Visual Sensor Pose Optimisation Using Rendering-based Visibility Models for Robust Cooperative Perception

Eduardo Arnold, Sajjad Mozaffari, Mehrdad Dianati and Paul Jennings

Abstract—Visual Sensor Networks can be used in a variety of perception applications such as infrastructure support for autonomous driving in complex road segments. The pose of the sensors in such networks directly determines the coverage of the environment and objects therein, which impacts the performance of applications such as object detection and tracking. Existing sensor pose optimisation methods in the literature either maximise the coverage of ground surfaces, or consider the visibility of the target objects as binary variables, which cannot represent various degrees of visibility. Such formulations cannot guarantee the visibility of the target objects as they fail to consider occlusions. This paper proposes two novel sensor pose optimisation methods, based on gradient-ascent and Integer Programming techniques, which maximise the visibility of multiple target objects in cluttered environments. Both methods consider a realistic visibility model based on a rendering engine that provides pixel-level visibility information about the target objects. The proposed methods are evaluated in a complex environment and compared to existing methods in the literature. The evaluation results indicate that explicitly modelling the visibility of target objects is critical to avoid occlusions in cluttered environments. Furthermore, both methods significantly outperform existing methods in terms of object visibility.

Index Terms—Pose Optimisation, Sensor Placement, Visual Sensor Networks, Rendering, Visibility Models.

1 INTRODUCTION

Visual sensor networks are used in a diverse set of applications such as surveillance [1], traffic monitoring and control [2], parking lot management [3] and indoors patient monitoring [4]. Recently, integrating such sensor networks to traffic infrastructure has been suggested as promising means to support autonomous driving functionality in complex urban zones to enable cooperative perception [5], [6]. In such setting, the infrastructure-based sensors, which may include cameras and lidars, augment vehicles’ onboard sensor data using emerging V2X technologies. The usage of these networks is expected to grow further as high resolution sensors become more affordable and new generations of highly reliable wireless communication systems become widely deployed [7]. When designing sensor networks, the choice of the number and pose of the sensors, i.e. their location and rotation angles, is critical in determining their coverage. This directly impacts the performance of object detection, classification, and tracking applications that use the data from these sensor networks.

The problem of optimising sensor poses for a network of sensors has been explored in the literature. A major category of the existing studies formulate this problem as a discrete optimisation problem where a finite set of possible sensor poses is considered and the target objects’ visibility is described by a set of binary variables [8], [9], [10], [11]. The problem is then solved by using various forms of Integer Programming (IP) solvers or heuristic methods [12] to either maximise the number of visible target objects (coverage) with a fixed number of sensors; or to minimise the number of sensors required to achieve a given coverage constraint. However, the majority of the applications that may use such sensor networks, e.g. object detection [6] and tracking [13], require a minimum level of visibility over the target objects which cannot be encoded by single binary variables. For example, an object may have different degrees of visibility due to occlusions and due to its position w.r.t. the sensors, which causes ambiguity in the assignment of a binary visibility variable.

Another category of the existing studies consider the optimisation of continuous sensor pose variables using simulated annealing [14], Broyden–Fletcher–Goldfarb–Shanno (BFGS) [15], particle swarm [16], evolutionary algorithms [17] and gradient-based optimisation [18]. Most of the studies in this category focus on maximising the coverage (visible ground area) of extensive 3D environments described by digital elevation maps. However, such formulation does not consider the distribution of objects in the environment, and instead, assume an object would be visible if it is within a region covered by the sensors. As a result, these studies fail to detect and prevent occlusions between objects since they do not explicitly model the visibility of the target objects. This becomes a limiting factor when considering cluttered environments such as traffic junctions with a significant number of vehicles and pedestrians.

The visibility models that have been used in the literature usually consider simplifying assumptions which hinder the applicability of such methods in many practical settings. Examples of such simplifications include the use of a 2D visibility model that does not take into account the sensors’
The problem of sensor pose optimisation has its historical origin in the field of computational geometry with the art-gallery problem [20], where the aim is to place a minimal number of sensors within a polygon environment in such a way that all points within the polygon are visible. Although further work extends the art-gallery problem to a 3D environment considering finite field-of-view and image quality metrics [21], it still falls short of providing realistic sensor and environmental models. Further efforts treat the pose optimisation problem as an extension of the maximum coverage problem, however use very simplistic sensor assumptions, such as radial sensor coverage in 2D environments [22].

One category of methods consider sensor poses as continuous variables which are optimised according to some objective function. Akbarzadeh et al. [15] propose a probabilistic visibility model using logistic functions conditioned on the distance and vertical/horizontal angles between the sensor and a target point. The authors then optimise the aggregated coverage over an environment described by a digital elevation map using simulated annealing and Broyden–Fletcher–Goldfarb–Shanno (BFGS) optimisation. In further work [18], the same authors propose a gradient-ascent optimisation to maximise the aggregated coverage using their previous visibility model. This method requires obtaining the analytical forms of the derivatives of sensor parameters. In contrast, we propose a gradient-ascent method that uses Automatic Differentiation (AD) [23] allowing efficient sensor pose optimisation without specifying the analytical forms of the derivatives.

Recent work by Saad et al. [17] uses a visibility model similar to [15] with a LoS formulation. The authors introduce constraints over sensors’ locations and detection requirements, which are application-specific, and optimise the sensor pose to achieve the detection requirements using a genetic algorithm. Temel et al. [24] uses a LoS binary visibility model and a stochastic Cat Swarm Optimisation (CSO) to maximise the coverage of a set of sensors. The aforementioned methods aim to maximise the coverage of extensive 3D environments represented by digital elevation maps and use LoS algorithms [15, 16] to detect occlusions in these elevation maps. However, digital elevation maps are not ideal to represent target objects due to their coarse spatial resolution, which may conceal the objects’ shapes. In this paper, we propose a novel visibility model, based on the depth buffer of a rendering framework [25], which allows to accurately and efficiently detect any occlusions using arbitrarily shaped environments and objects. Furthermore, we explicitly model the visibility of target objects using a
differentiable visibility score which is based on a realistic perspective camera model.

Given the difficulty of optimising the sensors’ pose as continuous variables, another category of methods consider a discrete approach, where a subset of candidate sensor poses must be chosen to maximise the binary visibility of target points [12], [14], [26], [27]. This formulation allows to solve the problem using various forms of Integer Programming (IP) solvers [26], including Branch-and-Bound methods [28]. In some cases, solving the IP problem can be computationally infeasible, particularly when the set of candidate sensors is large, and thus, approximated methods, such as Simulated Annealing [12], [14] and Markov-Chain Monte Carlo (MCMC) sampling strategies [12], [27], can be used. The drawback of the aforementioned approximated methods is that they cannot guarantee the optimality of the solution found.

The methods in the IP category consider the visibility of a target object as a binary variable (i.e. visible or invisible), which cannot represent different levels of visibility and may result in sub-optimal sensor poses. Consider, for example, two sensor poses that can observe a given target object; one of the poses is closer to the object and provides more information than the other; yet, both poses obtain the same binary visibility result, i.e. the object is visible. In contrast to methods in this category that assign a binary visibility for target objects, we propose a novel IP formulation that considers the number of points (or pixels) that each sensor cast over each object, obtained using a rendering framework. Our proposed IP formulation takes into account the effect of partial occlusions and guarantees a minimum visibility across all target objects.

2.2 Differentiable Rendering

A general renderer is a process that generates images, or pixel values, given 3D scene parameters, which include objects’ meshes and textures, camera and lighting parameters. In contrast, differentiable renderers are a subclass capable of providing the derivative of pixel values with respect to any of the aforementioned scene parameters [29]. Such formulation bridges the gap between 3D scene parameters and their 2D projections [30], allowing efficient gradient-based optimisation solutions to inverse-graphics problems such as 3D object reconstruction [31], object/camera pose estimation [32] and adversarial examples generation [33]. However, the potential of differentiable renderers is yet to be explored in the context of sensor pose optimisation for visual sensor networks. In this paper, we use PyTorch3D’s perspective camera models to create an end-to-end differentiable pipeline that can be optimised using gradient-ascent. This pipeline allows to directly optimise the sensors’ pose to maximise the visibility of multiple target objects, as described in Section 4. For a detailed review of differentiable renderers and applications the reader may refer to [30].

3 PROBLEM FORMULATION

This section firstly presents the formulation of the sensor pose optimisation problem upon which we base our gradient-based and Integer Programming (IP) methods in Sections [3] and [5], respectively. Next, a novel sensor pose parametrisation is introduced to constrain the sensor poses to feasible regions which are pre-defined according to the environment where the sensors are deployed.

The sensor network, depicted in Figure 1a, consists of a set of fixed infrastructure sensors $S$ that collectively observe a set of target objects, denoted by $O$, in a driving environment. Each target object is represented using a three-dimensional cuboid encoded by $o = (x, y, z, w, h, l, \theta) \in O$, where $x, y, z$ correspond to the 3D centroid of the box, while $w, h, l$ represent the box size and $\theta$ corresponds to the pitch angle (rotation around the vertical axis), as depicted in Figure 1b. The visibility of object $o$ by the sensor set $S$, denoted by $\text{vis}(o, S)$, is defined as the number of points, or pixels, that the set of sensors $S$ project onto the object’s surfaces. Visibility in this sense intuitively quantifies the information that sensors capture about each object and has shown to be correlated with the performance of perception tasks such as 3D object detection [6] and tracking [13]. This visibility metric is computed in two steps. First, the frame containing objects $O$ is rendered. Then, the depth-buffer from each sensor in $S$ is re-projected into 3D space, creating an aggregated point cloud $P(S)$, as described in Section 4.2 and illustrated in Figure 1. Finally, the visibility of each object $o \in O$ is obtained by counting the number of points of $P(S)$ that lie on the surface of each respective object:

$$\text{vis}(o, S) = \sum_{p \in P(S)} \begin{cases} 1, & \text{if } p \text{ on } o\text{'s surface} \\ 0, & \text{otherwise.} \end{cases}$$

This visibility metric provides pixel-level resolution which successfully captures the effects of total or partial occlusions caused by other target objects and by the environment. The environment model, denoted by $E$, can also be modified according to the application requirements. For example, it is possible to include static scene objects, such as buildings, lamp posts and trees, that may affect the visibility of target objects.

The formulation proposed so far considered a single, static configuration of target objects, denoted by $O$. However, driving environments are dynamic and typically contain moving vehicles and pedestrians. We account for dynamic environments by considering a set of $L$ static frames. Each frame contains a number of target objects with specific sizes, positions and orientations within the environment. The number of frames, denoted by $L$, must be chosen such that the distribution of objects over the collection of frames approximates the distribution of target objects’ in the application environment. For example, one can obtain a set of frames for driving environments using microscopic scale traffic simulation tools, such as SUMO [34] or through the empirical observation of the driving environment.

The underlying optimisation problem is to find the optimum poses for $N$ sensors, denoted by $S = \{s_1, \ldots, s_N\}$ that maximise the visibility of target objects across the $L$ frames. Formally, the optimal set of sensor poses is defined as

$$\hat{S} = \arg\max_{S} \min_{o \in O} \text{vis}(o, S),$$

where $O$ is the set of the objects across $L$ frames. In practice, each frame is rendered independently so that objects from
different frames do not occlude one another, but the optimisation is still performed across all frames.

It should be noted that one can alternatively maximise the mean visibility of the target objects, which can be formulated as \( \arg \max_S \frac{1}{M} \sum_{o \in O} \text{vis}(o, S) \). However, this may result in some of the objects having very low or zero visibility in the favour of others having unnecessarily large visibility. But maximising the minimum visibility biases the optimisation algorithm towards sensor poses that guarantee the visibility of all target objects.

### 3.1 Sensor Pose Parametrisation

Generally speaking, the pose of a sensor in a 3D environment can be described by the canonical six degrees-of-freedom parametrisation \( s = (x, y, z, \phi, \theta, \phi) \), where the \((x, y, z)\) represent the sensor position and \((\phi, \theta, \phi)\) its viewing angles. However, unconstrained optimisation under such parametrisation is seldom useful in practice as most environments have restrictions regarding sensors’ location, e.g. sensors must be mounted close to a wall, on lamp posts, and clear from a road, etc. To this end, we propose a continuous sensor pose parametrisation called virtual rail which imposes constraints over the sensors’ location without adding any penalty term to the optimisation objective function or requiring any changes to the optimisation process, such as gradient projection.

A virtual rail is defined by a line segment between two points in 3D space. The sensors can be placed at any point within this line segment, as illustrated in Figure 6. The viewing angles are described by the rotations along the X and Y axis, as we assume no rotation along the camera axis (Z). The pose of a sensor on a virtual rail between points \(p_1, p_2 \in \mathbb{R}^3\) has its pose fully determined by the parameters \(s = (t, \alpha, \beta)\) through the parametrisation

\[
(x, y, z) = p_1 + \sigma(t)(p_2 - p_1),
\]

\[
\begin{align*}
\phi &= 2\pi\sigma(\alpha), \\
\theta &= \pi\sigma(\beta), \\
\phi &= 0,
\end{align*}
\]

where

\[
\sigma(z) = \frac{1}{1 + e^{-z}},
\]

is the sigmoid function. This function enforces the bounds of position within the rail, i.e. \((x, y, z)\) on the line segment between \(p_1, p_2\), and viewing angles \(\varphi \in [0, 2\pi], \theta \in [0, \pi]\) for unbounded variables \(t, \alpha, \beta \in \mathbb{R}\).

This parametrisation allows the use of unbounded gradient optimisation with guaranteed constraints over the sensors’ poses. Note that the choice of the number and position of virtual rails are hyper-parameters defined to fit the needs of the application according to the complexity of the environment/task.

### 4 GRADIENT-BASED SENSOR POSE OPTIMISATION

This section describes the proposed gradient-based sensor pose optimisation for multi-object visibility maximisation.

The objective function proposed in Equation 2 is not differentiable w.r.t. the sensor pose parameters due to the non-continuity introduced by the threshold operation in \(\text{vis}(\cdot)\). Thus, gradient-based solutions cannot be applied to solve this optimisation problem. We, therefore, propose a processing pipeline featuring a differentiable objective function that approximates the objective function in Equation 2. A crucial element of this approximation is the visibility score, a continuous variable in the interval \([0, 1]\) that measures the visibility of a given 3D point w.r.t. a sensor. The processing pipeline considers the continuous visibility score of multiple points over each target object, which ensures the objects’ visibility and implicitly approximates the original problem in Section 3. It shall be noted that the visibility score is different from the visibility metric (Equation 1) in two ways: 1) the former is differentiable while the latter is not; 2) the former indicates the degree of visibility of a single point on a target object while the latter is the number of points on the surface of a target object. The proposed processing pipeline for the computation and optimisation of the objective function is depicted in Figure 2.

The processing pipeline consists of five stages.

1) a set of target points, denoted by \(T \in \mathbb{R}^{MF \times 3}\), is created by sampling \(F\) points from each of the \(M\) target objects. The points are randomly distributed along the objects’ surfaces proportionally to each surface area.

2) the points \(T\) are projected onto the image plane of each sensor and a visibility score is computed for each target point according to their position w.r.t. the visible frustum of the respective sensor, as described in Section 4.1.
3) an occlusion-aware visibility model, described in Section 4.2 is used to update the visibility score created in the previous stage. This is required since some projected points will be in the visible frustum of a given sensor but occluding objects prevent direct line-of-sight between the point and the sensor.

4) the objective function is computed as the mean visibility score of all points $T$ on target objects, as described in Section 4.3.

5) gradient-ascent is used to maximise the objective computed in the previous step, as described in Section 4.4.

The proposed processing pipeline can work for any continuous sensor pose parametrisation. In this paper, we consider the parametrisation proposed in Section 3.1 which constrains the sensor position to a line segment and allows for unconstrained gradient-based optimisation.

4.1 Visibility Model

In this section we propose a realistic visibility model based on the perspective camera model provided by PyTorch3D [25]. Built on top of PyTorch, this camera model provides differentiable transformations from the global coordinate system to the camera image plane which is fundamental for a fully differentiable pipeline. The cameras’ extrinsic matrix is determined by the pose of the sensors, specified by the set of parameters $S$, being optimised. It shall be noted that all cameras intrinsic properties are identical: 90-degree horizontal field-of-view, $D_{\text{near}} = 1m$, $D_{\text{far}} = 100m$ near and far clipping planes, respectively, and resolution of $W = 200 \times H = 200$ pixels. The resolution is kept relatively small in order to reduce the computational complexity of the rendering process, described in Section 4.2. Increasing the image resolution directly increases the visibility of the target objects as there will be a higher number of pixels/points per object. In practice, the sensor poses resulting from the optimisation process can be used for cameras with higher resolution, as long as they have the same aspect ratio and field-of-view.

The projection of a point $p = [x \ y \ z]^T \in \mathbb{R}^3$ in the global coordinate system into the image plane of sensor $s$ is given by

$$\begin{bmatrix} u \ v \ d \end{bmatrix} = M_3 M_e(s) \begin{bmatrix} x \\ y \\ z \end{bmatrix},$$

where $[u \ v \ d]^T$ are homogeneous coordinates that can be divided by $d$ to obtain the canonical form $[u \ v \ 1]^T$. Here, $u, v$ are the image plane coordinates in pixels, $d$ is the depth of the point in the view frustum and $M_3, M_e$ are the intrinsic and extrinsic camera matrices of sensor $s$, respectively. The point $p$ is within the visible frustum if and only if $W \geq u \geq 0, H \geq v \geq 0$ and $D_{\text{far}} \geq d \geq D_{\text{near}}$ where $W$ and $H$ are the image width and height in pixels. $D_{\text{near}}, D_{\text{far}}$ are the camera near and far clipping planes in meters, respectively.

The visibility of a given point from the perspective of a sensor $s$ is determined by verifying that the image plane projection of this point, given by Equation 5, satisfies the bounds described in the previous paragraph. If the bounds are satisfied, the point is considered visible, otherwise it is not. Since the threshold operations used to identify the visibility of a point are not differentiable, we opt to use the sigmoid function (Equation 4) as a differentiable approximation of the binary visibility. This continuous visibility score can be interpreted as a probabilistic visibility measure of a point, ranging from 0 (completely invisible) to 1 (completely visible). This is formulated by a window function as follows:

$$w(z, \gamma, z_0, z_1) = \sigma(\gamma(z - z_0)) - \sigma(\gamma(z - z_1)),$$

where $\gamma \in \mathbb{R}$ controls the rate of transition on the limits of the interval $[z_0, z_1]$, as illustrated in Figure 3. As $\gamma$ increases the window function tends to a binary threshold operation. However, this reduces the intervals with non-zero gradients, and consequently inhibits parameters updates through gradient optimisation. Empirical tests revealed that $\gamma = 1$ was the best out of the three tested values (0.1, 1, 10) for this hyper-parameter.

The visibility score of a point $p$ with image plane projection $[u_s d_s v_s d_s]^T$ is given by

$$\Psi(p, s) = w(u_s, \gamma, 0, W) \cdot w(v_s, \gamma, 0, H) \cdot w(d_s, \gamma, D_{\text{near}}, D_{\text{far}}).$$

(7)

This visibility model does not take into account occlusions caused by other objects or the environment since a point being within the visible frustum of a sensor is a required but not sufficient condition to guarantee direct line-of-sight visibility from the sensor to the point. We account for occlusion by proposing an occlusion aware visibility model in Section 4.2.

4.2 Occlusion Awareness

We verify line-of-sight visibility using the depth buffer generated by PyTorch3D’s rasteriser. This rasteriser transforms the meshes representing the environment and the target objects into a raster image with a corresponding depth buffer. When an object is projected to the image plane, the orthogonal distance between the object and the sensor is stored in the corresponding pixel position of the depth buffer. If another object is projected to the same pixel, the depth buffer keeps the smallest depth distance among the two. This solves the hidden surface problem in computer graphics, where some objects overlap over the sensor’s field-of-view and the closest objects occlude/hide other objects in the background. We use the same approach in our processing pipeline to determine if a given sensor has line-of-sight visibility of a point in 3D space.

Given a target point $p \in T$ and a sensor $s$, the point is considered to be occluded from the point of view of sensor $s$ if

$$|d_s - Z_s(u_s, v_s)| > \kappa,$$

where $[u_s v_s d_s]^T$ is the projection of $p$ on the image plane of $s$ according to Equation 5. Here, $Z_s(u_s, v_s)$ is the depth buffer of sensor $s$ at the pixel position $(u_s, v_s)$ and $\kappa$ is a threshold for the maximum disparity between the projection depth value and the depth buffer. In our experiments, we consider $\kappa = 0.5m$, which allows to accurately detect occlusions. Figure 4 illustrates this occlusion-aware visibility model for a visible and an occluded point. In this figure, the depth of a point projected on the image plane matches the
Fig. 2. Processing pipeline of the proposed Gradient-based sensor pose optimisation method. (a) an exemplar frame with two objects and a set $S$ of $N$ sensors, including an environmental model with an occluding block (in green). (b) the optimisation pipeline.

Fig. 3. Window function $w(z, \gamma, z_0 = 0, z_1 = 200)$ plotted for $z_0 = 0, z_1 = 200$ and varying values of $\gamma$.

depth buffer measurement at the corresponding pixel if the point is visible; if the point is occluded the depth buffer value will be smaller since there is another object closer to the sensor.

This occlusion-aware visibility model leads to an enhanced version of the visibility score of a point $p$ observed by sensor $s$, given by:

$$
\Psi(p, s) = \begin{cases} 
  w(u_s, \gamma, 0, W), & \text{if } |d_s - Z_s(u_s, v_s)| \leq \kappa \\
  w(v_s, \gamma, 0, H), & \text{if } |d_s - Z_s(u_s, v_s)| > \kappa \\
  0, & \text{otherwise}
\end{cases}
$$

(9)

In the case where $p$ is out of the visible frustum of sensor $s$, the visibility score is given by Equation 9. Note that if the point is occluded, there is no gradient signal to change the pose of the sensor in which the point is occluded. Yet, the occluded point can be targeted by other sensors in the network.

The depth buffer from each sensor is re-projected into 3D space, using the inverse of Equation 5, to create a 3D point cloud representing all points observed by the respective sensor. Effectively, the depth buffers from each sensor $s \in S$ are re-projected and aggregated into a fused point cloud $P(S)$, shown in Figure 7b. This fused point cloud is used to compute the visibility metric $vis(o, S)$, used by the IP method and during the system performance evaluation.

4.3 Objective Function

A target point $p$ may be observed by multiple sensors, thus, the overall visibility of a point by a set of sensors $S$ is computed as

$$
\Psi(p, S) = 1 - \prod_{s \in S} (1 - \Psi(p, s)).
$$

(10)

According to Equation 10 a point’s overall visibility score is forced to be 1 if at least one sensor has a visibility score of one. Conversely, sensors that cannot observe a point (zero visibility score) do not affect the overall visibility score. Furthermore, when multiple sensors observe the same point, the combined visibility score improves.

The proposed sensor pose optimisation model in this paper aims to maximise the mean visibility score across all
objects $O$ for a given set of sensors $S$. Hence, the following objective function is maximised in our gradient-based formulation:

$$\mathcal{L} = \frac{1}{|T|} \sum_{p \in T} \Psi(p, S),$$  \hspace{1cm} (11)

where $T$ is a set of randomly sampled target points from target objects' surfaces, and $\Psi(p, S)$ is the overall visibility score of point $p$ across all sensors $S$ according to Equation 10 considering the enhanced occlusion-aware visibility model, described by Equation 9.

4.4 Optimisation

We adopt the Adam optimiser [35] to allow per-parameter learning rate and adaptive gradient-scaling, which has been shown to stabilise and shorten the optimisation process. The optimiser uses a global learning rate of 0.1, and is executed for 20 iterations over the whole collection of frames. These optimisation hyper-parameters were determined empirically through experiments. Algorithm 1 describes the optimisation process for a set of frames and Table 1 specifies the input variables used in the algorithm.

The objective function in Equation 11 is maximised w.r.t. the continuous sensor pose parameters $(t, \alpha, \beta)$ described in Section 3.1. These parameters specify the pose of a sensor within a virtual rail. In an environment containing multiple virtual-rails, there must be an assignment between each sensor and the virtual-rail it belongs to. This assignment is represented by a discrete variable that maps each sensor to one of the virtual-rails and is also subject to optimisation. However, since it is a discrete variable, it cannot be part of the gradient-based optimisation process. We overcome this problem by performing multiple runs of the optimisation process, each with a random virtual-rail assignment, and reporting the best results across all runs in terms of the objective function.

The sensor poses are initialised using a uniform distribution on the interval $[-2, 2]$ over the parameter $t$, which controls the sensor position $(x, y, z)$ along the virtual-rail according to Equation 3. The limits of the uniform distribution are chosen such that the sensors initial position within the rail can be anywhere from $10\%$ to $90\%$ of the length of the rail. The viewing angles can be randomly initialised in the same fashion. However, there may be some prior information of the environment that can guide this decision. For example, in a traffic junction objects are likely to traverse the central area of the junction, thus, sensors could benefit by focusing towards the junction centre. Although this step is not strictly required, it introduces prior information into the problem which reduces the amount of time required to achieve satisfactory results in the optimisation process.

5 INTEGER PROGRAMMING-BASED SENSOR POSE OPTIMISATION

Integer Programming (IP) is an effective approach for solving optimisation problems where some or all of the variables are integers and may be subject to other constraints [28]. Applied to sensor pose optimisation, this formulation assumes that the optimal set of sensors are chosen from a finite set of sensor poses, called candidate poses. The

**Algorithm 1** Gradient-Ascent Sensor Pose Optimisation

**Input:** $N, O_1, O_2, \ldots, O_L, F, E, \text{virtualRails}$, epochs

**Output:** $\hat{S}$, minVisibility

**Initialisation:**

1: $S \leftarrow \emptyset$

2: for $i \leftarrow 1$ to $N$ do

3: $p_1, p_2 \leftarrow$ random virtual rail from virtualRails

4: Draw sample $t$ from Uniform $(-2, 2)$

5: Set $\alpha, \beta$ such that sensor focus on the centre of the junction

6: Set $s = f(p_1, p_2, t, \alpha, \beta)$ \{Alternative, sample them from the uniform distribution.\}

7: $S \leftarrow S \cup s$

8: end for

**Optimisation loop**

9: for $e \leftarrow 1$ to epochs do

10: $\mathcal{L} \leftarrow 0$

11: for $O \in \{O_1, \ldots, O_L\}$ do

12: $T \leftarrow$ sample $F$ points from each target objects $o \in O$ surfaces

13: $T' \leftarrow$ image plane projection of $p \in T$ for each sensor $s \in S$ according to Equation 5

14: $Z, P \leftarrow$ depth-buffer and reconstructed point-cloud from rasteriser as a function of $O, S, E$

15: $\Psi \leftarrow$ visibility score for each $p \in T'$ according to Equation 9

16: $\Psi_S \leftarrow$ overall visibility score over all sensors according to Equation 10

17: $\mathcal{L} \leftarrow \mathcal{L} + \text{mean} (\Psi_S)$

18: end for

19: minVisibilityMetric $\leftarrow \min_o \text{vis}(o, S) \forall o \in O_1 \cup \cdots \cup O_L$ \{Computes the visibility metric using the reconstructed point-cloud $P$\}

20: if minVisibilityMetric improved since last epoch then

21: $S \leftarrow S$

22: end if

23: Compute $\frac{\partial \mathcal{L}}{\partial S}$ using automatic differentiation

24: Update $S$ based on gradient-ascent update rule

25: end for

26: return $\hat{S}$, minVisibility

**Table 1** Description of variables in Algorithm 1

| Variable   | Description                                      | Value |
|------------|---------------------------------------------------|-------|
| $N$, $O_1, \ldots, O_L$ | Number of sensors                                    | 1-6   |
| $L$ | Sets of objects for each of the $L$ frames       |       |
| $F$ | Number of frames in the dataset                   | 1000  |
| $E$ | Number of target points sampled per object        | 400   |
| virtualRails | The set of virtual rails described by two end-points in $\mathbb{R}^3$ |       |
| epochs | Number of optimisation iterations | 20     |
problem is a combinatorial search to find the optimal subset of candidate poses that maximise an objective function. This objective function typically models the visibility of an area or objects. Additional constraints, such as the maximum number of sensors in the optimal set can be added to the problem formulation. This section describes how IP can be applied to solve the sensor pose optimisation problem formulated in Section 5.1. The objective is to find the subset of candidate sensor poses that maximises the minimum visibility metric of target objects. We firstly introduce a method for the discretisation of the sensor pose parameter space into a finite set of candidate poses. We then cast the base optimisation problem in Eq. 2 into an IP optimisation problem and present three approaches to solve it: a heuristic off-the-shelf solver and two approximate methods based on sampling strategies.

5.1 Discretising Pose Parameters

To apply Integer Programming to the sensor placement problem we need to discretise the sensor pose parameter space into a finite set of candidate sensor poses. We use the concept of virtual rails, described in Section 3.1, to create the set of candidate sensor poses by dividing each virtual rail into 10 equally spaced sensor positions. The horizontal viewing angles at each position is divided into 3 feasible angles. The vertical viewing angles at each position is divided into 3 feasible angles. To this end, the set of candidate sensor poses for a given virtual rail is $S' = \{e.0, \theta \in \{0.1, 0.2, \ldots, 1\}, \phi \in \{36, 72, \ldots, 360\}, \theta \in \{18, 36, 54\}\}$. For simplicity, in the rest of this paper we assume that $S'$ represents the union of candidate poses from all virtual rails and the number of candidate poses is given by $|S'| = N'$. Figure 6 illustrates the set of candidate poses $S'$ for a T-junction scenario.

5.2 IP Objective

The general sensor pose optimisation problem can be formulated as the following IP problem

$$
\max_{b_1, \ldots, b_N'} f(b_1, \ldots, b_N', o_1, \ldots, o_M) \quad \text{s.t.} \quad \sum_{i=1}^{N'} b_i \leq N, \tag{12}
$$

where $b_i$ is a binary variable indicating if the $i$-th sensor in the candidate set, denoted by $s_i \in S'$, is part of the optimal set. In other words, the sensor $s_i$ is part of the optimal set if $b_i$ is 1 and the optimal set of sensors is given by $S = \{s_i \in S' : b_i = 1 \ \forall i \in \{1, \ldots, N'\}\}$. The constraint guarantees that the maximum number of chosen sensors do not exceed $N$. The objective function $f(\cdot)$ represents the targets’ visibility, which depends on the choice of sensors $b_1, \ldots, b_N$, and the targets $o_1, \ldots, o_M$. Previous works [10], [12] define $f(\cdot)$ as the sum of binary visibilities of environment points. This is a poor estimate of target objects’ visibility since there are varying degrees of visibility which cannot be encoded as a binary variable. To address this problem, we propose a novel IP formulation that takes into account the visibility metric of a target object $o$ observed by a sensor $s$, $\text{vis}(o, \{s\})$, defined in Equation 1. The motivation is to to find the sensor set that maximise the minimum visibility metric among target objects. Hence, the equivalent IP problem is described by

$$
\max_{z,b_1,\ldots,b_N'} z \quad \text{s.t.} \quad \sum_{i=1}^{N'} b_i \text{vis}(o, \{s_i\}) \geq z \ \forall o \in O, \tag{13}
$$

where $z \in Z_{\geq 0}$ is the minimum visibility metric among target objects. The first constraint guarantees that $z$ is the minimum visibility metric among all objects. Note that the effect of multiple sensors observing a given object is cumulative w.r.t. the visibility metric, i.e. $\text{vis}(o, \{s_1, s_2\}) = \text{vis}(o, \{s_1\}) + \text{vis}(o, \{s_2\})$.

The formulation proposed so far considers a single frame, denoted by $O$, containing the target objects. This is extended to $L$ frames by rendering each frame individually, including the objects and the environmental model, for all candidate sensors. The visibility of an object $o$, as observed by sensor $s_i$, denoted by $\text{vis}(o, \{s_i\})$, is obtained by counting the number of points in the re-projected point cloud generated by sensor $s_i$ that are on the surface of the object $o$, as described in Section 3 and illustrated in Figure 5. In practice, the visibility of all objects are computed frame by frame, for each candidate sensor, prior to the optimisation and stored in a visibility matrix $V$. This allows to solve the IP problem in Eq. 13 for any number of sensors without recomputing the objects’ visibilities.

5.3 Heuristic Solution

IP problems are NP-complete [28], thus, finding the solution using exhaustive search is computationally expensive or even unfeasible when the search space is large. Particularly, the size of the search space of the IP problem in Eq. 13 is $\binom{N'}{N}$. For example, for a candidate set with $N' = 1500$ poses and a given number of sensors $N$, e.g. 6, the size of the search space is $\frac{1500}{6} \approx 317$. For this reason, there are multiple algorithms that attempt to solve the problem using heuristic methods such as cutting plane and branch-and-cut methods [28].

In this paper, we use the Coin-or Branch and Cut (CBC) open-source IP solver [35] and the python-mip wrapper [37] to solve the problem. This solver uses Linear Programming (LP) relaxation for continuous variables and applies branching and cutting plane methods where the integrality constraint does not hold. The solver cannot always guarantee the optimality of the solution, specially when exhaustive search is infeasible. Thus, the problem in Equation 13 is solved using the default optimisation settings until the optimal solution is found or the time since an improvement in the objective function exceeds a limit.

5.4 Approximate Solutions

Approximate solutions to the IP problem are often used for the camera placement problem when exact solutions cannot be obtained in feasible time [12]. As described in
Fig. 5. Illustration of the process of computing the visibility metric of object \( o_i \in O \) by each candidate sensor \( s_i \in S' \). The rendered point cloud naturally handles any occlusion caused by the environment model \( E \) and other target objects in the frame. The visibility of a given object is obtained by counting all points (represented by the blue dots) from the respective sensor that are on the surface of the respective object. The output is a visibility matrix \( V \) that depicts how many points each candidate sensor cast on each object in the frame, i.e. the object’s visibility. This process is repeated for all frames and the matrices computed for each frame are concatenated horizontally.

The solution set at iteration \( i \) is then set according to
\[
S_i = \begin{cases} 
S_{i-1}, & \text{if } u \leq r \\
S_i^*, & \text{otherwise},
\end{cases}
\]
where \( u \) is a sample from the uniform distribution \( U[0, 1] \). The algorithm is executed until the time since the last improvement in the objective function exceeds a limit.

### Algorithm 2 Naïve Sampling Approximate IP Solution

**Input:** \( S', N, O, \) maxTime

**Output:** \( S \)

1. \( z_{\text{best}} = 0 \)
2. \( S = \emptyset \)

**Sampling loop**

3. while \( \text{timeSinceLastImprovement} \leq \text{maxTime} \)
4. \( S = N \) samples from \( S' \) without replacement;
5. \( z = \min_{o \in O} \text{vis}(o, S) \)
6. if \( z \geq z_{\text{best}} \)
7. \( z_{\text{best}} = z \)
8. \( S = S' \)
9. reset timeSinceLastImprovement
10. end if
11. end while
12. return \( S \)

### 6 EVALUATION

This section describes the evaluation of the proposed sensor pose optimisation methods. First, the evaluation metrics are defined in Section 6.1. Next, the experiment setup is described, including details of the simulation scenario in Section 6.2. Then, a comparative evaluation between the methods proposed in this paper is presented in Section 6.3. Finally, a comparison of the proposed methods with existing works in the literature and a comparison of different visibility models are reported in Section 6.4 and 6.5, respectively.

#### 6.1 Evaluation Metrics

Existing studies in the literature assess sensor pose optimisation methods using the number of visible targets \( T \) or the mean ground area coverage \( M \), where
coverage is defined as the probability that an area is visible to a sensor. However, such metrics are unsuitable for object-centric visibility for two reasons. First, adopting a binary visibility for an object is a coarse measure, since an object can be visible to different degrees due to its distance from the sensors, due to occlusions and limited sensor field-of-view. Secondly, the coverage of a ground area does not guarantee that an object placed within this area will be visible, as occlusions may limit the object’s visibility. For the aforementioned reasons, in our analysis we evaluate a set of sensor poses $S$ based on the minimum visibility metric across all objects, denoted by $\min_{o \in O} \text{vis}(o, S)$. Recalling from Equation \ref{eq:visibility}, the visibility metric is defined as number of pixels that the set of sensors $S$ observe on the surface of a given target object. In addition to the minimum visibility metric, we compute the Empirical Cumulative Distribution Function (ECDF) of the visibility metric for all objects across frames, which provides broader insight into the visibility patterns across objects.

### 6.2 Evaluation Setup

The performance evaluation of the proposed sensor pose optimisation methods is carried out by simulating traffic on a T-junction environment. This is motivated by the challenging conditions faced in such environments. For safety reasons, it is critical to guarantee that all vehicles, i.e., target objects, are visible to the sensors. Yet, vehicles are subject to occlusions from other vehicles and buildings.

The driving environment is simulated using the CARLA open-source simulator \cite{Dosovitskiy17}. A typical urban T-junction is chosen from one of the existing maps in the simulation tool. It has an area of 80 x 40 meters with several tall buildings and road-side objects, such as trees, bus shelters and lamp-posts. Within this environment, a dataset consisting of 1000 frames is generated. Each frame is a snapshot of the environment at a particular time, containing the number of vehicles and their representation. The objects’ representation, as described in Section \ref{sec:objects}, defines their position, size and orientation in the environment.

The environment model, available through CARLA open-source assets, contains a high number of complex meshes that slow down the rendering process. For this reason, we opt to create a simplified version of the environment. To this end, we create cuboid meshes for the buildings near the junction, and represent vehicles as cuboids using the same dimensions of the original objects’ bounding boxes. This approximation significantly speeds up the rendering process without detrimental impact to the measurement of objects’ visibility metric. Figures \ref{fig:original} and \ref{fig:simplified} illustrate the original and simplified environment models, respectively.

Sensors placed in such driving environments must be placed by the road-side and clear from the road. This constraint is addressed by creating five virtual rails alongside the junction, each aligned with the curb over a segment of the junction, as illustrated in Figure \ref{fig:virtual_rails}. The parametrisation of the rails is application dependent and may need adjustment. In this application, the virtual rail configuration allows sensors to be positioned on existing road-side infrastructure, such as traffic lights. The virtual rails are positioned on a height of 5.2m above the ground, following the standards of public light infrastructure in the UK \cite{Highways17}. However, the height of each sensor could also be included in the optimisation process.

### 6.3 Comparative evaluation of the proposed methods

Table \ref{tab:comparison} shows the results comparing the gradient-based and IP optimisation methods in terms of the minimum object visibility metric and duration of the optimisation process for a varying number of sensors, denoted by $N$. The runtime performance of the IP methods does not include the time required to compute the visibility matrix, i.e., rendering 1000 frames for each of the 1500 candidate sensors poses, which took 28 hours. However, this process is only done once and the resulting visibility matrix is used by all IP methods for any number of sensors. None of the methods could find a pose for a single sensor that can observe all objects, thus, the results are reported for $N > 1$. The gradient-based method results are reported for the best out of 10 runs for each number of sensors. Each run has a random sensor-rail assignment and random sensor position initialisation, as described in Section \ref{sec:optimisation}. The best minimum visibility metric observed in each run is reported in Figure \ref{fig:visibility}.

The evaluation shows that the IP method consistently outperform the gradient-based method, which we believe is explained by two factors. First, the loss function being maximised in the gradient method is non-convex and presents local-maxima, which may result in sub-optimal
results. Secondly, the gradient-based method does not optimise the sensor-rail assignment. We circumvent the latter by performing multiple optimisation runs for the gradient-based method, each with a random sample of sensor-rail assignment. However, the variance of the visibility metric obtained across runs, observed in Figure 8, suggests that ten samples may not be enough to explore the sensor-rail assignment space. Including more samples of sensor-rail assignments requires more optimisation runs, which becomes time costly. On the other hand, the IP method handles the sensor-rail assignment naturally as the candidate sensor pose set includes sensor poses in all virtual rails.

Figure 9 shows the resulting sensor poses found by each method for different numbers of sensors and the associated ECDF of the visibility metric of the target objects for each set of sensor poses. The visibility metric distributions obtained with IP solutions show similar visibility patterns, except for \( N = 6 \) where the heuristic IP approach has a significant advantage over its counterparts. The distribution of visibilities for gradient-based solutions is significantly skewed towards smaller visibilities if compared to IP solutions. Particularly, for \( N = 5 \), approximately 80% of the objects have less than 1000 points when observed by the gradient-based solution, while only 40% of objects have less than 1000 points for the IP solutions.

6.4 Comparison with existing works

We compare our sensor pose optimisation methods with two existing works. Akbarzadeh et al. [18] maximise the coverage of a ground area using gradient-ascent and Zhao et al. [12] uses Integer Programming to maximise the number of target points visible in an environment. We reproduce these methods in the simulated T-junction environment considering the coverage of uniformly distributed points over the T-junction ground area. Note that these methods do not explicitly model the visibility of the target objects, instead they maximise the coverage of the ground area. The evaluation considers the ground surface coverage, i.e. the ratio of ground points that are visible to the sensors, and the minimum visibility of objects placed over this area. The results are reported in Table 3. These results show that the previous methods are successful in maximising the coverage of the T-junction’s ground area. However, this does not
Fig. 9. Resulting sensor poses and visibility distributions. The left column represents the perspective view of the junction showing the pose of the resulting set \( \hat{S} \). The right column shows the ECDF of object's visibility for the optimal set of sensors found by different methods. The colour of the sensors in the perspective view follows the legend of the ECDF plot. Each row describes the results for a given number of sensors, denoted by \( N \). Note that some of the camera poses are the same across methods and may appear as a single one, particularly for \( N = 2 \).
guarantee the visibility of target objects since occlusions between objects are a key factor in determining the visibility of objects in cluttered environments. This underpins the importance of explicitly considering the visibility of target objects in contrast to the coverage of ground areas.

6.5 Comparison between visibility models

We perform a study comparing the performance of the gradient-based method considering three different visibility models: our visibility model with and without occlusion awareness (Eq. 7 and 9 respectively) and the visibility model from Akbarzadeh et al. [18]. In this study, we consider \( N = 6 \) sensors and explicitly model the visibility of the target objects using the three aforementioned visibility models. Table 3 reports the results of this study. Our occlusion-aware visibility model achieves the best performance as it can realistically determine which points are visible and accordingly change the sensors’ pose to account for potential occlusions. This is highlighted in Figure 10 depicting the point clouds of target points, where the colour of each point encodes its visibility score, ranging from blue (invisible) to red (visible). Note that our occlusion-aware visibility model correctly identify non-visible parts of the objects due to occlusion (blue) or only partially visible (yellow). In contrast, the two other visibility models fail to identify areas of occlusion, mistakenly determining that all points are visible (red). As a result, the optimisation process cannot improve the visibility of such areas.

7 CONCLUSION

Sensor pose optimisation methods such as the ones proposed in this paper can guide the cost-effective deployment of visual sensor networks in traffic infrastructure to maximise the visibility of objects of interest. Such sensor network infrastructures can be used to increase the safety and efficiency of traffic monitoring systems and aid the automation of driving in complex road segments, particularly, in areas where accidents are more likely to happen.

Our systematic study, in addition to the proposition of novel approaches for sensor pose optimisation, reveals a number of key insights that can be useful for researchers and system designers. Firstly, explicit modelling of the visibility of the target objects is critical when optimising the poses of sensors, particularly in cluttered environments where sensors are prone to severe occlusions. Secondly, rendering-based visibility models can realistically determine the visibility of target objects at the pixel level and, thus, improve the pose optimisation process. Thirdly, the IP optimisation method seems to outperform the gradient-ascent method in terms of minimum object visibility, at the cost of increased computational time. The sensor pose optimisation methods proposed in this paper can guide the deployment of sensor networks in traffic infrastructure to maximise the visibility of objects of interest.

As a follow-on study, we believe that it can be interesting to investigate how to reduce the search space of the IP formulation, for example, by using heuristics to remove candidate sensor poses that have limited observability. Additionally, strategies to incorporate the discrete rail assignment variables directly into the gradient optimisation should be investigated, e.g., considering differentiable discrete distribution sampling via Gumbel-Softmax [40]. Other global-optimisation strategies could be used to circumvent the impact of local-minima in the gradient-ascent method.

**REFERENCES**

[1] M.-L. Wang, C.-C. Huang, and H.-Y. Lin, “An intelligent surveillance system based on an omnidirectional vision sensor,” in 2006 IEEE Conference on Cybernetics and Intelligent Systems. IEEE, 2006, pp. 1–6.

[2] M. F. Rachmadi, F. Al Aifi, W. Jatmiko, P. Marsanto, E. A. Manggala, M. A. Ma’sum, and A. Wibowo, “Adaptive traffic signal control system using camera sensor and embedded system,” in TENCON 2011 - 2011 IEEE Region 10 Conference, 2011, pp. 1261–1265.

[3] L. Baroffio, L. Bondi, M. Cesana, A. E. Redondi, and M. Tagliasacchi, “A visual sensor network for parking lot occupancy detection in smart cities,” in 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT). IEEE, 2015, pp. 745–750.

[4] M. Idoudi, E. Bourrennane, and K. Grayaa, “Wireless visual sensor network platform for indoor localization and tracking of a patient for rehabilitation task,” IEEE Sensors Journal, vol. 18, no. 14, pp. 5915–5928, 2018.

[5] Y. Wang, G. de Veciana, T. Shimizu, and H. Lu, “Deployment and performance of infrastructure to assist vehicular collaborative sensing,” in 2018 IEEE 87th Vehicular Technology Conference (VTC Spring). IEEE, 2018, pp. 1–5.

[6] E. Arnold, M. Dianati, R. de Temple, and S. Fallah, “Cooperative perception for 3d object detection in driving scenarios using infrastructure sensors,” IEEE Transactions on Intelligent Transportation Systems, pp. 1–13, 2020.

[7] N. Vo, T. Q. Duong, M. Guizani, and A. Kortun, “5G optimized caching and downlink resource sharing for smart cities,” IEEE Access, vol. 6, pp. 31 457–31 468, 2018.

[8] K. Chakrabarty, S. S. Iyengar, Hairong Qi, and Eungchun Cho, “Grid coverage for surveillance and target location in distributed sensor networks,” IEEE Transactions on Computers, vol. 51, no. 12, pp. 1448–1453, 2002.

[9] E. Horster and R. Lienhart, “On the optimal placement of multiple visual sensors,” in Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks, 2006, pp. 111–120.

[10] J. Gonzalez-Barbosa, T. Garcia-Ramirez, J. Salas, J. Hurtado-Ramos, and J. Rico-Jimenez, “Optimal camera placement for total coverage,” in 2009 IEEE International Conference on Robotics and Automation, 2009, pp. 844–848.

[11] J. Zhao, S.-C. S. Cheung, and T. Nguyen, “Optimal visual sensor network configuration.” 2009.
Fig. 10. Point clouds showing the target points over all objects in all the frames for three visibility models. 
(a) our visibility model including occlusion awareness, (b) our visibility model without occlusion awareness and (c) Akbarzadeh et al. [18]. The point colors indicate the visibility score \( \Psi \), ranging from blue (\( \Psi = 0 \), invisible) to red (\( \Psi = 1 \), visible). The white vertical pointer marks the position of the object with least visibility. Sensory poses are indicated by XYZ axis within coloured spheres.
autonomous driving applications at the Intelligent Vehicles group within WMG.

Sajjad Mozaffari is a PhD candidate with the Warwick Manufacturing Group (WMG) at University of Warwick, UK. He received the B.Sc. and M.Sc. degrees in Electrical Engineering at the University of Tehran, Iran in 2015 and 2018, respectively. His research interests include machine learning, computer vision, and connected and autonomous vehicles.

Mehrdad Dianati is a Professor of Autonomous and Connected Vehicles at Warwick Manufacturing Group (WMG), University of Warwick, as well as, a visiting professor at 5G Innovation Centre (5GIC), University of Surrey, where he was previously a Professor. He has been involved in a number of national and international projects as the project leader and work-package leader in recent years. Prior to his academic endeavour, he has worked in the industry for more than 9 years as senior software/hardware developer and Director of R&D. He frequently provide voluntary services to the research community in various editorial roles; for example, he has served as an associate editor for the IEEE Transactions on Vehicular Technology, IET Communications and Wiley’s Journal of Wireless Communications and Mobile.

Paul Jennings received a BA degree in physics from the University of Oxford in 1985 and an Engineering Doctorate from the University of Warwick in 1996. Since 1988 he has worked on industry-focused research for WMG at the University of Warwick. His current interests include: vehicle electrification, in particular energy management and storage; connected and autonomous vehicles, in particular the evaluation of their dependability; and user engagement in product and environment design, with a particular focus on automotive applications.