simulator AWSIM [13] as the experimental environment, which emulates a realistic scene that involve vehicles, sensors, driving environments,

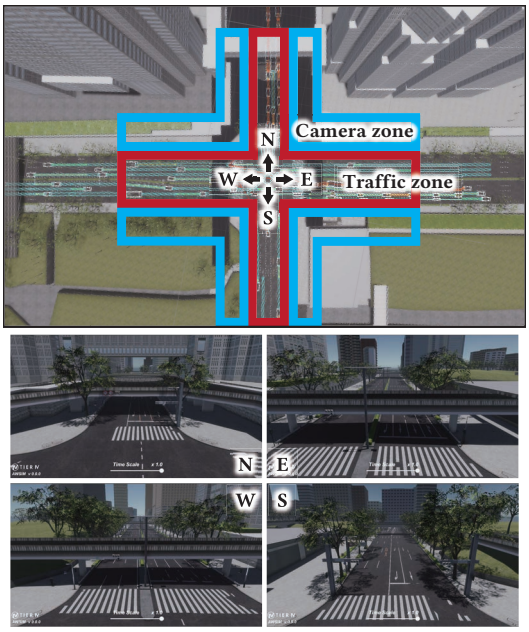

Fig. 6. Experiment scene for camera placement. The blue zones represent the allowable camera installation areas, whereas the red zone indicates the traffic zone where vehicles travel.

and traffic flows. We used a publicly available digital twin model of Nishi-Shinjuku and targeted an intersection and the surrounding area shown in Fig. 6. This scene is characterized by heavy traffic and complex environmental factors relevant to camera placement, including roadside trees, street lights, and spaces under elevated structures. Among vehicle types prepared by the simulator, we used a taxi, truck, hatchback, compact passenger car, and van. We customized the simulator to allow random color selection out of multiple colors for each vehicle to realize a more diverse traffic environment.

For each camera k, we imposed bounds on parameters $x_k, y_k, z_k, \theta_k, \phi_k, f_k$ as

$$-60 \le x_k \le 60,\ 0 \le y_k \le 4,\ -60 \le z_k \le 60, \tag{13}$$

$$0 \le \theta_k \le 30,\ -180 \le \phi_k \le 180,\ 10 \le f_k \le 30. \tag{14}$$

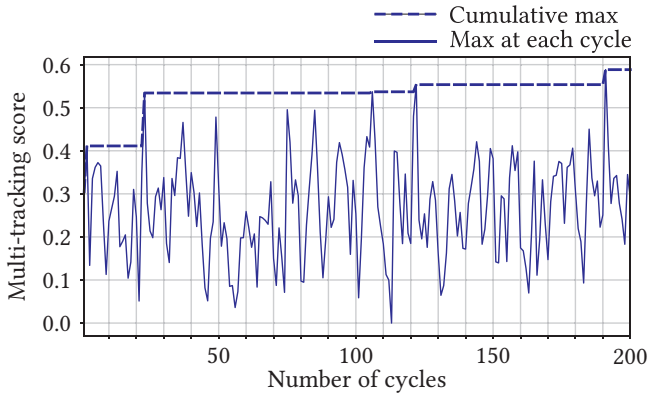

Fig. 7. Cumulative maximum multi-tracking score. The solid line indicates the maximum multi-tracking score in each cycle, and the dotted line shows the cumulative maximum multi-tracking score.

In addition, we imposed inequality constraints on the camera positions so that cameras are placed within the roadside areas indicated by the blue outlines in Fig. 6. We used YOLOv5 [11] for detection, ByteTrack [14] for single-camera tracking, and hierarchical agglomerative clustering with temporal and spatial constraints [15] for multi-camera tracking. We implemented Bayesian optimization using BoTorch [16], and used SobolQMCNormalSampler for sampling in multi-objective Bayesian optimization.

We set the number of cameras to be placed in the target space to $n = 4$, and used $l = 8, m = 2$ for observation augmentation. In each cycle, we simulated with the default settings (daylight, sunny) for 5 minutes and captured image sequences at 10 fps for each camera. The computer used in our experiments had the following specifications: CPU: Intel Core i9-13900K (32 Core, up to 5.8 GHz), GPU: NVIDIA RTX A6000, Memory: 128 GiB, OS: Ubuntu 22.04.3 LTS.

B. Results

1) Performance: To validate the effectiveness of the proposed method, we conducted experiments with 200 cycles under the conditions described above. Fig. 7 shows the transition of the maximum multi-tracking score y_{multi} in each cycle; the maximum value of 0.588 was recorded at cycle 192. This result indicates that the maximum multi-tracking score, shown in the blue dotted line in Fig. 7, increases as the number of cycles progresses, suggesting that better camera placements are being acquired. The non-monotonic fluctuations in the maximum are attributable to the exploration-oriented behavior of EHVI as the acquisition function.

Next, we analyze the camera placement by the proposed method through visualization. Fig. 8 shows the camera placement at cycle 192 that achieved the maximum multi-tracking score. Fig. 9 shows the field of view of each camera in this placement. These figures indicate that the four cameras capture a wide area of the intersection from different viewpoints, and are placed to reduce blind spots while keeping substantial overlapping to support robust tracking across cameras.

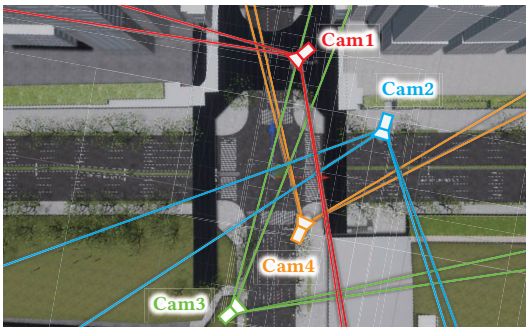

Fig. 8. Camera placement at the maximum multi-tracking score.

Fig. 9. Fields of view of each camera in the placement at the maximum multi-tracking score.

2) Comparison against Human Experts: To verify the effectiveness of the proposed method, we compared it with camera placements designed by six human participants, consisting of students and researchers specializing in computer vision.[2] The participants were first shown a bird's-eye view of the target intersection and briefed on the experimental setup, including the camera input parameters and how to adjust them. They were then instructed to place four cameras within five minutes to maximize the multi-tracking score defined in Eq.,(8). The five-minute window was found to be sufficient, as most participants completed their placement well before the time limit.

Table I summarizes the multi-tracking scores obtained by the participants' camera placement. Compared with participants, the best result of the proposed method was 0.588, exceeding the highest participant score of 0.510 by approximately 15%. Since the participants' average score was 0.444, the proposed method achieved a score roughly 32.4% higher than the average. These results show that the proposed method can propose camera placements with higher tracking

TABLE I

COMPARISON OF MULTI-TRACKING SCORES BETWEEN HUMAN EXPERTS (A–F) AND THE PROPOSED METHOD.

A	B	C	D	E	F	Proposed
0.484	0.510	0.388	0.497	0.361	0.422	**0.588**

[2]We consulted the institutional ethics review committee and were informed that ethical approval was not required as no personal information or information enabling the identification of individuals is included.

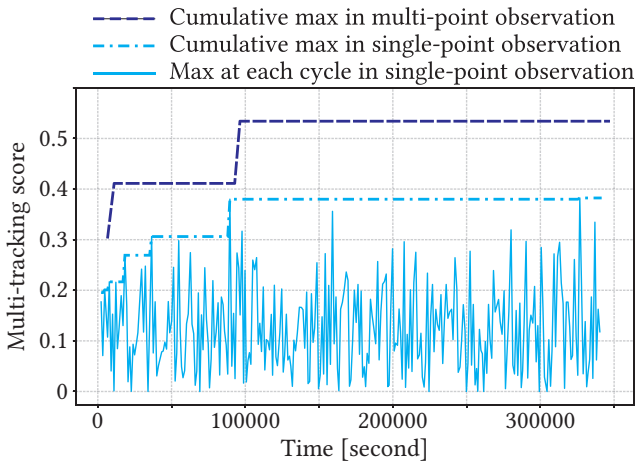

Fig. 10. Comparison of maximum multi-tracking scores between single-point and multi-point observations in the same timeframe.

performance than those designed intuitively by humans.

3) Ablation Study on Observation Augmentation: To evaluate the efficiency introduced by observation augmentation, we measured the time required for learning under single-point and multi-point observations. For estimating the time of each phase, we averaged 300 cycles for single-point observation and 80 cycles for multi-point observation. The results are shown in Table II. These results confirm that the multi-point observation method achieves more than a 4 times improvement in time efficiency compared to single-point observation, i.e., $17016/4158 \simeq 4$.

V. Conclusion

This paper addressed the optimal roadside camera placement problem for ITS, such as infrastructure-cooperative automated driving. We formulated it as an optimization problem that maximizes a detection-and-tracking-based metric computed via digital-twin-based simulation, and proposed a Bayesian approach to solve it. Unlike conventional methods that optimize sensor layouts to maximize spatial coverage, our method directly optimizes the metric built upon simulation-driven object detection and tracking, thereby reflecting both perception algorithm performance and site-specific conditions such as low-angle sunlight.

We validated the proposed method using a digital twin simulator, demonstrating that it derives a camera placement with a score exceeding human-designed placements and confirming the effectiveness of the approach. Furthermore, we proposed an observation augmentation strategy that alleviates the computational bottleneck in Bayesian optimization by placing more cameras than needed and augmenting observations per simulation run, achieving more than a 4 times efficiency improvement.

As future work, to reflect the sensing performance of real camera systems more accurately, we will incorporate simulation conditions that capture diverse factors affecting recognition performance in real environments (e.g., rain and low-angle sunlight). In addition, we aim to improve generality by extending the framework from camera-only sensing to multi-sensor configurations, including LiDAR, targeting sensing systems commonly adopted in practice.

TABLE II

Computation time [seconds] in single-point observation and multi-point observation to learn 15 observations.

t_a	t_b	t_c	t_d	T
663.1	324.8	139.3	7.3	17016

$t_{\tilde{a}}$	$t_{\tilde{b}}$	$t_{\tilde{c}}$	$t_{\tilde{d}}$	$\tilde{T}$
991.8	663.2	2373	130.1	4158

References

[1] J. O'Rourke, *Art gallery theorems and algorithms*. Oxford University Press, Inc., 1987.

[2] S. Indu, S. Chaudhury, N. Mittal, and A. Bhattacharyya, "Optimal sensor placement for surveillance of large spaces," in *Third ACM/IEEE International Conference on Distributed Smart Cameras*, 2009, pp. 1–8.

[3] A. Malhotra, D. Singh, T. Dadlani, and L. Y. Morales, "Optimizing camera placements for overlapped coverage with 3d camera projections," in *IEEE International Conference on Robotics and Automation*, 2022, pp. 5002–5009.

[4] J. Kim, Y. Ham, Y. Chung, and S. Chi, "Systematic camera placement framework for operation-level visual monitoring on construction jobsites," *Journal of Construction Engineering and Management*, vol. 145, no. 4, p. 04019019, 2019.

[5] S. Indu, S. Srivastava, and V. Sharma, "Optimal camera placement and orientation of a multi-camera system for self driving cars," in *4th International Conference on Vision, Image and Signal Processing*. Association for Computing Machinery, 2021.

[6] Y. Du, F. Wang, C. Zhao, Y. Zhu, and Y. Ji, "Quantifying the performance and optimizing the placement of roadside sensors for cooperative vehicle-infrastructure systems," *IET Intelligent Transport Systems*, vol. 16, no. 7, pp. 908–925, 2022.

[7] R. Vijay, J. Cherian, R. Riah, N. de Boer, and A. Choudhury, "Optimal placement of roadside infrastructure sensors towards safer autonomous vehicle deployments," *IEEE International Intelligent Transportation Systems Conference*, pp. 2589–2595, 2021.

[8] L. Kloeker, J. Quakernack, B. Lampe, and L. Eckstein, "Generic approach to optimized placement of smart roadside infrastructure sensors using 3d digital maps," in *IEEE International Conference on Intelligent Transportation Systems*, 2022, pp. 1311–1316.

[9] A. Qu, X. Huang, and D. Suo, "SEIP: Simulation-based Design and Evaluation of Infrastructure-based Collective Perception," in *IEEE International Conference on Intelligent Transportation Systems*, 2023, pp. 3871–3878.

[10] K. Li, Z. Dai, C. Zuo, X. Wang, H. Cui, H. Song, and M. Cui, "Scene adaptation in adverse conditions: a multi-sensor fusion framework for roadside traffic perception," *Journal of Intelligent Transportation Systems*, vol. 29, no. 6, pp. 698–718, 2025.

[11] G. Jocher, "Ultralytics YOLOv5," 2020.

[12] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The clear mot metrics," *EURASIP Journal on Image and Video Processing*, vol. 2008, no. 246309, 2008.

[13] Tier IV, Inc., "AWSIM." [Online]. Available: https://github.com/tier4/AWSIM

[14] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, "ByteTrack: Multi-Object Tracking by Associating Every Detection Box," in *European Conference on Computer Vision*, 2022.

[15] G. Szűcs, R. Borsodi, and D. Papp, "Multi-camera trajectory matching based on hierarchical clustering and constraints," *Multimedia Tools and Applications*, vol. 83, pp. 44 879–44 902, 2024.

[16] M. Balandat, B. Karrer, D. R. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy, "BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization," in *Advances in Neural Information Processing Systems*, 2020.