Received September 9, 2020, accepted September 26, 2020, date of publication October 8, 2020, date of current version October 21, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3029618

Effective Area Coverage of 2D and 3D Environments With Directional and Isotropic Sensors

DIEGO SUŠANJ, (Member, IEEE), DOMAGOJ PINČIĆ, AND KRISTIJAN LENAC, (Member, IEEE)
Faculty of Engineering, University of Rijeka, 51000 Rijeka, Croatia

Corresponding author: Kristijan Lenac (klenac@riteh.hr)

This work was supported in part by the European Regional Development Fund under Grant KK.01.1.1.01.0009 (DATACROSS), and in part by the University of Rijeka, Rijeka, Croatia, through the Research Project Embedded System for 3D Perception.

ABSTRACT The problem of finding the optimal deployment of sensors is becoming increasingly important with the growing expansion of the Internet of Things paradigm and increased usage of sensor networks in different applications. During the installation of sensor networks, sensor placement directly affects the performance of the system. The general problem of determining the position and orientation of the sensors with the goal of optimal coverage of a given environment is NP-hard. In this manuscript, an effective stochastic method for the placement of sensors in arbitrarily given two-dimensional and three-dimensional environments is proposed. The method uses models of generic isotropic and directional sensors with the defined probabilistic coverage. The optimization function combining the environment and sensor models based on the area coverage metric is proposed. Three optimization algorithms are compared with regard to obtained coverage score, execution time, and reliability, and the results are presented and discussed.

INDEX TERMS Sensor placement, sensor models, genetic algorithms.

I. INTRODUCTION

With the growing expansion of the Internet of Things (IoT) paradigm and increased usage of sensor networks in different applications the problem of finding the optimal deployment of sensors in the field relative to some specific performance metric is becoming increasingly important. One of the most important performance metrics is network coverage [1]. The coverage performance by the network of sensors depends collectively on the placement of individual sensors. Depending on the type of sensor, individual sensors may have different individual coverage models. For example, by optimally placing the visual sensors, the performance of the video surveillance system can be increased while simultaneously reducing the number of sensors required to cover a certain area, thus lowering the overall installation and operating costs. It may be important to cover specific parts of the environment of greater significance, such as passages, space around exposed artworks, and the like. When analyzing people's behavior, it is desirable to increase the coverage of certain areas, e.g. in front of product shelves, promotional posters, or showcases, to enable a more detailed analysis of people's interest in a particular product or type of advertisement.

The associate editor coordinating the review of this manuscript and approving it for publication was Xuxun Liu.

The problem of determining the optimal placement of sensors in an environment finds its basis in the problem of the Art Gallery Problem, a famous computational geometry problem posed by Victor Klee as *"What is the smallest number of guards needed to guard an art gallery?"* [2]. This problem has a basis in the visibility, a metric that gives quantitative information on how good the sensors detect and see the object in the environment, a common factor for the fields of robotics, computer vision, optimization, and computational graphics.

Klee's original Art Gallery Problem was to determine the covering positions for guards who can survey 360° around their fixed position, while the art gallery is described with a polygon. Four types of guards are defined: vertex, point, edge, and mobile guard. Vertex guard limits the positions of guards to polygon vertices, point guard limits the positions inside the polygon, while edge guard allows only placement along the polygon edges. The mobility guard is allowed to move along a sequence of closed line segments completely contained in the simple polygon. The term "polygon" is usually modified by "simple" to distinguish it from polygons that cross themselves [3]. This means that the original Art Gallery Problem assumes only the line of sight restriction: two points are visible if the line segment between them does not intersect any object. Some works extended the problem with the range and incidence constraints [4], but still kept the simple two-dimensional, polygon, approach.

These studies used a polygonal region in the plane as the model of the art gallery. The plane polygon only models the floor outline of the art gallery. It does not always provide adequate information about the complex spatial structure of the building. In many applications, knowledge of the spatial structure of the building is essential for deciding how the building should be monitored [5].

For three-dimensional environments, studies are usually based on elevation maps restricted to planar terrains [6] and orthogonal polyhedral environments [7]. The Art Gallery Problem, in both two-dimensional and three-dimensional environment, is proven to be NP-Hard, although, for some instances regarding two-dimensional problems and some three-dimensional with simple polygons, a solution can be found [3], [4], [8].

There are many real-life problems extrapolated from Art Gallery Problem making the mentioned problem more than a geometric exercise. Examples of the Art Gallery Problem usage are found in security applications (placing security cameras or guards) but also in other fields like placing TV cameras in a showroom, arranging the lighting sources in a room, or placing radar stations. Practical implementations of the Art Gallery Problem are increasing the complexity of the stated problem, as the real world has great variety in shapes of the polygons and a global solution that would suffice for all polygon types has not been found yet [2].

The first aspect of the problem is related to the type of used coverage, which directly affects the optimization process. There are multiple types of coverage metrics as the sensors are placed with a specific use in mind [9], [10]. The most widely used metric is the area coverage which is defined as the ratio of the area covered by the sensors to the total target area [11]–[13]. Special cases of area coverage include point and barrier coverage, where certain locations in the target environment have a higher importance in the coverage problem. The objective of the k-coverage is to cover each location in the target environment by at least k sensors. There are multiple reasons to use k-coverage, for example, robustness and positioning [14].

Based on the aforementioned Art Gallery Problem, it is evident that the second aspect of the problem relates to the dimension of the target environment. Even though two-dimensional environments represent a simpler computational problem, such an approximate approach can lead to errors. Those errors are the result of an attempt to model a real-world environment in a lower, two-dimensional space, and are caused by an overestimation in the calculation of area coverage. Three-dimensional environments represent a more faithful description of a real-world environment, which can lead to a much better approximation of sensor positions, but with the inherent problem of higher computational cost, i.e. slower computation.

In any case, when calculating the area coverage, the topography of the environment, and the existing obstacles that cover the sensory area of each sensor should be taken into account [15]. Existing research in this area has typically assumed two-dimensional environments [16]–[20] or approximated a three-dimensional problem with two-dimensional. The approximation approach used a fixed sensor height and limited search space to one plane [21], [22]. Throughout this manuscript, both two-dimensional and three-dimensional types of environments will be examined. Since a three-dimensional case expands the search space, thus significantly hardens the problem, emphasis will be on the aforementioned case.

A further aspect of the problem is related to the detection ability of the sensor [23]. The detection ability of the sensor describes if the sensor "sees" an object, in case of binary coverage, or how good it "sees" it, for probabilistic coverage. Detection is usually modeled using binary coverage for each of the sensors [17]–[20], [24], [25], but probabilistic coverage provides a more refined model [21], [26]–[28].

In addition to the detection capability, the shape of the area covered by the sensor is an important aspect. The coverage area of the sensor depends on the sensor type. The assumption of isotropic sensing ability is only true for some types of sensors (Bluetooth beacons, Wireless beacons...), while others have directional sensing ability (cameras, ultrasonic sensors...). Sensor, in the context of this manuscript, is the device that can receive or provide some kind of signal. Examples of sensors providing the signal are signal beacons, which emit a radio or light signal in their vicinity. An example of the latter one is the Camera, which receives the light signal and detects the visible light spectrum in the form of a video.

In this manuscript, an effective method for the problem of optimal placement of the sensors in a three-dimensional environment based on the probabilistic coverage model and stochastic approach is proposed. The proposed solution consists of the environment and sensor models combined with the optimization function derived from the area coverage metric. Any number of sensors can be used and placed in the environment using any algorithm for derivative-free, nonlinear, and constrained optimization. The stochastic approach is proposed, as an effective method for the proposed solution.

Through the definition of the environment and sensor models that roughly describe the real world the proposed solution makes the search for an optimal solution practical. Increasing the sensor coverage by optimally placing the sensors, improves the performance of the sensor network.

To validate the proposed solution experiments were performed using environments and sensors modeled based on their real-world physical characteristics. Models for both isotropic and directional sensors were proposed.

Three nature inspired genetic algorithms were compared with regard to their obtained coverage score, execution time, and reliability. Algorithms were compared in test cases with different environments, rasterization values, sensor types, and numbers of sensors.

To summarize, the following contributions are presented in this manuscript:

- An effective method for addressing the problem of optimal placement of sensors in a given environment
- Environment modeling based on real-world physical characteristics using both two-dimensional and three-dimensional representations
- Generic sensor models for isotropic and directional sensors with the probabilistic coverage
- Optimization function based on the area coverage metric
- Comparison between three nature-inspired genetic algorithms for the problem of sensor placement

The manuscript is organized as follows: in Section II and Section III proposed environment and sensor models are introduced, followed by the description of the optimization function in Section IV. After the optimization function, the experimental setup containing the chosen environment and sensor models, as well as the descriptions of the used algorithms and hardware is explained in Section V. Section VI contains a few chosen case studies of the optimization results followed by the results of the comparison between the selected algorithms. Finally, the concluding remarks are given in Section VII.

II. ENVIRONMENT MODELS

To evaluate the estimated positions of the sensors, environment models used to roughly describe the real world environments are defined using the free and occupied space. A rough model of the real world environment is defined by its ground plane layout. Free and occupied space can be arbitrary defined with the polygonal three-dimensional modeling. Environment models are defined in three-dimensional space, but if only a two-dimensional representation of the defined environment is needed their ground planes can be used.

To allow for the calculation of the space coverage, the environment is further split in voxels. The rasterization parameter defines the dimensions of one voxel and, as a product, the granularity of space coverage. The number of voxels in the environment, therefore, depends not only on the layout and dimensions of the free and occupied space but also on the rasterization parameter. The presupposition is, that the more voxels in an environment, the more accurate the description of the world is.

Using both two-dimensional and three-dimensional space representations, as well as different rasterization parameters, enables the evaluation of the performance of the optimization algorithms on test cases with different complexities.

III. SENSOR MODELS

Sensor models are defined through the specification of the rules for the visibility calculation for every voxel in the environment surrounding the sensor. As stated earlier, sensor models that use the probabilistic coverage model are proposed. There are two main types of sensors, based on their sensing ability. The first one is sensors with isotropic sensing ability, while the second is the sensors with directional one.

Visibility v_{vs} between each of the voxels v_i and sensors s_j is defined as a product of three base functions:

$$v_{vs}(v_i, s_j) = v_d(v_i, s_j) \cdot v_\varphi(v_i, s_j) \cdot v_\theta(v_i, s_j), \tag{1}$$

where v_d is the visibility value which depends on the distance between the sensor and the observed voxel, while the visibilities v_φ and v_θ are dependent on the azimuth and inclination angles between the sensor and the voxel. All three visibilities calculations depend on the used sensor type and will be described separately in the following subsections.

Visibility is only calculated if there is a line of sight between the voxel and the sensor. Distance d_{vs} between the voxel and the sensor used in distance visibility function v_d is calculated using euclidean distance formulation:

$$d_{vs}(v_i, s_j) = \sqrt{(v_{i_x} - s_{j_x})^2 + (v_{i_y} - s_{j_y})^2 + (v_{i_z} - s_{j_z})^2}. \tag{2}$$

Both azimuth φ and inclination θ, are calculated as angles in the spherical coordinate system:

$$\varphi_{vs}(v_i, s_j) = \arctan 2\left(v_{i_y} - s_{j_y}, v_{i_x} - s_{j_x}\right), \tag{3}$$

$$\theta_{vs}(v_i, s_j) = arcsin\left(\frac{v_{i_z} - s_{j_z}}{d_{vs}(v_i, s_j)}\right). \tag{4}$$

Those two angles are used in the calculation of the azimuth visibility v_φ and inclination visibility v_θ appropriately.

A. ISOTROPIC SENSORS

The isotropic sensors are ones that have the same uniform sensing ability, regardless of the signal direction. As such, all sensors with the isotropic antenna have the angular visibility of an observed voxel as a constant and defined as:

$$v_\varphi(v_i, s_j) = 1, \tag{5}$$

for the azimuth and as:

$$v_\theta(v_i, s_j) = 1, \tag{6}$$

for the inclination angle.

One of the most common representatives of the isotropic sensor is a radio transmitting beacon with an isotropic antenna that incessantly emits radio signals in its vicinity. In the following, we propose a simplified model for a radio transmitting beacon that does not take into account the influence of radio wave reflection or absorption.

Distance visibility v_d for this sensor type is modeled as Received signal strength indicator (RSSI) function:

$$rssi(v_i, s_j) = -20 \cdot log_{10}(d_{vs}(v_i, s_j)) + rssi_{1m}. \tag{7}$$

The received signal strength indicator is a value that depends both on the distance from the sensor as well as the signal strength at the distance of one meter $rssi_{1m}$. The received signal strength indicator value depending on the distance is shown in Figure 1.

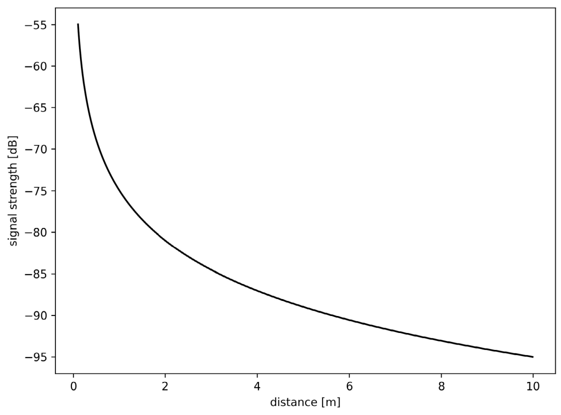


FIGURE 1. Received signal strength (*rssi*) depending on the distance between voxel and sensor.

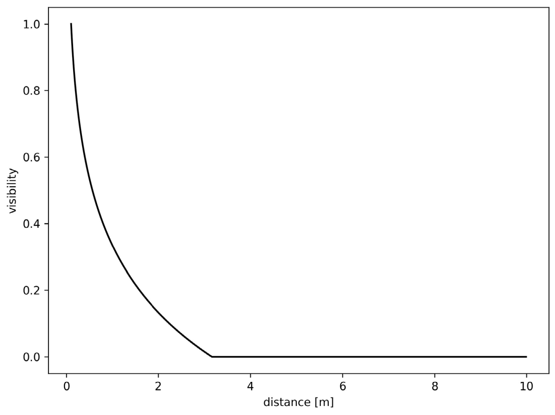


FIGURE 2. Distance visibility value (v_d) for modeled sensor.

As received signal strength indicator value drops as the distance increases, the signal strength drops below the strength of the noise signal. As the value of noise represents minimal usable value, a signal-to-noise ratio (SNR) is used. The signal-to-noise ratio is the difference between the received signal strength indicator and the noise values in decibels. As any value equal or below the noise level is considered not usable resulting value is set to zero for any signal-to-noise value not bigger than assumed noise value:

$$snr(v_i, s_j) = \begin{cases} rssi(v_i, s_j) - noise, & \text{iff } rssi(v_i, s_j) > noise \\ 0, & \text{otherwise} \end{cases}. \quad (8)$$

Calculation of the optimization function depends on the probabilistic visibility values, values between zero and one. To satisfy that requirement a signal-to-noise value is normalized using the $rssi_{max}$ value, a value of the received signal strength at the closest point. This normalized signal-to-noise ratio is the value used as distance visibility v_d, Figure 2, for this sensor:

$$v_d(v_i, s_j) = \frac{snr(v_i, s_j)}{rssi_{max} - noise}. \quad (9)$$

B. DIRECTIONAL SENSORS

The second model presented, representing the directional sensing group, is one of the stereo cameras. A Stereo Camera is a system of two RGB cameras in the same enclosure, usually lying in the same plane and on the fixed distance. Depth data, or distance from the sensor plane, is calculated through the process called Stereo matching.

Stereo matching is a process typically done on rectified images from both cameras [29]. The relative relationship between the pair of images is preserved after the rectification of both images in the form of a Q matrix:

$$Q = \begin{bmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 0 & f \\ 0 & 0 & \frac{-1}{T_x} & \frac{c_x - c'_x}{T_x} \end{bmatrix}. \quad (10)$$

In a Q matrix, a value T_x is the distance between cameras (baseline), usually in meters or millimeters. Values c_x and c_y are the coordinates in pixels of the principal point of the reference camera, usually a left one. Value c'_x is the x coordinate of the principal point of the non-reference camera. When both camera resolutions are equal and the normal rectification process is done, the c_x and c'_x will have the same values and the last element in the Q matrix will be zero. Value f is the focal length in pixels.

The key step of the stereo matching process is searching for matching pixels in both images corresponding to the same point in the scene [30]. For rectified images, those matching pixels are found on the same epipolar line, i.e. in the same row of the image. A search is only performed up to a maximum distance d_{max} from the reference pixel in one direction, in case no matching pixel is found, for example, due to occlusions. The distance between x coordinates of the matching pixels is called *disparity*.

From the *disparity* value and the values from Q matrix, a *depth* value can be calculated:

$$depth = \frac{T_x \cdot f}{disparity}. \quad (11)$$

As the disparity level is a discrete value, the error function used for distance visibility calculation is defined as the distance between two neighboring disparity levels. For the voxel at the specific distance d_{vs}, the error is calculated as the distance between which that specific distance is located:

$$error_d(v_i, s_j) = \frac{T_x \cdot f}{\lfloor disparity \rfloor} - \frac{T_x \cdot f}{\lceil disparity \rceil}, \quad (12)$$

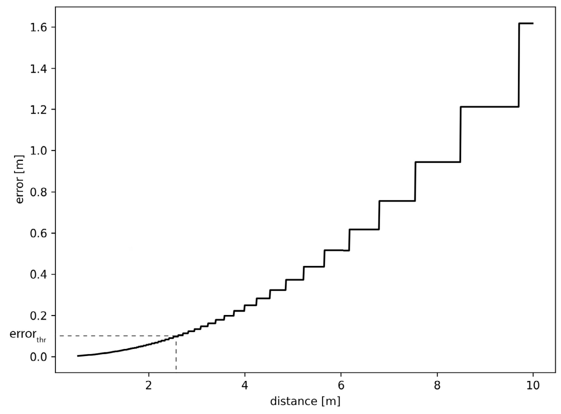


FIGURE 3. Error value ($error_d$) for stereo camera.

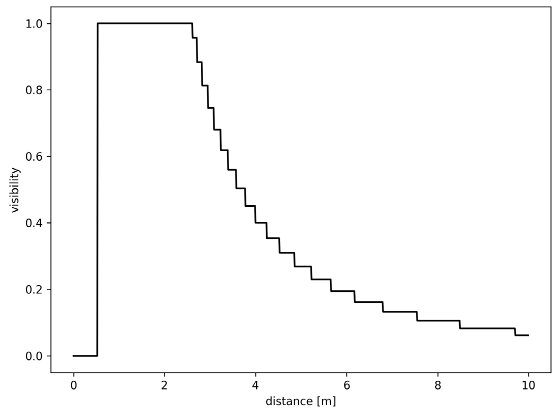


FIGURE 4. Distance visibility value (v_d) for stereo camera.

and then disparity calculation from the distance between the sensor and a voxel is added to the formula:

$$error_d(v_i, s_j) = \frac{T_x \cdot f}{\left\lfloor \frac{T_x \cdot f}{d_{vs}(v_i, s_j)} \right\rfloor} - \frac{T_x \cdot f}{\left\lceil \frac{T_x \cdot f}{d_{vs}(v_i, s_j)} \right\rceil}. \tag{13}$$

Error value for Stereo Camera depending on the distance is shown in Figure 3.

The measure used in the optimization function is visibility with the assumption of 1 being fully visible and 0 not visible. Depth data is usually used to detect objects in the scene. As such, during the modeling of the sensor, the minimal size of the object expected to be detected should be taken into account. If the distance between two consecutive disparity levels is smaller than a $error_{thr}$ value, a voxel is assumed to be fully visible. Furthermore, as this distance increases, the error rises and consequently, visibility value decreases, Figure 4:

$$v_d(v_i, s_j) = \begin{cases} 1, & \text{iff } error_d(v_i, s_j) < error_{thr} \\ \dfrac{error_{thr}}{error_d(v_i, s_j)}, & \text{otherwise .} \end{cases} \tag{14}$$

Coordinates of the voxels are converted from the environment's global coordinate system to the local coordinate system of the sensor using the pitch and yaw angles of the camera. Following the conversion, azimuth and inclination angles between the sensor and a voxel are defined using previously described formulas. The angular visibility model depends on two main characteristics of the sensor: field of view and the used stereo matching algorithm. Fields of view, both vertical and horizontal, affect the width of the visible angle spectrum, while the other features of the angle spectrum such as the slope of its boundaries depend on the used stereo matching process.

In Figure 5 an example of basic dimensions of one image is shown. Full thick lines represent image borders with the image width x_{dim} and image height y_{dim}. The thin dotted lines show the previously defined camera principal point C and its position in the image c_x and c_y. Thick dashed lines are separated from the vertical borders by maximum disparity d_{max}. There are two arrows marked with in_1 and in_2 pointing to two horizontal borders of the image as well as four marked with az_1, az_2, az_3, and az_4 pointing to two vertical image borders, together with two lines separated from those borders for d_{max}. Those six values, i.e. positions, will be used for angular visibility calculations, for both inclination and azimuth visibility respectively.

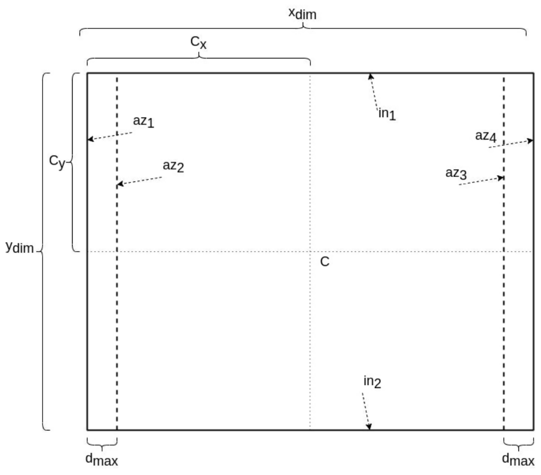


FIGURE 5. Representation of used values for angular visibility calculation.

As the basic stereo matching algorithm is trying to match the pixels in the same row, inclination dependent visibility has only two states: zero and one. Voxels with the inclination angle inside the vertical field of view domain are fully visible, while others, outside of the vertical field of view, are completely invisible. Two angles in_1 and in_2, calculated from those vertical image boundaries are defined as:

$$in_1 = \arctan 2(T_x \cdot (0 - c_y), T_x \cdot f), \tag{15}$$

and:

$$in_2 = \arctan 2(T_x \cdot (y_{dim} - c_y), T_x \cdot f), \tag{16}$$

where $0 - c_y$ is the distance in pixels of the upper vertical boundary from the principal point, while $y_{dim} - c_y$ is a distance between the lower vertical boundary and the principal point. Those two values are used as the limits of the vertical field of view in the inclination visibility calculation:

$$v_\theta(v_i, s_j) = \begin{cases} 1, & \text{iff } in_1 \leq \theta(v_i, s_j) < in_2 \\ 0, & \text{otherwise .} \end{cases} \tag{17}$$

Graph showing inclination visibility for proposed sensor is given in Figure 6.

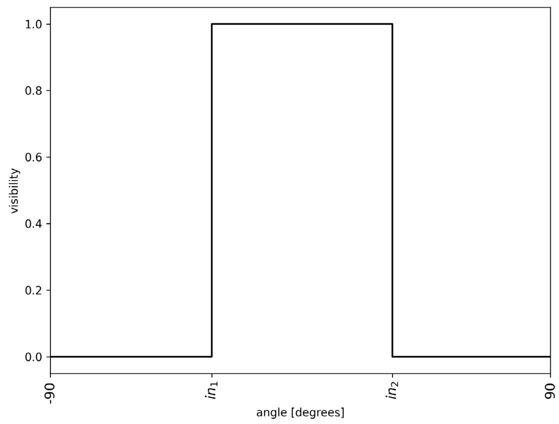


FIGURE 6. Inclination visibility value (v_θ) for stereo camera.

There is a small difference between inclination and azimuth visibility. Similar to the inclination based visibility, all the voxels with the azimuth outside of the horizontal field of view are not visible. All voxels whose corresponding pixels are inside the image and at most d_{max} from each edge are considered to be fully visible. The leftmost and the rightmost limits are image boundaries, defined as:

$$az_1 = \arctan 2(T_x \cdot (0 - c_x), T_x \cdot f), \tag{18}$$

and:

$$az_4 = \arctan 2(T_x \cdot (x_{dim} - c_x), T_x \cdot f). \tag{19}$$

Limits of the fully visible space are calculated on the distance of d_{max} from left edge:

$$az_2 = \arctan 2(T_x \cdot (d_{max} - c_x), T_x \cdot f), \tag{20}$$

and right edge:

$$az_3 = \arctan 2(T_x \cdot (x_{dim} - d_{max} - c_x), T_x \cdot f). \tag{21}$$

For those voxels that have corresponding pixels near the edges of the image, visibility is defined as the ratio between the distance from the edge and d_{max}:

$$v_\varphi(v_i, s_j) = \begin{cases} \dfrac{\varphi(v_i, s_j) - az_1}{az_2 - az_1} & \text{iff } az_1 \leq \varphi(v_i, s_j) < az_2 \\ 1, & \text{iff } az_2 \leq \varphi(v_i, s_j) < az_3 \\ \dfrac{\varphi(v_i, s_j) - az_4}{az_3 - az_4} & \text{iff } az_3 \leq \varphi(v_i, s_j) < az_4 \\ 0, & \text{otherwise .} \end{cases} \tag{22}$$

Such calculation is derived from the stereo matching algorithm and the smaller search space it has near the edges. The graph for the voxel azimuth visibility is shown in Figure 7.

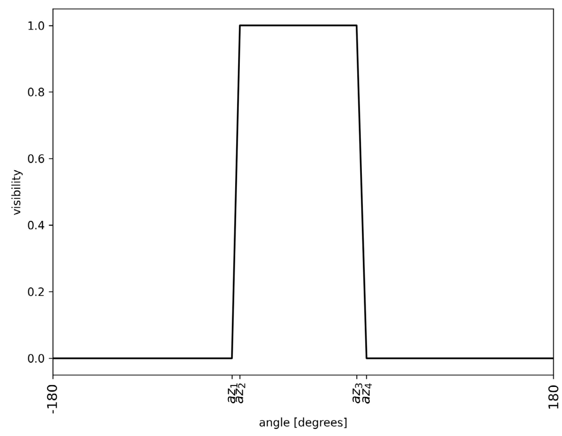


FIGURE 7. Azimuth visibility value (v_φ) for stereo camera.

IV. OPTIMIZATION FUNCTION

In order to enable the search for optimal positions of the sensors in a given environment, it is necessary to define an optimization function that evaluates the selected solution. That optimization function is a minimization function connecting the previously described environment and sensor models based on the chosen metric. The optimization function describes a single-objective, continuous problem. Any optimization algorithm with the ability of derivative-free, nonlinear, and constrained optimization can be used with it.

The metric used in the proposed solution is the area coverage metric, defined as the ratio of the area covered by the sensors to the total target area. The optimization algorithm aims to minimize the global loss value obtained as a result of the optimization function. Loss value is defined as the complement of the coverage.

Arising from the chosen coverage metric, global loss value L is defined as the arithmetic mean of the loss values for all n voxels in the environment V:

$$L(V, S) = \frac{1}{n} \sum_{i=1}^{n} L_v(v_i, S). \tag{23}$$

As one voxel can be visible from multiple sensors in the environment, a loss for each voxel, L_v, is defined as a product of the loss values from each of the m sensors in the sensor set S:

$$L_v(v_i, S) = \prod_{j=1}^{m} l_{vs}(v_i, s_j). \tag{24}$$

Loss value for each combination of the sensor and voxel in the environment, l_{vs}, is defined as the complement of that pair visibility, as previously defined in Equation 1:

$$l_{vs}(v_i, s_j) = 1 - v_{vs}(v_i, s_j). \tag{25}$$

For the definition of sensor positions, yaw and pitch angles, continuous values are used. Voxel values are defined with discrete positions derived from the environment using the defined rasterization parameter.

V. EXPERIMENTAL SETUP

A. ENVIRONMENTS

Three test environments were defined by varying their shape and the number and shape of the obstacles contained within. For each of the defined three-dimensional environments, their ground plane was used as a corresponding two-dimensional test case.

The first environment, Figure 8, is U-shaped, with no obstacles, and is representative of an environment where more than one sensor is needed to obtain a satisfactory coverage of the environment, due to the curvature present in the environment. In this scenario, one sensor might have a fairly large coverage of space, but it cannot cover parts of the environment that are hidden, due to the environment shape.

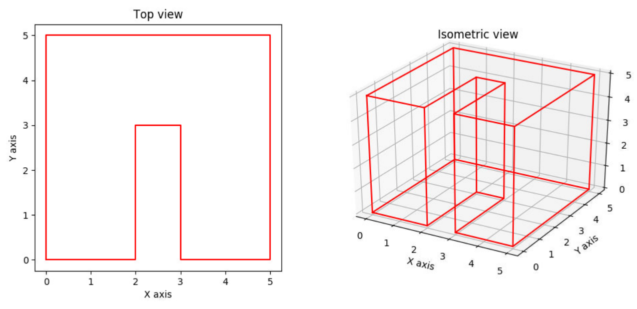


FIGURE 8. Environment 1 2*D* ground plan (left image) and 3*D* isometric view (right image).

The second environment, Figure 9, represents a simple quadratic shaped environment where multiple obstacles are present. Because of the obstacles, an individual sensor cannot have a large coverage of space. In three-dimensional space, obstacles have different heights, but they are diagonally symmetrical.

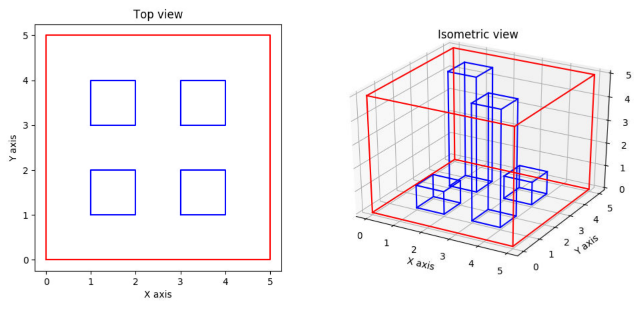


FIGURE 9. Environment 2 2*D* ground plan (left image) and 3*D* isometric view (right image).

The third environment, Figure 10, represents a scenario where multiple obstacles are isolating parts of the environment. The intention was to test the behavior of the optimization algorithm on the environment with the discontinuity, where the optimization algorithm is forced to either place the sensor in the enclosed part of the environment or to place sensors in the nonisolated parts of the environment. This way,

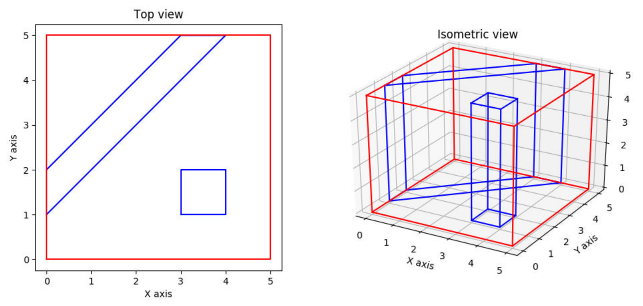


FIGURE 10. Environment 3 2*D* ground plan (left image) and 3*D* isometric view (right image).

an isolated part of the environment can be covered by a sensor only if it is contained inside that isolated space.

B. ENVIRONMENT RASTERIZATION

Rasterization values of $0.1m$, $0.5m$, and $1.0m$ were used to simulate different environment granularities. The chosen rasterization value determines the size of the voxel edges and as such the number of voxels in the simulated environment. The number of voxels for each of the environments in both two-dimensional and three-dimensional representation depending on the rasterization value is shown in Table 1.

TABLE 1. Number of voxels for each test environment and rasterization value.

Rasterization value [m]	Environment 1		Environment 2		Environment 3	
	2D	3D	2D	3D	2D	3D
0.1	2331	116550	2117	115530	2099	104950
0.5	115	1265	85	1079	88	968
1	36	216	20	152	23	138

C. SENSORS

In our test cases, two types of sensors are used, with isotropic and directional sensing abilities. For the isotropic sensor, a previously described model of a radio transmitting beacon instantiated with parameters derived from the specifications of the Estimote Bluetooth Low Energy sensor [31] was used. Parameters based on the ZED Stereo camera [32] were used to instantiate a model of a directional sensor. The number of sensors ranges from one to three sensors in a specific test case. The values of the used parameters are listed in Table 2.

TABLE 2. Parameters of the used sensors.

Isotropic sensor (Estimote Bluetooth Low Energy)		Directional sensor (ZED Stereo camera)	
$rssi_{1m}$	$-75dB$	T_x	$0.12m$
$noise$	$-85dB$	f	565.99
		$error_{thr}$	$0.1m$
		x_{dim}	1280
		y_{dim}	720
		d_{max}	128
		c_x	702.44
		c_y	351.20

D. OPTIMIZATION ALGORITHMS

As stated in Section IV, any optimization algorithm with the ability of derivative-free, nonlinear, and constrained optimization can be used for the minimization of the optimization function. Three nature-inspired genetic algorithms are used: Artificial Bee Colony Algorithm (ABC) [33], Fireworks Algorithm (FWA) [34] and Particle Swarm Optimization Algorithm (PSO) [35]. The mentioned algorithms are meta-heuristic, swarm intelligence based algorithms, designed for optimizing non-linear functions in a multidimensional space. In this manuscript, the optimization function described previously represents a non-linear minimization function. Sensor positions and angles are independent variables which are optimized. They represent a multidimensional search space, whose size varies not only based on the environment dimensionality but also on the number and type of sensors used.

The PSO algorithm was chosen because it is one of the most used swarm optimization algorithms, and was proved to be effective at many optimization problems. As an example of a more modern optimization algorithm, the ABC algorithm was used, which was proven to be more effective at specific optimization cases [33]. Furthermore, the FWA algorithm was used as an example of a recent swarm optimization algorithm, showing a significant improvement over the PSO algorithm [34]. Also, the same combination of algorithms was used in [36] for mobile robot path planning, which proves PSO, FWA, and ABC algorithms to be a viable choice for optimization problems in continuous space.

For every algorithm, the same maximum number of evaluations was defined and set to 100 evaluations per independent variable. Regarding the algorithm parameters, swarm size was determined empirically, by testing different swarm sizes $(10, 20, 30, 40, 50)$ on problems with two and nine independent variables. No statistically significant differences were found on the results comparing different swarm sizes, so the swarm size of 10 was used for all the algorithms. Furthermore, for the FWA and the PSO algorithms, other parameters were determined experimentally, namely for the FWA algorithm amplification and reduction coefficient were set to 10, and for the PSO algorithm cognitive and social components were set to 1.0 and inertial weight to 0.7.

E. HARDWARE SETUP

Considering the number of test cases and their computational complexity, powerful hardware was used to carry out all the calculations. The supercomputer "Bura" at the University of Rijeka was used [37]. More specifically, 40 nodes with 24 physical cores each were used in the experiments. To assure undisturbed processing and alleviate the comparison of execution times, each of the test cases was run on a dedicated physical core.

VI. EXPERIMENTAL RESULTS

As described earlier, three environments were used in the experiments. For every environment, a two-dimensional and a three-dimensional space models were generated, and for each of those spaces, three different rasterization values were defined. Also, two different types of sensors were being placed in the environment with the number of sensors varying from one to three sensors. Three different optimization algorithms were used. Furthermore, every test case with each of the algorithms was run for 100 iterations to account for the stochastic nature of the optimization algorithms. In total, considering all cases, the optimization was executed 32400 times.

The score was calculated as a complement of the loss value of the optimization function, $1 - L(V, S)$, for each execution of the optimization algorithm and for every optimization step, i.e. execution of the optimization function. Its value ranges between zero and one depending on how good the environment is covered by that particular sensor configuration.

Firstly, a few optimization results will be presented, followed by the results of the algorithm comparison over all the test cases.

A. CASE STUDIES

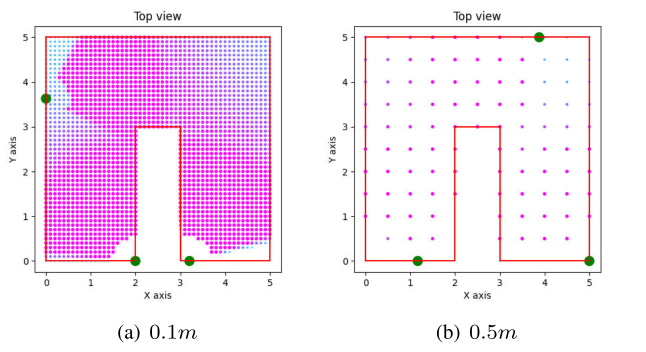
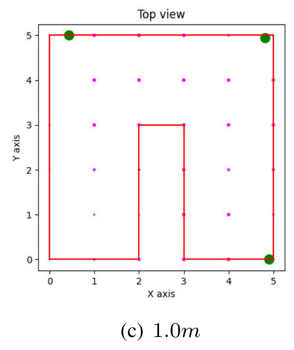
In all the figures in the following examples, green filled circles denote sensors, red lines are outlines of the free space, while blue ones are outlines of the obstacles. Voxels visible by one or more sensors are colored based on their visibility. Fully visible voxels, the ones with visibility of one, are colored with magenta, while the ones that are almost not visible, i.e. with the visibility of that voxel close to zero, from any sensor have been colored in cyan. Voxels that are not visible from any of the sensors are not shown. In the following examples, the ABC algorithm was used, since it produced the best overall coverage score.

The first example shows one of the proposed environments, Environment 1, described in the previous section. In Figure 11 coverage by three Stereo Camera sensors for three different rasterization values ($0.1m$, $0.5m$, and $1.0m$) is shown in a), b), and c) respectively.

Additionally, the influence of the different rasterization values on the reliability of optimization algorithms is shown in Figure 12, where, for the specific test case, three different rasterization values were used as a varying parameter, all else being equal. In this test case three Bluetooth Low Energy sensors were being placed in three-dimensional space in the Environment 3. Results from all iterations of the test case are shown for a specific rasterization value.

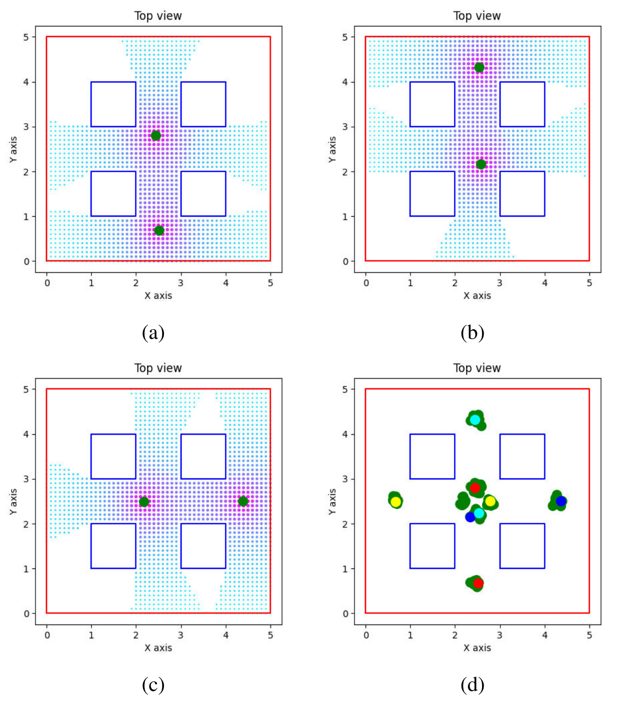
It is visible that the distribution of sensor positions largely depends on the rasterization value. For a large rasterization value ($1.0m$), the dispersion of sensor positions is large, while for smaller rasterization value ($0.1m$) distribution has significantly less dispersion.

The third example shows Environment 2 with the estimated positions for isotropic sensors. Specifically, two Bluetooth Low Energy sensors were placed in the two-dimensional environment rasterized with the $0.1m$ rasterization value. It is evident that for the placement of two sensors in the particular environment shown, different solutions with the same or similar score exist. In figure 13, a), b) and c) show

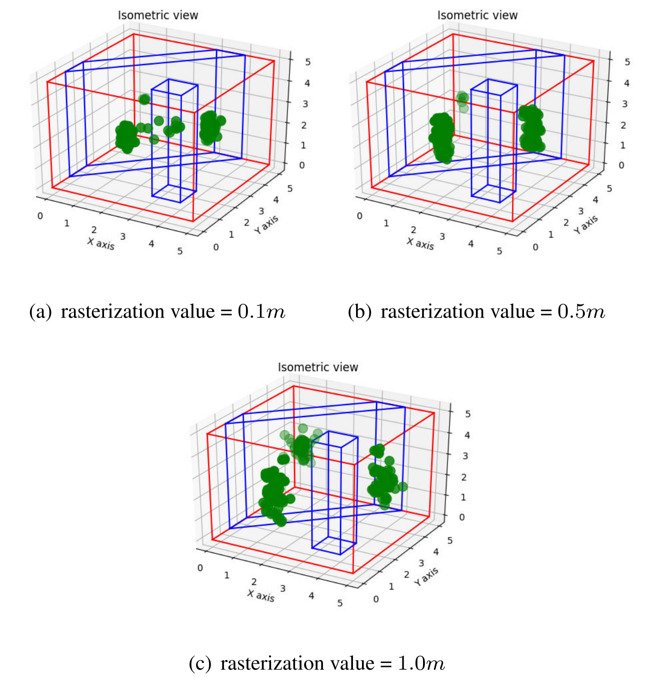
(a) $0.1m$ (b) $0.5m$ (c) $1.0m$

FIGURE 11. Example of the coverage for different rasterization values for two-dimensional environment 1.



(a) (b) (c) (d)

FIGURE 13. Example of different solutions for two-dimensional environment 2.



(a) rasterization value = $0.1m$ (b) rasterization value = $0.5m$ (c) rasterization value = $1.0m$

FIGURE 12. Example of the dispersion of estimated position, i.e. algorithm reliability, for different rasterization values for three-dimensional environment 3.

few such solutions found by the algorithm. Additionally, all positions of all iterations for this test case were combined in one view, d), which shows the resulting distribution of sensor positions in all iterations. Marked in cyan, blue, red, and yellow are results chosen from four random iterations, showing the general rule of sensors placement in this specific test case.

An example of a test case where there is no large variety of found solutions for sensor placement is shown in Figure 14. Two Bluetooth Low Energy sensors were placed in three-dimensional space of the Environment 2 with the rasterization set to $0.1m$.

In a), b), c), and d) two random iterations of the test case are shown and have similar sensor positions. To further examine how sensors were being placed across iterations a plot of all sensors positions in all iterations of the test case is shown in e), from a top view for easier visualization.

The next example is one with different numbers of Stereo Camera sensors being placed in the Environment 3 with the rasterization value of $0.1m$ and two-dimensional space.

In Figure 15 a) and b) one sensor was being placed, on c) and d) two sensors were placed, and in e) and f) three sensors were being placed. Due to the environment shape, it is visible that many near-optimal positions exist, for every number of sensors.

B. ALGORITHM COMPARISON

As mentioned earlier, 100 iterations of 108 distinct test cases were done for each of the algorithms. For each of the test cases, minimal, average, and maximum score and time were calculated. Both score and execution time are directly dependent on the size, shape, and rasterization parameter of the environment as well as the type and number of sensors. As such, a direct comparison between test cases is not viable.

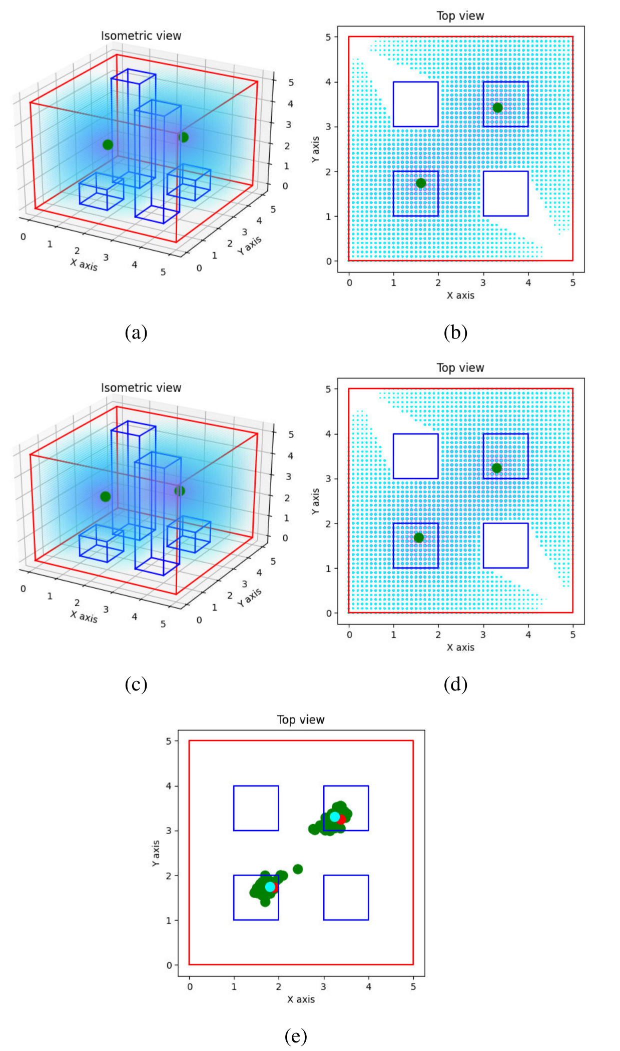


FIGURE 14. Example of different solutions for three-dimensional environment 2.

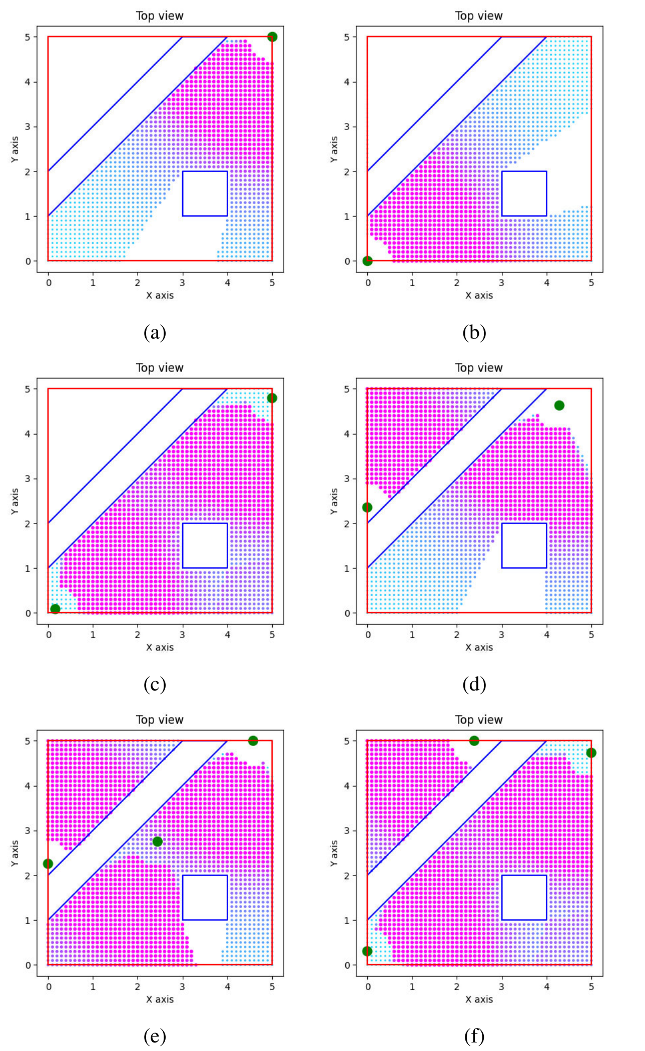


FIGURE 15. Example of solutions for different numbers of stereo camera sensors in two-dimensional environment 3.

For each algorithm, the scores and execution times for each test case presented in the following are calculated as the average over all the 100 iterations corresponding to that test case. In Table 3 the results for Environment 1 are shown. The results are similar for the other two considered environments and are omitted for brevity. Times are displayed in seconds and rounded to the closest integer value to simplify the display.

Due to the inability to directly compare the results of these 108 test cases, simple win counting is introduced. Win per each test case is credited to the algorithm having the best score in each of the comparable categories. Counting is done over three categories: lowest, *min*, average, *mean* and highest, *max*, coverage score over 100 iterations of each test case.

Score win counts for each algorithm per category are shown in Figure 16. From these results, an insight about the algorithm performance can be drawn. The ABC algorithm had the highest minimum score in 79 test cases as well as the highest average score in 76 test cases. The ABC algorithm had only 51 wins in the category of maximum scores, surpassed by the PSO algorithm with 57 wins. The PSO algorithm had a significantly lower number of wins in minimum and average score categories with 23 and 32 respectively. The FWA algorithm did not achieve significant results in any of the categories, achieving only 6, 0 and 1 wins.

A similar comparison was done on resulting times with the goal of having a lower execution time per category, Figure 17. In categories of average and maximum execution time, the FWA algorithm achieved wins in 84 and 75 respectively, while in the minimum category it was surpassed by the PSO algorithm with 54 wins compared to 49. The PSO algorithm was second with 20 wins in the categories of average and maximum time. The slowest algorithm was the ABC algorithm with 5, 4, and 13 wins per each of the categories.

Based on these win counts the ABC algorithm gave the best results overall but with the slowest time. The FWA algorithm finished the fastest but with the worst results. The

TABLE 3. Average scores and times for environment 1.

Rasterization value [m]	Sensor Type	Number	Scores ABC	Scores FWA	Scores PSO	Times [s] ABC	Times [s] FWA	Times [s] PSO
2D environment								
0.1	Bluetooth Low Energy	1	0.1440	0.1430	0.1445	210	211	200
		2	0.2715	0.2570	0.2723	855	867	834
		3	0.3660	0.3457	0.3664	1949	1976	1959
	Stereo Camera	1	0.4284	0.3676	0.3973	401	358	381
		2	0.6763	0.5672	0.5582	1614	1460	1536
		3	0.8147	0.6970	0.6776	3625	3292	3448
0.5	Bluetooth Low Energy	1	0.1299	0.1288	0.1302	11	11	10
		2	0.2454	0.2334	0.2460	44	44	43
		3	0.3345	0.3173	0.3350	101	99	100
	Stereo Camera	1	0.3827	0.3274	0.3470	20	18	19
		2	0.6122	0.5243	0.5207	81	74	76
		3	0.7738	0.6618	0.6455	182	167	172
1.0	Bluetooth Low Energy	1	0.1193	0.1153	0.1195	3	3	3
		2	0.2283	0.2096	0.2278	14	14	14
		3	0.3178	0.2900	0.3176	32	32	32
	Stereo Camera	1	0.3414	0.3044	0.3100	6	6	6
		2	0.5719	0.4969	0.4882	25	24	24
		3	0.7385	0.6238	0.6009	57	53	54
3D environment								
0.1	Bluetooth Low Energy	1	0.0792	0.0765	0.0795	18431	18461	17995
		2	0.1510	0.1369	0.1512	73940	75445	73961
		3	0.2085	0.1872	0.2087	168120	169525	166848
	Stereo Camera	1	0.2741	0.2262	0.2247	41398	36574	39299
		2	0.4715	0.3645	0.3243	166748	148269	156533
		3	0.5917	0.4580	0.4024	377526	332159	355918
0.5	Bluetooth Low Energy	1	0.0669	0.0651	0.0670	209	210	203
		2	0.1283	0.1169	0.1283	842	846	835
		3	0.1780	0.1621	0.1779	1894	1911	1902
	Stereo Camera	1	0.2389	0.2004	0.1951	461	410	433
		2	0.4220	0.3273	0.2939	1866	1661	1770
		3	0.5407	0.4071	0.3767	4198	3765	3945
1.0	Bluetooth Low Energy	1	0.0541	0.0534	0.0543	37	37	36
		2	0.1055	0.0977	0.1056	147	150	147
		3	0.1489	0.1364	0.1491	331	337	339
	Stereo Camera	1	0.2078	0.1749	0.1712	79	71	76
		2	0.3696	0.2868	0.2629	321	290	307
		3	0.4817	0.3680	0.3357	719	655	690

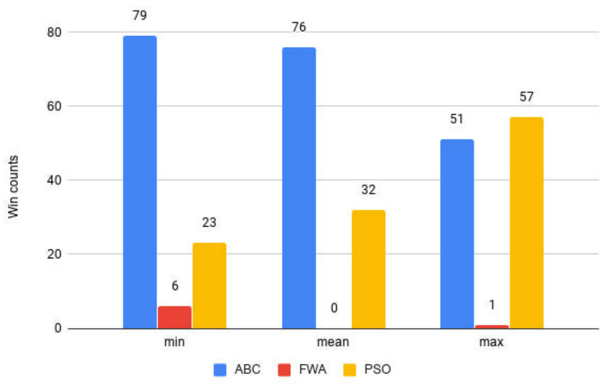


FIGURE 16. Score win counts for all test cases.

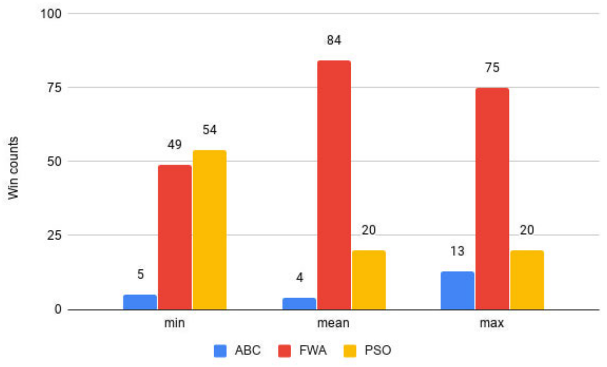
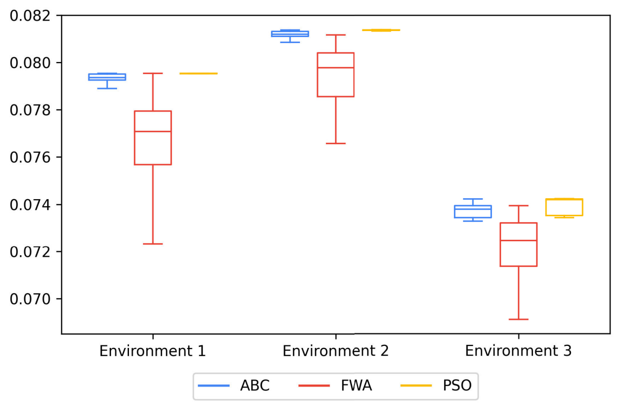


FIGURE 17. Time win counts for all test cases.

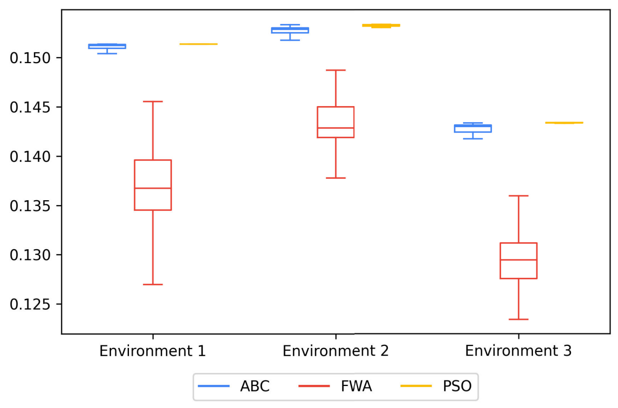
PSO algorithm stands in the middle, based on the number of wins both for the score and execution time.

To study the performance and reliability of the algorithms, they were compared over a few chosen test cases. As the focus of this research was on three-dimensional environments the results for three-dimensional space with the rasterization of $0.1m$ are shown in the form of the box plots.

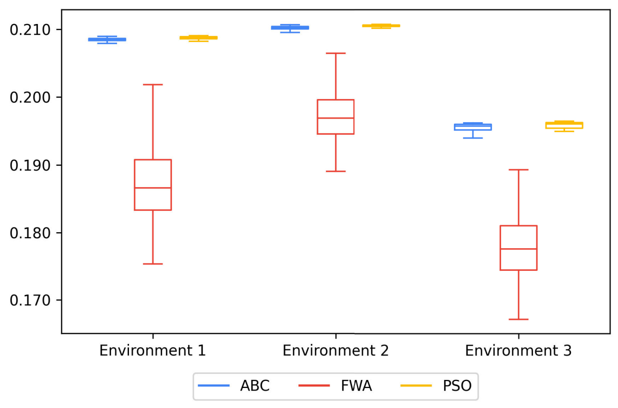
In each of the sub-figures shown in Figures 18 and 19 results for each of the algorithms for each of the environments are presented. In the Figure 18 results for the isotropic, Estimote Bluetooth Low Energy, sensor are shown while in



(a) 1 sensor

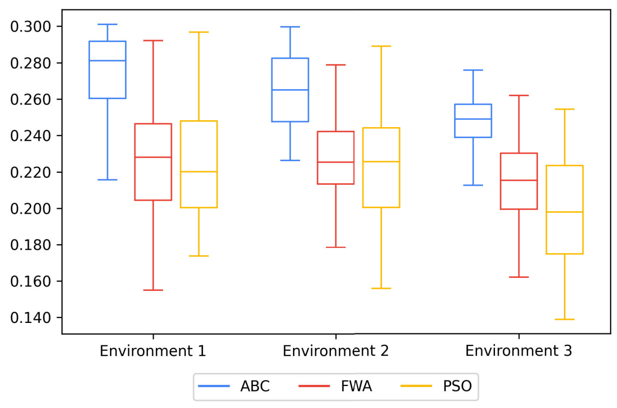


(b) 2 sensors

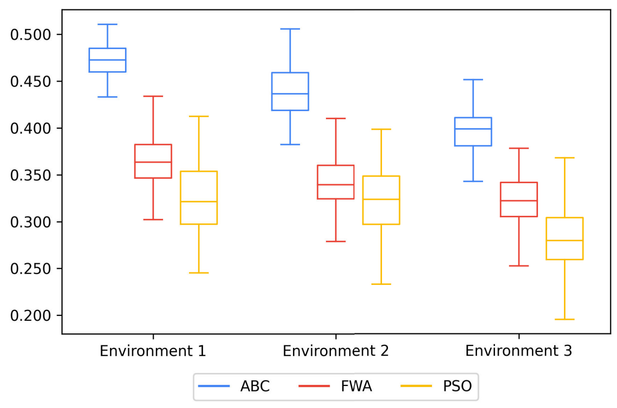


(c) 3 sensors

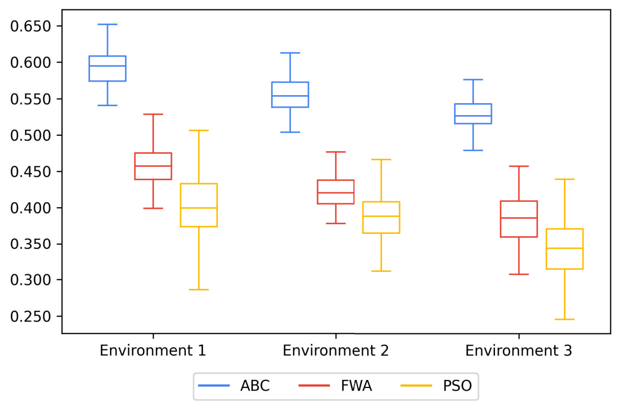
FIGURE 18. Results comparing algorithms placing estimote bluetooth low energy sensors for three-dimensional environments 1, 2, and 3.



(a) 1 sensor



(b) 2 sensors



(c) 3 sensors

FIGURE 19. Results comparing algorithms placing ZED stereo camera sensors for three-dimensional environments 1, 2, and 3.

the Figure 19 results for the directional, ZED Stereo Camera, sensor are shown. Each of the subfigures in both figures consists of results for the same number of sensors; for one sensor in the subfigures a), two sensors in the subfigures b), and for the three sensors in the subfigures c).

Results for these test cases are confirming the results of the score win counts. It is evident that the isotropic sensor results for the ABC and the PSO algorithms are comparable, while for the directional sensors results achieved by the PSO algorithm are significantly lower even compared to the FWA algorithm. The difference between the ABC and the PSO algorithm results for Bluetooth Low Energy and ZED Stereo Camera sensors is evident. This is probably due to the number of independent variables and will be explored in future work. The FWA algorithm is shown to be least reliable, showing the

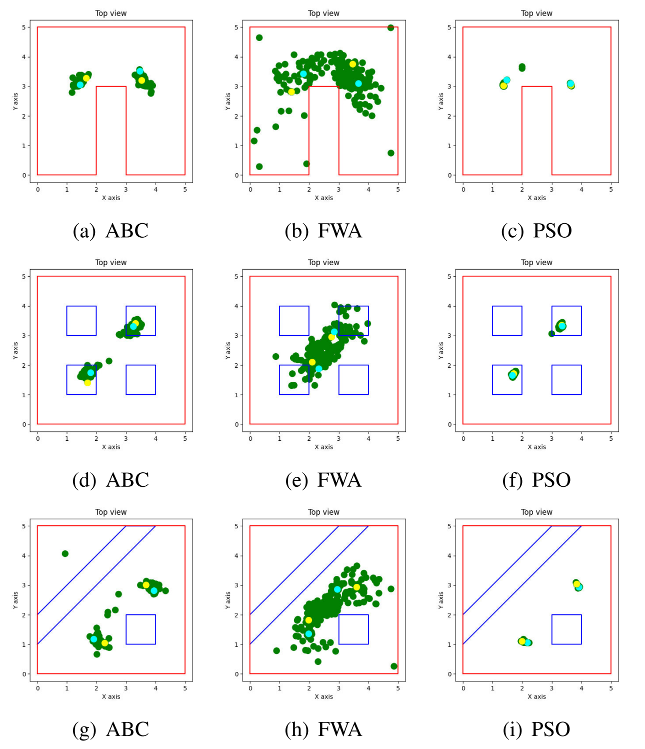


FIGURE 20. **Placement distributions for three example environments (rows) and algorithms (columns).**

high variance across all the iterations, along with significantly lower scores.

To investigate the distribution of sensors positions across all iterations of a specific test case in relation to the optimization algorithm used, three test cases are shown in Figure 20. For these test cases, all three environments were used, rasterization value was set to $0.1m$ and two Bluetooth Low Energy sensors were placed in three-dimensional space. For easier visualization, only the top view is shown. In the first row results for the Environment 1 are shown (sub-figures a), b), and c)), in the second row results for the Environment 2 (d), e) and f)) and in the third row results for the Environment 3 are shown (g), h), and i)), while in columns results for the ABC algorithm (a), d), and g)), the FWA algorithm (b), e), h)) and the PSO algorithm (c), f), i)) are shown, respectively.

From these results, it is noticeable that the FWA algorithm has the highest dispersion in positions, with many outliers, which contributes to our conclusion that the FWA algorithm is not among the best algorithms that were tested considering the score parameter. The ABC algorithm shows less dispersion in sensors positions and the PSO algorithm shows the least dispersion in sensor positions. This is consistent with the variance in the score of the algorithms shown in Figure 18 b), where the FWA algorithm has the highest variance, followed by the ABC and the PSO algorithms.

VII. CONCLUSION

In this manuscript, an effective method for addressing the problem of optimal placement of the isometric and directional sensors in an environment was proposed. The proposed solution incorporates environment models using both three-dimensional and two-dimensional representations as well as the sensor models with probabilistic coverage. Models for both environment and sensor are based on their real-world physical characteristics.

The proposed optimization function was combined with the proposed models to enable the maximization of the coverage of the environment by the sensors using the area coverage metric. The optimization function enables the use of any derivative-free, nonlinear, and constrained optimization algorithm.

A comparison of obtained coverage score and execution time for three optimization algorithms was performed. The ABC algorithm has shown to outperform its competitors, the FWA, and the PSO algorithms in the coverage score value. The FWA algorithm had the lowest execution times followed by the PSO and the ABC algorithms.

Times needed for the execution of three-dimensional test cases, especially ones with lower rasterization, proved to be notably higher than ones with a lower number of voxels. As shown in the previous section, some of the test cases started to take a few processing days compared to the mere seconds of the others, lower-dimensional ones.

Future research will include using different types of coverage, especially k-coverage, as well as adding different types of sensors. Further idea is to determine the optimal number of evaluations per independent variable, as well as finding the influence of the rasterization parameter on the optimization process. Finding values that provide a good ratio of the speed and the accuracy can speed the processing significantly with a low impact on the accuracy. Other optimization algorithms could be easily added and compared. One of the possibilities is to train an agent (sensor) model using reinforcement learning, which could lead to the possibility of finding optimal results within an unknown environment.

ACKNOWLEDGMENT

This research used supercomputer ''Bura'' from the Center for Advanced Computing and Modelling, University of Rijeka.

REFERENCES

[1] G. Zhang, B. Dong, and J. Zheng, ''Visual sensor placement and orientation optimization for surveillance systems,'' in *Proc. 10th Int. Conf. Broadband Wireless Comput., Commun. Appl. (BWCCA)*, Nov. 2015, pp. 1–5.

[2] G. Safak, *The Art-Gallery Problem: A Survey and an Extension*. Stockholm, Sweden: Skolan för datavetenskap och kommunikation, Kungliga Tekniska Högskolan, 2009.

[3] J. O'rourke, *Art Gallery Theorems and Algorithms*, vol. 57. Oxford, U.K.: Oxford Univ. Press, 1987.

[4] H. González-Banos, ''A randomized art-gallery algorithm for sensor placement,'' in *Proc. 17th Annu. Symp. Comput. Geometry (SCG)*, 2001, pp. 232–240.

[5] J. Marzal, ''The three-dimensional art gallery problem and its solutions,'' Ph.D. dissertation, School Inf. Technol., Murdoch Univ., Perth, WA, Australia, 2012.

[6] M. Marengoni, B. A. Draper, A. Hanson, and R. Sitaraman, ''A system to place observers on a polyhedral terrain in polynomial time,'' *Image Vis. Comput.*, vol. 18, no. 10, pp. 773–780, Jul. 2000.

[7] G. Viglietta, "Guarding and searching polyhedra," 2012, *arXiv:1211.2483*. [Online]. Available: http://arxiv.org/abs/1211.2483

[8] X. Li, W. Yu, X. Lin, and S. S. Iyengar, "On optimizing autonomous pipeline inspection," *IEEE Trans. Robot.*, vol. 28, no. 1, pp. 223–233, Feb. 2012, doi: 10.1109/TRO.2011.2169619.

[9] M. Cardei and J. Wu, "Energy-efficient coverage problems in wireless ad-hoc sensor networks," *Comput. Commun.*, vol. 29, no. 4, pp. 413–420, Feb. 2006.

[10] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava, "Coverage problems in wireless ad-hoc sensor networks," in *Proc. Conf. Comput. Commun., 20th Annu. Joint Conf. IEEE Comput. Commun. Soc. (IEEE INFOCOM)*, vol. 3, Apr. 2001, pp. 1380–1387.

[11] S. M. A. Salehizadeh, A. Dirafzoon, M. B. Menhaj, and A. Afshar, "Coverage in wireless sensor networks based on individual particle optimization," in *Proc. Int. Conf. Netw., Sens. Control (ICNSC)*, Apr. 2010, pp. 501–506.

[12] H. T. T. Binh, N. T. Hanh, L. Van Quan, and N. Dey, "Improved cuckoo search and chaotic flower pollination optimization algorithm for maximizing area coverage in wireless sensor networks," *Neural Comput. Appl.*, vol. 30, no. 7, pp. 2305–2317, Oct. 2018.

[13] F. Hržić, D. Sušanj, and K. Lenac, "Optimal beacon positioning for indoor drone navigation," in *Proc. 12th Annu. Baška GNSS Conf.*, 2018, pp. 109–117.

[14] M. Krishnan, V. Rajagopal, and S. Rathinasamy, "Performance evaluation of sensor deployment using optimization techniques and scheduling approach for K-coverage in WSNs," *Wireless Netw.*, vol. 24, no. 3, pp. 683–693, Apr. 2018.

[15] V. Akbarzadeh, A. H.-R. Ko, C. Gagné, and M. Parizeau, "Topography-aware sensor deployment optimization with CMA-ES," in *Proc. Int. Conf. Parallel Problem Solving Nature*. Berlin, Germany: Springer, 2010, pp. 141–150.

[16] S. S. Dhillon and K. Chakrabarty, "Sensor placement for effective coverage and surveillance in distributed sensor networks," in *Proc. IEEE Wireless Commun. Netw. (WCNC)*, vol. 3, Mar. 2003, pp. 1609–1614.

[17] E. Hörster and R. Lienhart, "On the optimal placement of multiple visual sensors," in *Proc. 4th ACM Int. Workshop Video Surveill. Sensor Netw. (VSSN)*, New York, NY, USA, 2006, p. 111, doi: 10.1145/1178782.1178800.

[18] R. Bodor, A. Drenner, P. Schrater, and N. Papanikolopoulos, "Optimal camera placement for automated surveillance tasks," *J. Intell. Robotic Syst.*, vol. 50, no. 3, pp. 257–295, Oct. 2007.

[19] J.-J. Gonzalez-Barbosa, T. Garcia-Ramirez, J. Salas, J.-B. Hurtado-Ramos, and J.-D.-J. Rico-Jimenez, "Optimal camera placement for total coverage," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2009, pp. 844–848.

[20] A. A. Altahir, V. S. Asirvadam, N. H. B. Hamid, P. Sebastian, N. B. Saad, R. B. Ibrahim, and S. C. Dass, "Optimizing visual sensor coverage overlaps for multiview surveillance systems," *IEEE Sensors J.*, vol. 18, no. 11, pp. 4544–4552, Jun. 2018.

[21] V. Akbarzadeh, J.-C. Lévesque, C. Gagné, and M. Parizeau, "Efficient sensor placement optimization using gradient descent and probabilistic coverage," *Sensors*, vol. 14, no. 8, pp. 15525–15552, Aug. 2014.

[22] Y. Morsly, N. Aouf, M. S. Djouadi, and M. Richardson, "Particle swarm optimization inspired probability algorithm for optimal camera network placement," *IEEE Sensors J.*, vol. 12, no. 5, pp. 1402–1412, May 2012.

[23] C.-F. Huang and Y.-C. Tseng, "The coverage problem in a wireless sensor network," *Mobile Netw. Appl.*, vol. 10, no. 4, pp. 519–528, 2005.

[24] A. A. Altahir, V. S. Asirvadam, N. H. Hamid, P. Sebastian, N. Saad, R. Ibrahim, and S. C. Dass, "Modeling multicamera coverage for placement optimization," *IEEE Sensors Lett.*, vol. 1, no. 6, pp. 1–4, Dec. 2017.

[25] M. Hefeeda and H. Ahmadi, "Energy-efficient protocol for deterministic and probabilistic coverage in sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, no. 5, pp. 579–593, May 2010.

[26] V. Akbarzadeh, C. Gagne, M. Parizeau, and M. A. Mostafavi, "Black-box optimization of sensor placement with elevation maps and probabilistic sensing models," in *Proc. IEEE Int. Symp. Robotic Sensors Environ. (ROSE)*, Sep. 2011, pp. 89–94.

[27] V. Akbarzadeh, C. Gagne, M. Parizeau, M. Argany, and M. A. Mostafavi, "Probabilistic sensing model for sensor placement optimization based on Line-of-Sight coverage," *IEEE Trans. Instrum. Meas.*, vol. 62, no. 2, pp. 293–303, Feb. 2013.

[28] M. Thiene, Z. S. Khodaei, and M. H. Aliabadi, "Optimal sensor placement for maximum area coverage (MAC) for damage localization in composite structures," *Smart Mater. Struct.*, vol. 25, no. 9, Sep. 2016, Art. no. 095037.

[29] D. V. Papadimitriou and T. J. Dennis, "Epipolar line estimation and rectification for stereo image pairs," *IEEE Trans. Image Process.*, vol. 5, no. 4, pp. 672–676, Apr. 1996.

[30] G. Medioni and R. Nevatia, "Segment-based stereo matching," *Comput. Vis., Graph., Image Process.*, vol. 31, no. 1, pp. 2–18, Jul. 1985.

[31] *Estimote Web Page*. Accessed: Sep. 8, 2020. [Online]. Available: https://estimote.com/

[32] *StereoLabs ZED Web Page*. Accessed: Sep. 8, 2020. [Online]. Available: https://www.stereolabs.com/zed/

[33] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm," *J. Global Optim.*, vol. 39, no. 3, pp. 459–471, Oct. 2007.

[34] Y. Tan and Y. Zhu, "Fireworks algorithm for optimization," in *Proc. Int. Conf. Swarm Intell.* Berlin, Germany: Springer, 2010, pp. 355–364.

[35] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE ICNN*, vol. 4, Nov./Dec. 1995, pp. 1942–1948.

[36] Y. Tan, *Handbook of Research on Design, Control, and Modeling of Swarm Robotics*. Hershey, PA, USA: IGI Global, 2015.

[37] *Supercomputer, Bura, University of Rijeka, Center for Advanced Computing and Modelling*. Accessed: Sep. 8, 2020. [Online]. Available: https://cnrm.uniri.hr/bura/

DIEGO SUŠANJ (Member, IEEE) received the B.S. and M.S. degrees in computer engineering from the Faculty of Engineering, University of Rijeka, in 2013 and 2015, respectively, where he is currently pursuing the Ph.D. degree in computer science.

He is currently a Teaching Assistant with the Faculty of Engineering, University of Rijeka. His research interests include computer vision, machine learning, image processing, and embedded systems.

Mr. Sušanj is a member of the IEEE Computer Society. He is currently serving as a Croatia Section Student Representative, the Vice-Chair of the CS Student and Young Professional Activities Committee, and the University of Rijeka Student Branch Counselor. His work was recognized by CS as he received the Upsilon Pi Epsilon Award in 2012 and the Richard E. Merwin Scholarship in 2014. He is one of the founders of the Riteh Drone Team and a member of the Royal Institute of Navigation.

DOMAGOJ PINČIĆ received the B.S. and M.S. degrees in computer engineering from the Faculty of Engineering, University of Rijeka, in 2015 and 2018, respectively, where he is currently pursuing the Ph.D. degree in computer science.

He is currently a Research Assistant with the Faculty of Engineering, University of Rijeka. His research interests include computer vision, machine learning, and image processing.

KRISTIJAN LENAC (Member, IEEE) received the Ph.D. degree from the University of Trieste, Italy, with a dissertation in the field of mobile robotics.

He is currently an Associate Professor with the Faculty of Engineering, University of Rijeka. He then worked for five years as the Project Manager and a Lead Developer with AIBSLab Engineering Company from Trieste leading several research and development projects commissioned by the industry. On his return to Croatia, he joined the Faculty of Engineering, University of Rijeka, as a Professor with the Computer Engineering Department. He is the Founder and the current Head of the Artificial Perception and Autonomous Systems Laboratory and one of the founders of the Riteh Drone Team. At the Centre for Artificial Intelligence and Cybersecurity (AIRI), he is the Founder and the current Head of the Laboratory for Applications of Blockchain Technology. His research interests include mobile robotics, satellite navigation, embedded systems, and blockchain. He is a member of the Royal Institute of Navigation.