Angle optimization in target tracking

Report

Author(s):
Gfeller, Beat; Mihalák, Matúš; Suri, Subhash; Vicari, Elias; Widmayer, Peter

Publication date:
2008

Permanent link:
https://doi.org/10.3929/ethz-a-006821121

Originally published in:
ETH Technical Report 592

Angle Optimization in Target Tracking*

ETH Technical Report 592

Beat Gfeller[†], Matúš Mihalák[†], Subhash Suri[‡§], Elias Vicari[†], and Peter Widmayer[†]

April 2008

Abstract

We consider the problem of tracking n targets in the plane using $2n$ cameras, where tracking each target requires two distinct cameras. A single camera (modeled as a point) sees a target point in a certain direction, ideally with unlimited precision, and thus two cameras (not collinear with the target) unambiguously determine the position of the target. In reality, due to the imprecision of the cameras, instead of a single viewing direction a target defines only a viewing cone, and so two cameras localize a target only within the intersection of two such cones. In general, the true localization error is a complicated function of the angle subtended by the two cameras at the target (the tracking angle), but a commonly accepted tenet is that an angle of 90° is close to the ideal. In this paper, we consider several algorithmic problems related to this so-called "focus of attention" problem. In particular, we show that the problem of deciding whether each of n given targets can be tracked with 90° is NP-complete. For the special case where the cameras are placed along a single line while the targets are located anywhere in the plane, we show a 2-approximation both for the sum of tracking angles and the bottleneck tracking angle (i.e., the smallest tracking angle) maximization problems (which is a natural goal whenever targets and cameras are far from each other). Lastly, for the uniform placement of cameras along the line, we further improve the result to a PTAS.

1 Introduction

We study the problem of tracking targets by a set of cameras in the plane. The position of a target can be estimated if two distinct cameras are dedicated to tracking the target. We consider simple low-resolution cameras with very limited image processing capabilities. The quality of the estimated position of a target depends mainly on the relative position of the target and the two cameras [1]. Figure 1 depicts two different tracking situations of target t with two cameras to illustrate this phenomenon. The field of view of a camera is a cone, a target can be tracked by the camera if it lies in that cone, and therefore any target to be tracked by two cameras needs to lie in the intersection of the two respective cones. The geometry of the situation indicates that tracking accuracy is best if the angle at the target is closest to 90°.

The cameras in our setting cannot move, but can freely choose their viewing direction. A pair of cameras can be dedicated to track one target. Thus, tracking n targets requires $2n$ cameras. We assume for the moment that not only the cameras, but also the targets are points in the plane in

*Work partially supported by the National Competence Center in Research on Mobile Information and Communication Systems NCCR-MICS, a center supported by the Swiss NSF under grant number 5005 – 67322, and by the Swiss SBF under contract no. C05.0047 within COST-295 (DYNAMO) of the European Union.

[†]Institute of Theoretical Computer Science, ETH Zurich, CH-8092 Zurich, Switzerland. {gfeller, mmihalak, vicariel, widmayer}@inf.ethz.ch

[‡]Department of Computer Science, University of California, Santa Barbara, USA suri@cs.ucsb.edu

[§]Work done while the author was a visiting professor at the Institute of Theoretical Computer Science, ETH, Zurich. The author wishes to acknowledge the support provided by the National Science Foundation under grants CNS-0626954 and CCF-0514738.

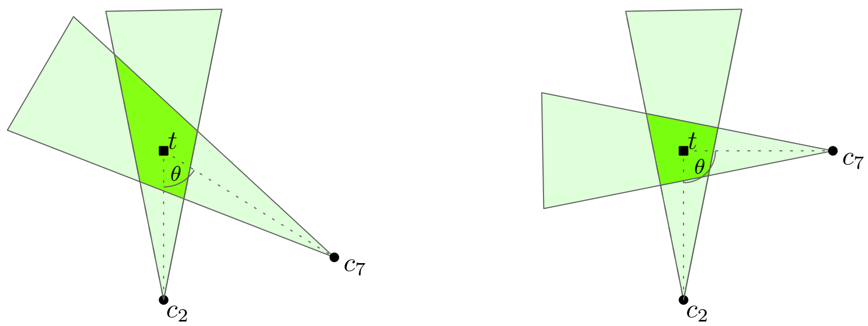

Figure 1: Two scenarios of tracking target t with tracking angle θ: less than 90° (left), and exactly 90° (right)

fixed positions. The *Focus of Attention* problem (FoA) for n targets and $2n$ cameras is to find a pairing of cameras and an assignment of camera pairs to targets that is optimum for some measure of tracking quality. In such an assignment, where each camera is assigned to exactly one target, and each target is assigned to two cameras, each target forms a triangle with its two assigned cameras. We evaluate the quality of the assignment as a function of the *tracking angles*, i.e., the angles that the triangles form at the targets. We consider three specific problems which belong to the following general family of combinatorial optimization problems:

Problem Family: Focus of Attention (FoA).
Input: A set T of n targets and a set C of $2n$ cameras, given as points in the plane.
Feasible solution: A camera assignment where each target is assigned two cameras, such that each camera is assigned to exactly one target.
Measure: A tracking angle for every triple consisting of a target and two assigned cameras.
Goal: Find a feasible solution which is optimal for one from a collection of objective functions defined on the set of tracking angles.

In this paper we consider the following objectives for the Focus of Attention problem. In the problem SumOfAngleDeviations the objective is to minimize the sum of the deviations of tracking angles from ninety degrees (90°); for SumOfAngles it is to maximize the sum of tracking angles, and for BottleneckAngle it is to maximize the minimum tracking angle. The latter two problems are interesting whenever targets and cameras tend to form small tracking angles, e.g. because the targets are far from cameras.

Notice that we assume the algorithms to know the exact position of the targets. In reality this is not possible for all scenarios. Nonetheless, the assumption is not a severe modeling simplification, if we assume that the targets are labeled and every camera can recognize the label of the tracked target. In such a situation every camera can look around (rotate the viewing focus) and for each target keep track of the angle under which the target is tracked. This information, together with the known positions of the cameras, allows to compute the tracking angle of any target and two cameras.

Related Work. Object (or target) tracking is an important task for environment surveillance and monitoring applications. It is a well established research subject [2] in the field of computer vision and image processing. Currently, multi-camera systems are being developed, where a certain depth information of objects needs to be computed for a given scene, ideally at low cost. The complexity of image processing and pattern recognition systems, and of the hardware is lowered when the images from cameras are obtained under a good tracking angle.

Isler et al. [3] were the first to consider this task as a combinatorial optimization problem. They

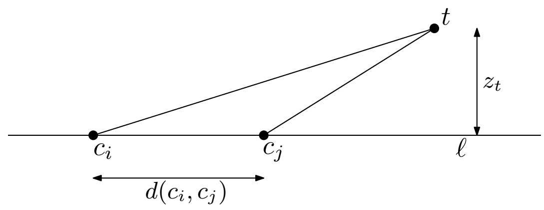

Figure 2: Illustration for the objective function of Isler et al. [3]

defined the *Focus of Attention* problem as a theoretical abstraction of the problem of lowering the computational costs of the depth estimation by assigning cameras to targets in an "optimal" way, and pointed out that for a very general problem setting (not in the plane) this comprises the classical NP-hard *3-Dimensional Matching* (3DM) problem as a special case. Therefore, the focus in [3] is on the problem version in which all cameras are restricted to lie on a single line ℓ. The objective is the aspect ratio $z_t/d(c_i, c_j)$, where z_t is the distance of target t from ℓ, and $d(c_i, c_j)$ is the distance between the two cameras c_i and c_j that are assigned to t (see Figure 2 for illustration). They give a 2-approximation for the problem of minimizing the sum of aspect ratios and for the problem of minimizing the maximum aspect ratio. Also, if the cameras are placed equidistantly on the line, they present a PTAS for the problem of maximizing the sum of aspect ratios. They also consider cameras on a circle and targets inside the circle with tracking cost being $1/\sin\theta$, where θ is the tracking angle, and deliver a 1.42-approximation for the problem of minimizing the sum of tracking costs, and for minimizing the maximum tracking cost.

Naturally, it is the powerful geometric structure that sets the Focus of Attention problems apart from more general assignment problems and makes it particularly interesting. If we would abandon the constraints that geometry imposes, Focus of Attention would belong to the class of *Multi-Index Assignment Problems* [4], where the well known NP-hard *3-Dimensional Matching* problem is a special problem in this class. Other NP-hard versions of multi-index assignment problems also focus on geometry, such as those aiming at the circumference or the area of a triangle formed by three assigned points in the plane [5]. An easy modification of the NP-hardness proof of the latter problem implies NP-hardness of SumOfAngles and BottleneckAngle [6].

Our Contribution. While aspect ratios of rectangles as in [3] might capture tracking quality very well in some cases, we believe that it will in general be more useful to optimize the most influential component of tracking quality directly, namely the tracking angles at the targets, even though this turns out to be surprisingly complicated.

We first show that the problem of minimizing the sum of the deviations of tracking angles from 90° (SumOfAngleDeviations) is NP-hard, and that it admits no (multiplicative) approximation. We then consider FoA that asks for a camera assignment with maximum sum of tracking angles (SumOfAngles), and FoA that asks for a camera assignment where the minimum tracking angle is maximized (BottleneckAngle). For cameras on a line, we present an algorithm that is a 2-approximation for both SumOfAngles and BottleneckAngle at the same time. This is the first constant approximation for the BottleneckAngle on a line, and for the case on a line, it also improves upon the previous $2 + 1/t$ approximation [7] (t is the size of the local neighborhood in the local-search algorithm) for SumOfAngles. For the special case where the spacing of the cameras on the line is totally regular, we present a PTAS for SumOfAngles.

2 NP-Hardness of SumOfAngleDeviations

In this section we consider the minimization FoA problem where the objective is the deviation of the tracking angle from 90°, and state that the problem is NP-hard, by showing that the corresponding decision problem OrthogonalAssignment is NP-complete.

OrthogonalAssignment: For a FoA problem, where every point has integer coordinates, decide whether there exists a camera assignment where every tracking angle is exactly 90°.

We reduce the following NP-complete RestrictedThreeDM problem to our OrthogonalAssignment problem.

RestrictedThreeDM: Given three disjoint sets X, Y, and Z, each with q elements, and a set $S \subseteq X \times Y \times Z$ such that every element of $X \cup Y \cup Z$ appears in at most three triples from S, decide whether there exists a subset $S' \subseteq S$ of size q such that each element of $X \cup Y \cup Z$ occurs in precisely one triple from S'.

Our proof is in the tradition of an NP-hardness proof of Spieksma and Woeginger [5] who showed that the following problem related to OrthogonalAssignment is NP-complete: Given sets of planar points A_1, A_2 and A_3, does there exist an assignment $X \subset A_1 \times A_2 \times A_3$ such that for every $x \in X$, the three points of x lie on a line? One can easily see that this result immediately implies that a "degenerate tracking of targets", where each target is collinear with the two cameras that track it (creating an angle of 0° or 180°) is NP-complete. However, the problem of tracking with 90° angles that we consider does not follow easily, but requires several new gadgets, constructions, and proof ideas.

Theorem 1 OrthogonalAssignment *is* NP-*complete.*

Proof. It easy to see that OrthogonalAssignment is in NP – guess the right assignment and check whether all target angles are 90 degrees. To complete the proof, we reduce the NP-complete problem RestrictedThreeDM to OrthogonalAssignment.

Consider an instance of RestrictedThreeDM, consisting of X, Y, Z and $S \subseteq X \times Y \times Z$ (with $|X| = |Y| = |Z| = q$ and $|S| = s$). Let us denote the elements of X, Y and Z as $x_1, x_2, \ldots$, $y_1, y_2, \ldots$, and $z_1, z_2, \ldots$. We construct three point sets C_1, C_2 and T which constitute an instance of OrthogonalAssignment with camera positions $C := C_1 \cup C_2$ and target positions T. This instance will be positive exactly if the RestrictedThreeDM instance is positive. All the constructed points will have integer coordinates with values between 0 and $p(q)$, where $p(\cdot)$ is a polynomial (which ensures that the reduction can be done in polynomial time).

Recall that each element in $X \cup Y \cup Z$ may occur in at most three triples $s \in S$. For ease of exposition, we assume in the following that each element occurs in exactly three triples. It is straightforward to adapt the proof to a situation with elements occurring also once or twice. We call every assignment of two cameras to a target, for which the tracking angle is 90 degrees, an an *orthogonal triple.*

First we construct for each triple $(x, y, z) \in S$ three points in the plane – two cameras c_x, c_z and one target t_y. In our construction, elements from Y will correspond to the constructed targets, and elements from X and Z will correspond to the constructed cameras. As every triple (x, y, z) is a candidate for a solution in RestrictedThreeDM, we make our three constructed points a candidate for a solution in OrthogonalAssignment – we place the points such that t_y sees the points c_x and c_z at the angle of 90 degrees – c_x and t_y on a horizontal line, and c_z below t_y, see Figure 3 for an illustration.

We call the points (cameras and targets) constructed from each triple $(x, y, z) \in S$ the *triple points.* Altogether we create $3s$ triple points (three for each triple). As we assume each element in $X \cup Y \cup Z$ appears exactly three times in S, each element $x \in X$, $y \in Y$ and $z \in Z$ leads to three points. We denote the three camera positions corresponding to $x_j \in X$ by $c^1_{x_j}$, $c^2_{x_j}$ and $c^3_{x_j}$. The three camera positions corresponding to $z_l \in Z$ are written as $c^1_{z_l}$, $c^2_{z_l}$ and $c^3_{z_l}$. The three target positions for each $y_k \in Y$ are denoted as $t^1_{y_k}$, $t^2_{y_k}$, and $t^3_{y_k}$. We put the cameras corresponding to $x_j \in X$ to set C_1, and cameras corresponding to $z_l \in Z$ to C_2.

The main difficulty in the construction is to avoid undesired possible orthogonal triples (i.e., (camera, target, camera) triples which do not correspond to any triple in the RestrictedThreeDM instance but whose triangle has a right angle at the target). We define the positions of the triple

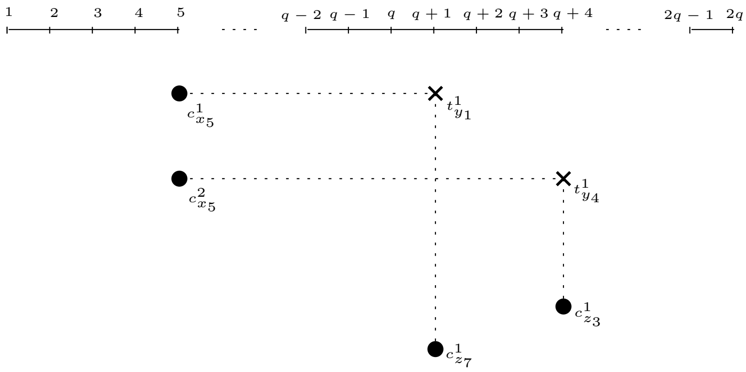

Figure 3: For each triple $(x, y, z) \in X \times Y \times Z$ there are two cameras c_x, c_z and one target t_y. The figure depicts a possible placement of points for triples (x_5, y_1, z_7) and (x_5, y_4, z_3)

points as follows. For each element $x_j \in X$, the points $c^1_{x_j}$, $c^2_{x_j}$, and $c^3_{x_j}$ have the same x-coordinate j. For each element $y_k \in Y$, the points $t^1_{y_k}$, $t^2_{y_k}$, and $t^3_{y_k}$ have the same x-coordinate $q + k$. The y-coordinates and the placement of points corresponding to elements from Z depends on the triples from S. We process the triples from S one by one, and place the corresponding cameras and targets as follows. Suppose we have considered $i - 1$ triples so far and now the ith triple (x_j, y_k, z_l) is considered. Let us assume that it is the fth occurrence of x_j, it is the gth occurrence of y_k, and it is the hth occurrence of z_l so far. The x-coordinates of $c^f_{x_j}$ and $t^g_{y_k}$ are j and $q+k$, respectively. The x-coordinate of $c^h_{z_l}$ is the same as the x-coordinate of $t^g_{y_k}$, i.e., $q + k$. We place the points $c^f_{x_j}$ and $t^g_{y_k}$ on the same horizontal line, i.e., their common y-coordinate is w (which we specify later on). The y-coordinate of $c^h_{z_l}$ is w'. We choose w to be the smallest integer such that the newly placed camera $c^f_{x_j}$ does not form an orthogonal triple with any two previously placed points, and such that the newly placed target $t^g_{y_k}$ cannot form an orthogonal triple with any two previously placed cameras, with the exception of "Z-type" cameras that correspond to an element from Z and that lie on the same x-coordinate as y^g_k (we will later see that this is not a problem). Similarly, we choose $w' \neq w$ to be the smallest integer such that $c^h_{z_l}$ does not form an "unwanted" orthogonal triple, i.e., an orthogonal triple with previously assigned points, with the exception of an assignment with a target with the same x-coordinate.

We can always find suitable w and w'. The reason is that any pair of a camera position c and a target position t only prevents placing cameras on the *excluding* line through t and orthogonal to line $\overline{ct}$. Similarly, any pair of cameras c_1 and c_2 only excludes points on their *excluding* Thales circle for placing new targets. The camera $c^f_{x_j}$ is placed on the vertical line $x = j$ and the target $t^g_{y_k}$ is placed on the vertical line $x = q+k$. Thus, the previously placed $2i-2$ cameras and $i-1$ targets form at most $(2i-2)(i-1)$ excluding lines and $\binom{2i-2}{2}$ excluding Thales circles that can together exclude at most $2\binom{2i-2}{2} + (2i-2)(i-1) = 6i^2 - 14i + 8$ choices for w. So w can be chosen within the range $1 \leq w \leq 6i^2 - 14i + 9$. Similarly for w', there are at most $(2i-2)(i-1) = 2i^2 - 4i + 2$ excluding lines that block the placement for $c^h_{z_l}$ on the line $x = q+k$ (and each of the lines intersect the line $x = q+k$ in exactly one point), and thus we can choose w' in the range $1 \leq w' \leq 2i^2-4i+4$ (w' has to be different from w).

Observe now that the constructed instance of OrthogonalAssignment so far admits an orthogonal triple for every target regardless of the original instance X, Y, Z, and S for RestrictedThreeDM. We therefore introduce additional cameras and targets (*gadget points* with polynomially bounded integer coordinates) which admit an orthogonal triple for every target if and only if there is a solution for the corresponding instance of RestrictedThreeDM. The gadget points assure that each point corresponding to some element in $X \cup Y \cup Z$ appears in the solution for OrthogonalAssignment at most once in an orthogonal triple of the form $(c^f_{x_i}, t^g_{y_k}, c^h_{z_l})$. This is achieved by introducing the gadget points as follows:

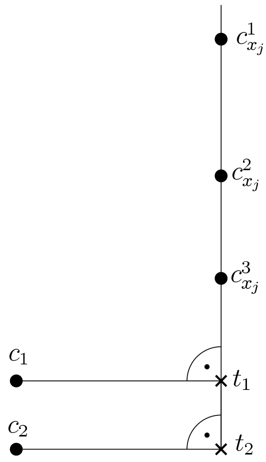

Figure 4: Gadget for the points corresponding to some $x_j \in X$.

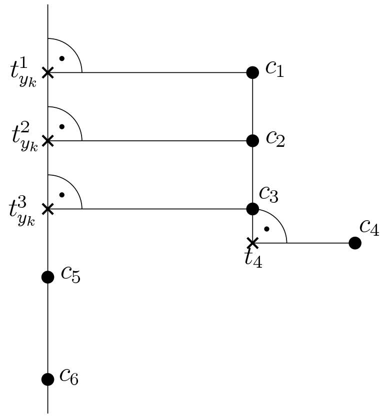

Figure 5: Gadget for the points corresponding to some $y_k \in Y$.

For each triple of points $c^1_{x_j}, c^2_{x_j}, c^3_{x_j}$ corresponding to some element $x_j \in X$ (which all have the same x-coordinate j, and only one of the three points can be used in an orthogonal triple formed by triple points only), we introduce two camera gadget points and two target gadget points as shown in Figure 4. Thus, we first place the two target points t_1 and t_2 having x-coordinate j and a y-coordinate which is not excluded by any Thales circle formed by a pair of already placed cameras. To make sure the camera positions c_1 and c_2 introduced on the same y-coordinate as t_1 and t_2, respectively, do not lie on a line already forbidden by a previous (camera, target) pair, we choose a y-coordinate larger than any y-coordinate used so far. Clearly, the y-coordinate can still be chosen to be an integer within a range polynomial in q. Similarly, the x-coordinates of c_1 and c_2 can be chosen in a polynomial range, such that c_1 and c_2 does not form an unwanted orthogonal triple (the only wanted triple are those formed by c_1, c_2, t_1, t_2, $c^1_{x_j}$, $c^2_{x_j}$, and $c^3_{x_j}$).

Similarly, for each of points $t^1_{y_k}, t^2_{y_k}, t^3_{y_k}$ corresponding to some element $y_k \in Y$ (which all have the same x-coordinate $q+k$), we introduce six camera-gadget points and one target-gadget points as shown in Figure 5. First, c_1, c_2 and c_3 are placed on the horizontal lines through the element points, and on a common x-coordinate which is not excluded by any previous points. Then, t_4 is placed with the same x-coordinate, and a non-conflicting y-coordinate, not lying on any line formed by a previous (camera,target) pair, such that c_4 can be placed at the same y-coordinate (with appropriate x-coordinate). To complete the gadget for $y_k \in Y$, we add two cameras c_5, c_6 on the x-coordinate $q+k$ and with non-conflicting y-coordinates. (By the same arguments as before, all the x-coordinates and y-coordinates can be found and are bounded in size by a polynomial in q).

Finally, the three points $c^1_{z_j}, c^2_{z_j}, c^3_{z_j}$ corresponding to some element $z_j \in Z$ lie on (possibly) arbitrary integer positions (within polynomial range). Still, we can introduce two camera-gadget points and two target-gadget points as shown in Figure 6: The first position we try for t_1 is $\overrightarrow{c^2_{z_l}} + \overrightarrow{c^1_{z_l} c^2_{z_l}}$. If this position is excluded, we consider $\overrightarrow{c^2_{z_l}} + 2 \cdot \overrightarrow{c^1_{z_l} c^2_{z_l}}$. In general, we try $\overrightarrow{c^2_{z_l}} + i \cdot \overrightarrow{c^1_{z_l} c^2_{z_l}}$ for increasing values of i until a position is found which does not cause a conflict. Note that these positions are all integer. When t_1 has been placed, we try positions $\overrightarrow{t_1} + i \cdot \overrightarrow{o}$ for increasing i for placing c_1, where $\overrightarrow{o}$ is the vector $\overrightarrow{c^1_{z_l} c^2_{z_l}}$ rotated by 90 degrees. Note that these positions are again integer. The remaining positions t_2 and c_2 can be chosen analogously.

It remains to show that the OrthogonalAssignment instance $\mathcal{I}_{90}$ we have described is indeed equivalent to the RestrictedThreeDM instance $\mathcal{I}_A$. By construction, it is clear that if there exists a perfect matching in $\mathcal{I}_A$, then there exists a solution for $\mathcal{I}_{90}$. For the other direction, assume that there is a solution for $\mathcal{I}_{90}$. In this solution, all gadget points must be part of exactly one (camera, target, camera) triple with an orthogonal angle at the target, which assures that for

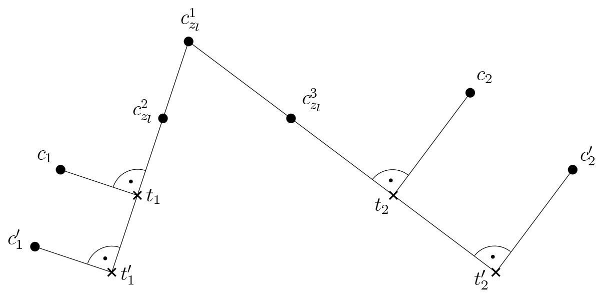

Figure 6: Gadget for the points corresponding to some $z_l \in Z$.

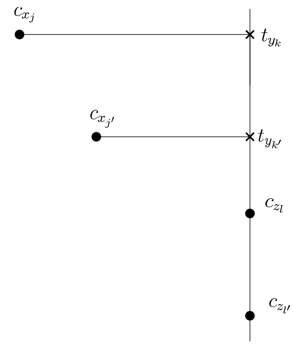

Figure 7: Forming a tuple corresponding to some $s \in S$.

each element in $X \cup Y \cup Z$, exactly one corresponding element point is part of a triple of element points. If all these triples correspond to some triple $s \in S$, then we have found a solution for $\mathcal{I}_A$. Otherwise, there exists a triple $(c_{x_j}, t_{y_k}, c_{z_l})$ in $\mathcal{I}_{90}$ not corresponding to any triple $s \in S$. As $\mathcal{I}_{90}$ matches all cameras and targets, there must exist a second triple $(c_{x_{j'}}, t_{y_{k'}}, c_{z_{l'}})$ such that $(c_{x_j}, t_{y_k}, c_{z_{l'}}) \in S$ (see Figure 7). Hence we can rearrange these two tuples into $(c_{x_j}, t_{y_k}, c_{z_{l'}})$ and $(c_{x_{j'}}, t_{y_{k'}}, c_{z_l})$, and obtain a solution where the number of triples not corresponding to some $s \in S$ is smaller than in the original solution. By repeating this process, we finally obtain a solution in which each (camera, target, camera)-triple corresponds to a triple in S, and thus yields a solution for $\mathcal{I}_A$. $\square$

Theorem 1 implies NP-hardness of all those FoA problems for which the objective function is optimum when the tracking angles equal 90°. These include the objectives of minimizing the sum/maximum of deviations of the tracking angles from 90°, or the goal of maximizing the sum/minimum of $\sin\theta_t$ over all tracking angles θ_t. Furthermore, the maximization FoA problem with the deviation of the tracking angle from 90° as the objective cannot be approximated, unless P = NP. We summarize this discussion:

Corollary 1 *Every problem from the family* FoA *for which the only optimum solution is a camera assignment with all tracking angles equal* 90° *is* NP*-hard.*

3 Maximizing the Sum/Min of Tracking Angles

In this section we consider the maximization FoA problems, where the objective is to obtain large tracking angles. In SumOfAngles we ask for a camera assignment such that the sum of the tracking angles is maximized. We also consider a bottleneck variant of the problem, BottleneckAngle, which asks for a camera assignment where the minimum tracking angle is maximized.

The approach to maximize tracking angles appears unreasonable whenever these angles get close to 180 degrees. However, it makes sense whenever targets are fairly far from cameras, i.e., for any assignment the tracking angle is always at most 90° (in other words, each target lies outside the Thales circle formed by any two cameras).

3.1 Cameras on a Line

We consider the scenario where the cameras are positioned on a horizontal line, and the targets are placed freely in the plane. We may assume, without loss of generality, that all targets lie above the line with cameras (otherwise we mirror the targets from below to the part above the line, with no change in the resulting assignment). An example of such a scenario is frontier monitoring, where the shape of the border can be approximated by a line. We present a 2-approximation algorithm for both SumOfAngles and BottleneckAngle. Note that the previous best approximation ratio for SumOfAngles was $(2+\epsilon)$ which was implied by the result of Arkin and Hassin [7], and nothing was known about the bottleneck version.

In the following, we denote by c_i both the i-th camera on the line, and its position, and assume that $c_1 < c_2 < \ldots < c_{2n}$. Note that we assume that no two cameras have the same position. This avoids complicated special cases, but our results still hold without this assumption.

We call the interval between two paired cameras the *baseline* of these cameras. Furthermore, we call a pairing of cameras *all-overlapping* if the baselines of any two camera pairs (of the chosen pairing) intersect. Observe that there always exists an optimum solution which uses an all-overlapping pairing – any two non-intersecting pairs can exchange their closest endpoints to create two intersecting pairs with larger baselines and thus larger tracking angles.

Lemma 1 *For both* SumOfAngles *and* BottleneckAngle *with cameras on a line, there exists an optimal solution with an all-overlapping pairing.*

Proof. For the sake of contradiction, consider an optimum solution with a maximum number of intersecting pairs, with at least two non-intersecting pairs (c_i, c_j) and (c_k, c_l), where $c_i < c_j < c_k < c_l$. By reassigning the target from (c_i, c_j) to (c_i, c_k) and the target from (c_k, c_l) to (c_j, c_l), a new solution is obtained in which both baselines are enlarged. Thus, in the new solution at least two more baselines (c_i, c_k) and (c_j, c_l) overlap, at least two angles increase, and no angle decreases, a contradiction. □

Note that for the objective of maximizing the sum of tracking angles, *every* optimal solution must have this property. We will make heavy use of the following consequence:

Corollary 2 *For both* SumOfAngles *and* BottleneckAngle *on a line, there exists an optimal solution where every left camera of a camera pair is among the leftmost n cameras, and every right camera of a camera pair is among the rightmost n cameras. These two groups of cameras can be separated by some point M on the line, such that $c_1 < c_2 < \ldots < c_n < M < c_{1+n} < c_{2+n} < \ldots < c_{2n}$.*

Approximation Algorithm.

Our approximation algorithm uses the structural properties of Lemma 1. We first create a simple *interleaved camera pairing* that proved its value in earlier work [3]: For $i = 1, \ldots, n$, pair camera c_i with camera c_{i+n}. Then we assign the interleaved camera pairs to the targets in an optimum way.

For SumOfAngles an optimum assignment of the camera pairs to the targets can be found by computing a maximum-weight perfect matching in a weighted complete bipartite graph where the camera pairs and the targets are the two vertex sets, and the weight of an edge between a pair and a target is the tracking angle of the triangle formed by the target and the pair of cameras. This matching can be computed in $O(|V|(|E| + |V| \log |V))$ time [8], which is $O(n^3)$ in our case, as we have a complete bipartite graph.

For BottleneckAngle, we can find an optimum assignment by a binary search for the maximum tracking angle (in the set of at most n^2 different tracking angles) for which a perfect matching exists. This means that for a tracking angle θ considered by binary search, all edges with value less than θ are discarded from the complete bipartite graph, and a maximum cardinality matching is computed. If the computed cardinality is less than n, we know that there is no camera assignment with interleaved cameras where the minimum tracking angle is at least θ, and the binary search proceeds with an angle smaller than θ, otherwise it proceeds with an angle larger than θ. We now describe the exact procedure in more details.

Given a weighted complete bipartite graph, we want to find a perfect matching for which the weight of the lightest matching edge is maximum. This problem is the so-called "bottleneck matching problem" or "max-min matching problem" described e.g. in [9], where an algorithm with running time $O(n^3)$ is given (for general bipartite graphs – the graph does not need to be complete).

The following idea however leads to a more efficient algorithm for our special setting: As the graph is complete, we know that a perfect matching exists. The idea is now to sequentially remove edges in increasing order of their weights, until the remaining graph does no longer have a perfect matching. In order to check whether a given graph has a perfect matching, we compute a matching of maximum cardinality of the graph. Note that for this computation, the weights can actually be ignored. A maximum cardinality matching on a bipartite graph can be computed in $O(m\sqrt{n})$ time [8]. In our case $m = n^2$ (n targets and n camera pairs), so we have $O(n^{5/2})$.

It is easy to see that if we remove edges in this order, until the remaining graph has no perfect matching anymore, then the last removed edge is the lightest edge in the desired matching.

Lemma 2 *Let $G = (V_1, V_2, E)$ be a weighted complete bipartite graph, and assume the edges in E are sorted in order of increasing weight, i.e., $w(e_1) \leq w(e_2) \leq \ldots \leq w(e_m)$. Then, define $G'(i)$ as the subgraph of G with the same vertices as G, but only with edges $e_i, e_{i+1}, \ldots, e_m$. Let i^* be the largest possible value such that $G'(i^*)$ has a perfect matching. Then, the bottleneck matching of G has objective value $w(e_{i^*})$.*

Proof. For the sake of contradiction, assume that the bottleneck matching of G has an objective value larger than $w(e_{i^*})$. Thus, none of the edges $e_1, e_2, \ldots, e_{i^*}$ is contained in that matching, and hence in $G'(i^*)$ there must be a perfect matching using only edges from $e_{i^*+1}, e_{i^*+2}, \ldots, e_m$, a contradiction. Similarly, if the bottleneck matching of G has an objective value smaller than $w(e_{i^*})$, then we get a contradiction because by the definition of i^* there exists a bottleneck matching in G which uses only edges from $e_{i^*}, \ldots, e_m$. □

This process of removing edges can be accelerated: instead of removing edges one by one, all below a threshold T are removed in one step (this can be done in constant time if one sorts the edges first). Initially, T is set to the median weight of the edge weights, and then a binary search is performed to find the smallest threshold which still permits a perfect matching. As the number of threshold values is at most $m = n^2$, the binary search requires $O(\log m) = O(\log n)$ steps. Thus, with this approach the total time required to find the max-min matching is $O(n^{5/2} \log n)$.

The aforementioned discussion shows that for any given camera pairing, we are able to efficiently find the best assignment of camera pairs to targets. Our algorithm, which we call Interleave, uses the interleaved pairing. In the following we analyze its approximation ratio.

Theorem 2 *For any solution O using an all-overlapping pairing, there exists a camera assignment with the interleaved pairing where each tracking angle is at least half of the corresponding tracking angle of the solution O.*

Proof. Let H denote the interleaved pairing, and let O be a target assignment using an all-overlapping pairing. We will show that any solution which uses an all-overlapping pairing can be transformed into a solution which uses the interleaved pairing H by a sequence of at most n steps of a particular nature: In each step, two pairs of cameras are chosen to create two new pairs under the following constraints: (1) One of the tracking angles may decrease by a factor of at most two, but from then on it stays the same throughout the rest of the transformation. (2) All other angles either stay the same or increase. The claim then follows from the constraints of the transformation. We will now show that for any solution O using an all-overlapping pairing, there exists a solution T using the interleaved pairing H, such that each tracking angle in T at any target is at least half of the corresponding tracking angle in O.

As O contains an all-overlapping pairing, there exists a point M on the line which separates the left ends of all baselines from the right ends of all baselines in O (see Figure 8). We prove the existence of the transformation by induction on the number k of targets in the instance. If $k = 1$, the claim is trivially true because O's pairing is equal to H (and thus no transformation is needed). For $k \geq 2$, let (c_1, c_{i+n}) be the baseline with leftmost starting point in O. If $i = 1$, then this camera pair is present in both O's pairing and in H, and by the induction hypothesis the

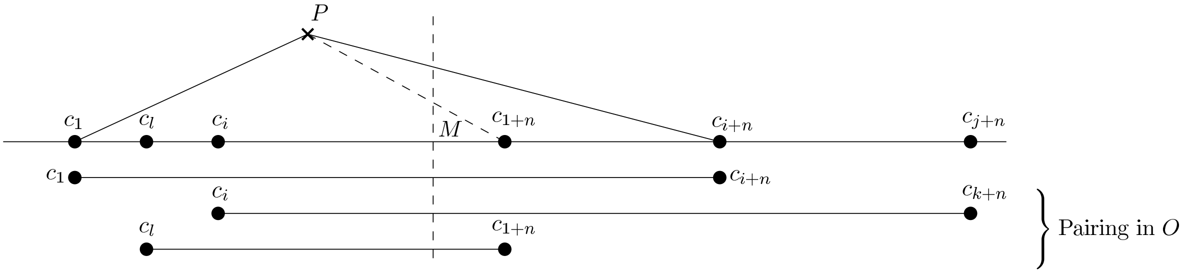

Figure 8: Comparing the camera pairings in O with those in H

desired transformation exists for the $k-1$ other targets and camera pairs. In the following, assume that $i \neq 1$. In H, c_1 and c_i will be paired with c_{1+n} and c_{i+n}, respectively. Considering O, let c_l be the camera paired with c_{1+n}, and let c_{j+n} be the camera paired with c_i.

Let P be the target which O assigns to the pair (c_1, c_{i+n}). We now distinguish two cases:

(A) The angle c_1, P, c_{1+n} is at least half of the angle c_1, P, c_{i+n}. Let Q be the target which O assigns to the pair (c_l, c_{1+n}). We create from O a new camera assignment O' by transforming the pairs (c_1, c_{i+n}), (c_l, c_{1+n}) into pairs (c_1, c_{1+n}), (c_l, c_{i+n}) and assigning these pairs to targets P and Q, respectively. Thus, the tracking angle at P in O' is at most cut in half, and the tracking angle at Q in O' increases, as the baseline (c_l, c_{i+n}) extends to the right (compared to (c_l, c_{1+n})).

(B) The angle c_i, P, c_{i+n} is at least half of the angle c_1, P, c_{i+n}. Let Q be the target which O assigns to the pair (c_i, c_{j+n}). We create from O a new camera assignment O' by transforming the pairs (c_1, c_{i+n}), (c_i, c_{j+n}) into pairs (c_1, c_{j+n}), (c_i, c_{i+n}), and assigning them to targets Q and P, respectively. The tracking angle at Q increases (as the baseline lengthens in the new assignment), and the tracking angle at P is at most cut in half.

As c_i and c_{1+n} both lie between c_1 and c_{i+n}, and c_i is to the left of c_{1+n}, at least one of these cases applies. In both cases, one angle increased, and one angle decreased by a factor of at most 2, but this angle uses a pair from the interleaved pairing. Thus, there remain $k-1$ camera pairs which are potentially not assigned to a camera pair from the interleaved pairing. By the induction hypothesis (applied on the $k-1$ targets and the cameras assigned to them) these targets can be assigned to camera pairs from the interleaved pairing such that each of these $k-1$ other angles are at most cut in half during the transformation. □

This theorem directly proves that our algorithm computes a 2-approximation for both the objective of maximizing the sum of all tracking angles, and the objective of maximizing the minimum tracking angle.

Corollary 3 *The algorithm* INTERLEAVE *computes a 2-approximation for both* SUMOFANGLES *and* BOTTLENECKANGLE.

It can be showed that the analysis of the approximation ratio of the algorithm is tight as can be shown by the following two examples. In both examples, there are four cameras placed symmetrically on a line ℓ. The two targets are placed in the "middle", t_1 above t_2. The distance between the cameras c_2 and c_3 is a small value ϵ. For SUMOFANGLES consider the left figure of Figure 9. Target t_1 is placed far from ℓ, such that the tracking angle of t_1 is negligible in any camera assignment. Thus, any solution using interleaved pairing incurs cost which is dominated by the tracking angle at t_2, which approaches (as $\epsilon \to 0$) half of the tracking angle of t_2 in an optimum solution which pairs c_1 and c_4, and assigns this pair to t_2. For BOTTLENECKANGLE consider the right figure of Figure 9. Target t_2 is placed at distance $\delta \ll \epsilon$ from ℓ, such that the angle $\angle(c_2 t_2 c_3)$ is bigger than the angle $\theta := \angle(c_1 t_1 c_4)$. In this case, any camera assignment using the interleaved pairing has the minimum tracking angle arbitrary close to $\theta/2$ (as $\epsilon \to 0$), while an optimum camera assignment (which pairs c_1 and c_4 together, and assigns this pair to t_1) has the minimum tracking angle equal to θ.

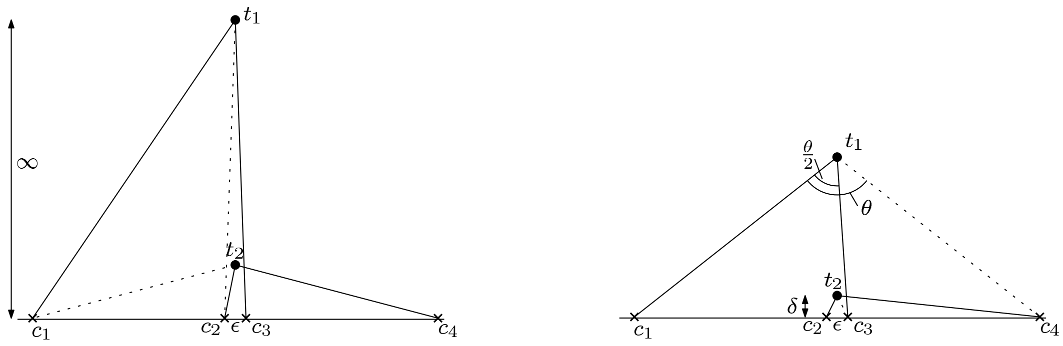

Figure 9: An example showing that the analysis of the algorithm INTERLEAVE is tight (left for SUMOFANGLES, right for BOTTLENECKANGLE)

3.2 Equidistant Cameras on a Line

We consider the special setting where the cameras lie on a (horizontal) line ℓ and the distance between any two neighboring cameras on the line is the same. Without loss of generality we assume unit distance. We consider the problem of maximizing the sum of tracking angles and present a PTAS for this problem.

We consider the cameras in the order as they appear on the line ℓ (from left to right). According to Corollary 2 we know that in every optimum camera assignment the first n cameras are paired with the last n cameras. Let L denote the first n cameras and R the last n cameras. We denote the cameras as they appear in the order on ℓ as $l_1, l_2, \ldots, l_n$ for cameras in L, and $r_1, r_2, \ldots, r_n$ for cameras in R. Hence, the distance between l_1 and r_n is $2n-1$.

The main idea of the algorithm is to partition L and R into k equally-sized sets $L_1, L_2, \ldots, L_k$ and $R_1, R_2, \ldots, R_k$, and to correctly guess what types of paired cameras an optimum solution OPT contains with respect to the partition, i.e., we want to know for every s and t how many pairs of OPT have a camera from L_s and a camera from R_t. Camera pair $\{l_i, r_j\}$ is called a *pair of type* (s,t) (with respect to the partition), if $l_i \in L_s$ and $r_j \in R_t$. There are k^2 different types of pairs. We can characterize every camera assignment by its types of the camera pairs – for each type (s,t) we know how many pairs are of this type. Let $m_{s,t}$ denote this number. For these k^2 numbers $m_{s,t}$, each $m_{s,t}$ is in the range $\{0, \ldots, \frac{n}{k}\}$. According to this classification, there are at most $(\frac{n}{k})^{k^2}$ different classes of camera pairings, which we call *camera-pairing types*. Thus, if k is a constant, the algorithm can enumerate all camera-pairing types in polynomial time. For each enumerated type of camera pairing the algorithm constructs some camera pairing of that type (if that is possible, otherwise it reports that no camera assignment using such a camera pairing type exists) and optimally assigns the targets to the camera pairs. At the end the algorithm outputs the best solution among all constructed camera assignments. The algorithm can create a camera pairing of a type specified by values $m_{s,t}$ in the following way: for every camera-pair type (s,t) it creates $m_{s,t}$ pairs of type (s,t) by pairing $m_{s,t}$ cameras from L_s with $m_{s,t}$ cameras from R_s. Clearly, if a camera pairing of the considered camera-pairing type exists, the algorithm finds one, otherwise the algorithm fails to create one, in which case it continues with the next enumerated camera-pairing.

In the following, we concentrate on the situation where the algorithm considers the same camera-pairing type as the OPT solution has. The algorithm creates some camera pairing of that type, and assigns the camera pairs to the targets in an optimum way by computing a maximum weight matching between the pairs and the targets. Let A denote the resulting camera assignment. We show that the sum of the tracking angles of A is a good approximation of the sum of tracking angles of OPT. We say that a camera pair $\{l_i, r_j\}$ is a *short pair*, if the distance between l_i and r_j is at most $\frac{n}{\sqrt{k}}$, otherwise we say it is a *long pair*. Observe that in any camera assignment there are a lot more long pairs than short ones, as there are at most $n/\sqrt{k}$ short pairs (every short pair

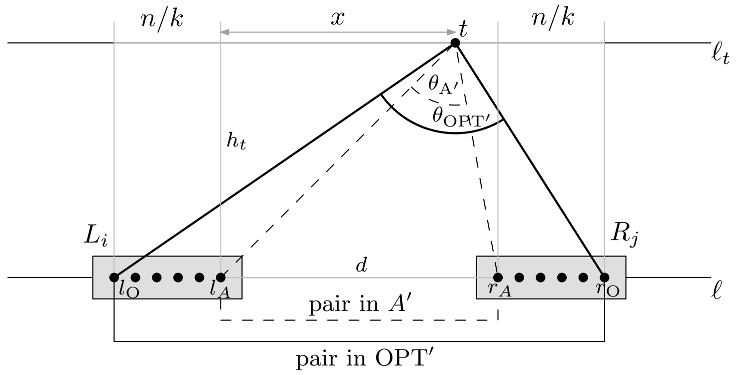

Figure 10: A long pair of type (i, j) in OPT$'$, and a long pair of type (i, j) in solution A'

has to have its left camera among the last $n\sqrt{k}$ cameras in L), and thus at least $n - n/\sqrt{k}$ long pairs. We show that the algorithm guarantees good tracking angles at long pairs. We further show that there exists a solution Q_{OPT} (quasi-optimum) which incurs most of the tracking profit at the long pairs, and which is not much worse than the optimum solution OPT. This then implies that the solution A computed by the algorithm is a good approximation of Q_{OPT}, and therefore it is a good approximation of OPT, too. In the following, if X is a camera assignment, we use $X|\mathrm{LONG}$ to denote the subset of X which consists of long pairs only. By $w(X)$ we denote the weight of X, i.e., the sum of tracking angles arising in X.

Let OPT$'$ denote an optimum solution for the problem of assigning all pairs of OPT to targets, and maximizing the sum of tracking angles at long pairs. Thus, OPT$'$, OPT, and A use the same type of camera-pairing. Consider a long pair $\{l_{\mathrm{O}}, r_{\mathrm{O}}\}$ of type (i, j) from OPT$'$. Let t be the target to which that pair is assigned, and let $\theta_{\mathrm{OPT}'}$ be the tracking angle of t in OPT$'$. See Figure 10 for illustration. Let $\{l_A, r_A\}$ be a (long) pair of type (i, j) that is created by the algorithm. In the camera assignment of A, $\{l_A, r_A\}$ is assigned to some target t'. We create a new camera assignment A' that uses the pairing of A, and the targets of long pairs of OPT$'$ in the following way: we match every long pair $\{l_A, r_A\}$ from A of type (i, j) with a long pair $\{l_{\mathrm{O}}, r_{\mathrm{O}}\}$ from OPT$'$ of the same type. Then, let $\{l_{\mathrm{O}}, r_{\mathrm{O}}\}$ be assigned to target t. We create A' by assigning the matched pair $\{l_A, r_A\}$ to t. Clearly, A' is a camera assignment of the same type as OPT' and A. Clearly, as A and A' are using the same pairs, the solution of A is at least as good as A', i.e., $w(A) \geq w(A')$. We now show that A' is a good approximation for OPT' on long pairs.

Let $\theta_{A'}$ denote the tracking angle of t in A'. We show that $\theta_{A'}$ is not much smaller (if at all) than $\theta_{\mathrm{OPT}'}$. Clearly, the worst case for the difference between the two angles is when l_{O} is the leftmost vertex in L_i, r_{O} is the rightmost vertex in R_j, and l_A is the rightmost vertex in L_i and r_A is the leftmost vertex in R_j. The distance between l_{O} and l_A is at most n/k (the size of L_i). Similarly for r_{O} and r_A, the distance between these two cameras is at most n/k. On the other hand, the distance between l_A and r_A is at least $n/\sqrt{k}$, because $\{l_A, r_A\}$ is a long pair. We can express the ratio $\theta_{A'}/\theta_{\mathrm{OPT}'}$ in the following way. Let x express the position of t on the line ℓ_t parallel to ℓ through t. We assume that $x = 0$ when t is exactly above l_A. Further, we denote by h_t the distance between ℓ and ℓ_t and by d the distance between l_A and r_A (cf. Figure 10). We can then express the ratio $\theta_{A'}/\theta_{\mathrm{OPT}}$ as follows:

$$\frac{\theta_{A'}}{\theta_{\mathrm{OPT}'}} = \frac{\arctan\left(\frac{x}{h_t}\right) + \arctan\left(\frac{d-x}{h_t}\right)}{\arctan\left(\frac{n/k+x}{h_t}\right) + \arctan\left(\frac{d+n/k-x}{h_t}\right)} \tag{1}$$

The analysis of the first and second derivative with respect to x of the previous function shows that $\theta_{A'}/\theta_{\mathrm{OPT}'}$ is minimal for $x = d/2$, i.e., when the target lies in the middle point of the segment

(l_A, r_A). Hence, by setting $x = d/2$ in (1) we get the following lower bound:

$$\frac{\theta_{A'}}{\theta_{\mathrm{OPT}'}} \geq \frac{\arctan\left(\frac{d}{2h_t}\right)}{\arctan\left(\frac{n}{kh_t} + \frac{d}{2h_t}\right)}.$$

We now distinguish two cases. First, if $h_t = o(n)$, then the two arguments inside the arctan functions of the previous term are unbounded as n grows (remember that $d \geq n/\sqrt{k}$). As arctan is a bounded function, $\theta_{A'}/\theta_{\mathrm{OPT}'}$ approaches 1 as n gets large. Therefore, given any ϵ and k, we can find n large enough, such that $\theta_{A'}/\theta_{\mathrm{OPT}'} \geq 1 - \epsilon$, as desired. Second, if $h_t = \Omega(n)$, the arguments inside the arctan functions of the previous term are bounded from above and may even approach zero as n goes to infinity (remember that $d \leq 2n - 1$), so we have to examine the behavior of the term in this case, and, as we will see, we also have to involve k. We denote $\alpha := \frac{d}{2h_t}$, and thus obtain $\theta_{A'}/\theta_{\mathrm{OPT}'} \geq \frac{\arctan(\alpha)}{\arctan\left(\frac{n}{kh_t}+\alpha\right)}$. We express the term $\frac{n}{kh_t}$ in terms of α (remember that $d \geq n/\sqrt{k}$):

$$\frac{n}{kh_t} = \frac{2dn}{2dkh_t} = \frac{2n}{dk}\alpha \leq \frac{2}{\sqrt{k}}\alpha.$$

Thus we have $\theta_{A'}/\theta_{\mathrm{OPT}'} \geq \frac{\arctan(\alpha)}{\arctan\left((1+\frac{2}{\sqrt{k}})\alpha\right)}$. The derivative of the last fraction with respect to α is positive for any $\alpha \geq 0$, and thus the last fraction is minimized when α approaches zero. The limit of that fraction, when α approaches zero, is $1/(1 + \frac{2}{\sqrt{k}}) = \frac{\sqrt{k}}{\sqrt{k}+2}$, and thus, for any fixed ϵ, setting k appropriately, and for n large enough, we get that $\theta_{A'}/\theta_{\mathrm{OPT}'} \geq 1 - \epsilon$.

In the remaining we show that there exists a camera assignment Q_{OPT} with the same camera-pairing type as OPT, which gains most of its profit (i.e., a $(1 - \delta)$ fraction of the total profit, $0 < \delta < 1$) on long pairs, and which is a good approximation to OPT. The optimum solution OPT has at most $n/\sqrt{k}$ short pairs. Let S denote the subset of OPT that contains the short pairs only. Let us consider those long pairs of OPT, where every long pair, considered as an interval on ℓ, fully contains every short pair. Let G denote the subset of OPT on such pairs. Observe that there are at least $n - 2n/\sqrt{k}$ pairs in G (there are at least $n - n/\sqrt{k}$ long pairs that contain the left camera of any short pair – the pairs formed by cameras $l_1, l_2, \ldots, n - n/\sqrt{k}$; among those pairs, at most $n/\sqrt{k}$ can be formed by cameras $r_1, r_2, \ldots, r_{n/\sqrt{k}}$). We split G into subsets of size $|S|$. Let $G_1, G_2, \ldots, G_z$ denote these sets, where $z = |G|/|S| \geq \sqrt{k} - 2$. For simplicity we assume that $|G|$ is divisible by $|S|$. As one can check, this is not a crucial assumption in our analysis. Let us order the sets G_i such that $w(G_1) \geq w(G_2) \geq \ldots w(G_z)$.

Observe first that if $w(G_z) \geq w(S)$, we get $w(\mathrm{OPT}) \geq \sum_i w(G_i) \geq z \cdot w(S)$, and thus $\frac{w(\mathrm{OPT})}{z} \geq w(S)$. As $w(\mathrm{OPT}|\mathrm{LONG})$ denotes the contribution of all long pairs to the total weight $w(\mathrm{OPT})$, we obtain $w(\mathrm{OPT}|\mathrm{LONG}) = w(\mathrm{OPT}) - w(S) \geq w(\mathrm{OPT}) - \frac{w(\mathrm{OPT})}{z} \geq \frac{z-1}{z} w(\mathrm{OPT}) \geq \frac{\sqrt{k}-3}{\sqrt{k}-2} w(\mathrm{OPT})$. Hence, setting k appropriately, OPT gains a $(1 - \delta)$ fraction of its profit on long pairs, and thus we can set $Q_{\mathrm{OPT}} := \mathrm{OPT}$.

Assume now that $w(S) > w(G_z)$. We create a new solution Q_{OPT}: we (arbitrarily) assign the pairs of G_z to targets of S and the pairs of S to targets of G_z. Let S' and G'_z denote the new assignments. Clearly, as $w(Q_{\mathrm{OPT}}) \leq w(\mathrm{OPT})$, we have $w(G'_z) + w(S') \leq w(G_z) + w(S)$, as only pairs in G_z and S have possibly been assigned to different targets. Observe now that $w(G'_z) > w(S)$, because in Q_{OPT} the pairs of G_z are assigned to the same targets as the pairs of S in OPT, and every pair from G_z fully contains every pair from S (if imagined as an interval), which makes every tracking angle of the respective target bigger in Q_{OPT}. Thus, using the last two inequalities, $w(S') < w(G_z)$. Hence, we also have that $w(S') < w(G_i)$, $i = 1, \ldots, z - 1$, and thus $w(S') \leq \frac{w(\mathrm{OPT})}{z}$. Applying the argumentation from above we get $w(Q_{\mathrm{OPT}}|\mathrm{LONG}) = w(Q_{\mathrm{OPT}}) - w(S') \geq \frac{z-1}{z} w(Q_{\mathrm{OPT}})$. Observe also that $w(Q_{\mathrm{OPT}}) \geq w(\mathrm{OPT}) - w(G_z)$. Thus, since

$w(\mathrm{OPT}) - w(G_z) \geq \frac{z-2}{z-1} w(\mathrm{OPT})$, we obtain $w(Q_{\mathrm{OPT}}) \geq \frac{z-2}{z-1} w(\mathrm{OPT})$, and hence $w(Q_{\mathrm{OPT}}|\mathrm{LONG}) \geq \frac{z-2}{z} w(\mathrm{OPT})$.

Putting the previously derived inequalities together, we obtain $w(A) \geq w(A') \geq (1-\epsilon) w(\mathrm{OPT}'|\mathrm{LONG}) \geq (1-\epsilon) w(Q_{\mathrm{OPT}}|\mathrm{LONG}) \geq (1-\epsilon)\frac{z-2}{z} w(\mathrm{OPT}) \geq (1-\epsilon)\frac{\sqrt{k}-4}{\sqrt{k}-2} w(\mathrm{OPT})$. Hence, for any given ϵ^*, we can find ϵ and k such that $(1-\epsilon)\frac{\sqrt{k}-4}{\sqrt{k}-2} \geq 1-\epsilon^*$, and thus $w(A) \geq (1-\epsilon^*)\mathrm{OPT}$. This yields:

Theorem 3 *There is a PTAS for* SumOfAngles *with equidistant cameras on a line.*

4 Conclusions

We have considered different variants of the "focus of attention" problem, where the objective function depends on the tracking angles. We have shown that the natural goal of assigning targets under 90° is (in general) an NP-complete problem. It remains an open problem whether the more restricted instances where the cameras are placed on a line can be solved in polynomial time. The hardness result shows that there is no approximation algorithm (unless P = NP) for the problem of minimizing the sum of deviations of tracking angles from 90°. In this context it would be interesting to consider different optimization goals which capture the optimality of tracking angles θ_i being 90° and which would allow a good approximation. The first candidate for such an objective could be $\sin\theta_i$. Any results for this or similar objective function would be interesting. Also for the objective functions considered in this paper there are unresolved questions. For example, we have only considered SumOfAngles on a line. For the general case, a simple greedy algorithm achieves a 3-approximation [4], and it remains open whether one could do better.

References

[1] Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. Second edn. Cambridge University Press, ISBN: 0521540518 (2004)

[2] Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Computing Surveys **38** (2006)

[3] Isler, V., Khanna, S., Spletzer, J., Taylor, C.J.: Target tracking with distributed sensors: the focus of attention problem. Computer Vision and Image Understanding **100** (2005) 225–247

[4] Spieksma, F.: Multi index assignment problems: complexity, approximation, applications. In: Nonlinear Assignment Problems. Kluwer (2000) 1–12

[5] Spieksma, F., Woeginger, G.: Geometric three-dimensional assignment problems. European Journal of Operational Research **91** (1996) 611–618(8)

[6] Goossens, D., Spieksma, F.: On the FOA-problem. Manuscript (2004)

[7] Arkin, E.M., Hassin, R.: On local search for weighted k-set packing. Mathematics of Operations Research **23** (1998) 640–648

[8] Galil, Z.: Efficient algorithms for finding maximum matching in graphs. ACM Comput. Surv. **18** (1986) 23–38

[9] Lawler, E.L.: Combinatorial Optimization: Networks and Matroids. Holt, Rinehart and Winston, New York (1976)