The Effect of Recovery Algorithms on Compressive Sensing Background Subtraction

Rhian Davies, Lyudmila Mihaylova, Nicos Pavlidis, Idris Eckley
STOR-i Centre for Doctoral Training, Lancaster University, Lancaster, UK

E-mails: *(r.davies3, mila.mihaylova, n.pavlidis, i.eckley) @lancaster.ac.uk*

***Abstract*—Background subtraction is a key method required to aid processing surveillance videos. Current methods require storing each pixel of every video frame, which can be wasteful as most of this information refers to the uninteresting background.**

Compressive sensing can offer an efficient solution by using the fact that foreground is often sparse in the spatial domain. By making this assumption and applying a specific recovery algorithm to a trained background, it is possible to reconstruct the foreground, using only a low dimensional representation of the difference between the current frame and the estimated background scene.

Although new compressive sensing background subtraction algorithms are being created, no study has been made of the effect of recovery algorithms on performance of background subtraction. This is considered by applying both Basis Pursuit and Orthogonal Matching Pursuit (OMP) to a standard test video, and comparing their accuracy.

I. Introduction

Surveillance cameras have become ubiquitous in many countries, constantly collecting a huge amount of data most of which is stored and never analysed. This is due to the challenge of converting a large volume of video data into useful information, particularly as several cameras may be acquiring data simultaneously. One of the main aims of collecting surveillance footage is to track an object or classify its behaviour, therefore the first step in video analysis is to identify the objects of interest from the background.

Background subtraction [**?**] is a method used to separate the foreground from the background of a video sequence. This consists of constructing and updating a model of the background and then subtracting it from the current frame. Background subtraction is not a new development and many methods for modelling a background scene exist, however traditional background subtraction methods are not very efficient. Most surveillance footage consists of a slowly adapting background scene with foreground appearing in a subset of the frames and when foreground does appear in the footage it usually takes up only a small percentage of the overall frame. Traditional background subtraction methods require that all pixels are acquired for each frame in order to correctly segment the two layers, however since foreground is sparse in most surveillance footage this can be seen as a waste of resources.

In order to combat this inefficiency, the single pixel camera (SPC) [**?**] was developed; a camera based on the theory of compressive sensing [**?**], [**?**], [**?**] and designed to acquire images directly in compressed format. Once surveillance footage is acquired using a camera such as the SPC, background subtraction can be performed on this low dimensional representation of the video frames. Later a recovery algorithm is used to decode a mask of the foreground into the correct dimension. The reason for using this technique is, due to the natural sparsity property of foreground, a mask can be reconstructed without ever storing the current frame, or background model in the full dimensional form. The computational savings available are significant, provided that the segmentation performed is accurate enough for the purpose of the application.

This paper discusses if the choice of recovery algorithm affects the performance of compressive sensing background subtraction. In particular two algorithms are investigated, Basis Pursuit [**?**] which is an algorithm based on convex optimisation [**?**] and a greedy algorithm called Orthogonal Matching Pursuit [**?**]. Results are presented from both algorithms on data which was gathered using a conventional camera but which have been simulated to mimic the acquisition process of compressive sensing imaging technology such as the SPC.

The rest of this paper is organised as follows; Section II discusses the related works in this area in brief before discussing the methodology behind compressive sensing and its application to background subtraction in Section III. Experimentation is conducted in Section IV and conclusions are given in Section V.

II. Related Works

There is a vast amount of research available in the literature detailing the many techniques for background subtraction; notable comparative studies include [**?**] and [**?**]. Generally, algorithms for background modelling can be categorised into pixel based and region based methods [**?**]. The latter may seem more intuitive as one expects foreground to be clustered and generally not exist as isolated pixels. Using knowledge of neighbouring pixels should therefore improve the classification of pixels into foreground or background, and region based techniques such as [**?**] and [**?**] do use this knowledge to segment the images into regions and then create a background model based on these regions. However, there are downsides to region based techniques, for example they are often a lot more complex to run than pixel based techniques, and are not generally robust to division of blocks.

Pixel-based methods such as approximate median filtering [**?**] assume that the pixels observed are classified as foreground independently of each other. These methods can often detect the contours of foreground well but may be susceptible to making false classifications, especially if the background model is not well tuned.

It is also possible to categorise these algorithms as recursive and non-recursive methods [**?**]. A non-recursive algorithm maintains a buffer or window of N previous video frames and estimates a background model based on the statistical properties of these frames. This sliding window is usually required to be fairly large in order to obtain accurate results, and therefore can quickly become computationally intensive. Recursive techniques update the background subtraction each time a new frame is stored and so there is no need to buffer previous frames. The method used in Section III-D to model the background is a pixel-based, recursive method.

More recently the new technology of Compressive Sensing [**?**], [**?**], [**?**] has been applied to the background subtraction problem. The theory of compressed sensing states that an undersampled signal can be recovered almost perfectly, given that the signal itself is sparse in some domain [**?**], i.e. a large proportion of the signal's elements are close to zero. In compressive sensing for video, a random matrix is used to encode a signal efficiently, and then a recovery algorithm is used to decode the signal and recover the required information fairly accurately.

Most compressed sensing background subtraction methods [**?**], [**?**], [**?**], [**?**] make the assumption that the majority of the pixels in a video represent the background, and therefore the foreground is sparse in the spatial domain. This key point allows us to encode a video into a low dimension and then search for a sparse solution, using the knowledge that the foreground mask should be suitably sparse.

The background subtraction problem is first expressed as a sparse signal recovery problem in [**?**]. Their work successfully recovers silhouettes of foreground activity by modelling a compressed form of the background and recovering a foreground mask directly from compressed measurements. Later [**?**] builds upon this work by proposing a method that adapts the encoding process between frames to incorporate the changing size of foreground. Unlike [**?**], the authors of [**?**] choose to use a static model for the background which could cause problems in the model when applying to real-life video.

More current work has started attempting to incorporate the structure of foreground into segmentation. It is generally expected that foreground will be clustered into particular shapes in most frames and does not consist of isolated pixels. Foreground often consists of humans, animals and vehicles, and so knowledge of this structure could be used to improve segmentation. Currently many different approaches are being investigated such as applying particle filters [**?**] and lattice based graphical models [**?**]. Although cluster type methods are not attempted in this work, it is a research area of interest for future investigation.

In this work, two recovery algorithms are compared, one greedy method and one based on convex optimisation when applied to a background subtraction algorithm.

III. Methodology

In this section, the main theory is introduced, starting with the explanation of sparse signals in Section III-A, progressing to the theory of Compressive Sensing in Section III-B with a focus on the two recovery algorithms of interest in Section III-C and finally notes on applying this theory to the background subtraction problem in Section III-D.

A. Sparse and Compressible Signals

A signal is known as being K-sparse if $\boldsymbol{x} \in R^N$ can be represented as a linear combination of K basis vectors [**?**]. The case of interest occurs when K is much smaller than N as this means that most of the coefficients are zero. No signal is ever truly sparse in the presence of noise and so some signals are described as approximately sparse, or compressible. If a signal is compressible there exist K large coefficients (with $K \ll N$) but the remaining $N-K$ coefficients are only required to be small and not necessarily zero. Fundamentally a signal is compressible if most of the information in the signal is represented by a few coefficients.

B. Compressive Sensing

According to the framework developed in [**?**], [**?**], [**?**], the measurement $\boldsymbol{y}$ is a linear function of the signal $\boldsymbol{x}$ as shown in Eq. (1). The number of measurements M in $\boldsymbol{y}$ is chosen to be smaller than N, so a measurement matrix $\boldsymbol{\Phi} \in R^{M \times N}$ is chosen, with $M \ll N$. Although it is known from linear algebra that there are infinitely many vectors $\boldsymbol{x}$ that can solve Eq. (1), the knowledge that the original signal $\boldsymbol{x}$ was sparse can be used to help the reconstruction process, given that certain conditions hold for $\boldsymbol{\Phi}$.

$$\boldsymbol{y} = \boldsymbol{\Phi x} \tag{1}$$

It is of vital importance that a stable measurement matrix $\boldsymbol{\Phi}$ is designed so that the signal information is not damaged by the dimensional reduction from $\boldsymbol{x} \in R^N$ to $\boldsymbol{y} \in R^M$. In order for the reconstruction problem to be well-conditioned, it is sufficient that $\boldsymbol{\Phi}$ holds the Restricted Isometry Property (RIP) [**?**] of order $2K$.

Definition 1. *A matrix $\boldsymbol{\Phi}$ satisfies the (RIP) of order K if there exists a $\delta_K \in (0,1)$ such that*

$$(1-\delta_k)\|\boldsymbol{x}\|_2^2 \leq \|\boldsymbol{\Phi x}\|_2^2 \leq (1+\delta_k)\|\boldsymbol{x}\|_2^2,$$

for all $\boldsymbol{x} \in \sum_K = \{\boldsymbol{x} : \|\boldsymbol{x}\|_0 \leq K\}$,

where $\|\boldsymbol{x}\|_0$ is the zero pseudo-norm defined as

$$\|\boldsymbol{x}\|_0 = \#(i|x_i \neq 0).$$

If $\boldsymbol{\Phi}$ satisfies the RIP with order $2K$, then $\boldsymbol{\Phi}$ approximately preserves the distance between any pair of K-sparse vectors. Unfortunately the task of checking that a matrix satisfies the RIP is a NP-hard problem, but fortunately the RIP will hold

true with high probability if $\mathbf{\Phi}$ is selected as a random matrix [**?**], and $M \geq cK \log \frac{N}{K}$, where c is a small constant.

C. Recovery Algorithms

Right at the heart of the compressive sensing theory is the ability for recovery algorithms to provide accurate signal estimations in an efficient manner. Recovery algorithms generally fall into two categories, those based on convex optimisation and greedy algorithms. One of the basic algorithms from each category are considered.

1) Convex Optimisation: Convex optimisation is a minimisation problem subject to a number of constraints where the functions involved are convex. According to [**?**], optimisation based on the ℓ_1 norm can exactly recover K-sparse signals and closely approximate compressible signals with high probability using only $M \geq cK \log \frac{N}{K}$ iid Gaussian measurements. The ℓ_1 norm of a vector $\boldsymbol{x}$ is the sum of the absolute values of the elements of $\boldsymbol{x}$ and is defined mathematically in Eq. (2).

$$\|\boldsymbol{x}\|_1 = \sum_{i=1}^{N} |x_i| \tag{2}$$

The use of ℓ_1 minimisation to promote sparsity is not a new idea, the links between ℓ_1 minimisation and sparsity were first noted in the field of geophysics when it was observed [**?**] that minimising the ℓ_1 norm could help detect sparse spike trends in earthquake data. The idea soon caught on, [**?**] utilised ℓ_1 methods in their work on extracting spike trains and eventually in 1996, ℓ_1 minimisation was approached in statistics as LASSO [**?**] (Least Absolute Shrinkage and Selection Operator) was developed. LASSO is a version of least squares with a constraint that the ℓ_1 norm, $\|x\|_1$ cannot be larger than some value ϵ. This penalty encourages more parameters to become zero as it increases, therefore promoting sparsity. Basis Pursuit was developed in [**?**] and is a type of ℓ_1 minimisation algorithm defined as

$$\min_{\boldsymbol{x}} ||\boldsymbol{x}||_1 \text{ subject to } \boldsymbol{y} = \mathbf{\Phi}\boldsymbol{x}. \tag{3}$$

This can be viewed as a least squares problem with an ℓ_1 regularizer. Basis Pursuit is considered to have polynomial complexity but in reality this is not always true for standard optimisation packages as they are not tailored for sparse signal recovery.

In Section IV the ℓ_l magic [**?**] implementation of Basis Pursuit is applied, which solves Eq. (3) by recasting the problem as a primal dual algorithm based on work by [**?**]. The Newton-Raphson method is then applied in order to detect an interior point at which the system is linearised and solved.

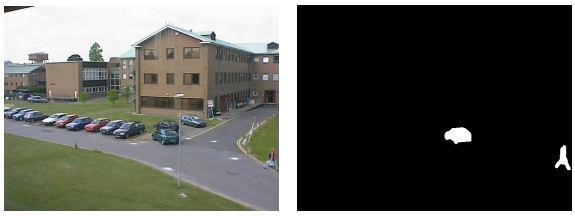
2) Greedy Algorithms: A greedy algorithm iteratively makes decisions based on some locally optimal solution. One of the simplest greedy algorithms suitable for the sparse signal approximation problem is Orthogonal Matching Pursuit. Some of the earliest Orthogonal Matching Pursuit algorithms can be found in [**?**], [**?**], although notable more recent work can be found in [**?**] and [**?**]. This method can often perform faster than Basis Pursuit due to its simplicity. It iteratively computes the local optimum solutions in the hope that these will lead to the global optimum solution. The algorithm determines the column of $\mathbf{\Phi}$ which is most correlated with $\boldsymbol{y}$, or which contributes to $\boldsymbol{y}$ most. This is repeated again by comparing the correlation between columns of $\mathbf{\Phi}$ with the signal residual, until it reaches some stopping criterion defined by the user. This can be described algorithmically as in Algorithm 1.

Algorithm 1 Orthogonal Matching Pursuit

Define the columns of $\mathbf{\Phi}$ to be $\varphi_1, \varphi_2, \ldots, \varphi_N$.
Require: $\boldsymbol{r_0} = \boldsymbol{y}, \Lambda_0 = \emptyset$ and iteration counter $i = 1$
  for $i < T$ **do**
    $\lambda_t = \text{argmax}_{\ j=1,\ldots,N} | < r_{t-1}, \varphi_j > |$
    {Find the index for the column of Φ with the greatest contribution.}
    $\Lambda_t = \Lambda_{t-1} \cup \lambda_t$, $\Phi_t = [\boldsymbol{\Phi_{t-1}}, \varphi_{\lambda_t}]$
    {Keeps track of the columns used.}
    $\boldsymbol{x_t} = \text{argmin}_{\boldsymbol{x}} ||\boldsymbol{y} - \boldsymbol{\Phi_t}\boldsymbol{x}||_2$
    {Updates the signal estimate.}
    $\boldsymbol{r_t} = \boldsymbol{y} - \boldsymbol{\Phi_t}\boldsymbol{x_t}$
    {Updates the measurement residual.}
  end for
  return $\hat{\boldsymbol{x}}$

In the algorithm, $\hat{\boldsymbol{x}}$ is updated for each iteration with the contributions from the columns of $\mathbf{\Phi}$ placed in the indexes stored in Λ_t. So if the algorithm is stopped after the Tth iteration for some positive integer T, $\hat{\boldsymbol{x}}$ will be a T-sparse vector. This emphasises how important it is to run the algorithm for the correct number of iterations, although generally this number is unknown.

Using greedy algorithms can be advantageous for the background subtraction problem as they can be both flexible and speedy. Greedy algorithms are less restrained to a particular form than in ℓ_1 minimisation, therefore greedy algorithms can incorporate constraints which do not fit naturally in a convex formulation. Also, when the signal $\boldsymbol{x}$ is exceptionally sparse, only a few iterations are required, which makes the whole process very fast. [**?**] claims that OMP can recover a K-sparse signal of size N by making only $\mathcal{O}(k \ln N)$ observations of the signal. However, the correlation between the sparsity of the foreground mask and the choice of the stopping criterion means that unless the stopping criterion is adaptive, the algorithm may struggle with varying sparsity as discussed in Section IV. As the signal length and number of measurements increases, so does the computational complexity - particularly in the identification stage. In this stage, the algorithm attempts to find the column φ of $\mathbf{\Phi}$ which is most strongly correlated with the residual. As the number of iterations increases, the sparsity of the signal estimate $\hat{\boldsymbol{x}}$ decreases. Therefore iterating for too long ($T \ll K$) not only leads to a poor estimate but unnecessary computational time.



(a) Test frame (b) Ground truth

Fig. 1: The spatial sparsity of foreground. A frame from the PETS data set [**?**] and the corresponding foreground in white. In this example, less than 1% of the frame is foreground, as N=442,368 and K=3862.

D. Background Subtraction with Compressive Sensing

A major assumption when applying compressive sensing techniques to foreground segmentation is the assumption that the foreground is sparse in the spatial domain [**?**]. This means that without having to apply any sort of sparsifying transformation, the foreground only takes up a small percentage of the number of pixels in the frame. Figure 1 shows a test frame and the true foreground or "Ground Truth" where white pixels represent the true foreground and black pixels represent the true background. The number of white pixels (3862) is much smaller than the number of black pixels (438,506) therefore indicating that the foreground is indeed sparse in the spatial domain. The precise sparsity of the foreground will inevitably vary in different videos and even between frames as discussed in [**?**].

Our method takes a vectorised form of the current frame $\boldsymbol{x_t}$, and acquires compressive measurements $\boldsymbol{y_t}$ of the frame using a Gaussian random matrix $\boldsymbol{\Phi}$. The foreground mask $\boldsymbol{x_t^f}$ is then reconstructed by applying a recovery algorithm to $(\boldsymbol{y_t} - \boldsymbol{y_t^b})$, where $\boldsymbol{y_t^b}$ is a compressed model for the background at time t. The silhouette is then thresholded to set any small values to zero, and the non-zero pixels are classified as foreground.

In order for this method to work well, it is important that a good model of the background is kept updated. Although a static background could be used for short indoor sequences, most real-world video sequences require a dynamic background model. [**?**] suggests that a background model should be able to adapt to deal with illumination changes, high frequency repetitive background objects and changes in background geometry. Illumination changes can be separated into gradual changes, or rapid changes. Gradual changes in illumination are mainly due to the light changing throughout the day. Rapid changes in illumination may occur when clouds cover the sun in outdoor scenes or a light being switched on or off in interior scenes. High frequency background objects are mainly found in outdoor footage such as leaves waving in the wind, water rippling or rainy weather. These small movements are actually part of the background so care should be taken to use an algorithm which detects them as such. Changes in background geometry refers to when part of the background begins to move. Examples of this are very common in real videos, such as a chair being repositioned in a room such as in [**?**] or a car driving into a car park as foreground, and then parking therefore becoming part of the background. In this report the background is modelled using an exponentially weighted moving average method as discussed in [**?**] and [**?**]. This is calculated as in Eq. (4).

$$\boldsymbol{y_t^b} = \alpha \boldsymbol{y_t} + (1-\alpha)\boldsymbol{y_{t-1}^b} \tag{4}$$

where $\alpha \in (0, 1)$ is a learning parameter. This background model was chosen, as it is not computationally heavy, and the single learning parameter α keeps tuning the model simple. The initial background $\boldsymbol{y_0^b}$ is calculated by taking an average of scenes from a training set. If this information is not available it is possible to use only the first frame in the data set, but this may not be as accurate. The background model learning parameter α is tuned so that it is sensitive to the challenges discussed above, specifically to avoid the change in geometry problems. This challenge is especially prominent in the experimentation in Section IV as the video used is CCTV footage from a car park.

IV. Performance Evaluation

The PETS 2001 dataset "camera1" [**?**], is used to compare implementation of the CS_{ℓ_1} and CS_{OMP} algorithms. A quantitative way to characterise the accuracy of these algorithms is to use two performance metrics; precision and recall. These measures calculate how accurate the segmentation is for a particular frame by comparing the estimated foreground and background with the true values calculated by a hand-segmented ground truth.

In order to quantify how well an algorithm is working, Type I errors and Type II errors must be considered. A Type I error is a false positive (FP), this is experienced when the algorithm classifies a pixel as foreground incorrectly. A Type II error is a false negative (FN), this where the algorithm has neglected to classify a pixel as foreground correctly. If the algorithm works perfectly there will only be True Positives (TP) which are correctly identified foreground pixels and True Negatives (TN), correctly identified background pixels. Recall is defined as the fraction of correctly identified foreground pixels over the number of ground truth foreground pixels which can be written mathematically as

$$\text{Recall} = \frac{TP}{TP+FN}. \tag{5}$$

Precision is defined to be the fraction of correctly identified foreground pixels over the number of detected foreground pixels in total, or when written mathematically

$$\text{Precision} = \frac{TP}{TP+FP}. \tag{6}$$

When the recovery algorithms provide an estimate of a foreground mask, the output is not binary but a probabilistic vector. In order to calculate the precision and recall values, a threshold must first be applied to the estimate. Precision-Recall curves can be used to display information relating to the algorithm's performance across a number of thresholds.

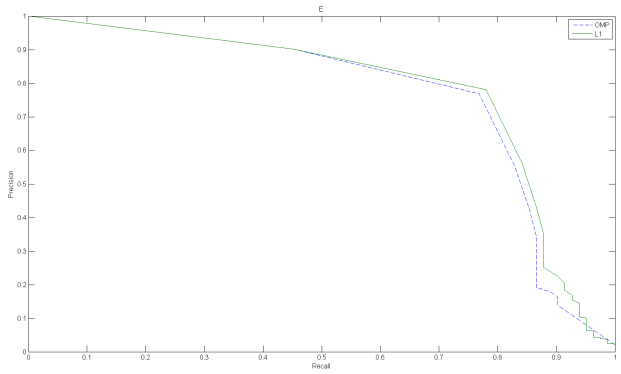


Fig. 2: Precision-Recall Curves for the 3rd test frame

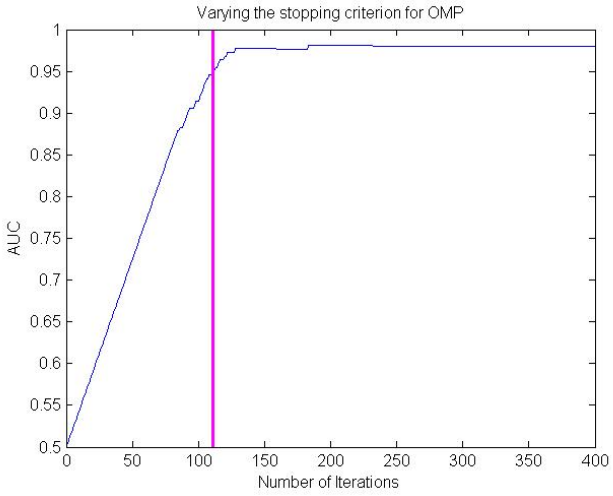
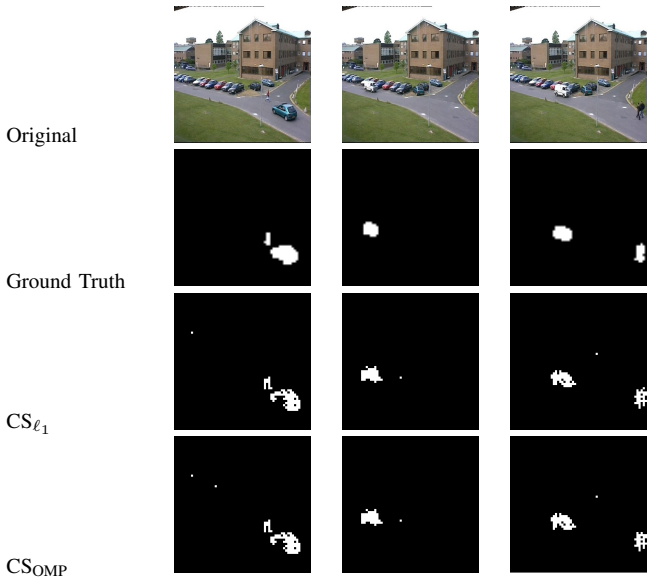


Fig. 3: Selection of the stopping criterion for OMP

In order to have a singular metric to rank the algorithms across a number of thresholds the Area Under Curve (AUC) [**?**] is used. This is equal to the probability that an algorithm will rank a randomly chosen positive instance higher than a randomly chosen negative one.

TABLE I: CS_{ℓ_1} and CS_{OMP} segmentation for 3 test scenes ($N = 4096, M = 2048, \alpha = 0.05$).



Foreground masks for three test frames can be found in Table I. These images have been thresholded so any value above 0.004 is classed as foreground and everything else is set to background. The performance of both algorithms is very similar, although the Precision-Recall curves in Figure 2 imply that CS_{ℓ_1} may be slightly outperforming CS_{OMP} across the different thresholds.

The effect of the stopping criterion for the Orthogonal Matching Pursuit can be seen in Figure 3. The stopping criterion is increased between 1 and 400 and the AUC is calculated at each stage. As discussed in Section III-C2, the selection of the stopping criterion can have an impact on the results; too small and it will underestimate the size of the foreground, too large and computational expense will increase greatly with no noticeable performance improvement. This can be seen in the steep increase of performance in Figure 3. Note that as the number of iterations reaches approximately the "true" sparsity 111 of the foreground, the performance increases. Ideally, the stopping criterion should be as small as possible to minimise the computational cost, but not at the expense of poor segmentation.

Another consideration is the effect of the choice of M on recovery accuracy. Ideally the number of M should be as small as possible in order to keep computations cost low, whilst still retaining a good enough quality of detection to suit the application, whether it be tracking, classification etc. Figure 4 shows the performance of both algorithms as the compression ratio $\frac{M}{N}$ increases, with respect to AUC. There is a sharp rise in performance as the compression ratio increases to 20%, and as we approach a compression rating of 30%, both algorithms are performing very well. Note that CS_{ℓ_1} is constantly outperforming CS_{OMP} in this experiment over all compression ratios.

V. Conclusions and Further Work

This paper presents results about the effects of different compressive sensing recovery algorithms on background subtraction for surveillance footage. In our experimentation Basis Pursuit outperforms Orthogonal Matching Pursuit across the foreground thresholds, although it is hard to distinguish them from foreground masks alone. The effect of the stopping criterion for OMP was seen to have a large impact on performance, which indicates the necessity for adaptive iterations in order to cope with varying sparsity in a video. It is also indicated that ideal boundaries for the compression ratio $\frac{M}{N}$ are between $25 - 35\%$. One important question to address in future is, is it possible to incorporate more prior information about surveillance footage and use this to aid the recovery process? In current methods, there is a failure to use the natural

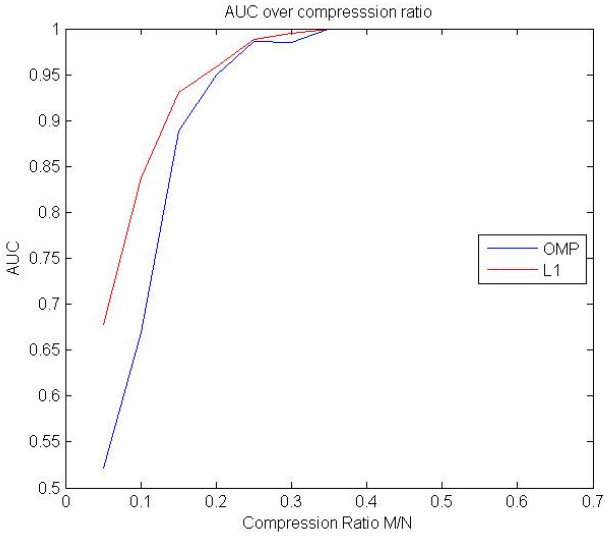


Fig. 4: AUC over Sparsity Ratio

structural properties of foreground, such as the expectation of foreground to take the shape of a person or vehicle. Structured sparsity has not been well explored in the compressive sensing background subtraction literature although a start in this area has been made by [**?**] and [**?**]. Future work seeks to apply a clustering method to these background subtraction algorithms and to build in a natural way to allow the sparsity constraint in the algorithms to vary between frames.

ACKNOWLEDGMENTS

Davies gratefully acknowledges the financial support of the EPSRC through a studentship within the STOR-i Centre for Doctoral Training, grant number EP/H023151/1.