**Journal of Mathematical Imaging and Vision manuscript No.**
(will be inserted by the editor)

# SCOOP: A Real-Time Sparsity Driven People Localization Algorithm

**Mohammad Golbabaee · Alexandre Alahi · Pierre Vandergheynst**

Received: date / Accepted: date

**Abstract** Detecting and tracking people in scenes monitored by cameras is an important step in many application scenarios such as surveillance, urban planning or behavioral studies to name a few. The amount of data produced by camera feeds is so large that it is also vital that these steps be performed with the utmost computational efficiency and often even real-time. We propose SCOOP, a novel algorithm that reliably localizes people in camera feeds, using only the output of a simple background removal technique. SCOOP can handle a single or many video feeds. At the heart of our technique there is a sparse model for binary motion detection maps that we solve with a novel greedy algorithm based on set covering. We study the convergence and performance of the algorithm under various degradation models such as noisy observations and crowded environments, and we provide mathematical and experimental evidence of both its efficiency and robustness using standard datasets. This clearly shows that SCOOP is a viable alternative to existing state-of-the-art people localization algorithms, with the marked advantage of real-time computations.

**Keywords** People Localization · Sparse Representation · Dictionary · Multi-View · Greedy Algorithm · Matching Pursuit · SCOOP · Group Testing

## 1 Introduction

The present paper deals with a simple but very important problem in computer vision: given a set of cameras observing a scene (there can be only one camera as an extreme example), we want to automatically detect pedestrians and locate them in the scene. The detection output would generally be used in a second step for tracking people, but we focus here on the detection/localization problem as it appears to be the main bottleneck (most computational complexity) of the whole system. This problem has numerous applications, surveillance being the most obvious, and has been the subject of intense research over the past decade. However, there remain two important challenges to most existing solutions:

- Robustness: due to occlusions and variable lighting conditions, existing algorithms tend to produce false or miss detections. Often, robustness is achieved at the expense of computationally complex scene modeling.
- Computational complexity: cameras operating at 25 frames per second or more generate tremendous amount of data. In order to achieve a real-time performance, existing algorithms have to sacrifice on robustness.

This trade-off between robustness and computational efficiency places unbearable constrains on real-world applications where both are desirable. The objective of this paper is thus to propose a solution to the people detection problem that would be at the same time robust and computationally efficient.

In a previous paper, we have proposed a model of motion detection maps based on the assumption that the number of people in the scene is much smaller than the total possible ground locations [1]. The model was relaxed into a Lasso-like problem [26] and solved with a re-weighted $\ell_1$ algorithm [5]. We showed that the resulting technique, deemed O-Lasso, reached state-of-the-art performances in

Part of this work was funded by the EU under research projects FET-OPEN 225913 (SMALL), ICT-216023 (APIDIS), and by the Swiss National Science Foundation under grant 200021-117884.

M. Golbabaee, A. Alahi, P. Vandergheynst
Signa Processing Lab (LTS2), EPFL, Station 11
CH-1015 Lausanne, Switzerland
Tel.: +41-21-6932657
E-mail:
{mohammad.golbabaei, alexandre.alahi, pierre.vandegheynst}@epfl.ch

terms of robustness. Unfortunately, O-Lasso is a computationally complex algorithm and, despite various optimizations, cannot reach a real-time operation. Acknowledging that the excellent robustness properties reported in [1] were due to the sparsity hypothesis, we conserve that part of the model but we propose a completely different way of exploiting it. First, where O-Lasso was based on complex floating point calculations, we derive a new regression model that involves only boolean arithmetics and takes full advantage of the binary output of basic motion detection algorithms. Second, instead of solving a difficult convex optimization problem with iterative shrinkage, we derive a novel greedy algorithm inspired by the set cover problem. This algorithm uses only binary operations and is therefore extremely efficient.

The relevance and performance of our model and algorithm are analyzed at two different levels. First, we draw a connection with group testing that allows us to study the mathematical properties of the model and sate the existence and uniqueness of solutions. We also show that these solutions can be recovered by a simple thresholding algorithm. Second, we extend these findings and propose a greedy heuristic that also incorporates physical constrains on the localization of detected people, the resulting algorithm is called Set Covering Object Occupancy Pursuit or SCOOP. We then study experimentally its performances: SCOOP matches O-Lasso in terms of robustness but at a fraction of the computational cost, easily reaching a real-time implementation.

## 2 Related Work

As hinted at above, the problem of detecting and localizing people in networks of camera has been the subject of an intense research activity. Let us review the main approaches leading to our own model. Detection can occur independently in each camera then fused across cameras [3, 25], or they can be detected concurrently in a unique referential [6, 21] since cameras are calibrated to match 3D points across image planes [20]. These approaches face difficulties to detect people that occlude each other and a good alternative is to fuse features extracted from all cameras in a unique referential and make the decision once all features are combined. The most commonly used features are the silhouettes extracted from all cameras using a motion detection algorithm and a reference background image. In [17] Khan and Shah project foreground silhouettes on a reference ground-plane given a global homography. By stacking and normalizing the obtained re-projected silouhettes, they construct a probability map. Alternatively probability maps over several planes can be estimated [8, 18], which provides more robustness. Eshel and Moses in [12] use a probability map at head level, but when the number of cameras is small the segmentation of grouped people becomes poor. Closer to our model, Fleuret *et al.* in [2, 13] use a dictionary of ideal silhouettes that is used to detect people in crowded environment. Rectangular shape prototypes are used to approximate the ideal human foreground silhouettes observed by the cameras. They then estimate the probability of occupancy map (POM) of the ground plane at each time. Recently, Alahi *et al.* in [1] have also proposed a dictionary based framework with a generative model to approximate foreground silhouettes and their model is the starting point of our investigations. The localization of people in the monitored scene arises as the solution to an inverse problem referred to as O-Lasso. It outperforms previous approaches in terms of detection rate but it is computationally costly for real-time applications. As a result, we propose in Section 3 a greedy approach that achieves the same detection rate but with a real-time performance.

## 3 Dictionary-Based Boolean Regression Model

Alahi *et al.* in [1] propose a sparsity driven framework that performs well with respect to the state-of-the-art. A key feature of this scheme is to cast the multi-view localization as a sparse linear inverse problem that is followed by a quantization step. A huge collection of silhouettes of an individual standing at various positions, that is called dictionary, is used for this purpose. Later, inspired by the recent massive developments in sparse linear approximation tools, an algorithm was proposed (O-Lasso) to approximate the foreground silhouettes by a small number of individuals. The main drawback of these schemes is their numerical complexity: they are based on iterative algorithms that converge slowly.

Our proposed approach is similar to the dictionary-based framework, however, we use a boolean (non-linear) regressive formulation to model the localization problem. Using boolean arithmetics, we design much faster and memory efficient approximation algorithms that are mainly rooted in the old literature of group testing and set cover. The following steps precisely describe our problem formulation:

### *3.0.1 Scene Discretization*

The ground plane is discretized into 2-D grid of $N$ cells (subregions). We assume each cell can be occupied by only one person at each time instance. To simplify notations, the 2-D grid is concatenated into a 1-D vector $x \in \{0,1\}^N$, whose elements are indicating the presence of a person in the corresponding cell (with Id= $i$) if $x_i = 1$. Typically, we refer to this vector as the *occupancy vector*. An adaptive sampling process can be used to discretize the ground plane into non-regularly spaced grid points to take advantage of the cameras' topology and the scene activity [1].

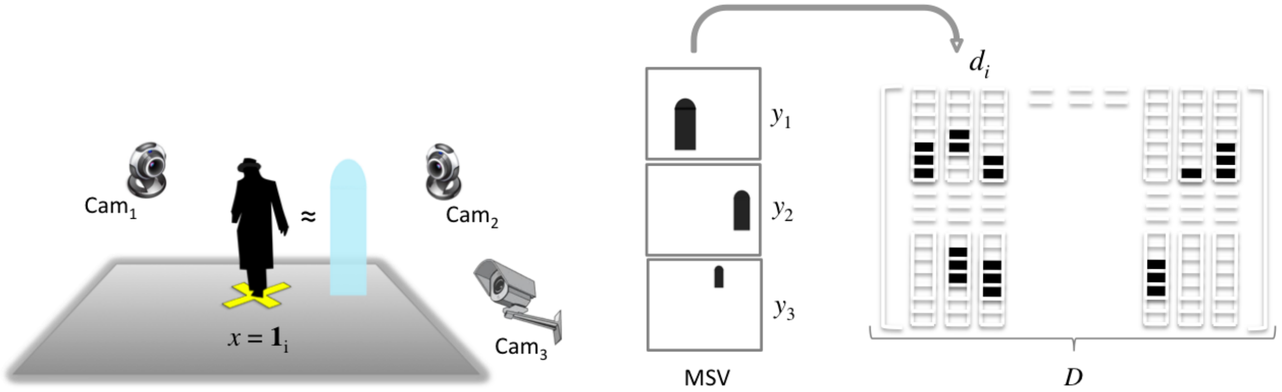


**Fig. 1** Dictionary reconstruction: an atom/column $d_i$ corresponds to the MSV of a half-rectangular-half-elliptical object approximating a person standing at position Id $= i$.

### *3.0.2 Foreground Silhouettes Arrangement*

A 2-D binary mask representing the foreground pixels observed by each camera $c$, is extracted given a background subtraction algorithm [22]. We used the well-known mixture of Gaussians to classify each pixel as foreground [24]. The foreground images are also rearranged by concatenation of the 2-D masks into binary vectors $y_c \in \{0,1\}^{M_c}$. Each of these vectors contain the extracted foreground silhouettes from the corresponding camera $c$. A foreground silhouette is a connected region of foreground pixels. As a result, $M_c$ denotes the resolution (number of pixels) of the $c^{th}$ camera. Further, we concatenate all these vectors into the Multi-Silhouette Vector (MSV):

$$y = (y_1^T, ..., y_C^T)^T \in \{0,1\}^M, \tag{1}$$

where, $M = \sum_{c=1}^{C} M_c$.

### *3.0.3 Dictionary Construction*

Imagine a person with a given volume walking in a scene. The shape observed by each camera can vary depending on the view-points and the behavior of that person. We approximate the shape of people with a half-rectangular-half-elliptical shape as in [1] (see Figure 1). We consider an average person with a height of 170 cm. A 3-D model is projected into all the camera views given the calibration data [16]. Such approximation is used to construct atoms of a dictionary $D \in \{0,1\}^{M \times N}$. Each column of the dictionary, i.e. each atom, represents the approximated multi-view silhouette observed at a given ground plane point. A typical dictionary contains a huge collection of atoms i.e., there is as many columns as the discretized ground plane points in the scene. Figure 1 demonstrates reconstruction of an atom of the dictionary used in people detection applications. Note that in urban scenes, dictionary atoms can be modified so that they can also represent approximate MSVs of various urban objects e.g., pedestrians, cars, buses, trucks, etc.

## 3.1 Boolean Regression Model

Suppose a single person occupies the scene. This refers to an occupancy vector $x$ with only one nonzero element whose index depends on the location of that person in the scene. Moreover, each of the cameras will capture only one silhouette (if they have a view over the position) whose size, position and possibly shape depends on the location. In more general cases, for a given configuration of $x$ with more nonzero elements (depending on the number of people and their positions), the resulting MSV may not be necessary unique. This non-uniqueness comes from the occlusions in the camera view which are highly dependent on the density of the crowd and the positions of the cameras.

Let us define the following *regression model* which describes the underlying correspondence between each occupancy vector and its resulting MSV:

$$y = D \cdot x \oplus z. \tag{2}$$

Note that here the operations are boolean i.e., sums and products correspond to OR and AND, and $\oplus$ denotes the bitwise XOR operation between two boolean vectors. The dictionary $D \in \{0,1\}^{M \times N}$, as previously defined, is a very huge matrix of silhouettes. Each column of $D$, say $d_i$, indicates the corresponding MSV of an average person who is standing at position $i$ in the scene. Finally, $z \in \{0,1\}^M$ denotes the noise vector that corrupts the MSV by both missing and extra foreground pixels. This may occur due to several reasons e.g., non ideal silhouette extraction, non ideal modeling of the dictionary atoms, shadows, reflections, etc. Figures 2(a)-2(d) provide an illustration of the regression model described by Equation (2).

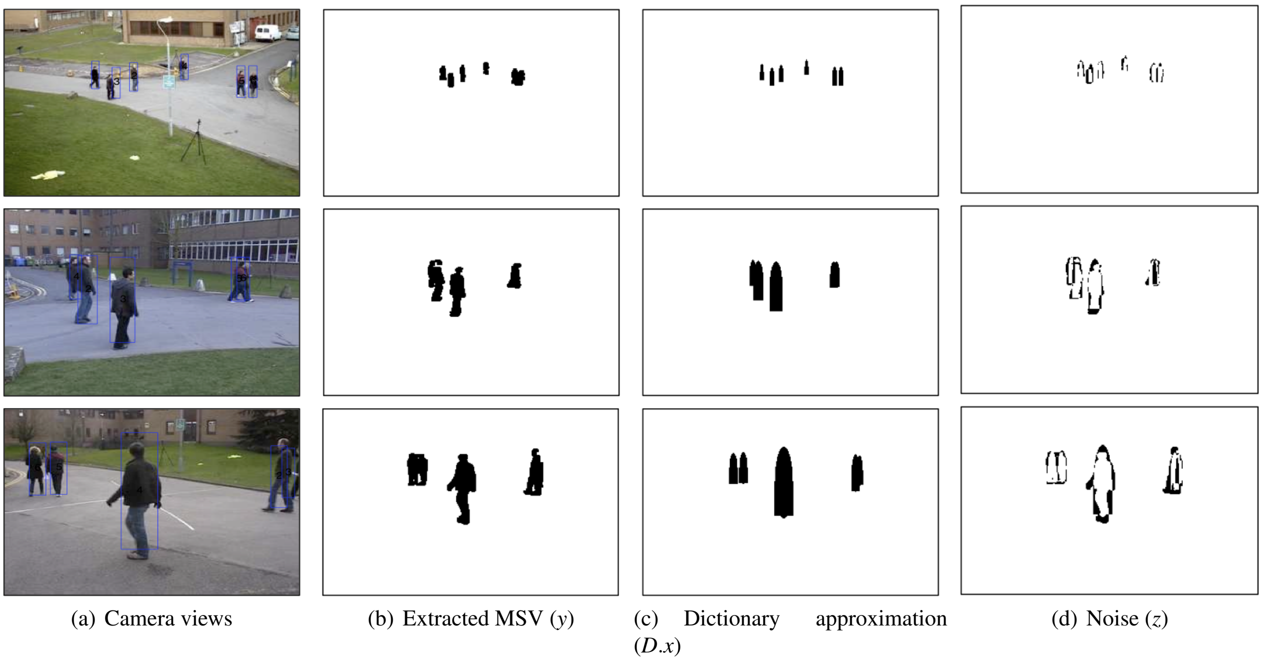

(a) Camera views (b) Extracted MSV ($y$) (c) Dictionary approximation ($D.x$) (d) Noise ($z$)

**Fig. 2** Illustration of the boolean regression model described by equation (2) for three camera views of a single frame of the PETS 2009 dataset.

Assume an occupancy vector $x$ representing $k$ individuals in a scene. The support of $x$ is the set $\mathscr{S}$ that contains indices of the $k$ nonzero elements,

$$\mathscr{S} = \text{supp}(x) := \{i : x_i = 1\}.$$

Equation (2) formulates the observed MSV as the boolean superposition of $k$ atoms (bitwise OR among the columns) of the dictionary, indexed by $\mathscr{S}$ and possibly corrupted by some noise,

$$y = \sum_{i \in \mathscr{S}} d_i \oplus z. \tag{3}$$

Recalling that each atom represents the silhouette of a single individual at a certain location, the use of the boolean operators in Equations (2) and (3), explicitly demonstrates the nonlinearity of the MSV model caused by the occlusions in scenes with more than a single person. We use later this model frequently, specially as one of the most important priors in order to infer the locations of the individuals given their foreground MSV.

## 4 Problem Statement and Connections with Group Testing

Considering descriptions in the previous section, the problem of detecting and localizing objects in a scene is equivalent to recover an occupancy vector from an inaccurate noisy MSV, provided with the knowledge of the dictionary $D$ that links them through Equation (2). Accordingly, we formulate people detection and localization as a non-linear inverse problem, in which one needs to identify the support set $\mathscr{S}$ that approximately leads to the observed MSV, even in presence of noise (see Equation (3)).

As one can observe, noise together with the non-linearity of the formulation can impose many different possible solutions to Equation (2). Hence, different occupancy vectors may result in the same MSV. This fact severely challenges the performance of any decoding algorithm to reliably detect and localize the objects e.g., a decoder may mistakenly add or neglect some individuals.

In the next part, we determine necessary and sufficient conditions in order to preserve the uniqueness of the solutions. Our theoretical analysis finds interesting and intuitive implications in multi-view people detection and localization problem. Moreover, a simple algorithmic approach called *Thresholding*, is introduced to recover the occupancy vector from the MSV and we theoretically show that it performs optimally i.e., if there exists a unique solution, Thresholding will recover it. We develop our results based on some popular existing tools in the well-established *group testing* literature and therefore we show how these two problems are related to each other.

The classical group testing problem which was introduced by Dorfman [9] finds its historical roots in World War II, when blood samples of many U.S. soldiers were examined to detect few cases of syphilis. The main idea is to pool the blood samples into certain groups and test the groups instead of one by one testing. The original problem can be formulated as Equation (2) in the noiseless case, where, $x$ contains $N$ blood samples with sparse nonzero elements indicating the infected cases and $D$ (called the *contact* matrix) determines the way of collecting $M < N$ group tests into a vector $y$. The question is, how to design a contact matrix and a recovery algorithm that can efficiently identify as many defective cases as possible? For more details see [10].

In our localization application, the design of the dictionary is however fixed by the number of the cameras, their relative positions, silhouette model and the density of the points in the scene. Thus, contrary to the group testing, our application deals mainly with the recovery problem rather than compression. Another difference is that the cameras are typically providing multi-view images with much higher resolution than the number of grid points in the scene i.e., $M \gg N$, which may help us compensate for the rather non-optimal design of the dictionary.

### 4.1 Uniqueness of the Representation

Given the regression model (2), there can be many realizations of the occupancy vector leading to the same MSV $y$, which makes any decoding scheme hopeless to recover the original $x$. There are two main reasons for this non-uniqueness: the presence of noise and non-linearity of the formulation. Particularly, in people detection applications, full occlusions may occur due the relative placement of the cameras and the people in the scene. Therefore, any decoder fails to decide correctly whether there are some individuals present at certain positions. This section defines precisely a set of conditions that avoids such uncertainties and guarantees the uniqueness of the solution to (2). These constraints provide an upperbound on the performance of any recovery scheme and in addition, they measure how efficiently they do perform. In the following, we define the notion of *disjunct* matrices that is often used in group testing literature [10] and it appears to be the key element of the theoretical framework that we establish in this paper.

**Definition 1** *A boolean matrix $D$ with $N$ columns $d_1,\dots,d_N$ is $(k,e)$-disjunct if for every subset $\mathscr{S} \subseteq \{1,\dots,N\}$ with cardinality $|\mathscr{S}| \le k$, and every $i \notin \mathscr{S}$, we have:*

$$\left|\operatorname{supp}(d_i)\setminus \bigcup_{j\in\mathscr{S}} \operatorname{supp}(d_j)\right| > e,$$

*where $|.|$ represents cardinality of a set, and $.\backslash.$ means set difference.*

Assuming an occupancy vector that is $k$-sparse ($x$ has less than or equal to $k$ nonzero elements) and a noise that flips at most $e$ bits of MSV i.e. $|\operatorname{supp}(z)| \le e$, the following proposition guarantees the uniqueness of the solution to (2):

**Proposition 1** *For any $(k,2e)$-disjunct dictionary Equation (2) implies a one-to-one mapping between all $k$-sparse $x$ and their corresponding $y$. Conversely, if every $k$-sparse $x$ is mapped to a distinct $y$ then $D$ must be $(k$-1, $2e)$-disjunct.*

*Proof* Assume two different $k$-sparse boolean vectors $x$ and $x'$ that are supported on sets $\mathscr{S}$ and $\mathscr{S}'$. Let $\bar{y}$ and $\bar{y}'$ be correspondingly the bitwise OR of the columns of $D$ indexed by $\mathscr{S}$ and $\mathscr{S}'$ i.e., $\bar{y} = D.x$ and $\bar{y}' = D.x'$ with boolean arithmetic. Choose a column of the dictionary $d_i$ so that, $i \in \mathscr{S}'$ but not in $\mathscr{S}$. Since $D$ is $(k,2e)$-disjunct, it implies that the support of $d_i$ has $2e+1$ elements that are not included in $\operatorname{supp}(\bar{y})$. Therefore, assuming noises that flip $e$ bits of $d_i$ and $e$ bits of $\bar{y}$, there will be still at least one element of $\operatorname{supp}(d_i)$ that is not included in the support of $y$ (recall that $y = \bar{y} \oplus z$), which makes $x$ and $x'$ distinguishable from their noisy MSVs.

For the converse, suppose $D$ is not a $(k$-1, $2e)$-disjunct matrix and pick a pair of a set $\mathscr{S} \subseteq [n]$ with $|\mathscr{S}| \le k-1$, and an index $i \notin \mathscr{S}$ that is a counterexample to the $(k$-1$, 2e)$-disjunctness. Assume vectors $x$ and $x'$ that are supported on $\mathscr{S}$ and $\mathscr{S} \cup \{i\}$ correspondingly. The noise can configure in an adversarial way so that, by flipping $e$ zero bits of $\bar{y} = D.x$ to one and $e$ one bits of $d_i$ to zero, the MSV outcome of $x$ and $x'$ becomes indistinguishable.

□

In our people detection application, disjunction is a measure of robustness against occlusions and noise. Intuitively, Proposition 1 implies that any person at any position must have a silhouette with enough distinguishable pixels to be robust against the noise and not be submerged (occluded) into the silhouettes of other people.

By increasing the number of people in the scene, the occlusions become more probable, thus, Proposition 1 can also be interpreted as an upperbound for the number of individuals that can be reliably localized by a fixed camera setup. Moreover, by changing the camera setup e.g., increasing the number of the cameras and properly selecting their positions with respect to the scene, one can design optimal dictionaries that are disjunct for larger number of people.

Note that in general, verifying whether a matrix is $(k,e)$-disjunct is a hard problem, which makes Proposition 1 impractical for large size setups. Moreover, Proposition 1 provides a *worst case* analysis for (2), since it guarantees the uniqueness for *all* occupancy vectors and it is robust against any *adversarial* noise setup. In practice, simulation results indicate a reliable recovery under much milder conditions than in Proposition 1, because the worst case situations are not very likely.

### 4.2 Decoding by Thresholding

In this part we introduce a simple algorithm for recovering the occupancy vector from the corresponding MSV. Note that a similar approach has been considered in [7], and in [23] and [15] for real-valued $x$. This algorithm works based on selecting atoms of $D$ whose supports are approximately included in $\text{supp}(y)$, that is the number of elements of $\text{supp}(d_i)$ not included in $\text{supp}(y)$ should not exceed a threshold.

**Thresholding:** *Select the columns of $D$ that satisfy the following equation:*

$$\Big|\text{supp}(d_i)\backslash\text{supp}(y)\Big| \leq e, \tag{4}$$

*and indicate their corresponding indices as the support of $x$.*

The following theorem characterizes the performance of Thresholding and highlights its optimality for solving the regression model (2) when $|\text{supp}(z)| \leq e$:

**Theorem 1** *Thresholding successfully recovers any $k$-sparse occupancy vector, if $D$ is $(k,2e)$-disjunct.*

*Proof* Assume a boolean vector $x$, supported on the set $\mathscr{S} \subseteq \{1,\ldots,N\}$ with $|\mathscr{S}| \leq k$. Since the noise has flipped at most $e$ bits of $y$, obviously every $i \in \mathscr{S}$ satisfies (4).

In addition, define $\bar{y}$ to be the bitwise OR of the columns of $D$ indexed by $\mathscr{S}$ i.e., the noiseless version of the MSV. Again by the assumption on the noise power, for any column of $D$ we have,

$$|\text{supp}(d_i)\backslash\text{supp}(\bar{y})| \leq |\text{supp}(d_i)\backslash\text{supp}(y)| + e.$$

Now, if any $i \notin \mathscr{S}$ satisfies (4), it implies $|\text{supp}(d_i)\backslash\text{supp}(\bar{y})| \leq 2e$, which violates the assumption that the dictionary is $(k,2e)$-disjunct. Therefore, thresholding recovers exactly the support of $x$.

□

Theorem 1 shows that Thresholding achieves the optimal bound of Proposition 1 as long as there is an unique solution to Equation (2). In addition, note that if the dictionary is not $(k,2e)$-disjunct then there may exist column indices $i \notin \text{supp}(x)$ that also satisfy (4), and therefore the recovery is not exact but contains the support of the original occupancy vector i.e., $\text{supp}(x) \subset \text{supp}(\widehat{x})$, where $\widehat{x}$ denotes the recovered occupancy vector.

In our application, the highest value $k$ to have a $(k,2e)$-disjunct dictionary is much smaller than the number of potential individuals in the scene. As a result, for typical populated scenes, the occlusions become more probable and therefore many different realizations of the occupancy vector solve (2) (no unique solution). In this case, by setting correctly the threshold value in (4), the algorithm does not miss any individual, but its performance is dramatically affected by many false positives.

In the next section, we consider an additional prior to better recover the exact solution. We assume that among all possible solutions, the one of interest is the sparsest one, and we design a real-time algorithm in order to recover it. Nevertheless, we keep taking advantage of the output of Thresholding as a very fast preprocessing step which efficiently refines the search space, and thus, accelerates the main step of the recovery algorithm.

## 5 Real-Time Sparsity Driven People Localization

In practice, the configuration of the cameras and the density of the people in the scene are such that full occlusions are inevitable and therefore, there is no unique solution to the localization problem. As an example, there might be many positions on scene, entirely covered by the people near to the cameras so that no decoder would be able to decide whether there are some individuals hidden there or not. The Thresholding algorithm defined in the previous section outputs a conservative solution that considers all those points as if they are occupied by people and thus it results in too many false positives that are not desired.

In this section and according to our hypothesis regarding the distribution of the individuals in the scene, we address this problem by selecting the *sparsest* solution to (2), i.e. among all possible solutions, we set our problem to the recovery of the sparsest occupancy vector $x$. In the noiseless case, we thus solve

$$\begin{aligned} \widehat{x} = \underset{x\in\{0,1\}^N}{\arg\min}\ & |\text{supp}(x)| \\ \text{subject to}\quad & \text{supp}(y) = \text{supp}(D\cdot x) \end{aligned} \tag{5}$$

and, in the noisy setup,

$$\begin{aligned} \widehat{x} = \underset{x\in\{0,1\}^N}{\arg\min}\ & |\text{supp}(x)| \\ \text{subject to}\quad & \Big|\text{supp}(y \oplus D\cdot x)\Big| \leq e. \end{aligned} \tag{6}$$

Note that, if disjunction holds, the solution of both problems coincide with the unique solution of (2) which can be simply identified by Thresholding. However, if $D$ is not $(k,2e)$-disjunct, this new approach neglects objects that might be fully occluded, and approximates the MSV by very few number of atoms of $D$ corresponding to the large-size silhouettes (i.e., people who are mainly in front of the scene).

Problem (5) is equivalent to the well-known *set cover* problem [27], which recovers the set of atoms with minimal cardinality, so that the union of its elements covers the support of $y$. In the noisy case (6), however, we relax the constraint since we are not interested in approximating the noise. Set cover is known as one of Karp's 21 NP-complete problems, thus designing feasible algorithms that can approximate the solution with polynomial time complexity is of high importance.

It has been shown in [27] that the simple *greedy* approach is indeed an effective way to approximate the solution of the set cover problem. This approach follows the heuristic of making the locally optimal choice at each iteration with the hope of finding the global optimum. For example, the greedy method proposed in [27] works iteratively and recovers one element of the support set (i.e., $\mathrm{supp}(x)$) per iteration. More precisely, at each iteration, the algorithm selects the index of the atom $i$ of the dictionary which contributes the most in energy of the MSV $y$ (i.e. the atom whose support shares the most common elements with $\mathrm{supp}(y)$). It then subtracts its contribution to update the remainder (initially MSV). This procedure continues until meeting the stopping criteria. This criteria can be either a prior knowledge on the sparsity level, or a threshold on the energy level of the remainder. For the first criteria the algorithm performs $k$ iterations and for the latter it runs until the remainder energy falls below a limit $e$.

It is noteworthy to mention that the same approach is extensively used in the compressed sensing and sparse approximation research literature because of the simplicity of the analysis and low computational cost [4, 14]. Among those, the Matching Pursuit (MP) algorithm [19] works quite similarly to the above-mentioned greedy algorithm, and its selection criteria is rephrased as choosing the atom having the highest coherence (i.e., inner product) with the remainder.

In the following part we introduce a novel method that extends the greedy approach described in [27] to approximate the noisy covering problem (6). We apply this approach to our localization problem and experimentally show that it outperforms the state-of-the-art algorithms, with a computational complexity amenable to real-time applications.

### 5.1 Set Covering Object Occupancy Pursuit: SCOOP

The proposed localization problem by its construction consists of non-normalized dictionaries i.e., the columns corresponding to the objects on the far back of the scene have much less energy than the ones in front because the corresponding targets appear smaller. As a result, the selection criteria based on the maximal common elements in the support (in the original set cover problem) or maximal coherence (like in MP) often chooses high energy columns that cover highly the MSV as well as many pixels out of the MSV support. For example, a person at the back of the scene with small silhouette is mistakenly approximated by a person in front with a much larger, but well covering silhouette. This indicates that some high energy columns of $D$, despite their good covering, are not fitting the silhouettes well enough. It is thus necessary to modify the algorithm to avoid such mistakes. We address this problem so that, at each iteration, the selecting criteria searches for the column $i$ that has the minimum difference with the remainder $r$ (i.e., MSV at initial step). Thus, the selected column, in addition to a good covering must fit well the MSV i.e., not contain many extra pixels out of the MSV support. In summary, an iteration of SCOOP selects a column index $i$ based on the following criteria:

$$i \Leftarrow \arg\min_i \Big\{ w \overbrace{\frac{|\mathrm{supp}(r)\backslash\mathrm{supp}(d_i)|}{|\mathrm{supp}(r)|}}^{\text{Covering factor}} + (1-w) \underbrace{\frac{|\mathrm{supp}(d_i)\backslash\mathrm{supp}(r)|}{|\mathrm{supp}(d_i)|}}_{\text{Fitting factor}} \Big\}, \tag{7}$$

where $0 \le w \le 1$ is a regularization factor to penalize the uncovered pixels in the remainder support and the extra covered pixels out of the remainder support. This brings a degree of freedom to the algorithm that balances between the covering and fitting factors of the columns.

Compared to simple Thresholding this criteria better respects the sparsity constraint as we now argue. Typically, the superposition of several atoms with poor coherence can have the same energy contribution in the MSV as a single atom with high covering factor. Practically this means that atoms approximating faraway people can be covered by few atoms corresponding to close-by people or atoms corresponding to cars or trucks can cover several atoms corresponding to people standing at the same location. The proposed selection criteria however promotes the selection of the largest atom in case of ambiguity. Typically, it prefers to select a single atom of a truck than several people in the scene. Likewise, it prefers to select one single close-by person instead of several faraway people. As a result, we can say that it promotes a sparse solution. Nevertheless, as mentioned above, a fitting factor is used to avoid selecting atoms with too many out-support pixels.

The full algorithm is presented in Algorithm 1 and coined as 'Set Covering Object Occupancy Pursuit' (SCOOP). As we can see, Thresholding is deployed as a *preprocessing* step in order to reduce the dimension of the search space $\mathscr{U}$ of all possible locations. When $|\mathscr{U}| \ll N$, this step can massively accelerate the main greedy pursuit.

Note that the *stopping criteria* can adopt three forms:

- If one knows a priori how many individuals are present in the scene (e.g., team sports like basketball, soccer,...), then SCOOP runs $k$ iterations to detect them.
- Given a good estimation of the noise power $e$, we apply the same criteria as in Algorithm 1.

**Algorithm 1:** Set Covering Object Occupancy Pursuit' (SCOOP)

```
Input: MSV signal $y$, Dictionary $D$, Error parameter $e$, RSS
       parameter $\tau$, Regularization parameter $w$.
Output: Support set $\widehat{\mathscr{S}}$ (equivalently the occupancy vector $\widehat{x}$).
Initialization:
$\widehat{\mathscr{S}} \Leftarrow \{\}, \mathscr{U} \Leftarrow \{\}, r \Leftarrow y, \widehat{y} \Leftarrow \mathbf{0}$
Preprocess:
for $(i = 1:N)$ do
    if $\left( \left|\mathrm{supp}(d_i) \backslash \mathrm{supp}(y)\right| \leq e \right)$
    $\mathscr{U} \Leftarrow \mathscr{U} \cup \{i\}$.
    end
end
Greedy Process:
while $(E > e)$ do
    $j \Leftarrow \arg\min_{j' \in \mathscr{U}} \left\{ w \frac{\left|\mathrm{supp}(r) \backslash \mathrm{supp}(d_{j'})\right|}{\left|\mathrm{supp}(r)\right|} + (1-w) \frac{\left|\mathrm{supp}(d_{j'}) \backslash \mathrm{supp}(r)\right|}{\left|\mathrm{supp}(d_{j'})\right|} \right\}$
    Updates:
    Recovered support: $\widehat{\mathscr{S}} \Leftarrow \widehat{\mathscr{S}} \cup \{j\}$
    Recovered MSV: $\mathrm{supp}(\hat{y}) \Leftarrow \mathrm{supp}(\hat{y}) \cup \mathrm{supp}(d_j)$
    Remainder: $\mathrm{supp}(r) \Leftarrow \mathrm{supp}(r) \backslash \mathrm{supp}(d_j)$
    Search space: $\mathscr{U} \Leftarrow \mathscr{U} \setminus \mathscr{N}_\tau(j)$
    Error: $E \Leftarrow \left|\mathrm{supp}(y \oplus \hat{y})\right|$
end
```

– If the decoder does not have access to any of those aforementioned priors, SCOOP continues the iterations until by adding the next index the outcome error $E$ (see Algorithm 1) starts to increase.

*Repulsive Spatial Sparsity (RSS)*

In many application (including people localization) there exists another sort of sparsity: *spatial sparsity*. Two individuals are separated by a minimum spatial distance related to the minimum surface occupied by a person on the ground e.g., here we choose 70 cm to be the average width of a standing person. This is what we refer to as the concept of *Repulsive Spatial Sparsity* (RSS) introduced in [1]. More precisely, if $i, j \in \mathrm{supp}(x)$ and $i \neq j$ then we must have,

$$\Delta_{i,j} := \|\mathbf{P}(i) - \mathbf{P}(j)\|_{\ell_2} > \tau, \tag{8}$$

where $\mathbf{P}(i)$ is the position of a point $i$ on the ground plane, and $\tau$ cm is the minimum spatial distance. Lets denote by $\mathscr{N}_\tau(i)$ the set of indices corresponding to positions that are $\tau$-close to $\mathbf{P}(i)$ i.e.,

$$\mathscr{N}_\tau(i) := \{j : \Delta_{i,j} \leq \tau\}. \tag{9}$$

Finding a sparse occupancy vector does not necessarily impose the constraint above. For this purpose, at each iteration, SCOOP excludes all the neighboring points $\mathscr{N}_\tau(.)$ of the selected atom from the search space and modifies $\mathscr{U}$. Note that, the neighboring sets can be computed offline once the dictionary is reconstructed, avoiding an additional complexity imposed by the sub-iterative scheme proposed in [1].

## 5.2 Complexity of SCOOP

All atoms selected by Thresholding satisfy inequality (4). This criteria is directly related to the boolean inner product, which counts number of elements that two $m$-dimensional boolean vectors share in their supports. We define the inner product of $y$ and $i$th column of $D$ as,

$$\begin{aligned} \langle d_i, y \rangle &:= \sum_j d_{ji} . y_j \\ &= \left|\mathrm{supp}(d_i) \cap \mathrm{supp}(y)\right|, \end{aligned}$$

with the predefined boolean arithmetic notations. Computing this inner product for all $N$ atoms of the dictionary leads the preprocessing step to complete with complexity $O(MN)$. The main greedy pursuit performs iteratively. Thanks to the preprocessing step, each iteration consists of searching over $u = |\mathscr{U}| \ll N$ atoms of the dictionary for finding the minimizer of (7). A simple computation indicates that we can rewrite (7) as

$$i \Leftarrow \arg\max_{i \in \mathscr{U}} \left\{ w \frac{\langle r, d_i \rangle}{\left|\mathrm{supp}(r)\right|} + (1-w) \frac{\langle r, d_i \rangle}{\left|\mathrm{supp}(d_i)\right|} \right\}. \tag{10}$$

Therefore, each iteration roughly consists of computing $u$ inner products between $m$-dimensional vectors, finding their maximum and modifying $\mathscr{U}$ by finding (and excluding) the neighbors of the selected atom i.e., complexity of $O(Mu + u \log u) \approx O(Mu)$. Now, if we consider a typical scene with $k$ individuals (known sparsity level e.g., in team sport activities), the whole complexity of SCOOP in Algorithm 1 mainly scales as $O(kMu)$.

Compared to O-Lasso, our method performs enormously faster so that, in typical problem sizes (i.e., typical $M$, $N$, $k$), it is able to detect and localize the objects of a frame in *real-time*. Mainly two reasons are behind the success of SCOOP: First, a novel formulation of the problem based on boolean regression model. Second, using the greedy approach to solve the localization problem. Unlike O-Lasso, our approach does not need to solve a sparsity-inducing convex optimization (reweighed $\ell_1$-minimization) with heavy computations. As a consequence, instead of performing many iterations to converge to a solution (like in O-Lasso), our approach identifies one individual per iteration. In addition, each iteration of SCOOP performs only basic boolean arithmetic operations that are cheap in terms of computational complexity and memory usage. In the next section we demonstrate by several simulations that this low complexity does not impair robustness.

## 6 Experimental Results and Comparisons

In this section we present results of two main sets of experiments that have been conducted to characterize the performance of our proposed method. *Real-world datasets* including the APIDIS[1] and the PETS 2009 benchmarks[2] are used in order to compare the performance of SCOOP to the state-of-the-art methods. In addition, we run several experiments on a *Synthetic dataset* generated by the same scene geometry as the APIDIS dataset, to analyze the performance of SCOOP as the problem size scales with the number of objects or size of the scene.

The APIDIS dataset consists of seven pseudo-synchronized cameras monitoring a basketball game (including one omnidirectional camera), whereas the PETS 2009 benchmark considers an outdoor scene containing multi-sensor sequences of different crowd activities filmed by multiple cameras. All videos are scaled to a QVGA resolution with approximately 25 fps, and for both datasets dictionaries have been constructed using camera calibration data as it was explained in section 3.0.3. We apply the adaptive non-regular grids as in [1] to discretize the scenes. The constructed dictionaries map the grid points that are densely sampling the scenes (e.g., every 10 cm in APIDIS dataset) into their approximate MSVs.

The performance of the detection process is quantitatively measured by computing the *Precision* and the *Recall* measures given by the following ratios:

$$Precision = \frac{TP}{TP+FP}, \quad Recall = \frac{TP}{TP+FN}, \tag{11}$$

where $TP$, $FP$ and and $FN$ are the number of True Positives, False Positives and False Negatives. A true positive is when a person is correctly located (at the resolution of the scene discretization e.g., within 10 cm distance accuracy in APIDIS dataset) on the ground plane. A confusion between two neighboring grids localizations counts as both a FP and a FN, indicating the strictness of our quality measures at such high-resolution scene sampling.

### 6.1 Synthetic-Noiseless Setup

Once the dictionary is constructed, we are able to synthesize foreground silhouettes with the same scene geometry as in the APIDIS dataset. We generate random occupancy vectors $x$ and the corresponding noiseless MSVs are computed by $y = D.x$. Our main goal here is to analyze the behavior of SCOOP in various problem sizes.

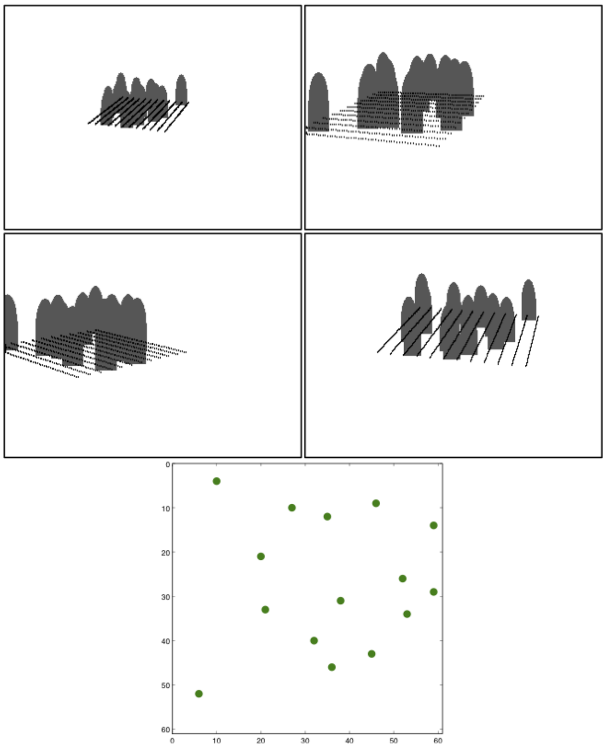


**Fig. 3** Demonstration of a densely populated synthetic scene: $k = 15$ synthetic half-rectangular-half-elliptical objects are randomly distributed on $N = 60 \times 60$ grid points. Silhouettes of the objects viewed by four cameras as well as their location on the grid are shown.

#### *6.1.1 Performance Comparisons*

First we choose a submatrix of the original dictionary which corresponds to a square subregion of the basketball court including $60 \times 60$ grid points ($\approx 6 \times 6 m^2$) observed by four planar cameras. MSVs are synthesized from occupancy vectors corresponding to $k$ individuals/objects randomly distributed in the scene, and the results are averaged over a hundred independent realizations. Figure 3 illustrates a realization of such scene and the half-rectangular-half-elliptical shaped silhouettes observed by four cameras. Note that, this is a very densely populated scene containing $k = 15$ people on a region smaller than one tenth of the basketball court.

In Figure 4 we compare Thresholding and SCOOP methods. We set the parameter $e = 0$ for both algorithms as we consider a noiseless scenario, and $w = 0.2$ for SCOOP.[3] Solid and dashed-line curves correspond to setups wherein any two individuals are separated by minimum distances of 100 cm and 70 cm, respectively. For both methods the average performance decreases as the scene becomes more dense i.e., more people/objects or less relative distance between them. We can observe that by increasing the number of in-

[1] Dataset is publicly available at `http://www.apidis.org/Dataset/`

[2] Dataset is publicly available at `http://winterpets09.net/`

[3] Here, unless otherwise specified, SCOOP applies the third stopping criteria.

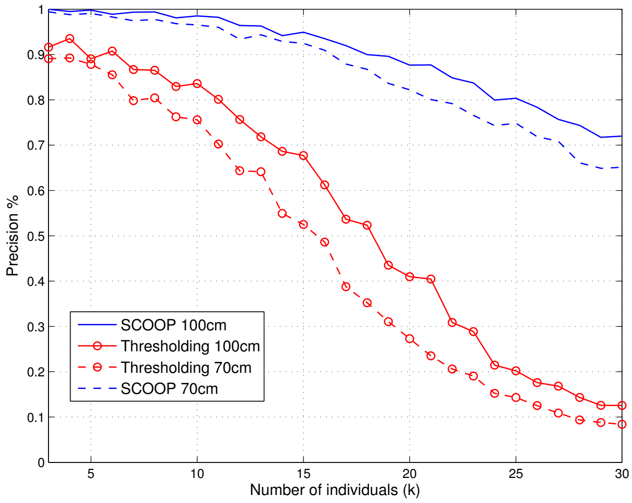


**Fig. 4** Comparing the precision of SCOOP and Thresholding on synthetic data. Solid and dashed-line curves correspond to setups wherein any two individuals are separated by minimum distances of 100 cm and 70 cm, respectively.

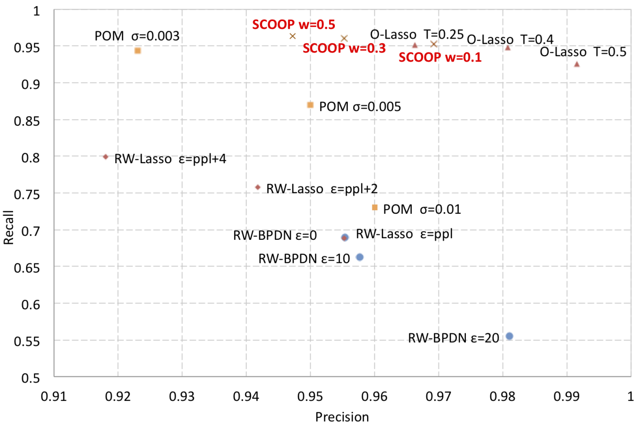


**Fig. 5** Precision and recall rate with the synthetic data given four cameras (Noiseless foreground silhouettes, however with possible occlusions). Our proposed approach SCOOP is compared with other sparsity driven formulation and the probability of occupancy (POM) approach presented by Fleuret *et al.* in [13].

dividuals, Thresholding reports many false positives (due to the ambiguity raised by many positions hidden from the camera views) resulting in a huge decrease in the method precision. In contrast, SCOOP discards many of those false positives by selecting a sparse occupancy vector, which makes the method more robust against densely populated scenes.

In another setup we compare the performances of SCOOP and the-state-of-the-art methods such as O-Lasso, RW-BPDN and RW-Lasso introduced in [1] as well as POM presented in [13]. Evaluations are done for various algorithms regularization parameters and over a wider region i.e., half of the basketball court. Synthetic foreground silhouettes are constructed as explained above with a spatial sparsity constraint; any two neighboring individuals have a minimum spatial distance of 70 cm with each other. Five to fifteen people are randomly triggered for each frame (few hundred frames are generated). As we can observe in Figure 5, SCOOP and O-Lasso have comparable performances and both outperform other methods such as RW-BPDN, RW-Lasso and POM. Remarkably, SCOOP has an extremely lower computational complexity compared to all other methods and it achieves a realtime performance that will be experimentally shown in the following section.

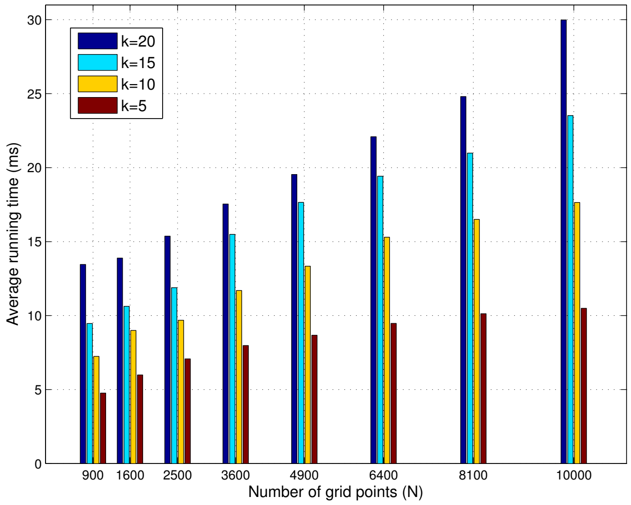


**Fig. 6** Average running time (per frame) of SCOOP for various problem sizes $k$ and $N$, using four cameras (synthetic data).

### 6.1.2 Computational Complexity

Figure 6 demonstrates the running time of SCOOP for different problem sizes, averaged over a hundred random synthetic frames. Increasing either $k$ or $N$ will naturally prolong the localization process, however, a remarkable observation is that, the complexity of SCOOP is less sensitive to the dimension of the scene than to the number of the individuals i.e., the computation time increases less by doubling $N$ with a fixed $k$ rather than vice-versa. This highlights the advantage of the preprocessing step in SCOOP which refines the search space $\mathscr{U}$ prior to the greedy pursuit i.e., in our experiments as $N$ grows, the dimension of the search space $u$ stays rather proportional to $k$ (with a constant factor between 1.5 to 5). The charts in Figure 7 are demonstrating this idea, where the preprocessing acceleration factor $(N/u)$ almost linearly increases as $N$ grows, and the steepest slope belongs to the less populated scenes $k = 5$.

We run these experiments in Matlab 7.1, on a modest MacBook Pro laptop (2.33 GHz Intel core 2 Duo CPU and 2GB RAM), and without any particular parallel optimization. Although our main focus was analyzing the general behavior of the graphs, we can observe that SCOOP performs

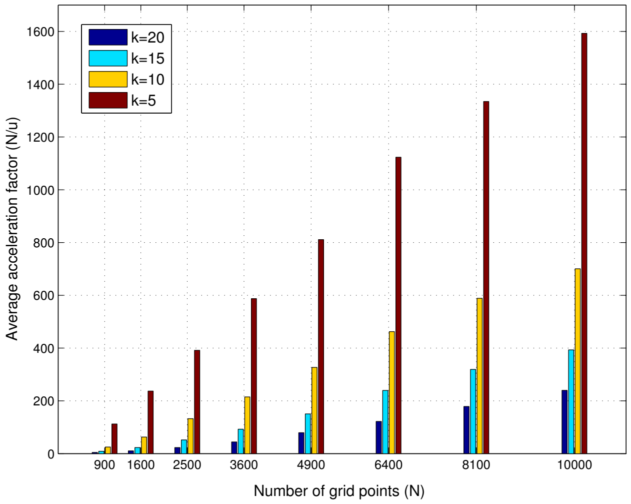


**Fig. 7** Average acceleration factor (per frame) of SCOOP for various problem sizes $k$ and $N$, using four cameras (synthetic data).

| Video Quality | Algos | Number of Individuals | | | |
|---|---|---|---|---|---|
| | | $k=5$ | $k=10$ | $k=15$ | $k=20$ |
| QVGA | SCOOP | 0.008 | 0.0117 | 0.015 | 0.0175 |
| | O-Lasso | 0.397 | 0.492 | 0.761 | 1.385 |
| VGA | SCOOP | 0.033 | 0.048 | 0.059 | 0.073 |
| | O-Lasso | 2.463 | 2.956 | 4.015 | 7.98 |

**Table 1** Average computational times of SCOOP and O-Lasso (in seconds) for different crowd densities and video qualities (Number of scene grid points $N = 3600$).

in real-time (see Figure 6). It has been shown in [1] that, by applying *adaptive non-regular* grids even huge practical setups such as the outdoor scenes in PETS 2009 benchmark can be efficiently sampled from $N \lesssim 6000$ points and therefore, by using SCOOP one can handle localization of 20 individuals (in such huge scenes) at the rate of approximately 45 fps.

Note that our implementation benefits from another sort of sparsity which appears to be essential for reducing even more the complexity of SCOOP. Since by construction each atom of the dictionary contains only MSV of a single object (which is a sparse vector in many positions), in many typical setups the dictionary matrix turns out to be sparse. In the above-mentioned experimental setups the nonzero elements are only a small fraction (about 1.2%) of the whole dictionary matrix of the APIDIS dataset. Considering this fact in the algorithm implementation significantly reduces the memory usage as well as the computational complexity of each iteration (i.e., the sparse matrix product), in order to perform much better than $O(Mu)$ (in fact, proportional to the number of the nonzero elements).

Finally, we compare the computation time of SCOOP with O-Lasso [1] (whose recall-precision performance is comparable to SCOOP) on a square subregion of $60 \times 60$ grid points and for various number of individuals. For the sake of fairness, both methods initially apply Thresholding, by setting $e = 0$ in (4), in order to reduce the search space dimension. Table 1 contains the computation times of SCOOP and O-Lasso (averaged over a hundred random synthetic frames) for the video qualities QVGA and VGA. We can see that for a given video sequence SCOOP runs between fifty to a hundred times faster than O-Lasso. The running times of both algorithms naturally increases using a higher video resolution. By using a more efficient implementation we can still predict a realtime performance for SCOOP at higher video resolutions than QVGA, however, our experiments in section 6.2 indicates that, the QVGA resolution can be sufficient for SCOOP to achieve a precise localization performance in real-life scenes.

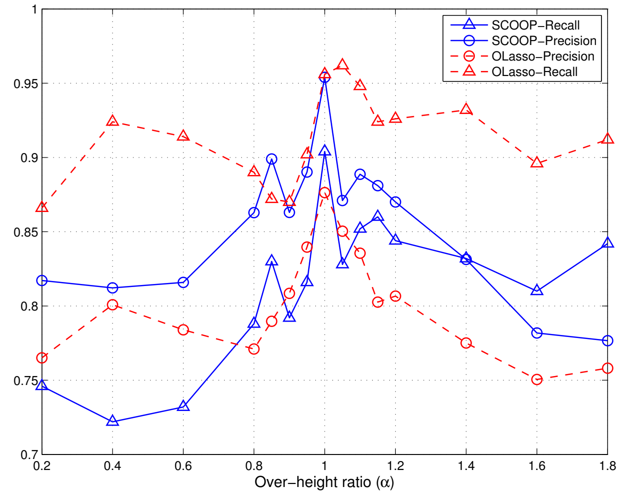


**Fig. 8** Precision and recall rates of SCOOP and O-Lasso with respect to the over-height ratio ($\alpha$) i.e., people's height (cm) divided by 170.

### *6.1.3 Stability against Height Variations*

We run a few simulations to measure the sensitivity of our algorithm with respect to the individuals having different heights than 170 cm. We construct several dictionaries (as previously mentioned in section 3.0.3) whose atoms simulate individuals with a wide variety of heights ranging from 34 cm to 306 cm (which can be sometimes even unrealistic). At a given height, we use the corresponding dictionary in order to generate a hundred video frames of five individuals who are randomly located in a subregion containing $N = 60 \times 60$ grid points. For the localization process, however, we use the previous "reference" dictionary with atoms of 170 cm height. The recall-precision performances of SCOOP and O-Lasso are depicted in Figure 8. As we can see, the performances of both methods decrease (at most) between 10-17%, indicating a rather stable localization even for dramatic height changes. Note that for each

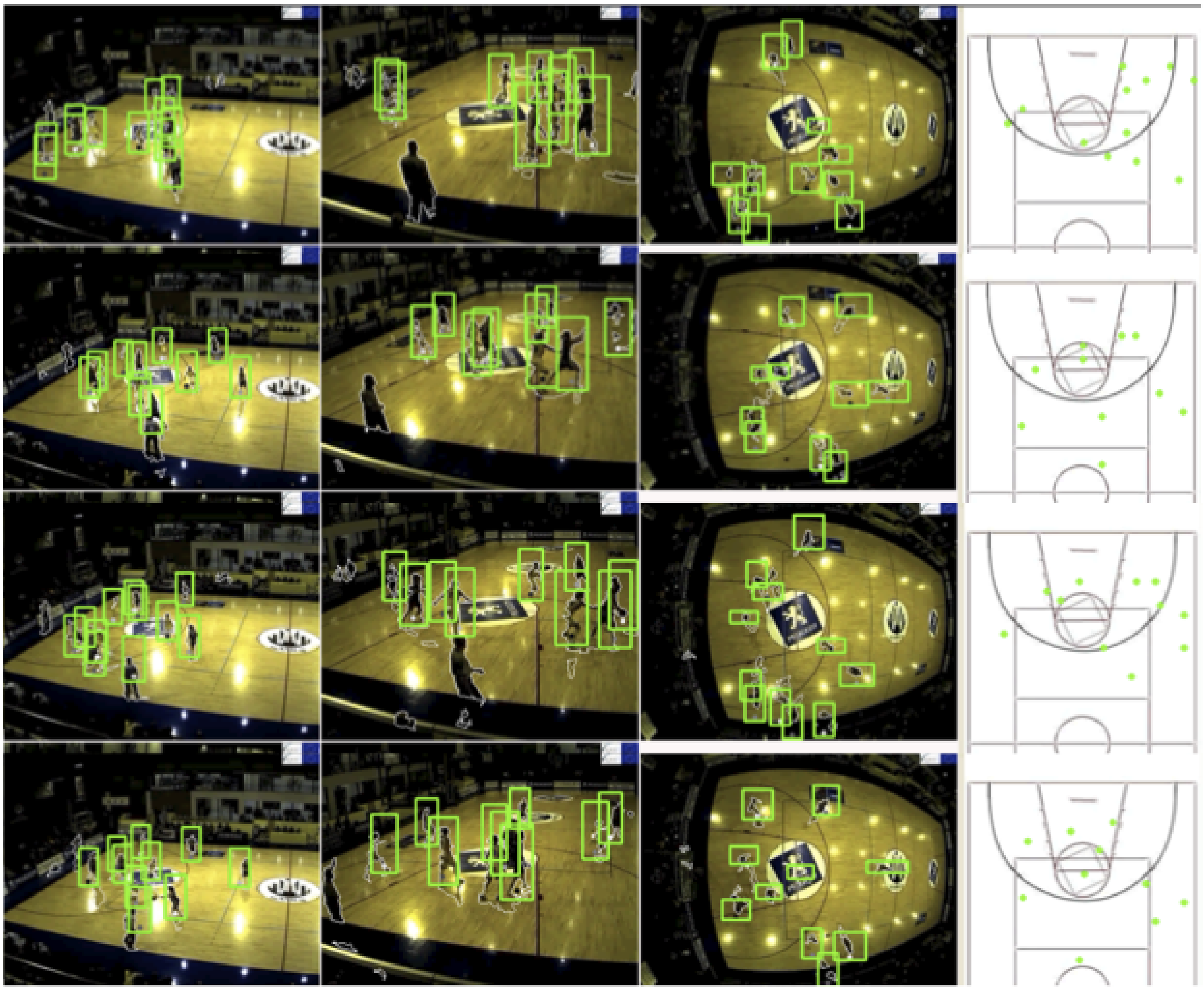


**Fig. 9** Detecting and localizing players in the APIDIS dataset using SCOOP: demonstration for four frames and three camera views per frame. Players' positions are marked for the left-half of the basketball court. White contours correspond to the degraded foreground silhouettes that are extracted and used for localization.

$0.2 \leq \alpha \leq 1.8$, that is the over-height ratio (the ratio of the heights of the individuals (cm) to 170 i.e., the height of the reference dictionary atoms), the Thresholding criteria in (4) is modified to

$$\frac{\langle d_i, y\rangle}{|\mathrm{supp}(d_i)|} \geq \min\{\alpha, 1\},$$

so that, the refined search space $\mathscr{U}$ contains the original solution. For the extreme height differences (in particular, for more than 50%), adding an extra atom at each iteration of SCOOP can increment the error term $E = |\mathrm{supp}(y \oplus \hat{y})|$ in Algorithm 1 and therefore, SCOOP may stop at the very early iterations and before localizing all individuals. In order to resolve this issue, here, we modify as well the third stopping criteria of SCOOP so that, the iterations continue as long as the newly chosen atom $d_i$ increases the error term less than $|1-\alpha| \times |\mathrm{supp}(d_i)|$.[4]

## 6.2 Real-World Datasets

As previously mentioned, in this part we compare the performance of SCOOP with the state-of-the-art methods of localization and for this purpose we consider two classes of real-world challenging datasets. Note that for both datasets the foreground silhouettes are extracted using the work of Stauffer and Grimson [24].

First, we consider a sequence of a basketball match from the APIDIS dataset and we evaluate the performance over

[4] Note that, $|.|$ denotes the absolute value of a scalar, whereas for a set (and as previously defined) it returns its cardinality.

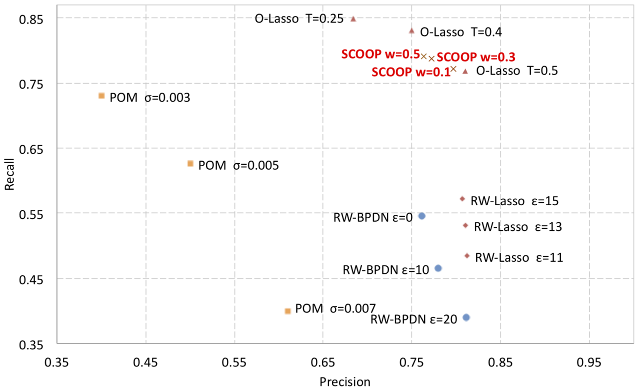


**Fig. 10** Precision and recall rate with the APIDIS dataset given four cameras (with severely degraded foreground silhouettes). Our proposed approach SCOOP is compared with other sparsity driven formulation and the probability of occupancy (POM) approach presented by Fleuret *et al.* in [13].

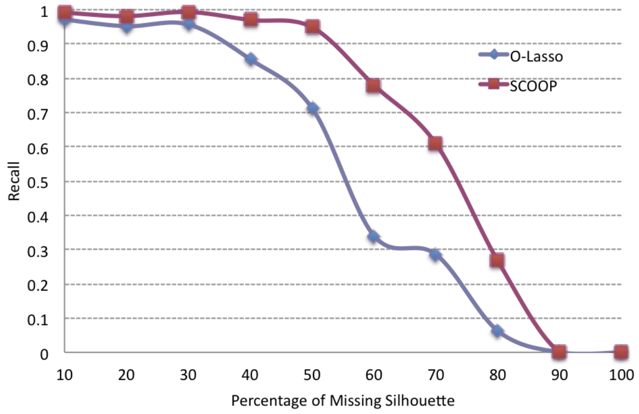


**Fig. 11** Recall rate with respect to the degradation on the foreground silhouettes.

the left-half of the basketball court where most of the cameras are located i.e., cameras id=1, 2, 5 and 7 (see Figure 9). This dataset has several challenges: Basketball players may have unexpected changes of behavior, e.g., running, jumping, crouching, sudden changes in the motion path, etc. Players can be either strongly grouped together or spatially scattered. Shadows and reflection of the players on the ground floor mislead many typical silhouette extraction techniques, and they output severely degraded MSVs often corrupted by many false positive pixels (i.e., noisy data).

We run several experiments on this dataset and we measure the performance of SCOOP together with several state-of-the-art methods of localization namely, the sparsity-driven convex-approach in [1] (RW-Lasso, RW-BPDN, O-Lasso) and the work of Fleuret *et al.* in [13] (referred to as POM). Results are reported in Figure 10 and provide a clear comparison between the performance of SCOOP and its counterparts; SCOOP outperforms RW-Lasso, RW-BPDN, POM, and record a similar performance as O-Lasso.

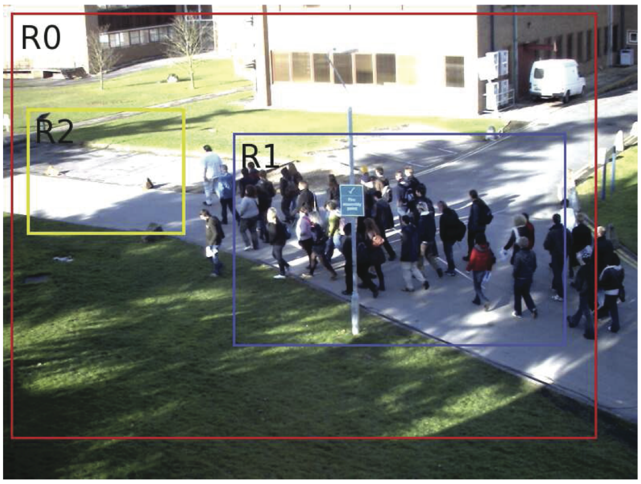


**Fig. 12** Three regions (R0, R1 and R2) are considered in the PETS 2009 dataset in order to report the number of detected people.

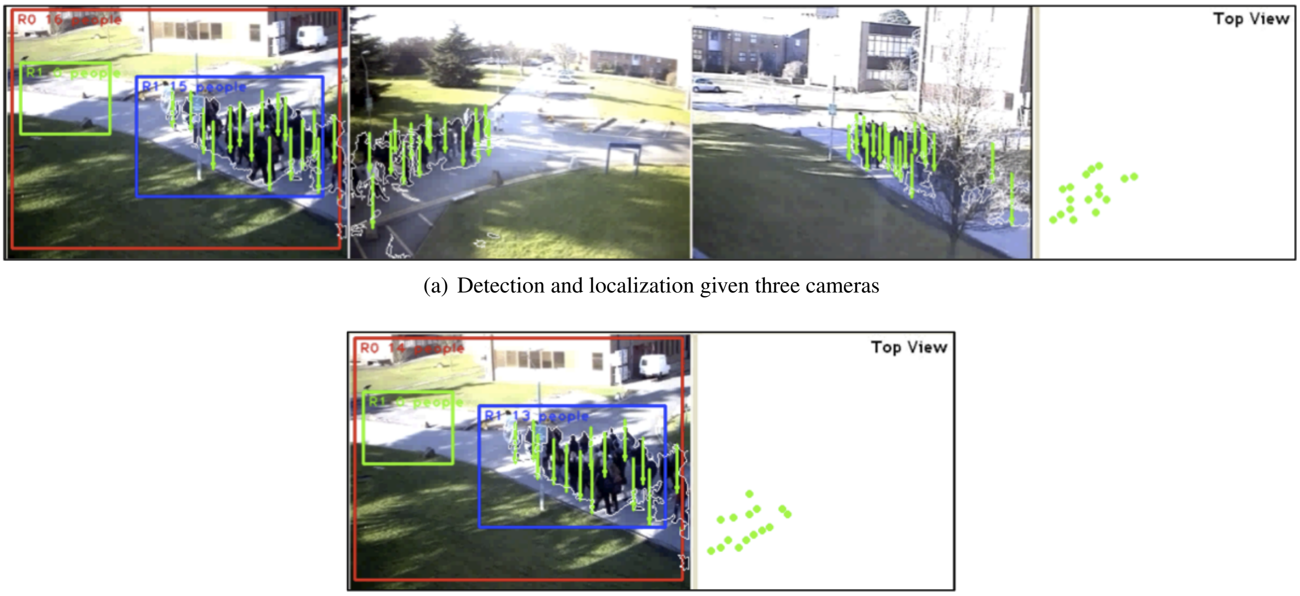
In addition to its precision, our method offers a huge acceleration in detection time. We measure the computation time of SCOOP as opposed to O-Lasso. Given the setup above with five cameras and a Matlab implementation for both algorithms, SCOOP locates the basketball players on average *a hundred times* faster than O-lasso.

Figure 11 compares the robustness of SCOOP to O-Lasso given degraded foreground silhouettes. We measure the recall rate with respect to the percentage of removed foreground silhouette (MSV) per person. Clearly, degrading the silhouettes, degrades the recall rate. However, it is interesting to notice that SCOOP is more robust to the degradation, i.e. for a given degradation, the recall rate is higher than with O-lasso. By removing half of the foreground silhouettes, we can still locate people 90% of the time.

In the next setup, we consider an outdoor scene monitored by three cameras using the PETS 2009 benchmark dataset (views are chosen from cameras Id=1, 2 and 3). The dataset consists of three levels of difficulty L1, L2 and L3, with various crowd activities (walking, running) and different densities up to approximately forty people in the scene. For evaluation and comparison purposes, we use the same performance evaluation proposed by the PETS organizers [11]. The number of people detected into predefined regions are reported (Figure 12 illustrates those regions). Results are demonstrated in Table 2 and we can observe that the proposed SCOOP algorithm is performing better than O-Lasso since the Average Frame Error (AFE) is reduced. Moreover, we can see that the solutions of SCOOP are less sparse than of O-Lasso (however, closer to the ground truth). Such gain in performance is justified by the boolean operators used by the greedy approach i.e., SCCOP is more precise with the non-linear phenomena occurring with the occlusions compared to O-lasso. Therefore, crowded scenes are better han-

| Seq | Algos | R0 | | | R1 | | | R2 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | GT | PC | AFE | GT | PC | AFE | GT | PC | AFE |
| S1-L1-2 | O-Lasso<br>SCOOP | 2861 | 2186\|2305<br>2669\|2753 | 4.2 \|3.8<br>**1.6**\|**2.4** | 1237 | 908\|981<br>1113\|1147 | 2.3\|1.9<br>**1.2**\|**1.2** | 1130 | 848\|895<br>1065\|1070 | 1.8\|1.8<br>**0.9**\|**1.3** |
| S1-L2-1 | O-Lasso<br>SCOOP | -<br>- | -<br>- | -<br>- | 1279 | 558\|710<br>832\|914 | 10.0\|8.0<br>**6.5**\|**5.7** | 1622 | 814\|940<br>1008\|1104 | 6.8\|6.0<br>**5.3**\|**4.7** |
| S1-L3-1 | O-Lasso<br>SCOOP | -<br>- | -<br>- | -<br>- | 230 | 163\|196<br>258\|231 | 2.0\|1.2<br>**0.9**\|**1.0** | -<br>- | -<br>- | -<br>- |
| S1-L3-2 | O-Lasso<br>SCOOP | -<br>- | -<br>- | -<br>- | 2632 | 2084\|1550<br>2717\|1647 | 10.1\|10.5<br>**8.8**\|**9.2** | -<br>- | -<br>- | -<br>- |

**Table 2** Performance of people counting on the scenarios proposed by the PETS organizers given 1 camera (id=1) and 3 cameras (id=1,2,3): GT = Ground Truth (number of people), PC = People Counted by SCOOP vs. O-Lasso, AFE = Average Frame Error. The dataset ”S1” is used with various level of difficulty (”L1”: medium density crowd, ”L2” and ”L3” correspond to high density of people.



(a) Detection and localization given three cameras

(b) Detection and localization given a single camera

**Fig. 13** Detecting and localizing people in a crowd using SCOOP, given (a) three camera views or (b) a single camera view: positions in the scene are marked in top view images. White contours correspond to the grouped and noisy foreground silhouettes that are extracted and used for localization.

dled by such approach. Figure 13 illustrates detection and localization of a densely populated crowd using SCOOP and for a single frame of the PETS 2009 dataset. We can observe that the extracted foreground silhouettes highlighted by the white contours are severely grouped together (many occlusions) leading to a very challenging localization scenario.

## 7 Conclusions

In this paper we introduced a novel algorithm (SCOOP) for people/object localization purposes. By considering various degradation models, we show that SCOOP performs as precise and robust as the state-of-the-art methods, however with the advantage of being much more numerically efficient, achieving a real-time performance. Design of this algorithm together with our mathematical analysis are the outcomes of a new formulation for the localization problem based on a boolean regression model. This model creates connections between our problem and the well-known group testing literature and later the set cover problem, allowing us using the powerful existing tools in their literature.

In the future, we intend to use SCOOP (for the localization step) together with a tracking block in order to build a complete system that is able to capture people’s motion behavior given a network of fixed cameras and analyze their trajectories, all in real-time. This system would find variety of applications in security and surveillance, psychological studies, team sport coaching and tactics, and market research.

In addition, we would like to study in further details multiple object recovery using dictionaries that are constructed from MSVs of various objects e.g., cars, bikes, buses, trucks and pedestrians that are appearing together in typical urban scenarios. There, designing a method that is able to robustly distinguish between multiple objects from their noisy MSVs (e.g., not estimating a crowd silhouette by a single car) is of high importance.

## References

1. Alahi, A., Jacques, L., Boursier, Y., Vandergheynst, P.: Sparsity driven people localization with a heterogeneous network of cameras. Journal of Mathematical Imaging and Vision pp. 1–20 (2011). URL `http://dx.doi.org/10.1007/s10851-010-0258-7`. 10.1007/s10851-010-0258-7
2. Berclaz, J., Fleuret, F., Fua, P.: Robust people tracking with global trajectory optimization. In: Conference on Computer Vision and Pattern Recognition (2006)
3. Black, J., Ellis, T., Rosin, P.: Multi view image surveillance and tracking. Proc. IEEE Workshop on Motion and Video Computing **00**, 169 (2002)
4. Blumensath, T., Davies, M.: Iterative thresholding for sparse approximations. Journal of Fourier Analysis and Applications **14**(5), 629–654 (2008)
5. Candès, E.J., Wakin, M., Boyd, S.: Enhancing sparsity by reweighted $\ell_1$ minimization. J. Fourier Anal. Appl. (2007). (to appear)
6. Caspi, Y., Simakov, D., Irani, M.: Feature-based sequence-to-sequence matching. International Journal of Computer Vision **68**(1), 53–64 (2006)
7. Cheraghchi, M., Hormati, A., Karbasi, A., Vetterli, M.: Group testing with probabilistic tests: Theory, design and application. IEEE Transactions on Information Theory (2010)
8. Delannay, D., Danhier, N., De Vleeschouwer, C.: Detection and recognition of sports(wo)man from multiple views. In: Proc. ACM/IEEE Int'l Conference on Distributed Smart Cameras. Como, Italy (2009)
9. Dorfman, R.: The detection of defective members of large populations. Annals of Mathematical Statistics **14**, 436–440 (1943)
10. Du, D.Z., Hwang, F.: Combinatorial Group Testing and its Applications. World Scientific Series on Applied Mathematics (1999)
11. Ellis, A., Shahrokni, A., Ferryman, J.: Overall evaluation of the pets2009 results. In: Proc. IEEE Int'l Workshop on Performance Evaluation of Tracking and Surveillance. Snowbird, Utah (2009)
12. Eshel, R., Moses, Y.: Homography based multiple camera detection and tracking of people in a dense crowd. In: Proc. IEEE Int'l Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)
13. Fleuret, F., Berclaz, J., Lengagne, R., Fua, P.: Multicamera people tracking with a probabilistic occupancy map. IEEE Trans. on Pattern Analysis and Machine Intelligence **30**(2), 267–282 (2008)
14. Fornasier, M., Rauhut, H.: Iterative thresholding algorithms. Applied and Computational Harmonic Analysis **25**(2), 187–208 (2008)
15. Golbabaee, M., Vandergheynst, P.: Average case analysis of sparse recovery with Thresholding: New bounds based on average dictionary coherence. In: IEEE Int. Conf. on Acoustics, Speech & Signal Processing (ICASSP) (2008)
16. Kannala, J., Brandt, S.: A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. IEEE Trans. on pattern analysis and machine intelligence **28**(8), 1335 (2006)
17. Khan, S., Shah, M.: A multiview approach to tracking people in crowded scenes using a planar homography constraint. In: Proc. European Conference on Computer Vision, pp. IV: 133–146 (2006)
18. Khan, S.M., Mubarak, S.: Tracking multiple occluding people by localizing on multiple scene planes. IEEE Trans. on Pattern Analysis and Machine Intelligence **31**(3), 505–519 (2009)
19. Mallat, S., Zhang, Z.: Matching pursuits with time-frequency dictionaries. IEEE Transactions on signal processing **41**(12), 3397–3415 (1993)
20. Mueller, K., Smolic, A., Droese, M., Voigt, P., Wienand, T.: Multi-texture modeling of 3d traffic scenes. In: Multimedia and Expo, 2003. ICME '03. Proceedings. 2003 International Conference on, vol. 1, pp. I – 657–60 vol.1 (2003). DOI 10.1109/ICME.2003.1221003
21. Orwell, J., Massey, S., Remagnino, P., Greenhill, D., Jones, G.A.: A multi-agent framework for visual surveillance. In: Proc. IEEE Int'l Conference on Image Analysis and Processing, p. 1104. IEEE Computer Society, Washington, DC, USA (1999)
22. Porikli, F.: Achieving real-time object detection and tracking under extreme conditions. Journal of Real-Time Image Processing **1**(1), 33–40 (2006)
23. Schnass, K., Vandergheynst, P.: Average performance analysis for thresholding. Signal Processing Letters, IEEE **14**(11), 828–831 (2007)
24. Stauffer, C., Grimson, W.: Adaptive background mixture models for real-time tracking. Proc. IEEE Int'l Conference on Computer Vision and Pattern Recognition **2**, 246–252 (1999)
25. Stauffer, C., Tieu, K.: Automated multi-camera planar tracking correspondence modeling. In: Proc. IEEE Int'l Conference on Computer Vision and Pattern Recognition, pp. I: 259–266 (2003)
26. Tibshirani, R.: Regression Shrinkage and Selection Via the Lasso. Journal of the Royal Statistical Society **58**(267-288) (1994)
27. Vazirani, V.: Approximation algorithms. Springer Verlag (2001)