abstract—We introduce the pan and scan problem, in which cameras are configured to observe multiple target locations. This is representative example within a broad family of problems in which multiple sensing devices are deployed, each in general observing multiple targets. A camera’s configuration here consists of its orientation and its zoom factor or field of view (its position is given); the quality of a target’s reading by a camera depends (inversely) on both the distance and field of view. After briefly discussing an easy setting in which a target accumulates measurement quality from all cameras observing it, we move on to a more challenging setting in which for each target only the best measurement of it is counted. Although both variants admit continuous solutions, we observe that we may restrict our attention to solutions based on pinned cones.

For a geometrically constrained setting, we give an optimal dynamic programming algorithm. For the unconstrained setting of this problem, we prove NP-hardness, present efficient centralized and distributed 2-approximation algorithms, and observe that a minimum quality threshold.

In full generality, the configuration of a camera will be characterized by the following tuple: \(\langle x, y, z, \theta, \phi, f \rangle\). The first three entries indicate the camera’s location in space. The targets will lie in the plane, but in some cases, as with UAVs or cameras deployed atop buildings, cameras may look down from on high. The fourth and fifth are angles indicating orientation. Cameras are directional, and may swivel (pan \(\phi\)); in the case of cameras positioned above the ground, there is a second orientation degree of freedom (tilt \(\theta\)). Finally, the last indicates the camera’s focal length or zoom factor. By shrinking a camera’s focal length (and thus enlarging its field of view), we allow the camera to observe more target locations, at the cost of correspondingly reducing the quality of the resulting image; conversely, by zooming in on a small area, the camera will record better quality images of the targets therein, while sacrificing targets lying outside this area. (Ideally, these targets will be covered by other cameras.) We now enumerate some example settings lying within this framework, in order of increasing generality:

- \(x, y, z > 0, \theta, \phi \text{ fixed; } f \text{ variable: } \text{UAVs hovering at fixed locations, facing downwards (predeployed disc-shaped sensors); for each sensor, we choose the radius.}\)
- \(x, y, z = 0, \theta = 90^{\circ} \text{ fixed; } \phi, f \text{ variable: ground-based cameras looking horizontally (predeployed cone-shaped sensors); for each sensor, we choose the angle } \phi \text{ and focal length } f.\)
- \(x, y, z \text{ fixed; } \theta, \phi, f \text{ variable: For each sensor, we also now choose orientation. Viewing regions are now conic sections.}\)
- \(x, y, z, \theta, \phi, f \text{ variable: We now may also choose the sensor locations.}\)

There are orthogonal options vis-a-vis mobility in the case of UAVs. Such vehicles may be deployable at will to arbitrary locations, or, if the UAVs are obliged to fulfill other responsibilities or, say, to maintain a certain topology, they may be limited to certain regions. Alternatively, the UAVs may be cycling over certain trajectories autonomously over time.

In the remainder of this paper, we introduce a specific geometric coverage problem inspired by an application involving cameras and targets, corresponding to the first special case listed above: cameras deployed to observe targets in the field may both swivel and adjust their focal length in order to observe one or more targets. That is, for each camera we
choose an orientation and a zoom factor \((x, y, z, \theta)\) are fixed, with \(z = 0\). By enlarging a camera’s focal length (and thus enlarging its field of view), we allow the camera to observe more target locations, at the cost of correspondingly reducing the quality of the resulting image; conversely, by zooming in on a small area, the camera will record better quality images of the targets therein, while sacrificing targets lying outside this area. The objective, informally speaking, is then to configure the cameras in order to observe as many targets as possible, with the highest possible precision.

A. Motivation and model

More formally, given are \(n\) camera locations in the field and \(m\) target locations to observe. (See Figure 1.) For each camera there are two parameters to set: the viewing direction \(\phi\) and the viewing angle \(\psi\), which assignment will allow the camera to observe all targets in the cone defined by angles \(\phi - \psi\) and \(\phi + \psi\) and the camera position \(P\). The quality of the a camera \(c\)’s observation of a visible target \(t\) depends on both the distance between \(c\) and \(t\) and on \(\psi\). The specific deterioration function may vary by camera hardware, but for a camera producing rectilinear images, the field of view [15] is proportional to the distance between camera and scene, specifically:

\[
\frac{o}{d} = \frac{i}{f}
\]

Here \(f\) is the focal length, \(i\) is the (constant) image size, and \(d\) is the distance. In this case, the object dimension, i.e., the field of view, is \(o = di/f\). An equation of this form holds in both horizontal and vertical dimensions. We are interested in the image quality, which is inverse to the object dimension in each direction, which then (for a camera \(s\) and a target \(t\)) depends on both \(d\) and \(f\):

\[
u(s, t) = f^2/d(s, t)^2
\]

That is, the observation quality varies inversely with the distance squared and directly with focal length squared. Increasing the focal length, however, decreases field of view \(o = di/f\). Thus there is a tradeoff between the angle of the viewing cone and the quality. Each such cone is infinite in area, but the imaging quality degrades as the cone stretches out to infinity. Narrowing the cone thus extends the range of imaging at a given quality farther out in the cone.

With this tradeoff between imaging quality and scope in place, we can now define an optimization problem. Given a set of target locations and a set of placed cameras, the goal is to configure cameras so as to maximize the total sensing quality. How to encode the assignments? Each possible direction/angle assignment for a camera will cover some contiguous subset of the targets. Although there are infinitely many possible angle/focal length choices, only a finite, polynomial number of such choices need be considered. Choosing the focal length and the angle \(\phi\) is equivalent to choosing the angles \(\phi, \psi\). For each camera, we may assume that these angles are tight in the following sense.

**Definition 1.1:** A pinned cone \(cone[j_1, j_2]\) (or just a cone when clear) is an assignment to a camera of angles so that its cone region intersects some targets \(j_1, j_2\) on both its boundaries (possibly the same target). Say a cone captures a target if the target lies within the cone region.

For each camera, we can restrict our attention to its \(O(m^2)\) possible pinned cones. (The value of any non-tight cone can only be improved by narrowing it until it goes tight.) In order to ensure finite utility values for all possible cones, we assume that each target is itself some distance \(C\) wide, so no cone’s angle will shrink to 0.

**ADD aggregation.** If a target is covered multiple times, a simple option for tallying its utility is simply to accumulate it in an additive way, i.e., a target’s total observation quality is:

\[
\sum_{k \in \gamma(j)} f_k^2/d(c(k), j)^2
\]

Here \(\gamma(j)\) is the set of all chosen cones \(k\) that capture target \(j\), and \(c(k)\) is the camera index underlying cone \(k\).

**Proposition 1.1:** The SUM formulation is solvable optimally in polynomial time.

**Proof:** For each camera, choose the highest value pinned cone, by enumeration. \qed

**MAX aggregation.** In some settings, however, accumulating utility from an arbitrary number of cameras is unrealistic. In the extreme case, no aggregation is permitted, in which case only one camera capturing a target within its cone as viewing the target would be counted when evaluating solution quality. Without loss of generality, it may be assumed that under a given cone configuration, each target is observed only by the camera producing the highest quality observation of it. In other words, in this formulation utility is aggregated by the MAX function rather than by SUM, i.e., a target’s solution quality is:

\[
\max_{k \in \gamma(j)} f_k^2/d(c(k), j)^2
\]

![Fig. 1. Three cameras configured to observe many targets.](image-url)
This setting is captured by IP formulation 1.

$$\max \sum_{j, k} x_{j, k} f^2_k / d(c(k), j)^2$$  \quad (1)$$

s.t. \quad \sum_{k} x_{j, k} \leq 1 \quad \forall j$$

\quad \quad \quad x_{j, k} \leq y_k \quad \forall j, k$$

\quad \quad \quad \sum_{k \in \sigma(i)} y_k \leq 1 \quad \forall i$$

\quad \quad \quad x_{j, k}, y_k \in \{0, 1\} \quad \forall j, k

Decision variable $x_{j, k}$ is 1 if target $j$ is covered by cone $k$, and 0 otherwise. The first constraint set ensures that we do not get credit for covering any target with more than one cone. Decision variable $y_k$ is 1 if cone $k$ is chosen, and 0 otherwise. The second ensures that we cover targets only with cones that are actually chosen. The third prevents our choosing more than one cone for any camera. Note that $c(k)$ is a constant specified by the problem instance.

### B. Contributions and generalizations

**Contributions.** We introduce several variants of the camera configuration problem. We show that the ADD variant and a geometrically constrained version of the MAX variant are both optimally solvable in polynomial time (Section II). We prove that the unconstrained MAX variant is NP-hard (Appendix A). We present two 2-approximation algorithms, one centralized and one distributed in an asynchronous model, and also note a setting that admits a PTAS (Appendix A). For a synchronized distributed setting, we give a 2-approximation protocol and a $(2\beta)/(1 - \alpha)$-approximation protocol (for all $0 \leq \alpha \leq 1$ and $\beta \geq 1$) with the stability feature that no target’s camera assignment changes more than $\log_3(m/\alpha)$ times (Section IV). We also discuss the running times of the algorithms and study the speed-ups that are possible in certain situations.

**Generalizations.** Although our motivating example problems above have their objective functions defined specifically in terms of distance squared and focal length squared, our algorithms presented below have more general guarantees. To obtain optimal algorithm correctness and constant approximation guarantees, we only assume that the quality of a camera’s measurement of a target declines monotonically as either the distance or the field of view grows. (Only in the case of the running time speed-ups do we assume that the measurement value is the product of separable monotonic functions of distance and field of view.) Moreover, although we focus on “cones” for concreteness, the approximation results extend to settings in which a camera’s possible sensing ranges may be other arbitrarily shaped regions. (In this case, a factor of $O(m^2)$ in the running times would be replaced by the number of possible configurations per camera.) Finally, all our results extend settings in which each target has a weight coefficient, representing its importance in a weighted sum, but we omit such weights for clarity of presentation.

### C. Notation

Throughout the paper we use $m$ to denote the number of targets and $n$ to denote the number of cameras. Thus there are $O(nm^2)$ pinned cones. We use variables $i, j, k$ to refer to indices of cameras, targets, and cones, respectively. We sometimes write $c(k)$ to indicate the camera underlying a given cone $k$. At a given point in time while an algorithm runs, $B(k)$ indicates the set of targets that would be (re)assigned to the camera underlying cone $k$, if cone $k$ were selected at that point.

### D. Related work

The pan and scan problem can be seen as a weighted, geometric variant of the Maximum Coverage problem, a dual problem to Set Cover, in which, given a set system and an integer bound $k$, the goal is to choose at most $k$ sets to cover a maximum-weight set of elements. The greedy algorithm provides a $1 - 1/e$ approximation, which is the best guarantee achievable unless P=NP [8].

A variant of Maximum Coverage was studied in [6], where the elements to be covered were unweighted and side constraints partitioned the available sets into groups from each of which at most one set could be chosen. They provide a 2-approximation greedy algorithm and state that $1 - 1/e$ is achievable through LP-rounding. Their version did not include element weights, and was non-geometric. Other related problems include the Multiple-Choice Knapsack problem, the Budgeted Maximum Coverage problem, and the Generalized Maximum Coverage problem [7].

One recent work motivated by similar applications is [4], in which directional antennas are configured in order to cover clients. The problems considered in that paper are similar in that the directional antenna can extend its range, distance-wise, by narrowing the angle at which it transmits. The specific optimization constraints and goals are quite different, however; they limit the number of targets covered per antenna or minimizing the number of antennas activated, for example. In other recent work [9], [10], approximation algorithms are given for several related coverage problem concerning directional sensors, including problems with goals to position sensors so that each point receives $k$ coverage and to minimize the time in which points are left uncovered.

Our problem differ crucially from these problems in that for us the quality of coverage depends on separating distance between camera and target. (We note also that, unlike ours, the NP-hardness proofs given in [9], [10] do not truly apply to the geometric problems studied, but rather involve settings in which a camera orientation may capture some arbitrary subset of the target points.) Heuristic algorithms for related coverage problems were also presented, without performance guarantees in [2], [13]. [1] studied the problem of placing directional sensors to cover (much of) a polygon, whereas we are concerned with orienting already-placed cameras in order to cover discrete targets.
More broadly speaking, our problem is a geometric coverage problem of a similar ilk of the one-dimensional and two-dimensional sensor radius problems (see [3] and references therein). Finally, measurement issues are dealt with from another, more physical perspective, in the statistics and engineering literatures (see [5] and references therein).

II. A CONSTRAINED, POLYNOMIAL-TIME SOLVABLE SETTING

In this section, we consider a constrained special case which, unlike the general two-dimensional setting discussed below, can be solved optimally in polynomial time. In this setting, there is a natural ordering on the targets, the cameras are interchangeable, and there always exist optimal solutions that are overlap-free.

Definition 2.1: We say a solution is overlap-free if no two of its pinned cones overlap unless one fully contains the other and that a setting is overlap-free if every instance admits an optimal solution which is overlap-free. Cameras are interchangeable if for each target set, all potential pinned cones covering it provide the same utility. The targets of an instance are naturally ordered, with ordering \( j_1, \ldots, j_n \), if for any three targets \( j_1, j_2, j_3 \) a cone cannot capture \( j_1, j_3 \) without also capturing \( j_2 \), and there exist optimal solutions in which no cone captures \( j_n, j_1 \), but no other cones (unless \( n = 2 \)).

One example setting satisfying these properties is when cameras are colocated at some point linearly separable from the target locations. Moreover, the setting of colocated cameras and targets located anywhere in the plane can also be solved by reduction to \( m \) separate naturally orderable subproblems.

Algorithm 1 Overlap-free DP (for target sequence \( 1, \ldots, m \))

1: for all \( 0 \leq j_3 \leq j \leq j_4 \leq m + 1 \), set \( cv[j_3, j_4, j] \)
2: the measurement quality for target \( j \) when observed by \( cone[j_3, j_4] \)
3: for all \( 0 \leq j_3 \leq j_1 \leq j_2 \leq j_4 \leq m + 1 \), set \( opt[j_1, j_2][j_3, j_4][i] \)
4: \( \leftarrow \sum_{j_1 \leq j_2 \leq j_3} cv[j_3, j_4, j] \)
5: for \( i = 1 \) to \( m \) do
6: for all \( j_3 \leq j_1 \leq j_2 \leq j_4 \) do
7: \( c_1 \leftarrow opt[j_1 + 1, j_2 - 1][j_1, j_2][i - 1] + cv[j_1, j_2, j_3] \)
8: \( c_2 \leftarrow \max_{0 \leq i' \leq i} \max_{j_1 \leq j_3' < j_2} \max_{j_4' \leq j_4} [opt[j_1, j_3'][j_3, j_4][i'] + opt[j_1, j_2][j_3, j_4][i - i']] + opt[j_1, j_2][j_3, j_4][i] \)
9: \( opt[j_1, j_2][j_3, j_4][i] \leftarrow \max\{c_1, c_2 \} \)
10: return \( opt[1, m][0, m + 1][n] \)

We solve this setting with a dynamic programming (DP) procedure (see Algorithm 1) which works recursively on the target sequence \( 1, \ldots, m \), computing optimal solutions for each combination an interval \([j_1, j_2]\), a superinterval \([j_3, j_4]\), and an integer \( i \) between 1 and \( n \), i.e. filling in a table with entries of the form \( opt[j_1, j_2][j_3, j_4][i] \). (Also used is a table \( cv[j_3, j_4, j] \), each entry of which corresponds to the measurement value for target \( j \) when observed by \( cone[j_3, j_4] \), if it captures that target, or 0 otherwise.) This subinstance corresponds to the the problem of covering interval \([j_1, j_2]\) with \( i \) cameras, plus the availability of a camera already assigned to the larger interval \([j_3, j_4]\). For convenience, we assume the existence of a zero-utility camera assigned to the interval \([0, n+1]\), which contains all genuine intervals.

For each such problem subinstance, there are two kinds of cases to consider. Either one of the \( i \) cameras is used to cover both targets \( j_1 \) and \( j_2 \), or not. Assume so. Then we consider \( opt[j_1 + 1, j_2 - 1][j_1, j_2][i - 1] \). Now assume not. Then we consider every possible way to carve up the interval \([j_1, j_2]\) into two pieces, say \([j_1, j']\) and \([j' + 1, j_2]\), combined with every possible way to divvy up the \( i \) cones between the two pieces, that is, the combination of \( opt[j_1, j'][j_3, j_4][i'] \) and \( opt[j' + 1, j_2][j_3, j_4][i - i'] \), for each \( j_1 \leq j' < j_2 \) and each \( 0 \leq i' \leq i \).

Since there are \( O(mn^k) \) instances, each taking time \( O(mn^k) \) to compute, total running time of the procedure is \( O(mn^{2n^2}) \).

Theorem 2.1: For every naturally ordered, interchangeable-cameras, overlap-free setting, Algorithm 1 produces an optimal solution.

Proof: (sketch) The proof of correctness is by induction on problem subinstances, over parameter \( i \), the number of cameras available for use in the subinstance. Assuming optimal values are correctly computed for all subinstances in which the number of cameras is less than \( i \), we correctly compute the optimal solution value for the current subinstance with \( i \) cameras, by construction.

Theorem 2.2: Naturally ordered and interchangeable cameras implies overlap-free.

Proof: Suppose that two pinned cones overlap without one containing the other, i.e., that one cone \( k_1 \) is defined by targets \( a, d \) and another \( k_2 \) defined by \( b, e \), where \( a < b < d < e \) (where the comparison operator indicates the order in which the targets appear on the circle). Suppose that lying between \( b \) and \( d \) are 0 or more targets \( c_1, \ldots, c_r \), each assigned to either \( k_1 \) or \( k_2 \). Among \( k_1 \) and \( k_2 \), let one have angle greater or equal to the other’s, say \( k_2 \). Then (re)assign \( b,d \), and each \( c_j \) to \( k_1 \), which will only increase the value of these reassigned targets, and shrink \( k_2 \) so that its coverage begins at the first target assigned to it following \( d \), which will only increase the value of \( k_2 \)’s targets.

Corollary 2.3: The setting with colocated cameras, all located at a point linearly separable from the targets, is polynomial-time solvable.

Proof: The natural ordering of the targets in this setting is the order in which a ray originating at the camera point, encounters the targets as it sweeps across the half-plane containing them.

Now consider a setting in which the cameras are colocated but the targets can lie anywhere in the plane. Although this
setting’s optimal solutions are overlap-free, the targets are not given as a linear sequence. We therefore solve the circular setting by solving $m$ all corresponding linear sequences.

**Corollary 2.4:** The setting with colocated cameras and targets located in the plane is polynomial-time solvable.

**Proof:** We execute the DP procedure $m$ times. In any optimal solution, there will be at least one pair of consecutive targets $j - 1, j$ that are not both covered by any one cone. Therefore for each value $j$ from 1 to $m$, we rename the target indices so that $j$ is the first, proceeding clockwise, and $j - 1$ is the last. We run the DP procedure on each such linear sequence, returning the best of these $m$ solutions as the result.

**Corollary 2.5:** Suppose that for each target $j$ there is a minimum acceptable utility value $b_j$ (and that there exist solutions obeying these constraints). Then all the results of this section still apply.

**Proof:** (sketch) Whenever $cv[j_3, j_4, j] < b_j$, reset $cv[j_3, j_4, j]$ to $-\infty$, which will have the effect of eliminating any solution in which $[j_3, j_4]$ is the best chosen cone capturing target $j$.

### III. Algorithms

In Appendix A we show the following, by reducing from a special case of Planar 3SAT [16].

**Theorem 3.1:** The two-dimensional camera configuration problem is NP-hard.

Moreover, in Appendix A, we show that under certain reasonable assumptions there exists a grid-shifting PTAS [12]. We turn now to approximation algorithms.

#### A. Combinatorial algorithms

In the problem instance, we have a total of $O(nm^2)$ pinned cones to choose among. The Greedy algorithm (see Algorithm 2) will repeatedly select a pinned cone that increases the current solution value by the maximum amount, i.e., when all captured targets that would benefit, whether not yet assigned or already assigned but receiving lower utility, are assigned to the newly chosen cone. Once a given camera is assigned a cone, all other cones for that camera are removed from consideration.

Extending the analysis of [6], we first name the cones that Greedy chooses $g_1, \ldots, g_n$, in order. Next, we name the cones of an optimal solution OPT $h_1, \ldots, h_n$, arranged so that the cameras of these cones appear in the same order as the cameras of the cones chosen by Greedy. Let $A_k = (a_{1,k}, \ldots, a_{m,k})$ be the vector of measurement quality increases for the $m$ targets, as a result of Greedy choosing the $k$th cone, and let $O_k = (o_{1,k}, \ldots, o_{m,k})$ be the coverage values for the $m$ targets by cone $h_k$. Let $ALG = (c_1, \ldots, c_m)$ and $OPT = (o_1, \ldots, o_m)$ be the final measurement qualities for the targets by Greedy and OPT, respectively.

**Lemma 3.2:** For each $k$, $\sum_j a_{j,k} \geq \sum_j \max\{o_{j,k} - c_j, 0\}$.

**Algorithm 2 Greedy**

1. $C \leftarrow$ all possible pinned cones
2. for each target $j$ and cone $k$, initialize $val(j, k)$ as the value of cone $k$ measuring target $j$
3. while $C \neq \emptyset$
4. for each pinned cone $k$ let $v_k = \sum_j val(j, k)$
5. select a most profitable pinned cone $k$, based at some camera $i$, and (re)assign each target $j$ such that $val(j, k) > 0$ to $k$
6. for each remaining cone $k' \in C$ and for each target $j$ now assigned to $k$, set $val(j, k') = \max\{val(j, k) - val(j, k'), 0\}$
7. remove from $C$ all cones based at $i$ and all cones of value zero
8. end while

**Proof:** (sketch; see also Lemma 4.1 below) If cone $g_k$ is the cone for camera $c(k)$ in OPT, then $g_k = h_k$ and so the inequality holds, so assume otherwise. Then Greedy chose a cone $g_k$ because the total increase $A_k'$ was larger than the increase would have been by choosing cone $o_k$, which increase is larger still than the of total gains of $o_k$ beyond the final values $c_j$.

**Theorem 3.3:** Algorithm 2 provides a 2-approximation.

**Proof:** Applying the lemma, we have:

$$\begin{align*}
ALG &= \sum_k A_k' = \sum_{k,j} a_{j,k}' \geq \sum_{j,k} \max\{o_{j,k} - c_j, 0\} \\
&\geq \sum_j \max\{\sum_k o_k - c_j, 0\} \\
&\geq OPT - ALG
\end{align*}$$

Therefore $ALG \geq OPT/2$.

Let $m$ be number of targets and $n$ be number of cameras. Then in a naive implementation of Algorithm 2, the initialization step and each iteration of the loop takes time $O(m^3)$, for a total of $O(n^2m^3)$. Using quality functions that degrade arbitrarily quickly with distance, it is easy to construct instances for which the approximation factor approaches 2.

Next we consider a distributed version of the greedy algorithm that assumes an asynchronous model in which cameras perform their cone-selection choices in arbitrary order, but we assume no two cameras will make their choices at precisely the same time.

**Proposition 3.4:** Algorithm 3 provides a 2-approximation.

**Proof:** Observe that the lemma still obtains because when camera $i$ chooses cone $k$, it does so because the benefit is greater or equal to choosing whatever cone OPT contains for $i$. The proof then goes through unchanged.

Each iteration of the loop completes in time $O(m^3)$, for a total of $O(n^2m^3)$ time in the naive implementation.

Although in principle we allow for cameras observing targets at arbitrarily large distances, we might reasonably truncate
B. Efficient implementation

We now explain how to implement the subroutine of choosing the best (current) cone for a given camera in time $O(m^2)$, on the assumption that the measurement quality for a cone/target pair is a multiplicative function of distance and zoom factor.

Fix a particular camera $i$, and renumber the targets $0$ to $m − 1$ as they are encountered by a ray originating at $i$ (the hand), sweeping around clockwise. We proceed in $m$ rounds, computing the values of $n$ cones each time. In round $j_1$, we compute the values of all cones picked out by targets $j_1$ and $j_2$ as $j_2$ varies (wrapping around modulo $m$) from $j_1 + 1$ to $j_1$, with the first cone observing all targets and the last observing only target $j_2$. Each new cone value is computed in constant time from the previous one, for a total of $O(m)$ per round.

Before beginning the rounds, we do some initial computation involving targets in order to support the cone evaluation. Let $q_j$ be the current quality value of the target $j$, and let $val(j, k) = 1/d_{j,k}f_k$ be the value with which cone $k$ can read target $j$ (if it captures it, or 0 otherwise), which is the product of the distance component and the product component. The value of cone $k$ will be the sum of the potential positive increases of these values, i.e., the sum of $D_k = \Delta_{j,k} = \max\{val(j, k) − q_k, 0\}$, summed over all the cone’s targets $j$ (i.e., those it captures), which is the same as the sum of the values $val(j, k) − q_k$, restricted to the cone’s targets that the cone would benefit. Call this set of targets $B_k$. This can be rewritten as $f_k\sum_{j\in B_k}1/d_{j,k} − \sum_{j\in B_k}q_k = f_kD_k − q_j$.

Each time the hand moves from one target to the next, one target is removed from evaluation, which means removing its $\Delta_{j,k}$. On the other hand, the value of the remaining targets increases, since the angle is now smaller. As the hand rotates around the camera, more and more targets depart from the cone. We keep stored the current values of $Q_k$ and $D_k$. To compensate for the higher quality readings of the remaining targets, after removing each lost target, it suffices to update $f_k$, $D_k$, and $Q_k$. Updating $f_k$ is done in constant time. To update the other two variables, we must add the corresponding sums restricted to the targets newly entering $B_k$.

But which targets enter $B_k$ each time the hand moves? A target is in $B_k$ exactly when $f_k/d_{j,k} − q_j > 0$, i.e., when $f_k > q_jd_{j,k} = T_j$, where we call $T_j$ the threshold for target $j$. (Note that within the computation for a single camera, $q_j$ and $d_{j,k}$ are constants.) We can compute this and sort the values $T_j$ in time $O(m \log m)$. Now consider the beginning of any particular round. All targets are captured by the cone. The members of $B_k$ are all targets for whom currently $T_j > f_k$. We maintain a pointer to the first target for which this is not so. Each time the hand advances, $f_k$ increases and we advance the pointer, adding targets to $B_k$, and updating $Q_k$ and $D_k$ until we reach a target whose threshold is not met. It therefore takes $O(m)$ time to update the $Q_k$ and $D_k$ values in each round, for a total of $O(m^2)$ time.

Thus we conclude the following.

*Theorem 3.5:* Let measurement quality be the product of functions of distance and zoom factor. Then Algorithms 2 and 3 can be implemented to run in times $O(n^2m^2)$ and $O(nm^2)$, respectively.

IV. DISTRIBUTED SYNCHRONOUS PROTOCOLS

As noted above, Algorithm 3 is a distributed algorithm in the sense that each camera can choose its configuration independently of other cameras, assuming no conflicts, i.e., assuming that each camera $c$’s decision is made quickly enough so that while $c$ is interrogating the cameras within its range, no other camera covers them simultaneously. In this section, we discuss more robust distributed protocols that maintain approximation guarantees but relax the no-collision assumption.

Algorithm 4 can be understood as a synchronous, rounds-based implementation of Algorithm 3 above, and so the
approximation guarantee remains unchanged. Unfortunately, a given target may be assigned as many as $n$ times over the course of the algorithm’s execution. The next algorithm is designed limit the number of reassignments while compromising the approximation guarantee only slightly. Note that with parameter values $\alpha = 0, \beta = 1$, its behavior is the same as Algorithm 4’s.

The following lemma is a generalization of Lemma 3.2 above.

**Lemma 4.1:** Let $\alpha = 0$ but allow $\beta \geq 1$. Then for each $k$, $\sum_j a'_j,k \geq \sum_j \max\{1/\beta o_j,k - c_j, 0\}$.

**Proof:** If the algorithm chooses the same cone for camera $k$ as OPT does, then for each target $j$ that $o_k$ covers, target $j$ is not assigned to camera $k$ only if it is already covered with value greater than $1/\beta$ the value it could receive from camera $k$. On the other hand, if the algorithm chooses a different cone for camera $k$, then this is because its choice of cone seemed sufficiently good in the following sense. Let $\text{val}(j)$ be the current coverage value for target $j$ at the time at which camera $k$ makes its choice. Let $a'_j,k, a_j,k, o_j,k,$ and $c_j$ be defined as in Section III, with the understanding that a target $j$ is assigned to cone $k$, and hence contributes to $a'_j,k$, only if it the value passes the tests based on $\alpha$ and $\beta$. Then the total increase due to ALG’s choice of cone for camera $k$ is

$$\sum_j a'_j,k = \sum_j \max\{a_j,k - \text{val}(j), 0\} \quad (2)$$

$$\geq \sum_j \max\{1/\beta o_j,k - \text{val}(j), 0\}$$

$$\geq \sum_j \max\{1/\beta o_j,k - c_j, 0\}$$

**Theorem 4.2:** Algorithm 5 provides a $(2\beta)/(1-\alpha)$-approximation guarantee.

**Proof:** Since at least one target can receive coverage value $e_{\max}$ but no target can receive greater than this, we have that $e_{\max} \leq \text{OPT} \leq m \cdot e_{\max}$. Suppose $\alpha = 0$ but possibly $\beta > 0$. Then by the lemma, we can expand the value of ALG as before yields the result that $ALG \geq OPT/\beta - ALG$, and so $ALG \geq OPT/(2\beta)$.

Now suppose $\alpha > 0$. The behavior of Algorithm 5 is unchanged if we eliminate all edges of value less than $\alpha \cdot e_{\max}/m$. Since each target gets coverage quality from only one camera, the optimal solution value decreases by at most $\sum_j \alpha \cdot e_{\max}/m = \alpha \cdot e_{\max} \leq \alpha \cdot \text{OPT}$. That is, the optimal solution value of the modified instance $\text{OPT}'$ is at least $(1-\alpha)\text{OPT}$. Applying the previous approximation guarantee to the modified instance, we conclude with a guarantee of $(1-\alpha)\text{OPT}/(2\beta)$.

**Proposition 4.3:** Algorithm 5 reassigns each target at most $\log_\beta(m/\alpha)$ times (where recall $m$ is the number of targets).

**Proof:** In the course of the algorithm, after initially being covered, a target’s coverage quality varies from a minimum of $\alpha \cdot e_{\max}/m$ to a maximum of $e_{\max}$, each time growing by a factor of at least $\beta$. The total number of reassignments is therefore at most $\log_\beta(e_{\max}/(\alpha e_{\max}/m)) = \log_\beta(m/\alpha)$.

For example, with $\alpha = 1/2$ and $\beta = 2$, we obtain an 8-approximation making at most $\log_2(m) + 1$ reassignment per target. In the worst case, Algorithms 4 and 5 can take $n$ rounds to complete. For graphs with bounded degrees, however, fewer rounds are required.

**Proposition 4.4:** Consider a bipartite graph whose vertices represent cameras and targets, and in which an edge appears exactly when a camera can potentially provide nonzero utility to a target. Assume the degrees of the camera nodes and target nodes are bounded by constants $\Delta_\text{c}$ and $\Delta_\text{t}$, respectively. Then Algorithms 4 and 5 both complete within $\log_{\Delta_\text{c}((\Delta_\text{t} - 1)(n)}$ rounds.

**Proof:** Consider a camera $i$ sending proposal in a given round. It sends proposals to at most $\Delta_\text{c}$, targets, each of which receives proposals from at most $\Delta_\text{t}$ cameras. Suppose camera $i$ receives its cone in this round. In the worst case, $c$ interferes with $\Delta_\text{c}((\Delta_\text{t} - 1)$ other cameras. More generally, in the worst case only one in $\Delta_\text{c}((\Delta_\text{t} - 1)$ remaining cameras receives a cone in each round.

Using the speed-up techniques of Section III-B, we conclude with the following.

**Proposition 4.5:** A camera’s work within each round can be implemented to run in time $O(n^2)$ or, given degree bound $\Delta_\text{c}$, time $O(\Delta_\text{c}^2)$.

---

**Algorithm 5** Bounded Reassignment Protocol (parameters $0 \leq \alpha \leq 1, \beta \geq 1$)

1: $C \leftarrow \{1, ..., n\}$
2: for each camera $i \in C$, compute the best value $e(i)$ of a cone based at $i$ observing a single target
3: perform a leader election protocol [11] in which the cameras broadcast their values $e(i)$ and all cameras learn $e_{\max} = \max_i \{e(i)\}$
4: while $C \neq \emptyset$ do
5: for each camera $i \in C$, let $c(i)$ be a best potential cone for $i$, based on $B(i)$, which is the set of targets lying inside $e(i)$ to which $c(i)$ would provide a coverage quality which is at least $\alpha \cdot e_{\max}/m$ and is at least $\beta$ times the target’s current coverage value
6: each camera $i \in C$ sends a coverage proposal to all targets $j \in B(i)$
7: each target that receives at least one proposal accepts the proposal coming from the camera of lowest index number
8: each camera $i$ that receives positive responses from all the targets it proposes to chooses cone $c(i)$ and has all targets in $B(i)$ assigned to it
9: end while
V. Open problems

One clear open problem is to find polynomial-time optimal algorithms, based e.g. on dynamic programming, for one-dimensional problem settings. In one natural one-dimensional setting, with cameras and targets appearing on the line, when a camera chooses some target to zoom in on, it observes that target and all subsequent targets, albeit with lower measurement quality.

More conceptually, the camera configuration problem could be generalized in two natural ways, aggregation functions and utility functions. We studied aggregation functions SUM and MAX, but other aggregation functions could be considered, including functions that may be tuned to trade-off between MAX, but other aggregation functions could be considered, utility functions. We studied aggregation functions SUM and be generalized in two natural ways, aggregation functions and.

REFERENCES

[2] Z. Charbiwala, Y. Kim, S. Zahedi, J. Friedman, and M. B. Srivastava. Orthogonally, including functions that may be tuned to trade-off between false-positive and false-negative errors (see [5]). Orthogonally, in some applications it might make sense to enforce limits (other than 1 or $n$) on the number of readings allowable for a given target or on the number of targets observable within a single cone.

APPENDIX

Let the two-dimensional camera configuration problem be defined as in the MAX formulation, except that focal length $f^2$ can be replaced (as discussed in Section I-B) with any monotonic function $f(j, k)$ of field of view and $d(c(k), j)^2$ becomes any faster growing monotonic function of distance, in the sense that increased distance cannot be entirely made up for by zooming in, which is a realistic assumption.

We reduce from a restriction of Planar 3SAT [16] in which each variable appears in at most three clauses, which remains NP-hard [17]. Given the bipartite graph corresponding to such a 3SAT instance, we produce a problem instance. All locations are chosen in such a way that “adjacent” cameras and targets lie exactly unit distance apart, with the exception of the cameras and targets composing the variable gadgets, which are positioned closer together. Let $u_1$ be the utility provided by a camera maximally zoomed in on a target at distance 1 away; let $u_{1/4}$ be defined similarly except with distance 1/4. We say that a variable target covered with utility $u_{1/4}$, or respectively a non-variable target covered with utility $u_1$, is covered perfectly. The problem instance is constructed so that given a positive 3SAT instance, in an optimal solution to the resulting instance, each target will be covered perfectly. In such a solution, each variable gadget will be oriented clockwise or counterclockwise, and each edge chain will either be oriented towards its clause or not.

For each clause we create a clause gadget as shown in Figure 2(a), in which a central target should be covered by one of three surrounding cameras, each located distance 1 away. Each camera is adjacent to a second target, which continues in an alternating chain of camera and target until reaching a variable gadget. (The chain can bend freely as long as the unit distance between adjacent nodes is maintained.) A variable gadget consists of three cameras arranged to form an equilateral triangle with side 1/2 and three targets, each group positioned to bisect one side of the triangle.

An edge gadget, corresponding to an appearance of this variable in the clause at the edge’s other end, is attached to the variable by positioning the edge’s final target collinear with one of the triangle’s sides, at distance 1/2 away. The dotted lines shown in Figure 2(c) are of unit length The target is positioned clockwise from its adjacent camera if the edge corresponds to a positive literal, and counter-clockwise otherwise. (The example in Figure 2(c) has two positive appearances and one negative.)

We now claim the construction works correctly.

Lemma A.1: The 3SAT instance is satisfiable iff in an optimal solution every target is covered perfectly.

Proof: First assume a solution with every target covered perfectly. Then each variable target must be covered by one of its two neighboring cameras. Due to the geometric nature of the variable gadget, deciding which camera will cover one target in a given variable determines which cameras will cover...
(a) Clause gadget.

(b) Variable gadget.

(c) Edge gadget.

Fig. 2. Gadgets used in the reduction from Planar 3SAT.

each of the other two targets, i.e., it determines an orientation of either clockwise or counter-clockwise for the variable. (The dotted lines shown in Figure 2(c) indicate a clockwise orientation.) The positioning of the edge chains as described above implies that a clockwise orientation will cover the final targets of adjacent positive edge chains, whereas a counter-clockwise orientation will cover the final targets of negative edge chains. In summary, each clause gadget must be covered perfectly, each edge chain is oriented, and each variable gadget has a consistent orientation. A satisfying assignment can then be read off from the variable orientations.

Now assume there is a satisfying assignment. For each clause, choose a variable appearing as a true literal in the clause under this assignment, and let the clause target be covered by the adjacent camera of the edge chain corresponding to this variable appearance. This choice orients the edge chain towards the clause. Orient the clause’s other two chains away from the clause. Now consider the resulting state of some variable gadget. If any of its appearances was used to make a clause true, then some of its edge chains are oriented away from it, meaning that these chains’ final targets are not covered. This determines an orientation for the variable gadget is chosen, either clockwise or counter-clockwise, in order to cover this target. A consistent orientation for each gadget is implied by the consistency of the satisfying assignment.

Thus we conclude that the problem is NP-hard.

We note that with the assumption of bounded measurement distance (i.e., that beyond a certain fixed separating distance, the observation quality falls to zero), plus the assumption that the graph of cameras and targets is drawn in a civilized manner [14] (with locations specified at finite precision and no two cameras or targets occupying exactly the same spot), a grid-shifting PTAS [12] can be obtained. We briefly sketch the details.

First, the existence of a global constant bounding the distance at which measurements of nonzero value are possible provides some locality, so for the targets within any enclosed area, any camera they are assigned to must lie sufficiently nearby. Second, the civilized-manner assumption means that within any region of constant area the number of cameras and targets will be bounded by a constant, and so the problem subinstance corresponding to that area can be solved in constant time. Together, these two assumptions allow us to implement a grid-shifting PTAS in which a grid, of precision based on the desired approximation guarantee, is lain on the plane. For each possible offset, we solve the problem subinstance contained within each cell of the grid, disregarding target-camera edges that cross the grid lines, unioning the results together. The best solution over all possible offsets is the solution returned. The grid is chosen to be sufficiently coarse so that the value of the optimal solution disregarding crossedges will be a good enough approximation, for at least one possible grid offset.

ACKNOWLEDGEMENTS

This research was sponsored by US Army Research Laboratory and the UK Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the US Government, the UK Ministry of Defence, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

We thank Mani Srivastava for useful discussions.