Block-sparse RPCA for Consistent Foreground Detection

Zhi Gao[1], Loong-Fah Cheong[2], and Mo Shan[2]

[1] Interactive and Digital Media Institute, National University of Singapore
gaozhinus@gmail.com
[2] ECE Department, National University of Singapore, Singapore
{eleclf,a0077940}@nus.edu.sg

Abstract. Recent evaluation of representative background subtraction techniques demonstrated the drawbacks of these methods, with hardly any approach being able to reach more than 50% precision at recall level higher than 90%. Challenges in realistic environment include illumination change causing complex intensity variation, background motions (trees, waves, etc.) whose magnitude can be greater than the foreground, poor image quality under low light, camouflage etc. Existing methods often handle only part of these challenges; we address all these challenges in a unified framework which makes little specific assumption of the background. We regard the observed image sequence as being made up of the sum of a low-rank background matrix and a sparse outlier matrix and solve the decomposition using the Robust Principal Component Analysis method. We dynamically estimate the support of the foreground regions via a motion saliency estimation step, so as to impose spatial coherence on these regions. Unlike smoothness constraint such as MRF, our method is able to obtain crisply defined foreground regions, and in general, handles large dynamic background motion much better. Extensive experiments on benchmark and additional challenging datasets demonstrate that our method significantly outperforms the state-of-the-art approaches and works effectively on a wide range of complex scenarios.

1 Introduction

Moving object detection plays a critically important role in applications such as video surveillance and smart environment. Despite much effort in this direction, a recent evaluation of major techniques for video surveillance [2] showed that hardly any approach can reach more than 50% precision at recall level higher than 90%. Designing an algorithm that is robust under a wide variety of scenes encountered in complex real-life applications remains an open problem.

For cameras that are mounted and are more or less stationary, background subtraction method is a major class of technique used to detect moving objects. Essentially in such methods, video frames are compared with a background model; changes are then identified as the foreground. The background models can be estimated via a parametric approach [15, 21] or a non-parametric approach

[7, 12], or just in the form of thresholding [8]. For a more detailed discussion of these techniques, readers can refer to a recent survey [3].

The reason why these methods fail to work well in realistic complex situation is that these methods often make overly restrictive assumptions about the background. In reality, the background itself can have complex changes. It might contain motion such as those caused by ripples on a lake, or swaying vegetation, which can cause false alarms. The motion of these backgrounds can be larger than that of the foreground. There could be sudden illumination change caused by cloud cover, causing complex intensity and shadow variation, or more gradual illumination change caused by the movement of the sun. During dawn and dusk hours, the image quality can be poor due to the low light condition. In view of these complex factors, it is very difficult to model the background well.

In this paper, we handle all these challenges by making very little specific assumptions about the background. The only assumption made about background is that any variation in its appearance (whether caused by intensity change, or non-stationary background) are highly constrained and can be captured by the low rank condition of a suitably formulated matrix. In its simplest form, we say that a matrix $\boldsymbol{M}$ composed of the observed vectorized image frames can be decomposed into a low-rank matrix $\boldsymbol{L}$ representing the background, and a sparse matrix $\boldsymbol{S}$ consisting of the moving objects treated as sparse outliers. Detecting the moving objects amounts to solving the following problem:

$$\min_{\boldsymbol{L},\boldsymbol{S}} \operatorname{rank}(\boldsymbol{L}) + \lambda ||\boldsymbol{S}||_0, \quad \text{s.t. } \boldsymbol{M} = \boldsymbol{L} + \boldsymbol{S} \tag{1}$$

where λ is an appropriate regularizing parameter. This is also known as the Robust Principal Component Analysis (RPCA) problem [4]. In the next section, we will first discuss the inadequacy of such formulation, and then introduce the necessary steps that will lead to substantially better results than other RPCA related formulations and other state-of-the-art background subtraction techniques.

2 Background and System Overview

In the earliest models using low rank matrix to represent background [4, 5], no prior knowledge on the spatial distribution of outliers was considered. In real videos, the foreground objects are usually spatially coherent clusters. Thus, contiguous regions should be preferably detected. Such prior has been incorporated into some recent works through the MRF prior [20]; however the result of imposing such smoothness constraint (even with the so-called discontinuity preserving prior such as those based on Potts model) is that foreground region tends to be over-smoothed. For instance, the detailed silhouette of the hands and legs of a moving person is usually sacrificed in favor of a more compact blob. The idea in our paper is related to the so-called block-sparsity or joint-sparsity measures to incorporate spatial prior. However, these works [6, 16] typically assume that the block structure is known. Rosenblum's [14] method does not require prior knowledge on the block size and block location, which are instead detected by

iteratively alternating between updating the block structure of the dictionary and updating the dictionary atoms to better fit the data. Nevertheless, both the number of blocks and the maximal block size are assumed to be known. In [9, 13], the sparsity structure is estimated automatically. However, in [9], parameter tuning is required to control the balance between the sparsity prior and the group clustering prior for different cases, and both these algorithms share the same limitation that training sequences composed of clean background are required. In contrast, our method estimates the sparsity structure automatically without a separate training phase.

Before describing how to automatically detect the blocks containing moving objects, we want to discuss the issue of scale. The scale issue is present in the preceding RPCA related formulation because there is no one value of λ, the regularizing parameter, that can handle foreground objects of all kind of sizes (λ controls the amount of outliers in the RPCA decomposition, and thus is related to the scale issue). This issue is in fact a perennial challenge in many segmentation problems. As an example, in the well-known Normalized Cut algorithm, there often cannot be a single correct segmentation of an image unless it has single prominent object. Let us take an example of a scene shown in Figure 4, the tree is much larger in size than the human, and its apparent image motion is also larger due to its proximity to the camera. For such a scene with prominent objects appearing at significantly different scales, having a single global parameter for segmenting the scene (whether in the sense of image segmentation, or in the present case, motion segmentation) is not even meaningful. While the block-sparsity approach to a certain extent can relieve this scale problem (by having blocks of different sizes), it does not fundamentally remove the problem, especially when there is a lot of large background motion.

The root of this problem lies in that the precise definition of the foreground target is intricately linked with the object of interest in the scene (i.e. one's purpose) and can be well defined only if the object of interest or its salient characteristics is known to us. However, knowing about the object of interest even before segmenting the scene seems to make the problem as one of many chicken-egg problems in computer vision, as we usually need to segment the scene to recognize the objects in it. So, how can we identify an object and its probable size even before segmenting it?

Clearly this must involve a feedback process, either implicitly or explicitly. In this paper, we put forth a hierarchical two-pass process to solve the aforementioned problems. The first-pass RPCA rapidly identifies the likely regions of foreground in a sub-sampled image. A simple motion consistency scheme is then used to measure the motion saliency of these foreground regions. Then in the second pass, a block-sparse RPCA imposes the spatial coherence of foreground objects in the outlier matrix $\boldsymbol{S}$, with the λ value set according to the motion saliency. Taking into account the motion salience and the block-sparse structure of the outlier matrix $\boldsymbol{S}$ makes the foreground detection robust against the clutter caused by the background motion, and largely invariant to object size, allowing us to return crisply defined foreground regions. As opposed to formulating

the whole problem into a single optimization function (as in the case of [9, 13]), we favor the explicit modeling of this feedback process. Not only this achieves greater modularity of different processes and ensures convergence, it also allows greater flexibility in the design of the motion consistency measure (other domain specific constraints can be readily accommodated too). This gives us a greater advantage when decomposing scenes with complex background motion, as can be seen from the experimental results later.

We test our algorithm on background with a wide range of dynamic texture motions with varying magnitude, and also on different sizes of foreground objects. Other challenging conditions as categorized in [2] (illumination change, high noise in dim environment, etc.) are also tested, using the evaluation data set of [2], as well as additional challenging sequences. In all cases, we are able to achieve accurate detection and clean delineation of targets, with significant improvement over those achieved by the state-of-the-art techniques.

3 Our Algorithm

3.1 First-pass RPCA

As discussed in the preceding section, our proposed approach is based on a two-pass RPCA process. First we make a rapid weak identification in the form of rough region(s) of interest by performing a first-pass RPCA on a scaled-down low resolution (sub-sampled at a four to one ratio). This step is based on a simple convex relaxation of equation (1):

$$\min_{\boldsymbol{L},\boldsymbol{S}} ||\boldsymbol{L}||_* + \lambda||\boldsymbol{S}||_1, \quad \text{s.t. } \boldsymbol{M} = \boldsymbol{L} + \boldsymbol{S} \tag{2}$$

where $||\boldsymbol{L}||_*$ is the nuclear norm of matrix $\boldsymbol{L}$, the sum of its singular values, and λ set at a value that ensures no genuine foreground regions will be missed. The inexact augmented Lagrange multiplier (ALM) method [10] is used to solve this problem. We find that the recommended value of $\lambda = 1/\sqrt{max(m,n)}$ (where $m \times n$ are the dimensions of $\boldsymbol{M}$) is adequate to identify all foreground regions, including possibly many background regions.

As we find in our experiments later, while the low-rank formulation of the background matrix is generally effective in absorbing many natural variations in the background (such as illumination change), the full power of the RPCA framework to achieve accurate decomposition can only be realized if we have a more subtle mechanism to handle the aforementioned issue of scale and to capture the salient aspect of the expected foreground motions. As it is, not only significant number of non-stationary background points are being deposited in the outlier matrix, faint trace of the foreground motion is often also retained in the background matrix (see the ghostly presence of the walking person in the inset of Figure 4, top right). It is evident that there is no single λ that can achieve a clean separation of the foreground and background regions.

Referring to the middle figure of Fig. 1, all those non-white pixels are the outliers estimated via the first-pass RPCA. These outliers typically comprise

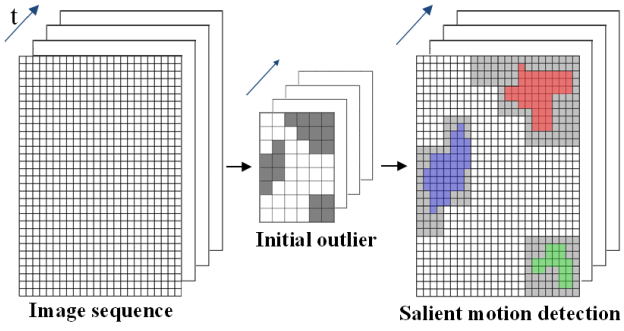


Fig. 1. Middle: Initial outlier detection via a first-pass RPCA on a sub-sampled image sequence. Right: The remaining computation is carried out in original resolution; gray pixels represent background objects performing non-consistent motion; those pixels in color denote foreground objects with salient motions.

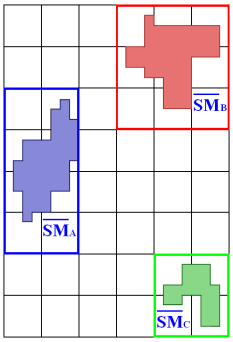


Fig. 2. Foreground regions with different motion saliency measure $\overline{SM}_A, \overline{SM}_B, \overline{SM}_C$. Here, each grid is not a pixel but a 4x4 block.

of both background and foreground objects. Note that pixels identified in this process as outliers correspond to a 4×4 blocks in the full resolution image. A simple block merging step is then carried out to group those spatially connected 4×4 blocks into a larger rectangular block that encompasses all of them (Fig. 2).

3.2 Motion saliency estimation

The likelihood of a block containing foreground motions is measured by a method similar to that proposed by [19]. The basic idea is to track all pixels within the blocks detected in the initial round of RPCA via dense optical flow. Only those trajectories that are moving in a consistent direction are retained as likely candidates of foreground motions. Each block is now weighted by the average distance over which the consistent trajectories in this block have traveled.

Our method differs from that of [19] in the following ways. Instead of using the KLT tracker, we applied the latest "Classic + NL" algorithm [17] to obtain dense optical flow, from which we can assess the motion consistency of an entire region rather than just a set of sparse features. It also suits our purpose that this algorithm is robust to ambient illumination change. Another difference with [19] is that we attempt to track a point for as long as possible, and compute the motion consistency based on the entire trajectory, rather than just on a window of about ten frames. This is important for handling "periodic" background motion whose period might be longer than ten frames. The details of the tracking procedure based on the optical flow field $(u, v)^T$ are described in the Appendix. Note that the tracking is done only at the pixel level. There is no tracking at the block level; thus, the saliency of the block is entirely dependent on the saliency of constituent trajectories passing through the block at that instant.

After tracking, dense point trajectories are obtained and we use $X_{l,j}, X_{l,j+1}, \ldots X_{l,k}$ to denote the successive position of points along the l-th trajectory from the j-th frame till the k-th frame ($j, k \in [1, n]$ depends on l). We first remove those trajectories that are too short, specifically, $k - j \leq 10$. We then adapt the algorithm of [19] to compute the motion saliency of the remaining trajectories, based on the consistency of the motion direction (see Algorithm 1).

Algorithm 1 Motion Saliency Computation.

Input: Trajectory $X_{l,j}, X_{l,j+1}, \ldots X_{l,k}$;
Output: Motion saliency of the trajectory;
1: Initialize counters: $P_u = P_v = N_u = N_v = 0$;
2: Counting consistency of horizontal flow direction u:
for $t = j$; $t <= k$; $t{+}{+}$
if $(u_t(X_{l,t}) > 0)\, P_u = P_u + 1$;
if $(u_t(X_{l,t}) < 0)\, N_u = N_u + 1$;
end for
3: Repeating previous operation on the vertical component v, and obtain P_v, N_v;
4: If one of P_u, P_v, N_u or N_v is greater than $0.8(k - j)$, the trajectory belongs to an object with salient motion, with its saliency measure given by $SM_l = \max|X_{l,t_1} - X_{l,t_2}|$, for all $(t_1, t_2 \in [j, k])$; otherwise mark the trajectory as inconsistent.

If a particular trajectory is moving in a consistent manner, the last step of Algorithm 1 defines the saliency measure SM to be proportional to the displacement travelled by that pixel. This serves to further enhance the detectability of those foreground motion moving in a slow but consistent manner for a sustained period of time. In general, slow motion causes a smaller rank change to the matrix L and is more liable to be missed if its associated λ is not suitably compensated. Conversely, inconsistent motions of the background that result in small local displacement will be further discounted if they are not already removed in the preceding step. Finally, for block i, we average the SM of all consistent trajectories that pass through this block, and denote the average as $\overline{SM}_i$.

3.3 Second-pass RPCA

Now that most of the non-stationary background motions have been filtered off or suppressed by the motion consistency check, we can afford to lower the regularizing parameter λ in the second-pass RPCA step. This would ensure that all the changes caused by the foreground motion will be entirely transferred to the outlier matrix and not leave any ghostly presence in the background, yet without incurring a large false positive rate. Thus, for all blocks, we lower λ by at least one order of magnitude compared to before. For all block i with consistent motion, we set $\lambda_i = \frac{0.1}{\sqrt{max(m,n)}} \frac{\overline{SM}_{min}}{\overline{SM}_i}$ where the last factor normalizes the $\overline{SM}_i$ computed for this block with respect to the minimum $\overline{SM}$ detected among all

blocks containing consistent motions. For blocks with no consistent motion, the λ_i's are set to arbitrarily large values.

With the location and size of the likely outlier blocks estimated, and each weighted by a different saliency measure (Fig. 2), we are ready to carry out the second pass RPCA.

$$\min_{\boldsymbol{L},\boldsymbol{S}} ||\boldsymbol{L}||_* + \sum_i \lambda_i ||\boldsymbol{P}_i(\boldsymbol{S})||_F, \text{ s.t. } \boldsymbol{M} = \boldsymbol{L} + \boldsymbol{S} \tag{3}$$

where $||\cdot||_F$ is the Frobenius norm of a matrix, and $\boldsymbol{P}_i$ is an operator that unstacks each column of $\boldsymbol{S}$, and then returns a matrix that represents block i. Essentially, this is the block-sparse version of the conventional RPCA. Equation (3) remains a convex optimization problem and we again solve it via the inexact ALM method. Due to space constraint, readers should refer to [10] for details. Briefly, the augmented Lagrangian function is defined as:

$$f(\boldsymbol{L},\boldsymbol{S},\boldsymbol{Y},\mu) = ||\boldsymbol{L}||_* + \sum_i \lambda_i ||\boldsymbol{P}_i(\boldsymbol{S})||_F + \langle \boldsymbol{Y}, \boldsymbol{M} - \boldsymbol{L} - \boldsymbol{S} \rangle + \frac{\mu}{2} ||\boldsymbol{M} - \boldsymbol{L} - \boldsymbol{S}||_F \tag{4}$$

where $\boldsymbol{Y}$ is the Lagrange multiplier, μ is a positive scalar. For this block-sparse RPCA, besides the usual soft-thresholding operator $S_{\epsilon_i}[\cdot]$ needed for the minimization with respect to $\boldsymbol{L}$, we need the following block shrinkage (BS) operator during the minimization with respect to $\boldsymbol{S}$:

$$BS_{\epsilon_i}[\boldsymbol{G}_i] = \begin{cases} \frac{||\boldsymbol{G}_i||_F - \epsilon_i}{||\boldsymbol{G}_i||_F} \boldsymbol{G}_i & \text{if } ||\boldsymbol{G}_i||_F > \epsilon_i, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

where $\boldsymbol{G}_i$ is a matrix representing the block i, and $\epsilon_i = \lambda_i \mu_k^{-1}$ is the corresponding threshold for this block. This shrinkage operator taken over all blocks has been proven to be the closed-form solution of the minimization with respect to $\boldsymbol{S}$ [18]. We summarize the algorithm as follows:

In Algorithm 2, $\mu_0 = 1.25/||\boldsymbol{M}||_2$, $\rho = 1.6$ and $J(\boldsymbol{M}) = max(||\boldsymbol{M}||_2, \lambda^{-1}||\boldsymbol{M}||_\infty)$. Herein, $||\cdot||_2$ and $||\cdot||_\infty$ are the spectral norm and the l_∞ matrix norm respectively, and λ is set at $0.1/\sqrt{max(m,n)}$. The criteria for convergence at step 2 is $||\boldsymbol{M} - \boldsymbol{L}_k - \boldsymbol{S}_k||_F / ||\boldsymbol{M}||_F < 10^{-7}$.

4 Experiments and Analysis

We now perform experiments on both real and synthetic sequences. In the former, only qualitative assessment can be made at a few particular frames. Nevertheless, as will be seen, the superiority of our method is quite clear by visual examination of the foreground detected. For the latter, the ground truth available from [2] allows us to quantitatively evaluate the performance over the entire sequence, in terms of *recall* and *precision*:

$$recall = \frac{\text{correctly classified foreground}}{\text{foreground in ground truth}}, \; precision = \frac{\text{correctly classified foreground}}{\text{pixels classified as foreground}}$$

Algorithm 2 Block-sparse RPCA via Inexact ALM.

Input: Matrix $\boldsymbol{M} \in \mathbb{R}^{m \times n}$, all salient blocks b_i, and the corresponding λ_i;
Output: Estimate of $(\boldsymbol{L}, \boldsymbol{S})$;
1: Initializing: $\boldsymbol{Y}_0 = \boldsymbol{M}/J(\boldsymbol{M})$, $\boldsymbol{S}_0 = 0$, $\mu_0 > 0$, $\rho > 1$, $k = 0$;
2: **while** not converged **do**
3: //Lines 4-5 solve $\boldsymbol{L}_{k+1}$=arg $\min_{\boldsymbol{L}} f(\boldsymbol{L}, \boldsymbol{S}_k, \boldsymbol{Y}_k, \mu_k)$, as equation (4).
4: $(\boldsymbol{U}, \boldsymbol{\Lambda}, \boldsymbol{V})$=svd$(\boldsymbol{M} - \boldsymbol{S}_k + \mu_k^{-1}\boldsymbol{Y}_k)$;
5: $\boldsymbol{L}_{k+1} = \boldsymbol{U}S_{\mu_k^{-1}}[\boldsymbol{\Lambda}]\boldsymbol{V}^T$.
6: //Line 7 solves $\boldsymbol{S}_{k+1}$=arg $\min_{\boldsymbol{S}} f(\boldsymbol{L}_{k+1}, \boldsymbol{S}, \boldsymbol{Y}_k, \mu_k)$.
7: Block-wise shrinkage: $\boldsymbol{S}_{k+1} = BS_{\lambda_i\mu_k^{-1}}[\boldsymbol{M} - \boldsymbol{L}_{k+1} + \mu_k^{-1}\boldsymbol{Y}_k]$;
8: $\boldsymbol{Y}_{k+1} = \boldsymbol{Y}_k + \mu_k(\boldsymbol{M} - \boldsymbol{L}_{k+1} - \boldsymbol{S}_{k+1})$, $\mu_{k+1} = min(\rho\mu_k, 10^7\mu_0)$.
9: $k \leftarrow k + 1$.
10: **end while**
11: Output $(\boldsymbol{L}_k, \boldsymbol{S}_k)$

We compare our algorithm with the three best performing algorithms in [2], namely *Zivkovic*[21], *Barnich*[1], *Maddalena*[11], following the naming convention of [2]. We augment the above list with the KDE (Kernel Density Estimate) non-parametric method [7], as it is proposed specifically to deal with complex dynamic scenes but has not been included in [2]. We also compare our algorithm with two RPCA-based methods as these methods are closest to ours in spirit: the PCP of [4], and the DECOLOR of [20] which combines RPCA and MRF, and is claimed to outperform PCP [1].

4.1 Qualitative Results

Fig.3 shows the detected foreground masks on eight real image sequences, the first four from public datasets used in previous works, followed by four collected by us to demonstrate specific issues addressed by our research.

The first two columns (people walking) are basically the baseline "easy" cases, where most algorithms performed well, though with different degrees of details in the foreground detected. The third column depicts a boat scene with wave motion. This presents difficulties for many algorithms; only ours and *Maddalena* performed relatively well. Note that both the large speedboat in the middle and the small speedboat in the far distance are detected by our method, demonstrating its scale-invariance. *Maddalena* did not manage to pick up the distant boat as clearly as ours. Note that there is also a distant sailboat which

[1] Note that for the PCP and our method, a thresholding step is required to produce the final foreground mask, as many entries in $\boldsymbol{S}$ may contain vanishingly small values. To obtain a threshold, we first identify the likely locations: those pixels whose corresponding entries in $\boldsymbol{S}$ have magnitudes less than half of the maximum magnitude of $\boldsymbol{S}$ are regarded as background, then obtain the difference between $\boldsymbol{M}$ and $\boldsymbol{L}$ at those tentatively identified background locations to estimate the expected level of noise. Finally, we set the threshold at the mean of the difference values plus three standard deviations of those difference values and apply it to $\boldsymbol{S}$.

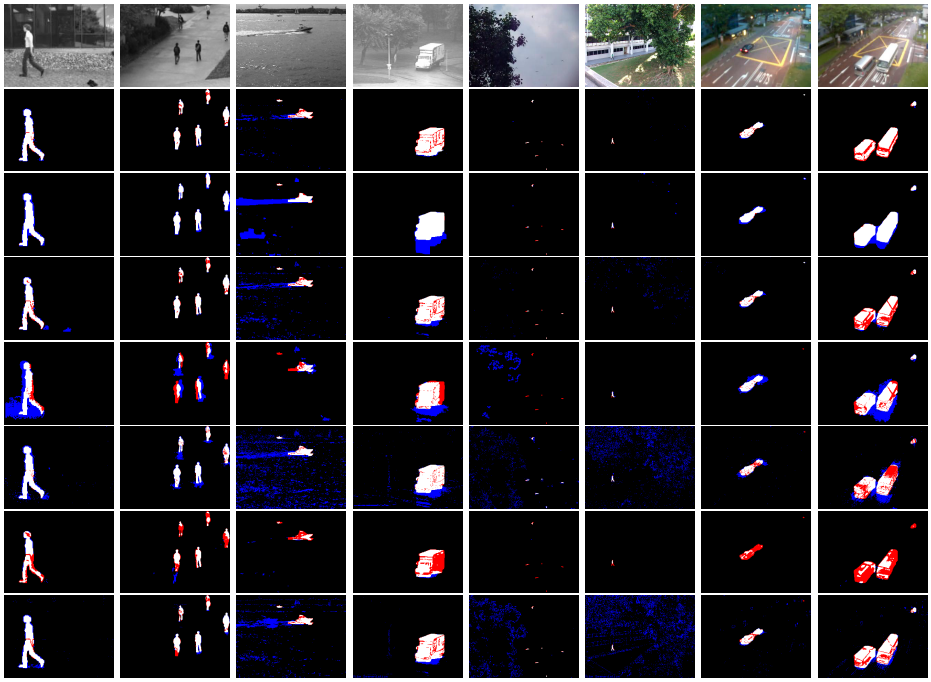


Fig. 3. Detected foreground mask of eight sequences depicted in each column. See online version for details. One image frame of the sequence is shown in the top row. Rows 2 to 8 depict the results of our method, DECOLOR, PCP, KDE, *Zivkovic*, *Maddalena*, *Barnich* respectively. For better visualization, we have manually tagged the foreground in these frames and adopt the following color coding scheme: white represents correctly detected foreground, red missing pixels, and blue false alarm.

moved few pixels over the entire sequence; we are hardly able to judge whether it actually moved or not and correspondingly, our algorithm returned a small motion saliency value for this sailboat and did not regard it as a foreground motion. Some methods detected this sailboat instead of the smaller speedboat. The fourth and the last columns depict rainy scenes where falling rain causes "motion" in the background. Evaluated over the two scenes, our method again produced the best results. *Maddalena* is able to remove the background clutter very well, but it tends to produce incomplete foreground object, as well as missing small moving object (last column). The fifth column depicts a swaying tree scene with various small birds flying. The three best performing methods in [2], namely, *Zivkovic*, *Barnich*, and *Maddalena*, all failed to do well. Either they picked up the tree motions, or failed to detect the birds. The KDE method that is introduced to handle dynamic background also failed quite badly in this sequence. The RPCA-based methods did better, with our method correctly picking out all the birds as foreground without any of the tree motion. These positive results also showed that our motion saliency measure can handle zigzagging ma-

noeuvres, as these swiftlets hurled themselves after evading insect preys. Most such evasive manoeuvres have an overall direction of move, that is, they result in a net movement in certain directions, thus meeting the criteria for motion saliency. The sixth column depicts the campus scene discussed in the preceding section, with a large swaying tree and a walking person. Due to its proximity to the camera, the magnitude of the tree motion is larger than that of the human. As can be seen from the results of our method, the human silhouette is cleanly delineated. For other methods, only the KDE and *Maddalena* did not have any false alarms, but the human silhouettes were not as cleanly detected. The problems faced by other RPCA-based method will be further commented upon in the next figure. Lastly, the seventh column depicts an evening scene with high image noise and flickering illumination caused by fluorescent lights. The sudden change in intensity caused by the varying illumination is not a problem for the RPCA-based methods, as the effect of the change is entirely captured by the low rank constraint. The KDE and *Maddalena* can also handle the sudden change in illumination well, but both of them missed the small car at the top right corner of the image. In both the seventh and the eighth columns, our method can also detect small camouflaged cars moving in the carpark behind the row of roadside trees (not shown). From the above, the qualitative conclusions that we can draw are: (1) despite the claim made in [2] that the best performing algorithms can handle dynamic background (see next set of experiment), this is not true when the background motion is large enough, as can be seen from some of the cases tested here; (2) While KDE and *Maddalena* can handle dynamic background well, they fail in some cases, and the foreground objects returned are often incorrect in shape, incomplete, or missed altogether (especially if the objects are small); (3) RPCA-based methods can handle various background changes quite well generally; these include illumination changes, changes caused by rain, tree and wave motions. Our method and PCP tend to produce cleanly delineated foreground shapes; this is typically not the case for DECOLOR due to the MRF smoothness prior imposed on the foreground shapes.

Despite the general success of the RPCA-based methods, PCP and to a lesser extent, DECOLOR have problems in setting a correct regularizing parameter that can handle regions or motions of varying scales, which is what we set out to overcome. We now explore this point in greater depth in Fig.4. It can be seen that no matter how the value of λ is chosen, we cannot obtain a simultaneously satisfactory background and foreground for PCP. On the one hand, when λ is small (middle column), no ghost of the foreground is detected in the recovered background but unfortunately, much clutter remains in the foreground. On the other hand, setting a somewhat larger λ (right column) has the undesirable effect of putting some of the genuine foreground in the background, even when the foreground itself still contains many false alarm. When viewed in video sequence, the effect of a ghostly presence walking in the background when the human is not cleanly removed is much clearer. For our results, there is no ghost and the motion of the swaying tree is largely retained in the background, creating a pleasing video edit that has little artifacts.

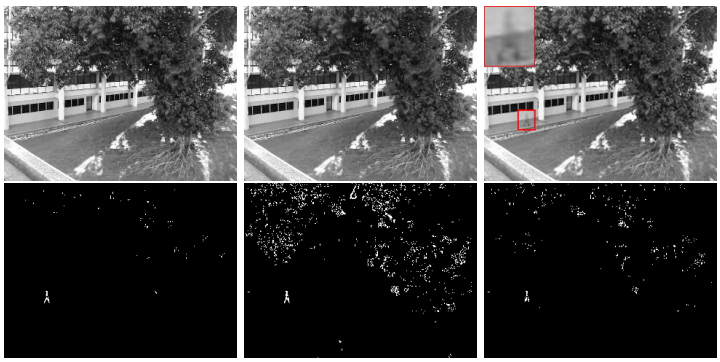
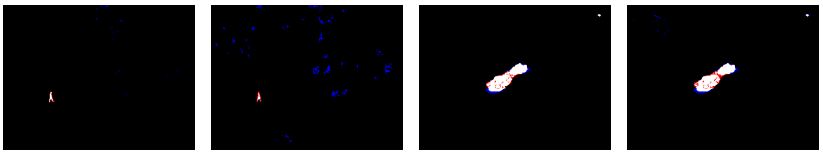


Fig. 4. Recovered background and foreground (top and bottom rows respectively). Left: our method; middle: PCP with $\lambda = 1/\sqrt{max(m,n)}$; right: PCP with $\lambda = 2/\sqrt{max(m,n)}$. Top right, inset: ghostly presence left in the background.



(a) (b) (c) (d)

Fig. 5. Detected foreground masks of our method ((a),(c)), and ReProCs ((b),(d)) on seq.6 and 8 in Fig.3.

The results show that the spatial contiguity prior that we incorporate in the form of blocks and the motion saliency measure have been quite effective in handling the aforementioned issues. Unlike DECOLOR which enforces the MRF constraint, our method is sensitive to small targets and details; it does not suffer from merging of adjacent objects nor inflation of foreground area.

Finally we compare our method against those that estimates the sparse outlier support automatically, in particular, ReProCS [13]. The comparison is done on sequences 6 and 8, as only they have clean background available for training. As shown in Fig.5, our method outperforms ReProCS in sequence 6 significantly and in sequence 8 slightly so. This corroborates our previous claim that our explicit two-pass method can handle complex background motion better.

4.2 Quantitative Results

The dataset in [2] is divided into nine test categories, with scenarios such as gradual and sudden illumination changes, camouflage, dynamic background, etc. The conclusion of that paper is that most techniques showed good performance for many challenges, but they failed in the more demanding experiments. *Light Switch* (sudden illumination change) and *Noisy Night* (dim ambient lighting)

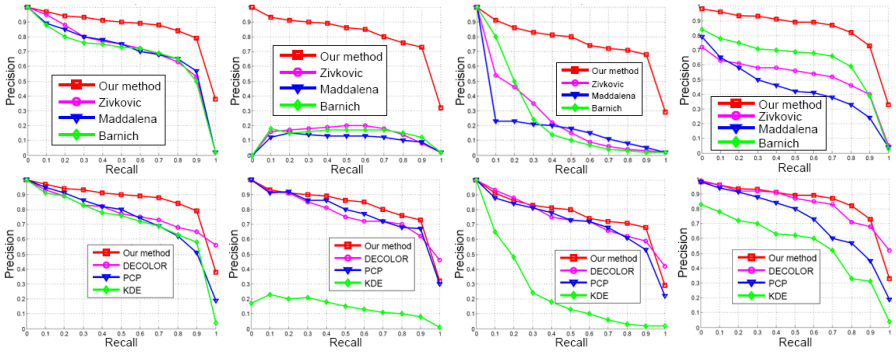


Fig. 6. Precision-recall charts of the performance of different methods with varying threshold. 1st column: Dynamic Background; 2nd column: Light Switch; 3rd column: Noisy Night; 4th column: Camouflage.

were clearly the most difficult sequences. Besides testing on these two sequences, we also include the cases of dynamic background and camouflage as these are the key focus of our paper. It is worth pointing out that the dynamic background motion created in the synthetic dataset is really quite small in magnitude. The relatively good performance reported in [2] is clearly not borne out by the results shown in our preceding qualitative experiments on real sequences, with its larger and more realistic motion. Other categories are either very easy for most techniques or peripheral to our concern here and are hence not included.

Fig.6 shows the precision-recall charts of the performance of different methods with varying thresholds. It is obvious that our method significantly outperforms other methods, reaching more than 70% precision at recall higher than 90%. As commented before, the dynamic background motion of these synthetic sequences is much smaller than some of the real sequences tested in Fig.3; otherwise, the superiority of our method in this case will be even more conspicuous. The RPCA-based methods generally perform better than the others; among these RPCA-based methods, our method is the clear winner. However, at high recall rates (greater than 90%), DECOLOR performs better than ours. This is due to the fact that the MRF-based smoothness prior of DECOLOR ensures that, as the threshold is varied, additional foreground points only grow around the core foreground regions already detected. This feature ensures that the additional points detected are not some background clutter; it is indeed a virtue against which we cannot prevail unless we are willing to forego the scale-independent detection ability of our algorithm. That is, if we accept that once beyond certain threshold, additional foreground points can only come in the blocks where currently detected foreground regions reside, we are essentially giving up on any remaining small, inconspicuous foreground objects that are still not detected. This can readily be done, if specific application needs so dictate. To conclude, the quantitative assessment further corroborates the results obtained in the qual-

itative experiments: our algorithm works well under a wide variety of scenes and significantly outperforms several state-of-the-art techniques.

5 Conclusion

To handle the complex scenarios encountered in background subtraction work, we propose a hierarchical RPCA process which makes little specific assumption about the background. The two-pass RPCA process is interleaved with a motion saliency estimation step that makes our method yield substantially better results than conventional RPCA. We are able to incorporate the spatial contiguity prior in the form of blocks whose size and locations are detected automatically, without having to resort to smoothness prior such as MRF, thereby fully realizing the potential of the low rank representation method to return scale-independent, crisply defined foreground regions. Extensive experiments on challenging videos and benchmark dataset demonstrate that our method outperforms various state-of-the-art approaches and works effectively on a wide range of complex scenarios.

6 Appendix: Point Tracking

Point $(x_t, y_t)^T$ can be tracked to the next frame by using the flow field $\boldsymbol{w}(x, y) := (u(x, y), v(x, y))^T$ that $(x_{t+1}, y_{t+1})^T = (x_t, y_t)^T + (u_t(x_t, y_t), v_t(x_t, y_t))^T$. Given that $(x_{t+1}, y_{t+1})^T$ usually ends up between grid points, we use bilinear interpolation to infer the flow. In new frame, we initialize new tracks at those points where no existing trajectories passed nearby (within 0.1 image pixel unit).

We stop the tracking of a point as soon as it gets occluded. We detect occlusions by checking that the backward flow vector should point in the inverse direction as the forward flow vector: $w_t(x_t, y_t) = -\hat{w}_t(x_t+u_t, y_t+v_t)$, where $\hat{w}_t(x, y) := (\hat{u}_t(x, y), \hat{v}_t(x, y))$ denotes the flow from frame $t+1$ to frame t. If the consistency requirement is broken, the point is either being occluded or the flow was not correctly estimated. Since small optical flow estimation errors are inevitable, we allow the following tolerance factor in the consistency check: $|\boldsymbol{w}(x, y) + \hat{\boldsymbol{w}}(x, y)|^2 < 0.01(|\boldsymbol{w}(x, y)|^2 + |\hat{\boldsymbol{w}}(x, y)|^2) + 0.2$.

For points on motion boundaries, the tracking may not be stable, with the undesirable result that the tracked points wander back and forth across the boundary. If not handled properly, such feature points might come to be regarded as undergoing inconsistent motion due to this drifting across the boundary. To ameliorate this effect, we perform two mitigating measures: Firstly, if the tracking of a point near an edge experiences substantial variation in its motion, we at once stop the tracking and re-initialize a new feature track from this frame onwards. Specifically, the conditions for stopping the tracking are:

$$|\nabla u|^2 + |\nabla v|^2 > 0.01|\boldsymbol{w}|^2 + 0.002, \text{ and } \sqrt{(\partial I/\partial x)^2 + (\partial I/\partial y)^2} > T_d$$

where T_d is the average edge strength. Secondly, before the second pass of RPCA, a border five pixels in width is added to each rectangular block, extending the coverage so that no outlier pixels are missed.

Acknowledgments. This work is supported by these grants: NRF2007IDM-IDM002-069 from MDA of Singapore, "NUS Living Lab" R-705-000-030-133, and "Detecting Moving Targets in Dynamic Background from a Mobile Platform" from DRTECH.

References

1. O. Barnich and M. Van Droogenbroeck. 2009. ViBE: A powerful random technique to estimate the background in video sequences. In *ICASSP*.
2. S. Brutzer, B. Hoferlin, and G. Heidemann. 2011. Evaluation of Background Subtraction Techniques for Video Surveillance. In *CVPR*.
3. A. Bugeau and P. Perez. 2009. Detection and segmentation of moving objects in complex scenes. *CVIU*, 113(4):459-476.
4. E. Candes, X. Li, Y. Ma, and J. Wright. 2011. Robust principal component analysis? *J. ACM*, 58(3):1-37.
5. V. Cevher, A. Sankaranarayanan, F. Duarte, D. Reddy, G. Baraniuk, and R. Chellappa. 2008. Compressive sensing for background subtraction. In *ECCV*.
6. Y. C. Eldar, P. Kuppinger, and H. Bolcskei. 2010. Block-sparse signals: Uncertainty relations and efficient recovery. *IEEE Trans. Sig. Proc*, 58(6):3042-3054.
7. A. Elgammal, D. Harwood, and L. Davis. 2000. Non-parametric model for background subtraction. In *ECCV*.
8. R. Fablet, P. Bouthemy, and M. Gelgon. 2005. A maximality principle applied to a contrario motion detection. In *ICIP*.
9. J. Huang, X. Huang, and D. Metaxas. 2009. Learning with dynamic group sparsity. In *ICCV*.
10. Z. Lin, M. Chen, and Y. Ma. 2009. The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrix. In *Arxiv preprint arxiv.org/abs/1009.5055*.
11. L. Maddalena and A. Petrosino. 2008. A self-organizing approach to background subtraction for visual surveillance applications. *IEEE Trans. Image. Proc*, 17(7):1168-1177.
12. A. Mittal and N. Paragios. 2004. Motion-based background subtraction using adaptative kernel density estimation. In *CVPR*.
13. C. Qiu and N. Vaswani. 2011. ReProCS: a missing link between recursive robust PCA and recursive sparse recovery in large but correlated noise. In *CoRR*.
14. K. Rosenblum, Z. Manor, and Y. Eldar. 2010. Dictionary optimization for block-sparse representations. In *AAAI*.
15. C. Stauffer and Y. Grimson. 1999. Adaptive background mixture models for real-time tracking. In *CVPR*.
16. M. Stojnic, F. Parvaresh, and B. Hassibi. 2009. On the reconstruction of block-sparse signals with an optimal number of measurements. *IEEE Trans. Sig. Proc*, 57(8):3075-3085.
17. D. Sun, S. Roth, and M. J. Black. 2010. Secrets of optical flow estimation and their principles. In *CVPR*.
18. G. Tang and A. Nehorai. 2011. Robust principal component analysis based on low-rank and block-sparse matrix decomposition. In *CISS*.
19. L. Wixson. 2000. Detecting salient motion by accumulating directionally-consistent flow. *PAMI*, 22(8):774-780.
20. X. Zhou, C. Yang, and W. Yu. 2012. Moving object detection by detecting contiguous outliers in the low-Rank representation. *Accepted by PAMI*.
21. Z. Zivkovic and F. van der Heijden. 2006. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters*, 27(7):773-780.