Visual Coverage Using Autonomous Mobile Robots for Search and Rescue Applications

A. Del Bue[1], M. Tamassia[1], F. Signorini[2], V. Murino[1] and A. Farinelli[2]

***Abstract*—This paper focuses on visual sensing of 3D large-scale environments. Specifically, we consider a setting where a group of robots equipped with a camera must fully cover a surrounding area. To address this problem we propose a novel descriptor for visual coverage that aims at measuring visual information of an area based on a regular discretization of the environment in voxels. Moreover, we propose an autonomous cooperative exploration approach which controls the robot movements so to maximize information accuracy (defined based on our visual coverage descriptor) and minimizing movement costs. Finally, we define a simulation scenario based on real visual data and on widely used robotic tools (such as ROS and Stage) to empirically evaluate our approach. Experimental results show that the proposed method outperforms a baseline random approach and an uncoordinated one, thus being a valid solution for visual coverage in large scale outdoor scenarios.**

I. INTRODUCTION

In recent years mobile robots emerged as a crucial technology for applications such as rescue operations in dangerous or hostile environments or for surveillance of safety critical areas. To date, much of the work related to this field focus either on building accurate maps of the environment by using sensors that provide dense measurements (such as laser range finder) [1], [2] or on methodologies for autonomous exploration [3], [4], [5], where robots must be able to plan their movements requiring only limited interactions with the human operator. In such regard, an interesting problem related to autonomous exploration is that of search, where robots explore the environment with the end goal of detecting possible victims that may require assistance[6], [7], [8].

Here we take a different perspectives on the problem and we focus on visual coverage of outdoor, large scale environments. Our aim is then to provide accurate 3D coverage of an unknown area by using a team of mobile robots equipped only with cameras. In particular, we focus on cameras mainly because they are very well suited for large scale environments, as their field of view typically span hundreds of meters. In such regard they might easily cover large extent of the surrounding environment with a relatively small cost. Moreover, cameras can provide crucial information for rescue operators to perform risk and damage assessment, or to detect possible intruders in areas with restricted access.

Now, visual coverage based on cameras has been addressed from different perspective in several fields such as sensors network, computer vision and robotics. The recent comprehensive review of Mavrinac and Chen [9] reveals that, even if the camera coverage problem is an active field, many problems are still unsolved. This is clearly given by the fact that the coverage problem in 3D is ill-posed for several reasons: lack of a priori information about the 3D structure of the scene, possible occluders and misaligned readings in multi-sensor/multi-robot configurations. Moreover visual sensors often present different "dis-functions" compared to standard range sensors such as laser range finders. For instance, since 3D reconstruction is typically based on the initial detection and matching of feature points, the camera might be too far and the resolution too small in order to extract visual features. Similarly the camera might be too close to the object (focus problem). Finally, cameras have also a Field of View (FoV) which restrict the number of points visible in the scene.

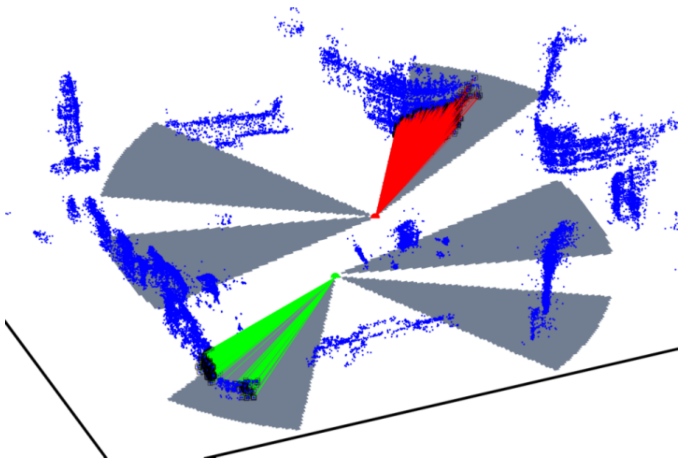


Fig. 1. The figure shows two robots (red and green) moving in a large-scale outdoor 3D map. The grey areas represent the coverage of the past and present moves while the black rays show the visible 3D points from each robot. The map has been obtained by processing images acquired in Trafalgar Square, London, UK [10].

Given this scenario, here we consider a set of mobile robots that move in an outdoor environment and our goal is to drive the robots so to maximize the coverage of the inspected area. Figure, 1 provides an overview of our application scenario, where two robots navigate in a large 3D environment and acquire visual observations. The 3D scene is built by using data from [11], [10] which are obtained by performing large scale 3D reconstruction on a collection of images[1].

In more details, we propose a measure of 3D coverage based on a regular discretization of the environment in voxels, and we devise an automatic approach to decide which is the best position that each robot should reach to maximize information accuracy while minimizing movement costs.

[1]A. Del Bue, M. Tamassia and V. Murino are with the Pattern Analysis and Computer Vision Department (PAVIS), Istituto Italiano di Tecnologia (IIT), Genova, Italy.

[2]F. Signorini and A. Farinelli are with the Department of Computer Science, University of Verona, Italy.

[1]The 3D data used in this paper is available online at http://grail.cs.washington.edu/projects/bal/ and http://www.diegm.uniud.it/fusiello/demo/samantha/

Specifically, this paper makes three main contributions to the state of the art: First, we propose a novel coverage metric for 3D environments. Such metric takes into account the average penetration of the camera bundle of rays towards the 3D points which fall within each voxel, where a voxel is the discretization unit of the map. Second, we provide a cooperative strategy to drive a team of robots in the environment so to maximize the coverage metric described above while minimizing movement costs. Exploration is formalized as an optimization problem and our strategy is a greedy strategy that guides the robots in the map towards positions that are most promising for information acquisition. Robots share their observations and operate on the basis of a global visual coverage map that fuses such observations. Third, we empirically evaluate our approach by using ROS (for the robot navigation stack) and Stage for simulating 2D navigation of the robots. The 3D observations are obtained by a custom simulation environment that provides to the robot the observed features.

Our empirical evaluation validates the approach in three large-scale datasets (Trafalgar Square, Piazza San Marco and Piazza Bra) [11], [10]. Moreover, we compare our approach to a baseline randomized method (where the next position is randomly chosen) and to an uncoordinated approach (where each robot chooses the next position maximising its own coverage map without sharing any observations with its team mates). The results allow us to draw encouraging conclusions on the effectiveness and applicability of our method.

The rest of the paper is organized as follows. Section II presents the related works that deal with similar coverage problems. Section III describes our coverage descriptor and Section IV our approach to single and multi-robot coverage. Section V describes our empirical settings and achieved results. Finally Section VI presents conclusions and discussions for future work.

II. Related work

In recent years, there has been a growing interest towards autonomous robotic systems that can accomplish sensing and surveying tasks in their surrounding environments. Application scenarios of such autonomous systems range from surveillance and security [12], [1], to environmental surveying [13]. In particular, a large body of work focuses on reconstructing the 3D structure of the environment which is a key information for several tasks such as victim detection and localization in search and rescue [8], [6], [7]. In this scenario, most previous approaches focus on the use of dense sensors that can provide accurate information from the environments such as 2D or 3D laser range finders [14], [15], [16], or more recently the Kinect system [6]. Much of this work is based on exploration strategies for single and multi-robot systems that aim at maximising the information that each robot can acquire given the next possible moves. In particular, the idea of frontier based exploration, originally proposed by Yamauchi [5], is a widely used approach to address autonomous exploration and information gathering problems. For example, Burgard et al. [2] propose a multi-robot exploration approach where robots cooperatively choose next sensing positions by considering both the utility (in terms of information) of frontier points as well as the cost that robot would incur to reach such position. In a similar way, we also base our approach on a utility function that consider both possible information acquired by robots and movement costs. However, in contrast to such approach, we do not use a frontier based method to allocate utility to points. This is because our goal is not only to explore the whole environment but to provide an accurate visual coverage of the surroundings (i.e., to maximize our coverage descriptor). In this perspective, the work of Stachniss and Burgard [17] proposes an autonomous approach for exploration that considers coverage maps: an extension of occupancy maps that maintain occupancy probability for each map cell. Based on such representation, authors propose a decision-theoretic approach for autonomous exploration based on the concept of information gain. With respect to this method, we consider a different concept of coverage, as we are interested in *visual* coverage which measures the number of 3D visual features observed by the robots rather than a probabilistic measure of occupancy.

Another important strand of works focuses on 3D digitalization of objects or environments and proposes the use of dense sensors such as 2D/3D laser or the Kinect systems. For example, Whaite and Ferrie in [14] propose an exploration approach for a robotic arm equipped with a 2D laser scanner to reconstruct the structure of 3D objects. Surmann and colleagues in [15] propose an approach to determine the next best view of a mobile platform for digitalization of 3D indoor environments using a 3D laser scanner. Finally, Dornhege and Kleiner in [6] propose a frontier-like exploration strategy for a 3D environment based on the Kinect system, focusing on unstructured scenarios (typical of rescue applications). With respect to previous approaches, here we focus on visual coverage, hence explicitly restricting our attention to cameras. Moreover, our goal is not to provide an accurate digitalization of the environment but to capture as many visual information as possible. Consequently, a crucial point for our approach is to propose a *visual* coverage descriptor which is then used by our autonomous exploration strategy. Finally, unlike most previous approaches we focus on large-scale outdoor environments.

III. The Visual Coverage Descriptor

The two main elements of our approach are the definition of suitable metrics for visual coverage and the definition of a *utility* function that controls robots' movements considering both the *reward* in terms of increased degree of visual coverage and the *cost* for movements.

In this Section, we start by proposing a visual coverage descriptor that can measure coverage for each voxel of the 3D map. This 3D map is given by large scale 3D reconstruction algorithms using multiple images [10], [11]. In our simulation scenario, the robot starts from an initial position with no a-priori information on the map and by moving in a new position it checks if a set of points is visible given a visibility model (Sec. III-B). At each movement new points become visible given the robot location and orientation in the map. In such context, our general idea for the descriptor is that a certain 3D volume is covered if it is possible to *view through it*, hence the measure of coverage is related to how much of the voxel volume is "penetrated" by the bundle of rays towards the 3D points which fall inside the camera field of view.

In general a 3D point is considered visible if the point is inside the camera field of view, the range distance, and it is not too close to the camera (focus effect). These criteria are the one used in the most complex modelling scenarios for sensor camera networks [9]. If a 3D point satisfies the field of view, camera range and focus constraints, it is considered as observed in our model. In what follows, we first describe our visibility model and then propose our coverage descriptor.

A. The camera visibility model

In order to formally describe the concept of coverage with visual sensors we first need to define our mathematical and geometric settings. We adopt the *General Camera Model* [18], [19] which defines the imaging model as rays travelling in a straight line, as this is a convenient formalisation for modelling coverage using ray bundles. We also assume that the robot position $\mathbf{t}_n$ in the world coordinates coincides with the camera optical center. The 3D position of a generic 2D image point l onto the camera plane is defined in the camera coordinate system as:

$$\mathbf{o}_{nl} = \mathtt{R}^T \mathtt{K}^{-1} \begin{bmatrix} \mathbf{w}_{nl} \\ 1 \end{bmatrix}, \tag{1}$$

where $\mathtt{R}$ is the camera rotation, $\mathtt{K}$ is the 3×3 intrinsic camera parameter matrix and $\mathbf{w}_{nl}$ contains the 2D point coordinates. Given the robot orientation, we align $\mathbf{o}_{nl}$ in the world coordinate system such that:

$$\boldsymbol{\eta}_{nl} = [\mathtt{R}_{yaw}(\varphi) \mid \mathbf{t}_n] \begin{bmatrix} \mathbf{o}_{nl} \\ 1 \end{bmatrix},$$

where $\mathtt{R}_{yaw}$ is the orientation of robot as a rotation on the z-world axis and $\mathbf{t}_n$ is the camera translation with respect to the world origin[2]. Now it is possible to derive the direction of the ray projected from the camera center and passing through a generic 2D point on the image plane as the unit-vector:

$$\mathbf{r}_{nl} = \frac{\mathbf{t}_n - \boldsymbol{\eta}_{nl}}{|\mathbf{t}_n - \boldsymbol{\eta}_{nl}|}. \tag{2}$$

Thus we can now define the line crossing the 3D space starting from the camera as $\mathbf{r}_{nl}$.

Given our camera model, we need to define a set of criteria to compute the visibility descriptor for each camera position and orientation. As mentioned before, we consider the field of view, resolution and focus.

Field of View. We model the field of view by first defining the angle between the ray $\mathbf{r}_{nl}$ and the ray departing from the camera center and intersecting the principal point $\mathbf{q}$ given by $\mathbf{r}_{nq}$. Such angle for each 3D point l is defined as:

$$\boldsymbol{\phi}(l) = \arccos(\mathbf{r}_{nq} \cdot \mathbf{r}_{nl}). \tag{3}$$

Now given a robot orientation φ, the set $\Lambda^{fov}_{\varphi} \subset \Lambda$ of the 3D reconstructed points that are visible to the camera, can be defined as:

$$\Lambda^{fov}_{\varphi} = \{\mathbf{x}_l \in \Lambda \mid \left(0 < \boldsymbol{\phi}(l) < \frac{fov}{2}\right),\ l \in \{1, \ldots, |\Lambda|\}, \tag{4}$$

where *fov* is the *field of view* of the camera. This value was set to 30° in the experiments but it might be customised given the specific camera model.

[2]Notice that here we constrain the robot to move on a planar surface. However our approach can be easily generalized for unconstrained 3D motion.

Resolution. Camera resolution is closely related to the concept of range in classical laser systems. In practice, to reconstruct a 3D point from image observations, it is necessary to match 2D feature points from different views. If a feature is too far from the camera center it might not have enough image support to detect the point in the image and subsequently to match it in other views. In most systems, this effect is modelled as a distance constraint that limits the visibility of faraway points [20], [21]. Thus we can define such distance from the current camera position $\mathbf{t}_n$ to the l-th 3D point $\mathbf{x}_l$ as:

$$\epsilon_{ln} = |\mathbf{x}_l - \mathbf{t}_n|.$$

The points $\Lambda^{res}_{\varphi} \subset \Lambda$ that have enough resolution to be detected can be defined as:

$$\Lambda^{res}_{\mathbf{t}} = \{\mathbf{x}_l \in \Lambda \mid \epsilon_{ln} < \delta_{max}\},\ l \in \{1, \ldots, |\Lambda|\}. \tag{5}$$

where δ_{max} is the the maximum range. An analysis of image feature detectors recall with respect to resolution can be found in [22] and it can be used as a guideline for setting the parameter δ_{max}.

Focus. Likewise, the scene has to be imaged at the proper focus in order to avoid misdetection of the 2D image features. In practice this constraint rules out elements in the scene that are too close to the camera. This can be implemented as a minimum range constraint [23] such that:

$$\Lambda^{foc}_{\mathbf{t}} = \{\mathbf{x}_l \in \Lambda \mid \epsilon_{ln} > \delta_{min}\},\ l \in \{1, \ldots, |\Lambda|\}. \tag{6}$$

These three constraints as defined in Eq. (4), (5), (6) gives the visibility of a 3D point given a certain camera position and orientation such that:

$$\Lambda^{vis} = \Lambda^{fov}_{\varphi} \bigcap \Lambda^{res}_{\mathbf{t}} \bigcap \Lambda^{foc}_{\mathbf{t}}. \tag{7}$$

B. Computing the coverage descriptor

Having defined the visibility model for the 3D points we can now define the descriptor for each voxel $\mathbf{v}_k$ in the map. First, let us consider the set $\Lambda_{\mathbf{v}_k}$ of the *visible* 3D points for each voxel such that:

$$\Lambda_{\mathbf{v}_k} = \{\mathbf{x}_l \in \Lambda^{vis} \mid \mathbf{x}_l \in \mu_{\mathbf{v}_k}\},$$

where $\mu_{\mathbf{v}_k}$ represents the boundaries of the voxel $\mathbf{v}_k$.

The intersection of the line projection from the camera center to each reconstructed 3D point $\mathbf{x}_l \in \Lambda_{\mathbf{v}_k}$ localises the point $\mathbf{x}_{k_l}$ on the voxel $\mathbf{v}_k$'s boundaries.

To give our definition of *coverage*, it is necessary to include also the information regarding how much the intersecting ray "penetrates" each voxel. We define such length as the Euclidean distance between point $\mathbf{x}_{k_l}$ and $\mathbf{x}_l$ normalized by the maximum length d_{k_l} of the ray intersecting the voxel, i.e. $p_{k_l} = |\mathbf{x}_{k_l} - \mathbf{x}_l| / d_{k_l}$. This creates a vector $\mathbf{d}_k = [p_{k_1}, \ldots, p_{k_{|\Lambda_{\mathbf{v}_k}|}}]^T$ measuring the penetration of the bundle rays inside the voxel.

We now define the *coverage* metric for each voxel $\mathbf{v}_k \in \mathcal{V}$ as the average penetration of the camera bundle of rays towards

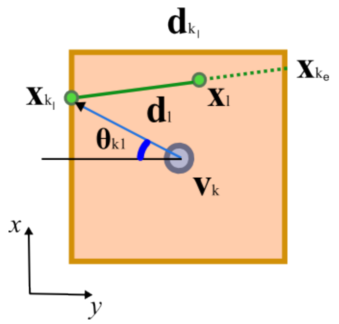


Fig. 2. The image shows an x, y view of the metric 3D space with the length of the bundle ray going through the voxel $\mathbf{v}_k$. $\mathbf{x}_{k_l}$ and $\mathbf{x}_{k_e}$ are the entry and exit points respectively, with an overall lenght d_{k_l}. $\mathbf{x}_l$ is the reconstructed 3D point and the distance $d_l = |\mathbf{x}_{k_l} - \mathbf{x}_l|$ represents how much the ray penetrates the voxel.

the 3D points $\mathbf{x}_l \in \Lambda_{\mathbf{v}_k}$ which fall within its boundaries. The coverage descriptor is given by:

$$\psi_{\mathbf{v}_k} := \begin{cases} \left(\dfrac{\sum_{l=1}^{|\Lambda_{\mathbf{v}_k}|} \frac{|\mathbf{x}_{k_l} - \mathbf{x}_l|}{d_{k_l}}}{|\Lambda_{\mathbf{v}_k}|} \right), & \text{if } |\Lambda_{\mathbf{v}_k}| > 0, \\ 0, & \text{if } |\Lambda_{\mathbf{v}_k}| = 0. \end{cases} \tag{8}$$

Thus, each time the camera is located in a position on the map from which it has to (re)-compute the coverage (say at the position $\mathbf{t}$ assuming orientation φ), the algorithm computes a vector $\boldsymbol{\xi}_{\mathbf{t}}^{\varphi}$ as:

$$\boldsymbol{\xi}_{\mathbf{t}}^{\varphi} = \begin{bmatrix} \psi_{\mathbf{v}_1} \\ \vdots \\ \psi_{\mathbf{v}_{|\mathcal{V}|}} \end{bmatrix},$$

which stores for each voxel the average degree of coverage.

Since the camera is moving in the map, we need to keep track of all the voxels that has been seen and to update the coverage values. We assume that the camera stops in D different positions $\mathbf{p}_i = [\mathbf{t}_i, \varphi_i]$ of the map, to re-calculate the coverage and we indicate with $\mathbf{P} = \{\mathbf{p}_1, \cdots, \mathbf{p}_D\}$ the vector of all such positions. We can now define:

$$\mathcal{V}_{\text{full}}^{\mathbf{p}_i} = \{\mathbf{v}_k \in \mathcal{V} \mid \psi_{\mathbf{v}_k}^{\mathbf{p}_i} > 0\}$$
$$\mathcal{V}_{\text{empty}}^{\mathbf{p}_i} = \{\mathbf{v}_k \in \mathcal{V} \mid \psi_{\mathbf{v}_k}^{\mathbf{p}_i} = 0\},$$

where $\psi_{\mathbf{v}_k}^{\mathbf{p}_i}$ represents the value of our coverage descriptor for voxel $\mathbf{v}_k$ (see Equation (8)) when observed from position $\mathbf{p}_i$. Moreover, we keep track of all the full and empty voxels seen from $\mathbf{p}_1$ to $\mathbf{p}_D$ in the map by taking the union of the voxels observed in each position:

$$\mathcal{F}_{\mathbf{P}} = \bigcup_{\mathbf{p}_i \in P} \mathcal{V}_{\text{full}}^{\mathbf{p}_i},$$
$$\mathcal{E}_{\mathbf{P}} = \bigcup_{\mathbf{p}_i \in P} \mathcal{V}_{\text{empty}}^{\mathbf{p}_i}.$$

Finally, we define $\psi_{\mathbf{v}_k}^{\mathbf{P}}$ as the coverage value for $\mathbf{v}_k$ calculated given the vector $\mathbf{P}$ of camera positions. In particular, for each voxel, we store the average coverage value calculated so far considering only the positions from which we have a positive value (i.e., we consider only the camera positions from which the voxel is visible):

$$\psi_{\mathbf{v}_k}^{\mathbf{P}} = \begin{cases} \dfrac{\sum_{i \in P_k^+} \psi_{\mathbf{v}_k}^{\mathbf{p}_i}}{|P_k^+|}, & \text{if } \mathbf{v}_k \in \mathcal{F}_{\mathbf{P}} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

where $P_k^+ = \{\mathbf{p}_j \in \mathbf{P} : \psi_{\mathbf{v}_k}^{\mathbf{p}_j} > 0\}$ represents the vector of position from which we have positive observations of voxel $\mathbf{v}_k$ (i.e., observations that give a positive value for the descriptor).

IV. Coverage Approach

Having detailed our coverage descriptor, we now describe our approach to drive the robots so to maximize coverage and minimize movement costs. The basic idea is to devise a utility function that, based on the level of *visual coverage* for all the voxels that we observed so far, drives the robot towards points in the map that can have useful visual information (i.e., that avoids voxels that have a high level of coverage). In the following, we first define the utility function and how we select the next position that the robot should visit based on such function, then we detail our approach for Multi-Robot coverage

A. Single robot utility function

Our *utility* function aims at estimating the reward in terms of *visual coverage* that the mobile robot would have for a specific position assuming a particular orientation. Our goal is then to find the next robot position that maximizes such reward. Notice that, since we do not know the visual features in the environment, but only what we already observed, we must estimate the value of the reward for a future position given the known data only.

Now, given the complexity of the 3D structure for our reference applications, it is not easy to capture such reward in a closed form solution. Hence we proceed by computing an estimation of the gain for each possible future position of the robot. However, for computational reasons, we restrict this calculation to a square *local grid* of n voxel per side, constructed around the current robot location. Moreover, for each such position we consider only the set Θ of the eight principal angles as possible orientations for the robot. Here we define the camera position and orientation with the vector $\mathbf{t}_i$ and $\varphi_i \in \Theta$ respectively. Then the vector $\mathbf{n}_i = [\mathbf{t}_i, \varphi_i]$ represents the generic next possible position for the robot.

The estimated coverage value associated to $\mathbf{n}_i$ is given by the value $LimCov$. Notice that this value for a fully covered voxel corresponds to $LimCov = 0.5$. In fact for each visual point that is inside a voxel, if we observed that point from all the possible directions, we would obtain an average ray penetration of 0.5. This can be briefly explained if we call $0 \leq f \leq 1$ the ray penetration ratio for a given point inside a voxel from a given observation point. Now, if we consider a second observation point which is opposite to the first one with respect to the point itself we can easily verify that the ray penetration ratio is now $1 - f$. For this reason a fully covered voxel will have an average value of $\frac{f+(1-f)}{2} = 0.5$. We then assume that a new observation for any full (i.e., $\mathcal{F}_{\mathbf{P}}$) or unseen voxel (i.e., $\mathcal{V} \setminus (\mathcal{F}_{\mathbf{P}} \cup \mathcal{E}_{\mathbf{P}})$) will give a value of 0.5, while observations for empty voxels (i.e., $\mathcal{E}_{\mathbf{P}}$), will provide no further information.

Next, we define the set $\mathcal{V}^{\mathbf{n}_i} \subseteq \mathcal{V}$ of voxels that are visible from position $\mathbf{n}_i$. In more detail, we have:

$$\mathcal{V}^{\mathbf{n}_i} = \{\mathbf{v}_k \in \mathcal{V} | \left(0 < \phi(k) < \frac{fov}{2}\right) \wedge (\delta_{min} < \epsilon_{ik} < \delta_{max})\},$$

where, as in Equations (4) (5) (6), we implement the FoV, resolution and focus constraints using the next position $\mathbf{t}_i$ instead of the current camera position and the center of the k-th voxel $\mathbf{v}_k \in \mathcal{V}$. We can now define $\Psi^{\mathbf{P}}_{\mathbf{n}_i}$ as the vector of size $|\mathcal{V}^{\mathbf{n}_i}|$ which stores our estimated coverage values for all the voxels in $\mathcal{V}^{\mathbf{n}_i}$ as follows:

$$\Psi^{\mathbf{P}}_{\mathbf{n}_i} = \frac{\left|P^+_k\right| \psi^{\mathbf{P}}_{\mathbf{v}_k} + \hat{\psi}_{\mathbf{v}_k}}{\left|P^+_k\right| + 1}, \tag{10}$$

where $\psi^{\mathbf{P}}_{\mathbf{v}_k}$ has been defined by Equation (9). The value $\hat{\psi}_{\mathbf{v}_k}$ is the expected value of a new observation for voxel $\mathbf{v}_k$ and is given by:

$$\hat{\psi}_{\mathbf{v}_k} := \begin{cases} 0 & \text{if } \mathbf{v}_k \in \mathcal{E}_{\mathbf{P}} \\ LimCov & \text{otherwise.} \end{cases} \tag{11}$$

We can now compute the reward function associated to $\mathbf{n}_i$ as follows:

$$rew^P(\mathbf{n}_i) = \sum_{k \in \mathcal{V}^{\mathbf{n}_i}} \delta_{\mathbf{P}}(k), \tag{12}$$

and

$$\delta_{\mathbf{P}}(k) = |\Psi^{\mathbf{P}}_{\mathbf{n}_i} - \psi^{\mathbf{P}}_{\mathbf{v}_k}| = \frac{|\hat{\psi}_{\mathbf{v}_k} - \psi^{\mathbf{P}}_{\mathbf{v}_k}|}{|P^+_k| + 1}. \tag{13}$$

The intuitive explanation is that we want to focus on positions of the local grid that avoid observing again voxels that are already well covered or empty, while at the same time we want to explore new voxels. Specifically, the definition of $\delta^{\mathbf{n}_i}_{\mathbf{P}}(k)$ aims at measuring the difference that a new observation would make to the level of coverage of a voxel, and the final equation can be easily derived by substituting Eq. (10) in (13). Such term shows that a new observation is more important if the average coverage value of the voxel is far from the expected value of such new observation and if the voxel has been observed few times (i.e., $|P^+|$ is small). Moreover, for unobserved voxels this term reduces to $LimCov$ which correctly shows that we give high expected gain when exploring new voxels. Finally, the gain correctly reduces to 0 for the empty voxels that have been already observed, as we have no interest in gaining new observations for such voxels

Figure 3 reports a visual representation of $rew^P(\mathbf{n}_i)$ computed given a specific robot position $\mathbf{t}_i$ and considering a constant φ_i. Figure 3 super-imposes a color coded representation of $rew^P(\mathbf{n}_i)$ on a 3D portion of the map where colors represents reward values (dark is low and light is high).

To have an effective exploration, we must also consider the movement costs for the robot and favour exploration policies that reduce such costs. Hence we define a cost function $moveCost$ that expresses the cost incurred by the robot to reach the next position. In more detail, $moveCost(\mathbf{n}_c, \mathbf{n}_i)$ estimates the travel distance from $\mathbf{n}_c = (\mathbf{t}_c, \varphi_c)$ to $\mathbf{n}_i = (\mathbf{t}_i, \varphi_i)$ by considering the Euclidean distance between $\mathbf{t}_c$ and $\mathbf{t}_i$ and the rotation cost to reach φ_i from φ_c.

Finally, we define a *utility* function that combines two objectives: i) maximise $rew^P(\mathbf{n}_i)$, and ii) minimise $moveCost(\mathbf{n}_c, \mathbf{n}_i)$. In this work, we aggregate these two objectives with a weighted sum, as this is the most straightforward and widely used approach to address multi-objective optimization[3]. Consequently, our utility function is expressed as follows:

$$util^P(\mathbf{n}_c, \mathbf{n}_i) = \underbrace{\alpha \cdot rew^P(\mathbf{n}_i)}_{reward} - \underbrace{moveCost(\mathbf{n}_c, \mathbf{n}_i)}_{cost}, \tag{14}$$

where α is a scaling factor to weight differently the two components and must be tuned to address the trade-off between gaining new information and minimizing robot movements.

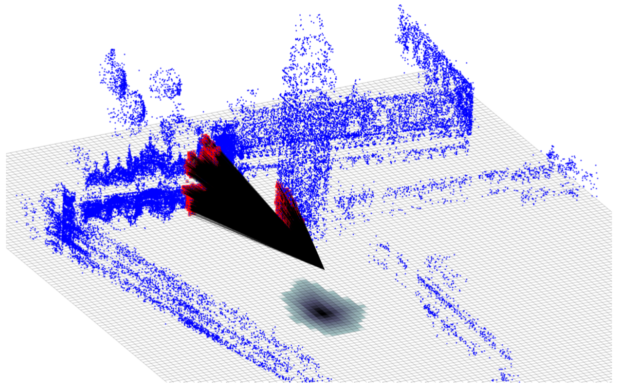


Fig. 3. Visual representation of the $rew^P(\mathbf{n}_i)$ function for each voxel and constant φ_i. The colors from dark to light represents reward values ranging from lower to higher respectively. The angle φ_i for which the reward function is displayed is at the same orientation of the previous move. As a result, we have higher reward (and gain in coverage) if we do sideways moves instead of backward/forward. Moreover the right direction is more interesting since the most of the observed points are located on the front-left area of the camera.

B. Multi-Robot Visual Coverage

Our approach to Multi-Robot coverage is based on a greedy method that merges information from the robotic platforms and assigns each robot to the best position given the current visual information acquired by all team members. In more detail, the global visual map is maintained by a central controller that merges information coming from the different robotic platforms. Each robot, sends the observed features to such centralized controller and since robots have homogeneous sensors, the controller updates $\psi^{\mathbf{P}}_{\mathbf{v}_k}$ for each specific voxel $\mathbf{v}_k$ by using Equation (9). Each robot then queries the controller to obtain the next target point. When the target point is reached by the robot it sends new acquired visual information to the controller and asks for a new target point. To compute the next target point the controller determines $\mathbf{n}^* = \arg\max_{\mathbf{n}_i} util^P(\mathbf{n}_c, \mathbf{n}_i)$. Notice that, following the classification provided in [25], our approach can be considered as a centralized weakly-coordinated approach, where robots exchange only their current state (position and observation) and the next position to go to is computed on the merged visual information provided by all robots. While this approach could poorly perform when robots start from the same position, our empirical results show that if the initial positions are taken from a random distribution, our approach is able to provide

[3]We note that more advanced techniques (such as the ones proposed in [24]) could be applied but leave this as a future direction.

good performance significantly outperforming a completely randomized approach and an uncoordinated method.

V. Experiments

In this section we detail the empirical evaluation of our approach. Specifically, we first introduce our methodology, then we describe our empirical settings and finally we discuss the results.

A. Empirical methodology

Our empirical methodology is based on a 3D simulation environment where a set of mobile cameras navigate in a given 2D map using *Stage 2D*. The 3D environment is based on real 3D reconstructions from images and the 2D map is built by pre-processing the same data. In more detail, we simulate a mobile robotic platform that is able to localize and navigate autonomously in the 2D map. To this end, we adopt *ROS (Robot Operating System)* to control our simulated platforms. For the simulation of the 2D navigation environment we use *Stage 2D*, a ROS module that simulates virtual 2D worlds and mobile platforms with sensors and actuators for which various level of noise can be considered (in this work we use the gaussian odometry error provided by Stage 2D with variance of 0.2 for all the variables[4]). As for sensing, each mobile platform is equipped with a laser range finder (used only for navigation and simulated by Stage 2D) and with a fixed camera, heading in the front direction of the robot (used for visual coverage and simulated using ad-hoc procedures described below). As for the navigation control stack, we use the *amcl* ROS module for localization and *move base* ROS module for path planning and motion control.

As mentioned before, we developed specific procedures to run our tests. These procedures are implemented in MATLAB: part of them are used to simulate the visibility model in our synthetic world, and the rest of them are used to update the coverage descriptor and to use it to compute the utility function.

For what concerns the visibility model, it is necessary to simulate the observations of cameras given the robot movements and to compute the associated gain for visual coverage. For this reason, specific procedures implement the visibility constraints: they define the set of 3D points that are visible to each camera, given the camera position and orientation. This information is the input of other procedures that use the visible features to update the coverage descriptor (defined in Section III), which is then used to compute the utility function. Finally, another set of procedures are dedicated to the computation of the utility function: when the robot pools for a new position to move to, these procedures use the information stored in the coverage descriptor to compute the utility function which can be used to create a ranking of desiderable positions and to choose the one with the highest value. The MATLAB code is executed by ROS (C++) routines via the MATLAB Engine[5].

[4]Check the odometery model parameters at: http://www.ros.org/wiki/amcl for further details.

[5]A set of API that allow C/C++/Fortran code to execute and query an instance of MATLAB. For more info: http://www.mathworks.it/it/help/matlab/matlab_external/using-matlab-engine.html

In order to test the system in a realistic scenario, we use 3D reconstructions of three different environments obtained from real world images [11], [10]: Trafalgar Square (London, United Kingdom), Piazza San Marco (Venice, Italy) and Piazza Bra (Verona, Italy). We pre-process these maps to remove outliers and to convert them in a suitable format to be used by Stage 2D[6]. Notice that, the 2D map is used only for navigation (i.e., path planning and localization) and that we assume planar movements for robots.

B. Empirical setting

The information that we use to evaluate the performance of our algorithm are the total number of explored voxels and the sum of coverage level of every voxel. In more detail, the metric measure used in the experiments is defined as the ratio between the total number of voxels observed during the exploration and the distance travelled so far by the robots. This gives us an estimate on how well the robots make use of their energy.

The first phase of our experiments consists in tuning the parameter α, used to compute the utility function. The value α expresses the reward that has the "same" value for moving 1 meter or rotating by 180 degrees. Such tuning phase was performed by trial and error on all the scenarios. This was achieved by restricting with incremental steps the parameters range so to reach satisfactory values for each of them. This phase showed that, on average, the best results were obtained with $\alpha = 15$; thus this is the value we used for all the other tests.

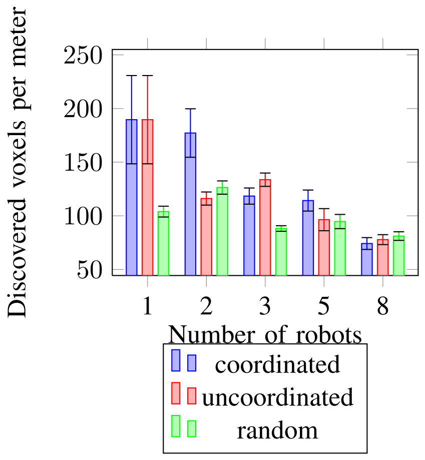
The core of our experiments consists in testing our algorithm by evaluating the above mentioned metrics. In more details, we compare our coordinated and uncoordinated approaches against a baseline method based on random movements of the robot which randomly picks a location inside the local grid as the next one to visit. In all the experiments the initial positions of the robots are randomly chosen from a selected area in the center of each map, and we run 10 repetitions for each experimental configuration (i.e., for each map and robot number). The exploration is performed until each robot reaches a fixed maximum travel distance (i.e. until the battery is too low).

C. Results

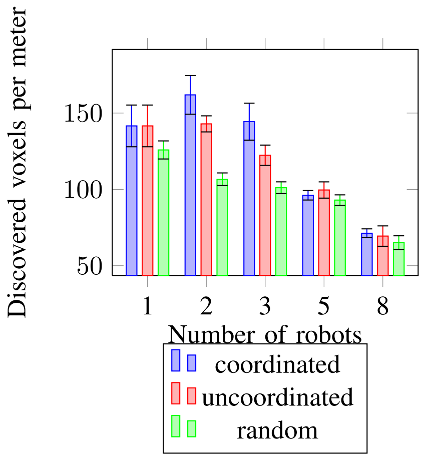
The results show that our cooperative and uncooperative approaches to visual coverage both outperforms the random baseline method. In more detail, Figure 4 reports the average number of voxels observed per meter (along with the standard error). For every map the coordinated algorithm achieves the best results. Furthermore, our algorithm achieves the best results for the average occupancy; that is, its average occupancy level is the closest to 0.5, as reported in Table I.

It is worth noticing, that the number of explored voxels per meter decreases as the number of robots increases (see again Figure 4). This is because, when the number of robot increases, each robot has less chances of observing new voxels. In other words, the robots can easily cover the whole map

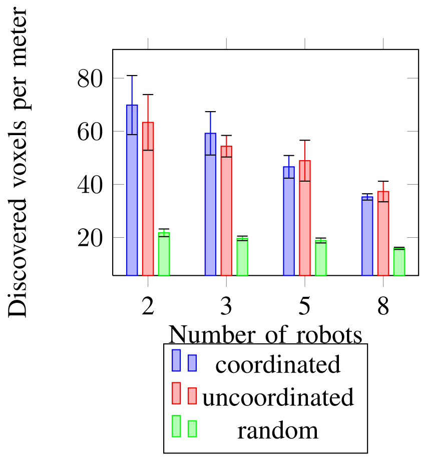
[6]We divide the 3D map in voxels and the 2D map in corresponding squares, we then consider a square to be occupied if the corresponding voxel contains a number of 3D points that is greater than a given threshold (which is specific for every map).



(a) Results for Piazza Bra.



(b) Results for Trafalgar Square.



(c) Results for Piazza San Marco.

Fig. 4. The plots show the voxels observed per meter. The bars represent the mean values while the whiskers represent the standard error.

with few moves and hence each of them observes less new voxels. Finally, by analysing data of the robots' trajectories, we also noticed that our algorithm performs much more rotations than the random approach (i.e., three times more). This is an interesting result, as far as in our setting, rotating allows to discover new voxels using a much lower amount of "battery".

map	**random**	**uncoordinated**	**coord**
Piazza Bra	0.00015	0.00977	0.01351
Trafalgar Square	0.0206	0.00244	0.01163
Piazza San Marco	0.0544	0.02842	0.02651

TABLE I. AVERAGE OCCUPANCY DEVIATION FROM $LimCov = 0.5$ FOR THE CASE OF 2 ROBOTS.

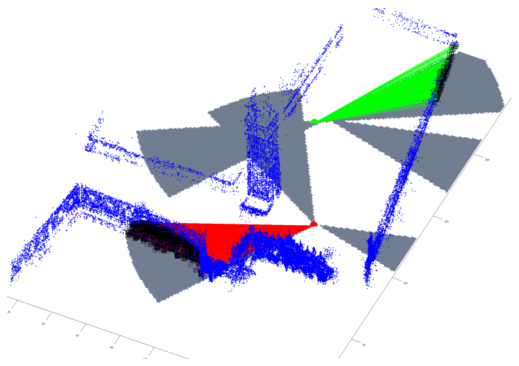


Fig. 5. Visual representation of the third move for two robots (red and green) in Piazza San Marco. The grey areas represent the coverage of the past and present moves (best viewed in color).

Figure 5 presents an exemplar navigation scenario with two robots (green and red) for Piazza San Marco. Here we visualise in gray the overall coverage at each movement and the respective visible points for the last move. Finally Figure 6 shows five moves of the coordinated strategy against the random one for the Trafalgar Square scenario with two robots. The coordinated approach movements clearly show a better coverage of the area. Qualitatively, a larger part of the map is coloured in grey denoting a better coverage. Moreover, with respect to the random robots, the coordinated ones avoid movements that overlap their field of view hence making a better use of energy (i.e., minimizing movement costs).

VI. Conclusions and Future Work

This paper proposes a novel perspective for visual coverage of large-scale 3D environments. Specifically, we define a descriptor that measures the amount of information acquired by a visual sensor. We also provide an autonomous cooperative control method for mobile robots that aims at maximising acquired visual information while minimizing movement costs. Moreover, we developed a simulation environment to evaluate visual coverage approaches that uses real images from large-scale outdoor scenarios (i.e., Trafalgar Square in London, Piazza San Marco in Venice and Piazza Bra in Verona) for simulating 3D visual sensing, and which employs widely used robotic tools (such as ROS and Stage) for 2D navigation. When evaluated in such environments, our approach gives evident benefits over a random baseline method and over an uncoordinated approach.

We believe that this work is a first and significant step towards the understanding of the 3D visual coverage problem for autonomous mobile robots and that it can have a significant impact on robotic systems for rescue and security applications, where collecting 3D information for large scale outdoor environments is crucial. Our future work in this space includes considering more complex aspects of the multi-view geometry problem, such as modelling possible mismatches of 2D image features due, for example, to wide baselines among different robot moves [26]. Moreover, we plan to investigate decentralized approaches to multi-robot coverage, where each robot maintains its own visual map and exchanges with peers only relevant information. In this regard, we might also apply cooperative strategies driving robots under different tasks such as one maximizing 3D reconstruction quality and the other maximizing area coverage.

References

[1] S. Thrun, S. Thayer, W. Whittaker, C. Baker, W. Burgard, D. Ferguson, D. Hahnel, D. Montemerlo, A. Morris, Z. Omohundro, C. Reverte, and

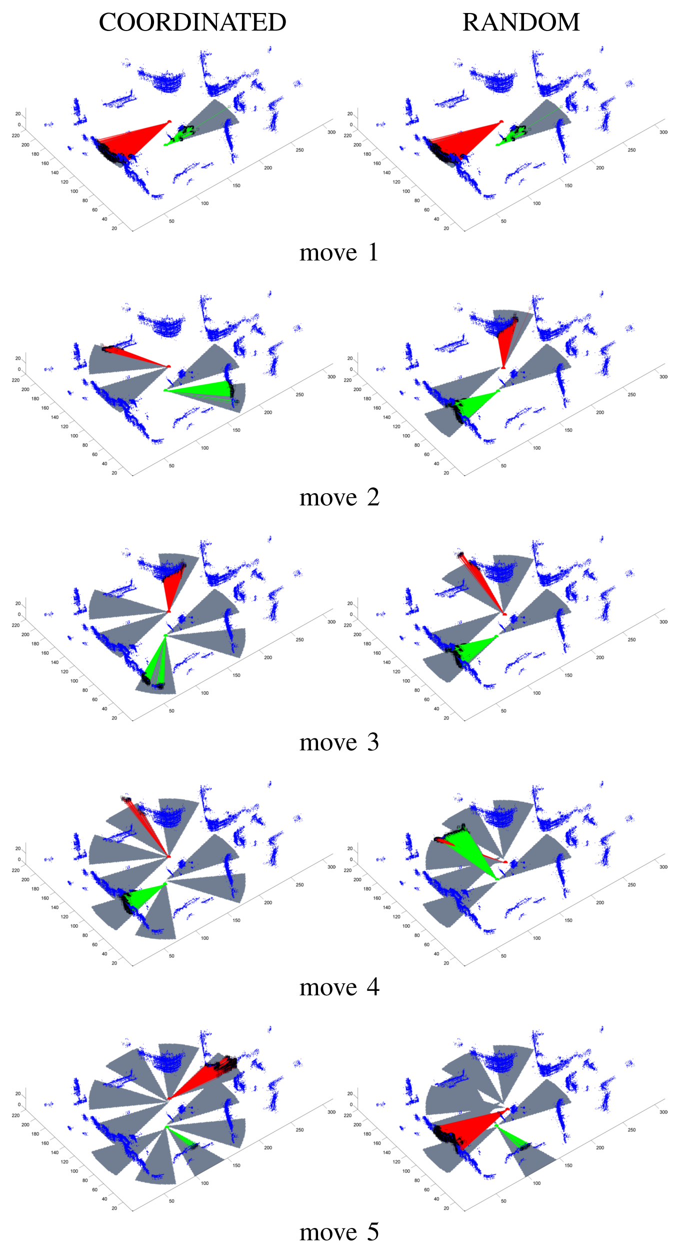


Fig. 6. The figure shows a comparison between coordinated and random approach for the Trafalgar Square scenario for two robots. The grey areas represent the coverage of the past and present moves while the black rays show the visible 3D points from each robot (best viewed in color).

W. W, “Autonomous exploration and mapping of abandoned mines,” *Robotics Automation Magazine, IEEE*, vol. 11, no. 4, pp. 79–91, 2005.

[2] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, “Coordinated multi-robot exploration,” *Robotics, IEEE Transactions on*, vol. 21, no. 3, pp. 376–386, 2005.

[3] J. C. Latombe, *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[4] C. Stachniss, G. Grisetti, and W. Burgard, “Information gain-based exploration using rao-blackwellized particle filters,” in *Proceedings of Robotics: Science and Systems (RSS)*, Cambridge, MA, USA, 2005.

[5] B. Yamauchi, “A frontier-based approach for autonomous exploration,” in *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 1997, pp. 146–151.

[6] C. Dornhege and A. Kleiner, “A frontier-void-based approach for autonomous exploration in 3d,” in *Safety, Security, and Rescue Robotics (SSRR), 2011 IEEE International Symposium on*, 2011, pp. 351–356.

[7] G.-J. M. Kruijff, V. Tretyakov, T. Linder, F. Pirri, M. Gianni, P. Papadakis, M. Pizzoli, A. Sinha, E. Pianese, S. Corrao, F. Priori, S. Febrini, and S. Angeletti, “Rescue robots at earthquake-hit mirandola, italy: a field report,” in *IEEE International Symposium on Safety, Security and Rescue Robotics*, 2012.

[8] Y. Baudoin, D. Doroftei, G. De Cubber, S. A. Berrabah, C. Pinzon, F. Warlet, J. Gancet, E. Motard, M. Ilzkovitz, L. Nalpantidis, and A. Gasteratos, “View-finder: robotics assistance to fire-fighting services and crisis management,” in *IEEE International Symposium on Safety, Security and Rescue Robotics*. IEEE, 2009, pp. 1–6.

[9] A. Mavrinac and X. Chen, “Modeling coverage in camera networks: A survey,” *International Journal of Computer Vision*, vol. 101, no. 1, pp. 205–226, 2013.

[10] S. Agarwal, N. Snavely, S. Seitz, and R. Szeliski, “Bundle adjustment in the large,” *Computer Vision–ECCV 2010*, pp. 29–42, 2010.

[11] R. Gherardi, M. Farenzena, and A. Fusiello, “Improving the efficiency of hierarchical structure-and-motion,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 2010.

[12] M. A. Hsieh, A. Cowley, J. F. Keller, L. Chaimowicz, B. Grocholsky, V. Kumar, C. J. Taylor, Y. Endo, R. C. Arkin, B. Jung *et al.*, “Adaptive teams of autonomous aerial and ground robots for situational awareness,” *Journal of Field Robotics*, vol. 24, pp. 991–1014, 2007.

[13] A. Singh, A. Krause, C. Guestrin, W. Kaiser, and M. Batalin, “Efficient planning of informative paths for multiple robots,” in *Proc. International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[14] P. Whaite and F. Ferrie, “Autonomous exploration: driven by uncertainty,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, no. 3, pp. 193–205, 2007.

[15] H. Surmann, A. Nüchter, and J. Hertzberg, “An autonomous mobile robot with a 3d laser range finder for 3d exploration and digitalization of indoor environments,” *Robotics and Autonomous Systems*, 2003.

[16] D. Joho, C. Stachniss, P. Pfaff, and W. Burgard, “Autonomous exploration for 3D map learning,” in *Autonome Mobile Systeme (AMS)*, K. Berns and T. Luksch, Eds. Springer, 2007, pp. 22–28.

[17] C. Stachniss and W. Burgard, “Exploring unknown environments with mobile robots using coverage maps,” in *Proc. of the International Conference on Artificial Intelligence (IJCAI)*, 2003.

[18] P. Sturm, “Multi-view geometry for general camera models,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2005, pp. 206–212.

[19] G. Schweighofer and A. Pinz, “Fast and globally convergent structure and motion estimation for general camera models,” in *Proc. British Machine Vision Conference (BMVC)*, 2006.

[20] A. Mittal and L. S. Davis, “A general method for sensor planning in multi-sensor systems: Extension to random occlusion,” *International Journal of Computer Vision*, vol. 76, no. 1, pp. 31–52, 2008.

[21] Y. Yao, C.-H. Chen, B. Abidi, D. Page, A. Koschan, and M. Abidi, “Sensor planning for automated and persistent object tracking with multiple cameras,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[22] D. Q. Huynh, A. Saini, and W. Liu, “Evaluation of three local descriptors on low resolution images for robot navigation,” in *24th International Conference on Image and Vision Computing New Zealand, 2009*, 2009, pp. 113–118.

[23] J. Park, P. C. Bhat, and A. C. Kak, “A look-up table based approach for solving the camera selection problem in large camera networks,” in *Proceedings of the International Workshop on Distributed Smart Cameras (DCS06)*, 2006.

[24] F. Amigoni and A. Gallo, “A multi-objective exploration strategy for mobile robots,” in *IEEE International Conference on Robotics and Automation*, 2005, pp. 3850–3855.

[25] A. Farinelli, L. Iocchi, and D. Nardi, “Multi robot systems: A classification focused on coordination,” *IEEE Transactions on System Man and Cybernetics, part B*, vol. 34, no. 5, pp. 2015–2028, 2004.

[26] H. Aanæs, A. L. Dahl, and K. S. Pedersen, “Interesting interest points: A comparative study of interest point performance on a unique data set,” *International Journal of Computer Vision*, vol. 97, no. 1, pp. 18–35, 2012.