Online Multi-object k-coverage with Mobile Smart Cameras

Lukas Esterle
Aston Lab for Intelligent Collectives Engineering (ALICE)
Aston University, Birmingham, UK
l.esterle@aston.ac.uk

Peter R. Lewis
Aston Lab for Intelligent Collectives Engineering (ALICE)
Aston University, Birmingham, UK
p.lewis@aston.ac.uk

ABSTRACT

In this paper, we combine k-coverage with the Cooperative Multi-robot Observation of Multiple Moving Targets problem, defining the new problem of *online multi-object k-coverage*. We demonstrate the benefits of mobility in tackling this and propose a decentralised multi-camera coordination that improves this further. We show that coordination exploiting shared visual features is more effective than coordination based on Euclidean distance. When coordinating k-coverage in a distributed way, our results suggest that the design of coordination mechanisms should shift towards decisions being made by potential responders with up-to-date knowledge of their own state, rather than a coordinating camera.

KEYWORDS

distributed k-coverage, CMOMMT, mobile smart cameras, dynamic reconfiguration, distributed control, distributed coordination

1 INTRODUCTION

The ability of smart cameras to pre-process visual information on site is often exploited, however little work has been done on a major benefit of smart camera networks: to operate effectively and efficiently without central coordination. In large networks of mobile smart cameras, decentralised coordination becomes more pressing as cameras can relocate to areas where communication with a central component might not be possible, for example due to lack of mobile network signal. Nevertheless, mobile smart cameras allow for i) rapid deployment in unknown environments without existing infrastructure, and ii) adaptation to unforeseen and rapidly unfolding situations.

In this paper we are interested in how to achieve high levels of k-coverage of a number of target objects, measured online over time, when both, cameras and objects, can move. For sensor networks in general, k-coverage is achieved when a monitored region is covered by at least k sensors [5], and is used as a measure in one-shot coverage optimisation problems. When measured over time as targets move, this gives rise to an online version of k-coverage, where it is generally not possible to ensure all targets are k-covered on an ongoing basis (e.g., as objects disperse or evade sensors, or as sensors fail), and instead we are faced with the objective of maximising the number of targets for which the network achieves k-coverage, over time. We call this *online multi-object k-coverage*.

The problem studied here extends the Cooperative Multi-Robot Observation of Multiple Moving Targets problem (CMOMMT) [12] to consider: (i) the case when the set of objects to cover changes over time, (ii) directional rather than omnidirectional cameras, and most importantly (iii) how to achieve k-coverage in this context [5].

In networks of static smart cameras, Esterle et al. [1, 3] showed how the efficiency of decentralised coordination (in terms of task performance and resource overhead) could be improved by cameras learning the neighbourhood relations between their fields of view (FoVs) at runtime. Here, we ask if this might aid the coordination of a network of mobile cameras. Our research questions are:

(1) Assuming each camera is controlled locally by an agent with access to only local information, to what extent can a network of mobile cameras operating in a distributed fashion maximise online multi-object k-coverage?
(2) If camera agents can exchange information locally through message passing to coordinate their movements, can online multi-object k-coverage be improved?
(3) Can learning neighbourhood relations between cameras benefit this, in terms of either performance in achieving online multi-object k-coverage, or the efficiency trade-off between this performance and the resource overhead (specifically, movement) involved in achieving it?

The experimental simulation study presented in this paper demonstrates that the addition of local multi-camera coordination improves online multi-object k-coverage by a network of mobile cameras, in a variety of different scenarios. This was found to be true across a range of different coordination schemes, even some that appear counter-intuitive. We also discuss the potential improvements in network-wide multi-object k-coverage when learning neighbourhood relations during runtime. Neighbourhood relations prove preferable to Euclidean distance when selecting communication partners, although, selecting random partners is as effective over time. However, our results further suggest that the responding camera is best placed to decide which object to cover.

The next two sections of this paper present a formal problem definition and related work. Section 4 addresses research question 1, evaluating distributed control with no communication or coordination. Section 5 turns to research questions 2 and 3, presenting several variants of a decentralised coordination approach, based on inter-camera "calls for help", various response models, and how cameras can learn neighbourhood relations during runtime. Sections 6 and 7 present and discuss results respectively.

2 PROBLEM STATEMENT

We consider a set of n mobile cameras $C = \{c_1, c2, ..., c_n\}$ and a set of objects $O = \{o_1, o_2, ..., o_m\}$. At present we consider only the 2D

ICDSC'17, Stanford, CA
© 2017 Copyright held by the owner/author(s). Publication rights licensed to ACM.
978-1-4503-5487-5/17/09…$15.00
DOI: 10.1145/3131885.3131909

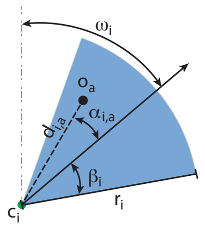


Figure 1: Illustration of an object in a camera's FoV. FoV is illustrated in blue with a range r_i, an orientation ω_i, and angle β_i on both sides of ω_i. The object is at angle $\alpha_{i,a}$ to the camera's orientation and distance $d_{i,a}$.

case, such that each object and each camera has an x_i, y_i location, $\vec{x}_i = (x_i, y_i)$. Each camera c_i can move with a velocity $\vec{s}_i$ and rotate with an angular velocity u_i. Furthermore, cameras communicate via message passing. We define a subset $P \subseteq O$ of all known objects, where objects of interest are denoted $p_1, p_2, ..., p_l \in P$. Objects in O can become important and unimportant to the network at any time. One can think of this process being driven by an operator or a specific camera identifying interesting behaviour, thus rendering an object as important (in P) or unimportant (in O but not in P).

Each camera c_i has its own field of view (FoV) f_i which is modelled as a cone and defined by a range r_i, an angle ω_i defining the viewing orientation relative to a fixed reference point, and an angle β_i defining the width on either side of ω_i. The range of a camera is limited by the distance at which an object can be detected and identified on-board the camera. Therefore a camera's state is defined as $c_i = \langle \vec{x}_i, \vec{s}_i, u_i, \omega_i, r_i, \beta_i \rangle$. This defines a snapshot at a particular point in time and can be further indexed by t to represent the camera's state over time. Specifically, the discrete-time behaviour of a camera c_i can be defined as

$$\vec{x}_i(t+1) = \vec{x}_i(t) + \vec{s}_i(t)$$
$$\omega_i(t+1) = \omega_i(t) + u_i(t) \tag{1}$$

The velocities $\vec{s}_i$ and u_i are controlled by an internal agent at each time t with the aim of achieving the current objective (e.g. follow object, move to cover object, move back to original location).

We consider an object o_a to be covered at a given time t, if the object is geometrically within f_i:

$$cov(o_a, f_i, t) = \begin{cases} 1, & \text{if } d_{i,a} \leq r_i \;\&\; |\alpha_{i,a}| \leq |\beta_i| \\ 0, & \text{otherwise,} \end{cases}$$

where $d_{i,a}$ is the Euclidean distance and $\alpha_{i,a}$ is the angle between the object o_a and the camera c_i. This is illustrated in Figure 1.

We consider an object o_j to be k-covered at a given time t if

$$kcov(o_a, k, t) = \begin{cases} 1, & if \ \sum_{i=1}^{n} cov(o_a, f_i, t) \geq k \\ 0, & otherwhise. \end{cases}$$

In our scenarios, each object has a 5% chance of becoming important at any time step, and then remains important for a random number of time steps drawn from the uniform distribution [5, 100].

For a given value of k, provided by an operator and known to all cameras, the aim is to maximise the k-coverage of objects over time, while minimising the total amount of camera movement.

$$\sum_{t=1}^{T} \sum_{a=1}^{m} kcov(o_a, k, t)$$

where T represents a finite time period of interest.

3 RELATED WORK

The presented work extends the Cooperative Multi-robot Observation of Multiple Moving Targets (CMOMMT) which was introduced as an NP-hard problem by Parker and Emmons [12]. They generate artificial forcefields for each object of interest which attracts individual robots to targets. Werger and Matarić [15] extend this towards W-CMOMMT (weighted CMOMMT) giving a weight to each target. They then use BLE (Broadcast of local eligibility) to directly coordinate tasks among the robots. Jung and Sukhatme [6] use learn densities of sensors and targets to steer individual robots to not sufficiently covered areas. This essentially leads to higher coverage of the available targets. Kollin and Carpin [8] performs target loss prediction to decide on when to call for help. They use broadcasting in order to ensure continues 1-coverage different objects. However, their main concern is to maximise overall coverage rather than covering objects with multiple sensors at once.

Covering objects with multiple sensors has received quite some attention as k-coverage in sensor networks. The idea is to have redundant measurements of a specific location [5]. This redundancy allows for resource constraint sensor networks to cover areas better or allow for sleep-cycles effectively prolonging network lifetime [9]. Liu et al. [10] specifically define directional k-coverage for visual sensors and discuss the benefits of k-coverage using cameras.

Fusco and Gupta [4] propose a simple greedy algorithm to optimally place and orient directed sensor for k-coverage of static objects in the environment. Micheloni et al. [11] identify activity density maps determine areas highly frequented by target objects and use an expected-maximization process to define optimal orientations of PTZ cameras. CMOMMT and coverage optimisation in camera networks has been researched quite intensively [14, 13, 7]. However, the problem of k-coverage with unknown number of targets using mobile smart cameras only received little interest yet.

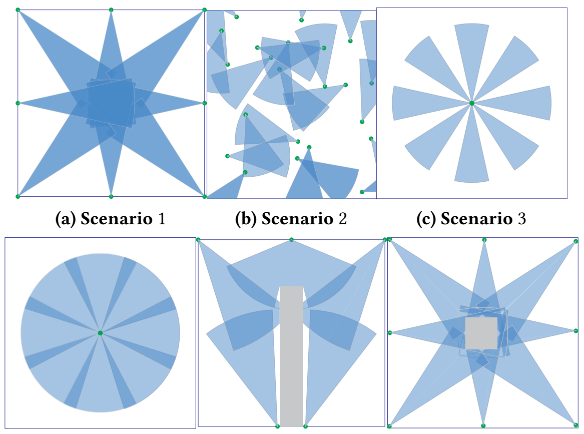
Our work is based on the ideas proposed by Esterle et al. [1, 3], who introduced a market-based approach in combination with artificial pheromones to efficiently coordinate tracking tasks in smart camera networks with limited resources. We transfer these ideas to networks of mobile smart cameras and the problem of distributed k-coverage.

4 BASELINE BEHAVIOUR AND RESULTS

First, we are interested in establishing the levels of online multi-object k-coverage achievable by distributed control, where there is no coordination between cameras. This forms a baseline for comparison against later approaches, where inter-camera communication is used as a basis for decision-making and online learning. There are two sources of change that the cameras must adapt to, over time. First, the set of objects to cover may change in its membership, and second, the physical positions of objects change.

To evaluate distributed control approaches in the context of these dynamics, we constructed six qualitatively different scenarios in the CamSim [2] smart camera network simulator. These are depicted in Figure 2. Objects and cameras move in a straight line according to a random vector, bouncing back in a random fashion upon reaching the boundary. Cameras can move as fast as objects and can turn fast enough to keep passing objects within their FoV.

We study three non-communicative baseline control approaches:


(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

(d) Scenario 4 (e) Scenario 5 (f) Scenario 6

Figure 2: Evaluated scenarios. Green dots represent cameras and blue cones their respective FoVs. Gray blocks illustrate opaque walls/areas.

(1) *Fixed locations*: Each camera has an assigned fixed location corresponding to its start position (see Figure 2).
(2) *Random movement*: Cameras have a starting location and move with a random vector. Their orientation changes by bouncing off from the boundary of the environment.
(3) *Random and following*: As *random movement* but changes velocity in order to follow important objects.

Figure 3 shows results for these three approaches. All graphs show mean results over 30 independent runs, while error bars represent one standard deviation. $T = 1000$ time steps, and $k = 3$.

It is clear that using *fixed locations* may achieve some very small k-coverage if there is an initial overlap. Pure *random movement* generally performs poorly since cameras do not follow important objects, thus k-coverage happens only by accident for a very limited time. *Random and following* performs by far the best at k-coverage, as well as at overall coverage of important objects (more than 75% on average in each scenario). While *fixed locations* can achieve this as well for overall coverage, this is highly dependent on their geometric overall coverage of the scene.

5 DECENTRALISED COORDINATION

To achieve a more coordinated approach to online multi-object k-coverage, while still retaining the decentralisation property, we next investigate a number of inter-camera communication schemes. Cameras notify others of important objects within their FoV, sending a "call for help". Upon receiving this, a camera decides how to react, based on their local information. Cameras use message passing to communicate. While we could consider simply broadcasting all information to all cameras, such a number of requests would be prohibitively expensive to process on board resource constrained cameras. The outline coordination algorithm is described in Algorithm 1, from which two important questions arise:

(1) To avoid broadcasting to all other cameras, how should individual cameras focus their communication efforts (i.e., on the most relevant recipients)?

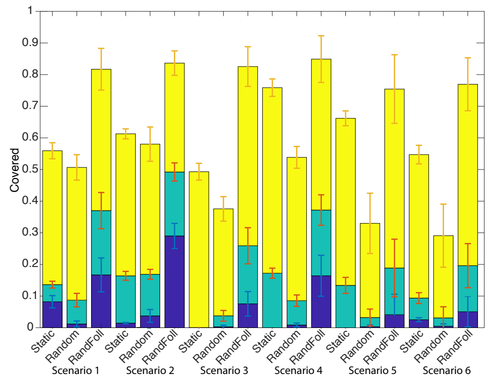


Figure 3: Illustration of baseline approaches *fixed locations*, *random movement*, and *random movement and following* for all 6 scenarios. The top yellow bar represents 1-coverage, second green bar illustrates 2-coverage, and bottom blue bars represent objects being at least k-covered.

Algorithm 1: Query-based dynamic k-coverage algorithm

```
1  foreach $c_i \in C$ at time $t$ do
2      foreach $o_a \in O$ & $cov(o_a, f_i, t)$ do
3          if $o_a \in P$ then
4              Notify other cameras as described in Section 5.1
5              Adjust camera to cover object $o_a$ (Equation 1)
6          else
7              if Received notification then
8                  React to notification (Section 5.3)
9              else
10                 Baseline behaviour (Section 4)
11             end
12         end
13     end
14 end
```

(2) How should a camera receiving such a "call for help" react? I.e., when and where should it move, or how should it issue follow-on communications?

5.1 Baseline Communication Strategies

In addition to a baseline broadcast behaviour, we evaluate three strategies for the targeting of inter-camera messages: k-CLOSEST relies on the Euclidean distance between cameras, notifying only the $k-1$ closest cameras to the camera's own location. k-FURTHEST notifies the $k-1$ furthest cameras from the notifying camera. Finally, k-RANDOM does not rely on the distance between cameras, but communicates with a random set of cameras in the network.

Both k-FURTHEST and k-CLOSEST are implemented in CamSim such that they use global information about the location of individual cameras. More realistically, k-CLOSEST could be interpreted as using short-range wireless technology. However, it is not necessary to explore the implementation feasibility of this further, as the results reported in this paper show that communicating purely based on distance can be improved upon using other methods.

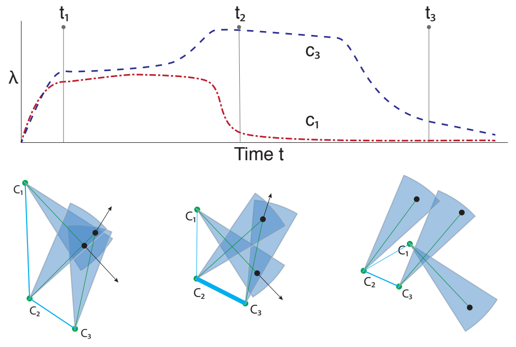


Figure 4: Illustration of the overlap vision graph with 3 cameras and the local view of the graph on camera c_2 over time.

5.2 Learning Overlap Relations for Targeted Communication

Often, cameras observe the same area. While each camera could analyse its entire FoV and identify potential overlaps with others, we rely on commonly observed moving objects to identify such areas. Since we are interested in the impact of this approach on the network, in CamSim we assume objects can be perfectly tracked and re-identified by different cameras.

Knowledge about commonly tracked objects is built up by each individual camera. We model this information as a graph, that can be interpreted as a *neighbourhood relation* between the two cameras. Inspired by the pheromones used in the foraging process of ants and the work by Esterle et al. [1, 3] on static smart camera networks, an edge is reinforced when two cameras observe the same object. At each discrete time step when a camera c_i observes a common object o_a with another camera c_j, we increase the value on a local link $\lambda(i, j)$. Since both cameras are aware of the shared object, $\lambda(i, a)$ and $\lambda(j, i)$ increase equally. To allow the network to adapt, we also artificially "evaporate" each link at a base level over time. To account for changes in distance between cameras, camera movement leads to further evaporation. In our simulation study, we reinforce links by a $\delta = 1$ per object per time step. Each link strength λ evaporates per time step with a base factor of 0.995 and an additional factor of 0.95 if the camera changes its pose or location. We selected δ and the evaporation factor to fit to our scenarios. Changing the values of them allows to adapt to the dynamics of the environment as it will increase or decrease the duration to *unlearn* previously learned links.

Figure 4 illustrates an example where two cameras are moving in the same direction, while a third moves perpendicularly. Black dots with arrows indicate objects and their direction of movement. Green dots with blue cones show cameras with their respective FoVs. Green lines indicate which camera follows which object. Blue lines between the cameras illustrate the overlap relations for c_2, and the graph above shows the strength of these over time. The link to both other cameras increases quickly, but decreases for c_1 as the shared object moves out of the FoV of c_2. The link to c_3 is reinforced until after t_2, as an object is visible to both cameras.

We use this local neighbourhood information to focus requests of other cameras, based on their potential overlap. As discussed in the context of static cameras by Esterle et al. [3], in contrast to using the Euclidean distance, cameras physically close but separated by an obstacle (e.g., a wall) would not be contacted. Further, the decay in the overlap graph means links also represent recency. The following communication strategies use the overlap graph:

k-Best picks the $k-1$ strongest links from the overlap graph and communicates with these cameras.

Step generates a probability of camera c_i communicating with camera c_j by thresholding the link strength. This is defined as:

$$P_{\text{Step}}(i, j) = \begin{cases} 1, & \text{if } \lambda_{i,j} > \theta \\ \eta, & \text{otherwise,} \end{cases} \tag{2}$$

Smooth interprets the strength of a link to another camera as a relative probability of communicating with it. This is defined as:

$$P_{\text{Smooth}}(i, j) = \frac{1 + \lambda(i, j)}{1 + \lambda(i, j^*)} \tag{3}$$

where

$$j^* = \underset{l}{\operatorname{argmax}}\, \lambda_{i,l}, \forall l \in c_1 \dots c_n.$$

Step and Smooth were first proposed by Esterle et al. [3]. We set the Step threshold $\theta = 10$ and residual probability $\eta = 0.2$.

5.3 Response models

When receiving to a request, the camera could simply oblige, and move to cover the object of interest. However, this might lead to over-provisioning on individual objects as well as losing objects currently covered by some cameras. In addition, this might lead to cameras constantly trying to cover different objects, getting stuck in-between both of them, as they always follow the most recent request. Therefore, in addition to this basic strategy (termed *Newest-Nearest*) we devised three additional response models:

(1) *Newest-Nearest* (NN): The attempts to cover to the most recently requested object from another camera. If there are multiple requests, it will choose the nearest.
(2) *Available* (AV): The camera will use the *Newest-Nearest* response model if and only if the camera is currently not occupied by covering/following a different object.
(3) *Graph* (GR): The camera considers the learnt *overlap* graph, attempting to cover the object requested by the camera with the strongest link. As with *Available*, if the camera is occupied, it will continue to cover its current object.
(4) *Received calls* (RE): In contrast to the other response models, this one considers currently covering cameras for a given object. Here, the camera will provision the object with the least requests as this corresponds to a small number of cameras currently observing this object. As before, this is only done, if the camera is currently available.

6 RESULTS

We analyse our communication strategies in combination with our proposed response models using the previously discussed scenarios.All experiments have been performed 30 times and the mean results and standard deviations are shown. Based on its performance with respect to k-coverage in the baseline experiments reported in Section 4, *random with following* was used as the default behaviour a camera reverts to when no notifications are received. Figure 5 shows the achieved network-wide coverage of objects using the different approaches. We included the results achieved when only using baseline behaviour and it is very clear that adding communication to the baseline behaviour increased online multi-object k-coverage. This is regardless of which communication strategy or response model is used. Even counter-intuitive communication strategies, such as k-FURTHEST, achieve better results than no communication indicating the benefits of our notification approach.

While STEP and SMOOTH perform very well, the importance of *response models* becomes clear. While all response models perform better in covering objects than any non-coordinated approach, *Graph* does not perform very well in achieving k-coverage. Nevertheless, it still achieves high coverage with one camera (1-coverage) throughout all experiments. With respect to communication strategies, the performance of *k-Random* and *Received Calls* seems counter-intuitive. While it only communicates with random cameras, rather than those that might be close-by, it still achieves very high k-coverage. This is due to the fact that *k-Random* communicates only with $k-1$ other cameras per timestep, sampled randomly each time. Eventually, this leads to communications with all cameras. However, cameras are only requested gradually, leaving the rest of the cameras available to provision other calls. *Broadcast*, in comparison, communicates with all cameras at once, making them effectively unavailable to other calls. STEP and SMOOTH, on the other hand, will continuously call the same cameras with only a small probability of calling others.

Results presented in Figure 6 show the average trade-off between higher movement and the achieved proportion of k-covered objects. All results are normalised (Coverage was normalised by the number of objects that have been covered at least once). The benefit of *Available* as well as the drawback of higher movement when using *Received Calls* becomes apparent. Again, the importance of the response model over the actual communication strategy become evident. The response model *Available* usually requires only about 40% of the movement in comparison to *Received calls* while both of them achieve about the same average coverage (between 45% and 55%). The trade-off becomes more pronounced when obstacles are in the experimental scenario as obstacles may affect the shortest routes for contacted cameras towards objects to cover.

7 DISCUSSION

In this paper we present a novel approach for k-coverage of moving objects in a confined scenario. Our approach uses knowledge of commonly observed objects to focus communication efforts. We were able to show the benefits of notifying cameras about objects of interest. Our results clearly indicate the benefits of coordinated notification as well as provisioning within the network. However, in our approach cameras currently only rely on local information and do not update each other on currently observed objects. We speculate that including this information exchange will improve the network-wide performance even further. In real world deployments, one will also have to take account of the time a detector/tracker requires to identify important objects.

Coming back to our initial research questions: (i) as shown in Figure 3, a small improvement in k-coverage can be achieved for a given number of cameras when those cameras can move, even in an uncoordinated fashion; (ii) as Figure 5 shows, the addition of local multi-camera coordination improves this further, and this is true for all the evaluated coordination schemes, even some that appear counter-intuitive; (iii) selecting communication partners based on neighbourhood relations was an improvement relative to selecting them based on Euclidean distance. However, a surprising result was that notifying random cameras often performed equally as well. Further analysis revealed that approaches based on neighbourhood relations focussed notifications, where a higher distribution would have been preferable. This broader communication is achieved over time by the *k-random* strategy. This stands in contrast to static networks where it has previously been shown that focussed communication is highly beneficial. Importantly, we have not disproved the benefits of communicating using neighbourhood relations in mobile camera network coordination.

However, cameras need to be able to make decisions based on rapidly changing states of themselves and their peers, and incorporate this information into their coordination behaviour. How to achieve this remains an open challenge. As a starting point, in our experiments, the choice of response model had a greater impact on the performance than the choice of communication strategy. This suggests that the camera responding is best placed to decide whether it should attend or not, as it has the most up-to-date information about its current state. The challenge is then how to enhance response behaviour with knowledge of the state of nearby cameras. Then, it may turn out that a simple random communication strategy is sufficient, when used with such behaviour.

ACKNOWLEDGMENT

This work was supported by the SOLOMON project (Self-Organisation and Learning Online in Mobile Observation Networks) funded by the European Union H2020 Programme under grant agreement no 705020.

REFERENCES

[1] L. Esterle, P. R. Lewis, M. Bogdanski, B. Rinner, and X. Yao. A socio-economic approach to online vision graph generation and handover in distributed smart camera networks. In *Int. Conf. on Distributed Smart Cameras*, pages 1–6, 2011.

[2] L. Esterle, P. R. Lewis, H. Caine, X. Yao, and B. Rinner. Camsim: A distributed smart camera network simulator. In *Int. Conf. on Self-Adaptive and Self-Organizing Systems Workshops*, pages 19–20, 2013.

[3] L. Esterle, P. R. Lewis, X. Yao, and B. Rinner. Socio-economic vision graph generation and handover in distributed smart camera networks. *Trans. on Sensor Networks*, 10(2):20, 2014.

[4] G. Fusco and H. Gupta. Selection and orientation of directional sensors for coverage maximization. In *Communications Society Conf. on Sensor, Mesh and Ad Hoc Communications and Networks*, pages 1–9, 2009.

[5] C.-F. Huang and Y.-C. Tseng. The coverage problem in a wireless sensor network. *Mobile Networks and Applications*, 10(4):519–528, 2005.

[6] B. Jung and G. S. Sukhatme. Cooperative multi-robot target tracking. In *Distributed Autonomous Robotic Systems 7*, pages 81–90. 2006.

[7] A. Khan, B. Rinner, and A. Cavallaro. Cooperative robots to observe moving targets: Review. *Trans. on Cybernetics*, PP(99):1–12, 2016.

[8] A. Kolling and S. Carpin. Cooperative observation of multiple moving targets: an algorithm and its formalization. *The Int. J. of Robotics Res.*, 26(9):935–953, 2007.

[9] S. Kumar, T. H. Lai, and J. Balogh. On k-coverage in a mostly sleeping sensor network. In *Int. Conf. on Mobile Computing and Networking*, pages 144–158, 2004.

[10] L. Liu, H. Ma, and X. Zhang. On directional k-coverage analysis of randomly deployed camera sensor networks. In *Int. Conf. on Comm.*, pages 2707–2711, 2008.

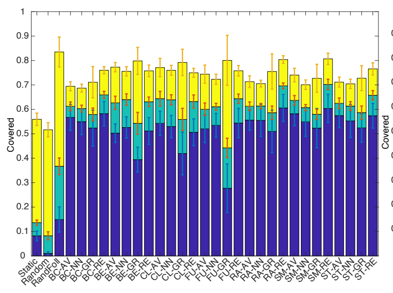
[11] C. Micheloni, B. Rinner, and G. L. Foresti. Video analysis in pan-tilt-zoom camera networks. *Signal Processing Magazine*, 27(5):78–90, 2010.

[12] L. E. Parker and B. A. Emmons. Cooperative multi-robot observation of multiple moving targets. In *Int. Conf. on Robotics and Automation*, volume 3, pages 2082–2089 vol.3, 1997.

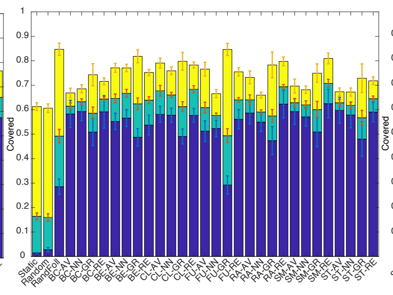
[13] C. Piciarelli, L. Esterle, A. Khan, B. Rinner, and G. L. Foresti. Dynamic reconfiguration in camera networks: A short survey. *Trans. on Circuits and Systems for Video Technology*, 26(5):965–977, 2016.

[14] J. C. SanMiguel, C. Micheloni, K. Shoop, G. L. Foresti, and A. Cavallaro. Self-reconfigurable smart camera networks. *Computer*, 47(5):67–73, 2014.

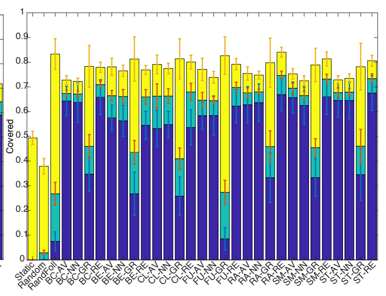
[15] B. B. Werger and M. J. Matarić. From insect to internet: Situated control for networked robot teams. *Annals of Math. and A. I.*, 31(1):173–197, 2001.



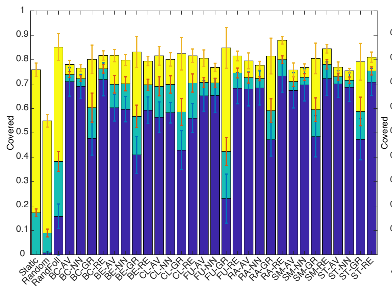
(a) Scenario 1



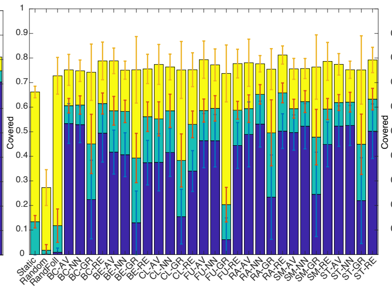
(b) Scenario 2



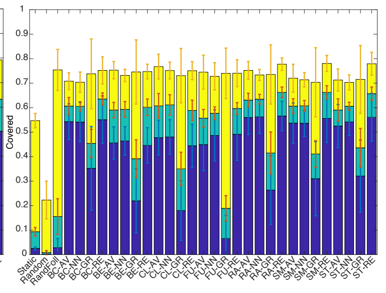
(c) Scenario 3



(d) Scenario 4

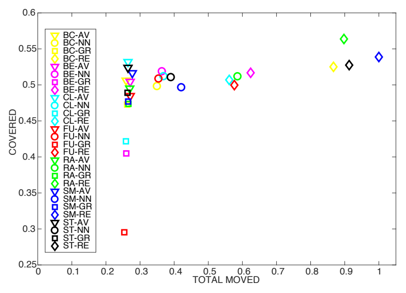


(e) Scenario 5

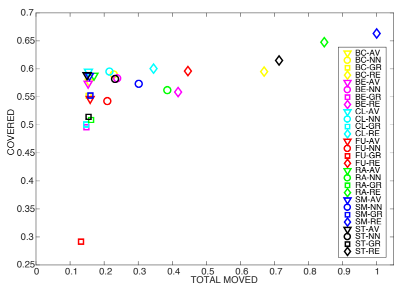


(f) Scenario 6

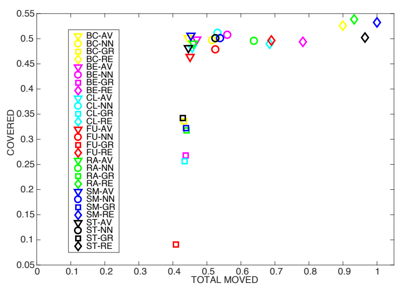
Figure 5: Comparison of the different approaches in terms of k-coverage with default behaviour *random movement with following*. The top yellow bar represents mean 1-coverage, second green bar illustrates mean 2-coverage, and bottom blue coverage represents mean k+-coverage and corresponding standard deviations over 30 runs. Static represents *Fixed lcoations*, Random is for *Random movement*, RandFoll is for *Random and following* baseline behaviour. Communication strategies are BC for Broadcast, BE for k-BEST, CL for k-CLOSEST, FU for k-FURTHEST, RA for k-RANDOM, SM for SMOOTH, and ST for STEP. Response models are AV for *Available*, NN for *Newest-Nearest*, GR for *Graph*, and RE for *Received calls.*



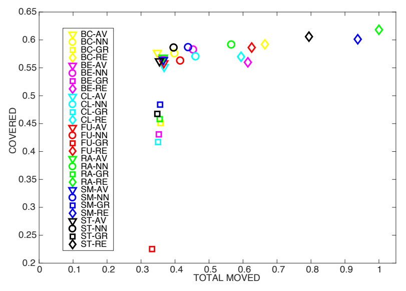
(a) Scenario 1



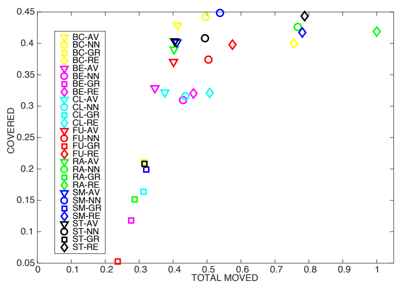
(b) Scenario 2



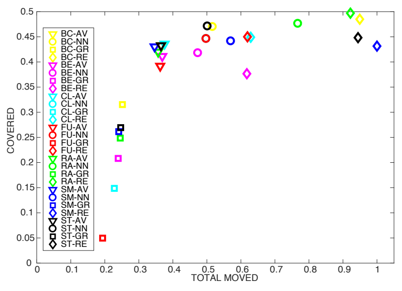
(c) Scenario 3



(d) Scenario 4



(e) Scenario 5



(f) Scenario 6

Figure 6: Trade-off of movement vs. achieved proportional coverage of previously seen objects for Scenario 1, 2 and 6.