# An online background subtraction algorithm deployed on a NAO humanoid robot based monitoring system

**Article** *in* Robotics and Autonomous Systems · September 2016

DOI: 10.1016/j.robot.2016.08.013

**5 authors**, including:

Yang Hu

University of Kent

**10** PUBLICATIONS **150** CITATIONS

SEE PROFILE

Konstantinos Sirlantzis

University of Kent

**79** PUBLICATIONS **569** CITATIONS

SEE PROFILE

Gareth Howells

University of Kent

**204** PUBLICATIONS **1,602** CITATIONS

SEE PROFILE

Nicolas Ragot

ESIGELEC

**28** PUBLICATIONS **80** CITATIONS

SEE PROFILE

Robotics and Autonomous Systems 85 (2016) 37–47

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

# An online background subtraction algorithm deployed on a NAO humanoid robot based monitoring system

Yang Hu[a,*], Konstantinos Sirlantzis[a], Gareth Howells[a], Nicolas Ragot[b], Paul Rodríguez[c]

[a] University of Kent, UK
[b] ESIGELEC, France
[c] Pontificia Universidad Católica del Perú, Peru

## HIGHLIGHTS

- A online background subtraction algorithm detecting contiguous foreground.
- A weighted linear regression formulating background model and foreground detection.
- A Nao humanoid robot based monitoring system.
- Implement the background subtraction algorithm on the monitoring system.
- Experiments on both benchmark data and Nao robot data.

## ARTICLE INFO

*Article history:*
Received 22 July 2015
Received in revised form 1 July 2016
Accepted 24 August 2016
Available online 1 September 2016

*Keywords:*
Background subtraction
Contiguity
NAO humanoid robot
Monitoring system

## ABSTRACT

In this paper, we design a fast background subtraction algorithm and deploy this algorithm on a monitoring system based on NAO humanoid robot. The proposed algorithm detects a contiguous foreground via a contiguously weighted linear regression (CWLR) model. It consists of a background model and a foreground model. The background model is a regression based low rank model. It seeks a low rank background subspace and represents the background as the linear combination of the basis spanning the subspace. The foreground model promotes the contiguity in the foreground detection. It encourages the foreground to be detected as whole regions rather than separated pixels. We formulate the background and foreground model into a contiguously weighted linear regression problem. This problem can be solved efficiently via an alternating optimization approach which includes continuous and discrete variables. Given an image sequence, we use the first few frames to incrementally initialize the background subspace, and we determine the background and foreground in the following frames in an online scheme using the proposed CWLR model, with the background subspace continuously updated using the detected background information. The proposed algorithm is implemented by Python on a NAO humanoid robot based monitoring system. This system consists of a control station and a Nao robot. The Nao robot acts as a mobile probe. It captures image sequence and sends it to the control station. The control station serves as a control terminal. It sends commands to control the behavior of Nao robot, and it processes the image data sent by Nao. This system can be used for living environment monitoring and form the basis for many vision-based applications like fall detection and scene understanding. The experimental comparisons with most recent algorithms on both benchmark dataset and NAO captures demonstrate the high effectiveness of the proposed algorithm.

## 1. Introduction

The position of moving objects is an important knowledge for many algorithms and applications based on the humanoid robot. For example, in path planing, it helps the robot to avoid potential obstacles; in navigation, it allows the robot to adjust its velocity; in scene understanding, motion can be a useful prior to distinguish between some objects in the scene; in fall detection, it provides the potential position where the fall can happen. Therefore, a fast and accurate algorithm for moving object detection is desirable in robotic based systems.

A widely used solution for moving object detection on robots is to equip the robot with a video camera, view the moving object

[*] Corresponding author.
*E-mail address:* yh94@kent.ac.uk (Y. Hu).

http://dx.doi.org/10.1016/j.robot.2016.08.013
0921-8890/© 2016 Published by Elsevier B.V.

as the foreground in image sequence and perform background subtraction algorithms on the captured frames. Generally, the framework of background subtraction algorithms includes two components: a background model and a foreground model. The background model estimates the potential background in image sequences, and the foreground model detects foreground regions by comparing between captured frames and the estimated background. Currently, although a large number of algorithms have been proposed for background subtraction [1–16], some problems remain open for a background subtraction algorithm designed for robots. One main challenge is the varying working environment. The working environment of robots changes more frequently compared to static cameras in fixed positions, since a robot can be deployed to any possible positions for monitoring, and the environment like lighting condition, size of moving object and background dynamic can vary in different positions. Therefore, it is a challenging problem to design a background subtraction algorithm for a robot to distinguish between the foreground motion and background changing in varying environment. Another problem is the memory and computational cost. Most of recent algorithms like [13,14] use a batch scheme to handle the background changing. They model the background using the information from the whole image sequence. It improves the detection accuracy at the cost of more memory and computation requirement. However, in a robot based monitoring system, the background subtraction algorithm needs to run in an online scheme with the speed satisfying the requirement of specific applications. It is difficult to reduce the computational and memory cost of the algorithm while preserving a high accuracy.

Recently, low rank model has shown its potential to address the above challenges. The representative algorithms are Principal Component Pursuit (PCP) based algorithms [13,14] and PCP-like online algorithms [17–22]. PCP [13] assumes the potential background images of an image sequence lie in a low rank subspace, and the moving objects (foreground) is spatially sparse. It decomposes the image sequence into a low rank component as the background and a sparse component as the foreground, achieving an impressive perform. Also, recent research has shown that a contiguous foreground prior can be incorporated with the low rank background model, achieving a promising performance for foreground detection [14]. Furthermore, to speed up these PCP-based algorithms, a large number of PCP-like [17–22] online algorithms have been proposed. Some of these PCP-like online algorithms have achieved a real time speed [19,21,22].

The above PCP-based and PCP-like algorithms have shown their power in background subtraction. This is due to several advantages of low rank model. First, the assumption for background in low rank model is simple yet effective. The only assumption of low rank model for background is that the background with local and global variations can be represented by a low rank matrix [13]. The effectiveness of this simple assumption has been demonstrated by the success of batch methods using low rank model to estimate background [13,14,23].

Second, low rank model is a frame level model which can better capture background variation. In comparison to pixel level models which consider the pixels in each frame independently [1–5,7–9], low rank model views the pixels in a frame as a whole, hence it is able to capture the inner correlations between frames, better modeling the background variation.

Third, low rank model can be integrated with other models, combining the advantages of different models. Typical examples are some PCP-like online algorithms [17–20]. These algorithms incorporate representative background model [6,10–12] into low rank model [13,14], achieving excellent performance.

The aforementioned advantages make low rank model an excellent foundation to develop algorithms for background subtraction in challenging environments. However, some problems still exist and restrict the application of low rank based background subtraction algorithms in real robotic systems. First, it has been shown that the contiguous foreground prior contributes significantly to the accuracy of foreground detection [14,20], but using this prior also induces heavy computational burden. It influences the speed of algorithm, even in an online scheme [20]. Second, currently, little research examines the low rank model based background subtraction algorithm in real robotic systems. Little knowledge and study exist for the performance of these algorithms in real applications.

In this paper, we propose a fast online background subtraction algorithm with contiguous foreground prior for background subtraction, and we implement the proposed algorithm in a Nao humanoid robot [24,25] based monitoring system to examine its performance in real applications.[1] The algorithm includes a background model and a foreground model. For background modeling, we use a regression based low rank model. We assume that the potential background images lie in a low rank subspace, so we expect that the background of one frame can be represented as the linear combination of the basis spanning the subspace. Therefore, we represent the background of the current frame as the linear combination of the estimated background of previous frames. In foreground model, we use a contiguity constraint encouraging the foreground to be detected as contiguous regions rather than separated pixels. The proposed algorithm can be solved cheaply in terms of computational load.

The proposed Nao based monitoring system consists of two parts: Nao robot and a control station. Nao robot works as a mobile probe. It can move to a specific position, capturing image sequence using the equipped camera and sending the captures back to the control station for further processing. The control station serves as a control terminal. It sends commands to Nao robot to control its behavior, and it processes the image data sent by Nao robot. We implement the proposed background subtraction algorithm on the control station to detect the moving objects in the images sent by Nao. The proposed monitoring system can be used for living environment monitoring and form the basis for many vision-based applications like fall detection.

The rest of this paper is organized as follows. In Section 2, we introduce the related algorithms. We revisit the state-of-the-art PCP-based and PCP-like algorithms for background subtraction. In Section 3, we describe our background subtraction algorithm. In Section 4, we present the details of the proposed Nao humanoid robot based monitoring system and the implementation of the proposed background subtraction algorithm on this monitoring system. In Section 5, we report the experimental results. Finally, in Section 6, we conclude the paper.

## 2. The related algorithms

The related algorithms to the proposed algorithm are PCP-based and PCP-like algorithms [13,14,17–23]. PCP [13] is a batch method. It assumes the potential background of image sequences lies in a low rank subspace, and the foreground region is spatially sparse. Therefore, it decomposes image sequences into a low rank component as background and a sparse component as the foreground. However, due to its batch scheme, PCP has high computational cost and memory requirement, and it is difficult to apply PCP in robot based monitoring systems.

Since low rank background assumption has shown its power in background modeling, lots of PCP-based and PCP-like algorithms are proposed to detect moving objects in an online scheme to improve the efficiency [17–22]. Some algorithms perform PCP

[1] This is an extension of our algorithm in [26].

incrementally to reduce the computational cost of computing the low rank component. Representative algorithms are [21,22]. These algorithms rewrite batch PCP algorithm into an incremental form, and this incremental form can be well solved using some rank-1 SVD operations including increment, downdate and replacing. Other algorithms use a representative low rank background model to reduce the computation cost. That is, instead of directly computing the low rank component of image sequence, these algorithms seeks some basis spanning a low rank background subspace, using the linear combinations of these basis to represent the background, and updating these basis frame by frame to capture the background dynamic. ReProCs [17] is a recursive robust PCP algorithm with the assumption of a known model for the trajectories of moving objects. GRASTA [19] constraints the background subspace to be a Grassmannian manifold and updates it on the Grassmannian. A $\ell_1$ norm based cost function is used to achieve a robust representation. pROST [18] is similar to [19]. It uses $\ell_p$ norm ($p < 1$) instead of $\ell_1$ norm in the cost function to better model dynamic backgrounds.

The above works show that the background can be well-modeled using the low rank assumption in PCP, but the assumption of sparse foreground in PCP is sometimes not enough to model the moving objects. Usually, the moving objects are not only sparse in pixel-level but also are contiguous regions. Therefore, some researches [14,20] incorporate a contiguity foreground prior into the framework of PCP or PCP-like algorithms. DECOLOR [14] is a PCP-based batch algorithm. It imposes a Markov random field (MRF) based model on the sparse component of PCP, promoting the contiguity in the foreground detection result. GOSUS [20] is an online PCP-like algorithm using a representative background model. Similar to [19,18], it constraints the background subspace on a Grassmannian manifold, and it uses a group sparsity model to encourage the contiguity in the foreground detection. It is shown that the contiguity prior is able to significantly improve the accuracy of foreground detection, but heavy computational cost is also induced by this prior, leading to a reduced speed.

Also, although the aforementioned PCP-based and PCP-liked algorithms demonstrate impressive performance and illustrate the power of low rank models in background subtraction, little research studies the performance of these algorithms in a real system in real applications.

## 3. The proposed online background subtraction algorithm

### 3.1. Problem formulation

The problem of online background subtraction can be posed as follows. In a video sequence, given a current input frame $\boldsymbol{y} = [y_1, y_2, \ldots, y_n]^{\mathrm{T}} \in \mathbb{R}^n$ and the background model estimated using the previous frames, we aim to compute a foreground mask $\boldsymbol{s} = [s_1, s_2, \ldots, s_n]^{\mathrm{T}} \in \{0, 1\}^n$ for $\boldsymbol{y}$, where $s_i = 1$ if $y_i$ is detected as foreground, otherwise $s_i = 0$.

In background model, we assume the potential background images of a sequence lie in a low rank subspace. Therefore, we represent the background of current frame using the basis spanning the subspace. Specifically, denote the estimated background of the nearest $k$ frames prior to the current frame by $\boldsymbol{B} = [\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k] \in \mathbb{R}^{n\times k}$ ($\boldsymbol{b}_i \in R^n$ is a column vector of the estimated background of one of the frames), we represent the background of the current frame by $\boldsymbol{Bx}$, where $\boldsymbol{x} \in \mathbb{R}^k$ is a column vector of coefficients.

In foreground model, we use two priors for $\boldsymbol{s}$. The first one is the sparsity prior, as generally used in many low rank based background subtraction algorithms [13,17–22]. The sparsity prior assumes that $\boldsymbol{s}$ is sparse (most of the elements in $\boldsymbol{s}$ are 0). It means the foreground region is small comparing to the background. The other prior is the contiguity prior. It restricts that in $\boldsymbol{s}$, the elements with the same value should be distributed as groups. It promotes the contiguity of the detected foreground regions.

We formulate the above background and foreground model into three terms: a background fidelity term, a foreground sparsity term and a foreground contiguity term.

**Background fidelity term.** The background fidelity term is defined as follows:

$$f_1(\boldsymbol{x}, \boldsymbol{s}) = \frac{1}{2}\|(\mathbf{1} - \boldsymbol{s}) \otimes (\boldsymbol{y} - \boldsymbol{Bx})\|_2^2 \tag{1}$$

where $\mathbf{1} \in R^n$ is a column vector of ones and $\otimes$ is element-wise multiplication operator. Minimizing $f_1(\boldsymbol{x}, \boldsymbol{s})$ restricts that all background pixels are represented as the linear combination of the basis in low rank background subspace ($\boldsymbol{B}$).

**Foreground sparsity term.** The foreground sparsity term is defined as follows:

$$f_2(\boldsymbol{s}) = \|\boldsymbol{s}\|_1. \tag{2}$$

As shown in literature [13,14], $\ell_1$-norm is able to induce sparsity. Hence, minimizing $f_2(\boldsymbol{s})$ leads to a sparse $\boldsymbol{s}$.

**Foreground contiguity term.** The foreground contiguity term is defined as follows:

$$f_3(\boldsymbol{s}) = \sum_i \sum_{j\in N^i} \left|s_i - s_j\right| = \|\boldsymbol{Gs}\|_1 \tag{3}$$

where $N^i$ denotes the neighboring pixels of pixel $i$, and $\boldsymbol{G}$ is a matrix indicating the neighborship of all pixels. Minimizing $f_3(\boldsymbol{s})$ encourages neighboring pixels in $\boldsymbol{s}$ to have the same value. Thus, the detected foreground will be distributed as groups of pixels.

Combining the three terms, our formulation has the following form:

$$\arg\min_{\boldsymbol{s},\boldsymbol{x}} \frac{1}{2}\|(\mathbf{1} - \boldsymbol{s}) \otimes (\boldsymbol{y} - \boldsymbol{Bx})\|_2^2 + \alpha\|\boldsymbol{s}\|_1 + \beta\|\boldsymbol{Gs}\|_1 \tag{4}$$

where $\alpha$ and $\beta$ are penalty parameters. We call it a contiguously weighted linear regression (CWLR), because the first term is a weighted linear regression, and the third term constrains the weights to be spatially contiguous.

It can be seen that the model in Eq. (4) essentially consists of several state-of-the-art techniques used in existing low rank models for background subtraction. The first term of Eq. (4) is a regression based low rank background model (please also see Section 3.3). It is inspired by recent algorithms in [17–20] which combine low rank background model [13,14] with representative background model [6,10–12] for background modeling. The second term of Eq. (4) uses $\ell_1$-norm to induce the sparsity in foreground detection. The effectiveness of this sparse foreground prior has been demonstrated by batch low rank models in background subtraction [13,14,23]. The third term promotes contiguity in foreground detections. This term is motivated by [14,20] which show that the performance can be improved by promoting the detected foreground to be contiguous. By combining these state-of-the-art techniques, we expect that the proposed model is able to achieve an improved performance compared to existing low rank models.

### 3.2. Algorithm

Eq. (4) has discrete variables in $\boldsymbol{s}$ and it is not convex in both $\boldsymbol{s}$ and $\boldsymbol{x}$. As a result, it is difficult to solve $\boldsymbol{s}$ and $\boldsymbol{x}$ jointly. Instead, we seek a solution by optimizing over $\boldsymbol{s}$ and $\boldsymbol{x}$ alternatively. It leads to a $\boldsymbol{x}$-subproblem and a $\boldsymbol{s}$-subproblem.

***x*-subproblem.** Let $\bar{\boldsymbol{s}} = \mathbf{1} - \boldsymbol{s} \in \{0, 1\}^n$. With $\boldsymbol{s}$ fixed, minimizing Eq. (4) leads to the following problem:

$$\arg\min_{\boldsymbol{x}} \|\bar{\boldsymbol{s}} \otimes (\boldsymbol{y} - \boldsymbol{Bx})\|_2^2 \tag{5}$$

$\bar{\mathbf{s}}$ is binary, so we remove from $\mathbf{y}$ and $\mathbf{B}$ the rows corresponding to the zero elements of $\bar{\mathbf{s}}$, and solve a linear system to seek $\mathbf{x}$.

**$\mathbf{s}$-subproblem**. Let $\mathbf{e} = [e_1, e_2, \dots, e_n]^{\mathrm{T}} = \mathbf{y} - \mathbf{Bx} \in \mathbb{R}^n$. With $\mathbf{x}$ fixed, we rewrite the objective function in Eq. (4):

$$\begin{aligned}&\frac{1}{2}\|(\mathbf{1}-\mathbf{s})\otimes(\mathbf{y}-\mathbf{Bx})\|_2^2+\alpha\|\mathbf{s}\|_1+\beta\|\mathbf{Gs}\|_1\\&\quad=\frac{1}{2}\sum_i(\alpha-e_i^2)s_i+\beta\sum_i\sum_{j\in N^i}\left|s_i-s_j\right|+\frac{1}{2}\sum_i e_i^2\end{aligned}\tag{6}$$

where the third term can be ignored since it is a constant with $\mathbf{x}$ fixed. Eq. (6) is a first order Markov random field with binary labels [27]. It can be solved using graph cuts [28].

**Convergence analysis**. With the parameters $\alpha$ and $\beta$ fixed, the algorithm will converge to a local minimum. This is demonstrated as follows. Let $f(\mathbf{x}, \mathbf{s})$ be the objective function in Eq. (4). Let $\mathbf{x}^i$ be the $\mathbf{x}$ obtained in the $i$th iteration. Let $\mathbf{s}^i$ be the $\mathbf{s}$ obtained in the $i$th iteration. We have:

**(1)** alternating between $\mathbf{x}$-subproblem and $\mathbf{s}$-subproblem leads to a sequence of monotonically decreasing objective function values. As a proof, $f\left(\mathbf{x}^i, \mathbf{s}^i\right) \le f\left(\mathbf{x}^i, \mathbf{s}^{i-1}\right) \le f\left(\mathbf{x}^{i-1}, \mathbf{s}^{i-1}\right)$ can be obtained as follows:

$$f\left(\mathbf{x}^i, \mathbf{s}^{i-1}\right) = \arg\min_{\mathbf{x}} f\left(\mathbf{x}, \mathbf{s}^{i-1}\right) \le f\left(\mathbf{x}^{i-1}, \mathbf{s}^{i-1}\right) \tag{7}$$

$$f\left(\mathbf{x}^i, \mathbf{s}^i\right) = \arg\min_{\mathbf{s}} f\left(\mathbf{x}^i, \mathbf{s}\right) \le f\left(\mathbf{x}^i, \mathbf{s}^{i-1}\right). \tag{8}$$

**(2)** $f(\mathbf{x}, \mathbf{s})$ is lower bounded ($f(\mathbf{x}, \mathbf{s}) \ge 0$).

Therefore, according to above (1) and (2), the convergence of proposed algorithm is guaranteed. Furthermore, since $f(\mathbf{x}, \mathbf{s})$ is not convex in both $\mathbf{x}$ and $\mathbf{s}$, we can obtain that the proposed algorithm will converge to a local minimum. In our experiment, the algorithm usually converges in 5–10 iterations.

**Parameter setting**. There are 3 parameters in the proposed algorithm: $k$ which is the basis number in $\mathbf{B}$, $\alpha$ and $\beta$ in Eq. (4). We set $k$ to 10 empirically (see Section 5.2).

As for $\alpha$ and $\beta$, borrowing an idea from [14], we update them as follows. In the first iteration, $\alpha$ is set to a large value, $\alpha = 0.5\sigma^2$ where $\sigma$ is the standard deviation of the current frame. The reason is that $\mathbf{Bx}$ is an inaccurate background estimation at the beginning of the algorithm (foreground is not fully masked). It will lead to an inaccurate estimation of $\mathbf{s}$. Therefore, we apply a large penalty resulting a conservative estimation of $\mathbf{s}$. In each iteration, $\alpha$ is reduced by a factor of 0.5, since, along with more foreground region is found, $\mathbf{Bx}$ becomes more accurate and we relax the penalty to encourage more foreground detection. $\beta$ is set to $5\alpha$ in each iteration.

### 3.3. Background updating

After the foreground of the current frame $\mathbf{y}$ is detected, a key problem to achieve an online scheme is how to update the low rank background subspace $\mathbf{B}$ for the detections in the following frames. In this paper, we update $\mathbf{B}$ based on an incremental PCP algorithm. Let $\mathbf{Y}' \in \mathbb{R}^{n\times k}$ be a matrix including the nearest $k$ frames to $\mathbf{y}$ with each of its column being a frame. Assuming we have the low rank approximation of $\mathbf{Y}'$, we seek the low rank approximation of $[\mathbf{Y}', \mathbf{y}]$ by the incremental PCP algorithm in [21,22]. After the low rank approximation of $\mathbf{y}$ is computed, we estimate the background of $\mathbf{y}$ as a new basis for $\mathbf{B}$ by preserving the detected background pixels in $\mathbf{y}$ and replacing the detected foreground pixels by their low rank approximations. Then, we use this new basis to replace the basis corresponding to the oldest frame in $\mathbf{B}$.

Now, the problem becomes how to find an initial low rank background subspace so that we can update it frame by frame using the above described method. We employ the incremental initialization algorithm in [21,22] to seek this subspace as well as the low rank approximation of some initial frames (*i.e.* initial $\mathbf{Y}'$). This algorithm works in an incremental scheme so it is much faster than the batch initialization.

We note that a more efficient way to update the background is to use $\mathbf{Bx}$. We do not choose this way because it exploits little new information additional to the current background subspace. In our experiment, we find that updating $\mathbf{B}$ via the former method leads to a better performance.

## 4. Implementation of the proposed algorithm on a Nao humanoid robot based monitoring system

In this section, we present a Nao humanoid robot based monitoring system where the proposed algorithm is implemented. We firstly give a brief introduction of Nao humanoid robot. Then, we describe the hardware design of the monitoring system and the software implementation.

### 4.1. Introduction of Nao humanoid robot

Nao [24,25] is the product of Aldebaran Robotics—SAS (Limited Company). It has a medium size with a height of 57.3 cm, a width of 27.3 cm and a weight less than 4.3 kg. A wide class of sensors locates around the body of Nao. These sensors allow Nao to capture the around environment, making it a powerful medium to develop applications. An illustration of Nao's shape and sensor equipment is shown in Fig. 1. In this subsection, we introduce some key modules of Nao related to our implementation. We refer the readers with interest to [24,25].

**Motherboard**. The motherboard of Nao locates in the head. This motherboard is equipped with an Intel ATOM 1.6 GHz CPU with 1 GB RAM, and additional 2 GB flash memory and 8 GB micro SDHC are available. The CPU runs a Linux kernel. It manages the combination, vision and other sensor modules of Nao.

**Motion**. The body of Nao has 25 degrees of freedom (DOF), 11 DOF for the lower body part and 14 DOF for the upper body part. All body parts are driven by electric motors and actuators. One ARM microcontroller locates at the chest distributes information to all the actuator module microcontrollers. The ARM microcontroller communicates with the motherboard via a USB-2 bus.

**Vision**. NAO has two cameras. One locates in the forehead and the other one locates at the month level. The forehead camera scans the horizontal direction, while the month camera focuses on the floor. The two cameras use MT9M114 image sensor. They support a wide range of resolution up to 960 × 1280 with a focus range from 30 cm to infinity. In video capture, the highest frame rate is 30 fps.

**Connectivity**. Nao supports Wi-Fi IEEE 802.11 b/g/n standard and Ethernet connection. Therefore, it is possible to send command to Nao and retrieve the data captured by the sensors on Nao through a control station within the same network.

**Battery**. The battery locates at the back of Nao. It is a Lithium ion battery with a charge duration less than 3 h. The fully charged battery can support around 60 min of use.

### 4.2. System overview

The Nao humanoid robot based monitoring system consists two components: a control station and the Nao robot. The Nao robot works as an adaptive mobile monitor. It is able to walk to the target monitoring point, adapt its camera towards target area, capture image sequence using it camera, and send the captured images to the control station for further processing. The control station accounts for control and image processing task. It sends

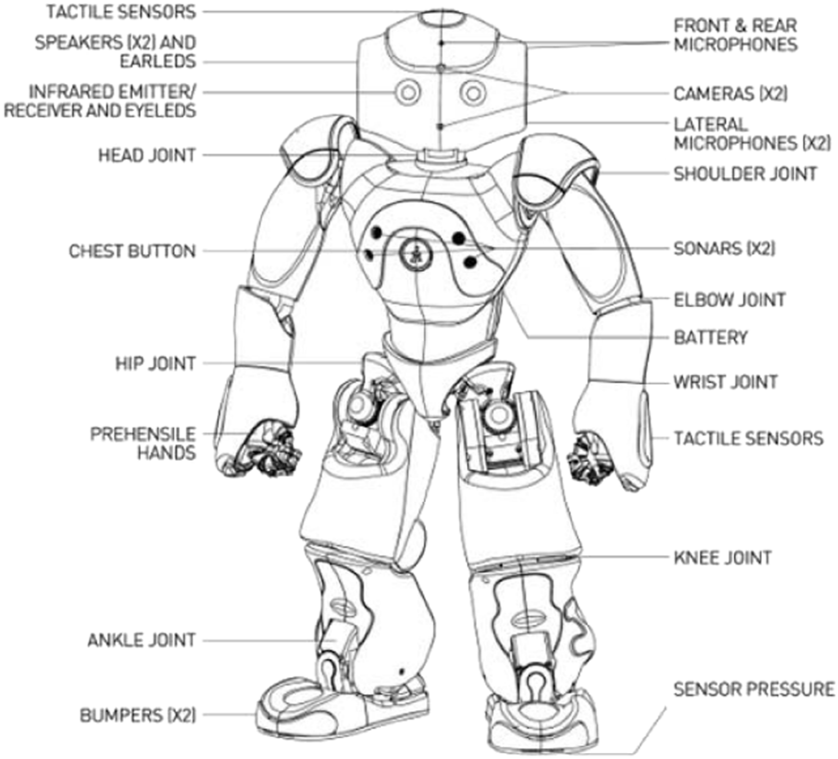


**Fig. 1.** An illustration of the shape and structure of Nao humanoid robot [25].

commands to Nao, controlling Nao to walk to the monitoring point, adjusting its camera direction and beginning to capture image sequences. Also, the control station runs the background subtraction algorithm to analyze the image sequence sent back by Nao. The position of moving objects can be used as the foundation for many applications like fall detection and scene understanding.

### 4.3. Hardware configuration

In the current implementation, we use a Samsung laptop as the control station. This laptop has Intel i5-3210M quad-core 2.50 GHz CPU and 6 GB RAM. It runs a Linux 64 bit system. As for Nao, we use its top camera for image capture. The Nao robot and the control station can be connected by either Ethernet or Wi-Fi to transmit the commands and images.

### 4.4. Algorithm implementation

The algorithm in this system has two parts. The first part is the algorithm for the functionality of Nao, including walking, adjusting camera direction and capturing image. The second part is the algorithm of background subtraction as described in Section 3. For the first part, we employ the NAOqi framework [29] developed by Aldebaran robotics. NAOqi includes a set of APIs which controls the behavior of Nao, including walking, turning head and capturing image sequence.

As for the background subtraction part, we write and run the proposed background subtraction algorithm in Python. The code consists of two parts: initialization and background subtraction. The function call map of our implementation is shown in Fig. 2, and we explain each function as follows.

**initIncPCP:** this function implements the fast initialization algorithm described in [21,22] and used in Section 3.3.

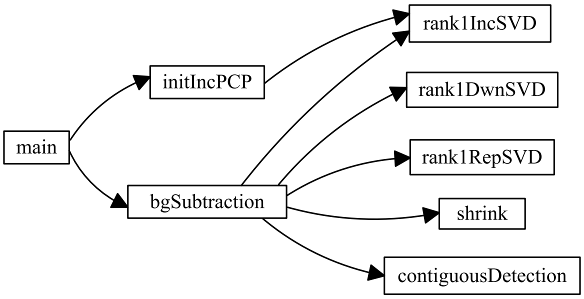


**Fig. 2.** Function call graph of our implementation.

**bgSubtraction:** this function implements the background subtraction algorithm described in Section 3.

**contiguousDetection:** this function solves the contiguously weight linear regression problem using the algorithm in Section 3.2 for contiguous foreground detection.

**rank1IncSVD:** this function implements the rank 1 SVD increment algorithm.

**rank1DwnSVD:** this function implements the rank 1 SVD downdate algorithm.

**rank1RepSVD:** this function implements the rank 1 replacing algorithm.

**shrink:** this function implements the soft-thresholding shrinkage operation, shrink $(\boldsymbol{x}, \varepsilon) = \text{sign}(\boldsymbol{x}) \max(|\boldsymbol{x}| - \varepsilon)$.

We would like to note that in the background subtraction part (bgSubtraction), the rank 1 SVD increment (rank1IncSVD), downdate (rank1DwnSVD) and replacing algorithm (rank1RepSVD), and soft-thresholding shrinkage operation (shrink) are used in the algorithm in [22,21], and they are employed for the background updating in the proposed algorithm (see Section 3.3).

## 5. Experiment

We perform experiments on both benchmark dataset and Nao captured data. The experiment on benchmark dataset studies the effect of the proposed CWLR model as a general method for background subtraction, while the experiment on Nao data investigates on the performance of CWLR on the Nao humanoid robot based monitoring system.

### 5.1. Evaluation

We evaluate the performance of algorithms by accuracy and speed. The accuracy is evaluated by $F$-measure defined as follows:

$$F = \frac{2\times\text{precision}\times\text{recall}}{\text{precision} + \text{recall}} \tag{9}$$

where precision $=\frac{\text{TP}}{\text{TP+FN}}$ and recall $=\frac{\text{TP}}{\text{TP+FP}}$; TP, FN and FP are the number of true positive, false negative and false positive pixels, respectively.

Generally, the $F$-measure is used to evaluate the pixel-level accuracy of foreground detection. It is reasonable for some applications requiring a pixel-level accuracy like motion segmentation. However, in some applications like fall detection [30], the moving objects are sometimes extracted at a region level using a bounding box for the following processing. Although the detection result of different algorithms varies at the pixel level, such differences at the pixel-level may have little influence on the bounding box. Therefore, we also use a block-wise $F$-measure to evaluate the accuracy of the bounding box found by each algorithm. Specifically, we use the bounding box of the connected components in the ground truth image as ground truth bounding box. We use the bounding box of the connected components in the foreground mask computed by each algorithm as the detected bounding box. The block-wise $F$-measure calculates the $F$-measure between the ground truth bounding box and the detected bounding box using similar way to Eq. (9).

We use frames per second (FPS) to evaluate the speed.

### 5.2. Experiment on benchmark data

*Dataset*. We use the I2R dataset[2] [31] as the benchmark dataset. It includes 9 challenging videos: Bootstrap (120 × 160 × 3057 frames, crowd scene), Campus (128 × 160 × 1439 frames, waving trees), Curtain (128 × 160 × 2964 frames, waving curtain), Escalator (130 × 160 × 3417 frames, moving escalator), Fountain (128 × 160 × 523 frames, fountain water), hall (144 × 176 × 3548 frames, crowd scene), Lobby (128 × 160 × 1546 frames, switching light), ShoppingMall (256 × 320 × 1286 frames, crowd scene), WaterSurface (128 × 160 × 633 frames, water surface). Ground truth of some frames is provided in the dataset. On each sequence, we use the first 200 frames for training (initialization) and perform background subtraction on the remaining frames.

*Comparison methods*. We report comparisons with the following algorithms: mixture of Gaussian (MoG) [2] as the baseline; GRASTA[3] [19], GOSUS[4] [20] and incPCP[5] [21,22] as recent online low rank based algorithms. Moreover, we compare with two batch low rank methods: PCP[6] [13] as a traditional low rank batch algorithm; DECOLOR[7] [14] as an improved PCP with the foreground contiguity prior.

Most of the comparison methods need a threshold to produce foreground mask. For these algorithms, we use the first image with ground truth in each video as the training image; we choose the threshold to maximize the $F$-measure on the training image, and fix this threshold for the other images. For other parameters in different algorithms (including $k$ in ours), we find an optimal option using the training images and fix it for all the videos. We use fixed parameters for all the videos, since it is closer to the scenario of real applications, especially for a robotic based vision system working in varying environment.

*Results and discussions*. We run all the algorithms on a desktop with Intel i5-3470 quad-core 3.20 GHz CPU, 16 GB RAM, WIN7 64 bit system. Since all the algorithms for comparison to ours are implemented in Matlab, we also implement CWLR in Matlab for a fair comparison in terms of speed. Moreover, to study the speed of CWLR in the real system in Section 4, we report the speed of the CWLR implemented in Python and running on the control station as described in Section 4.3 and Section 4.4 using the same image sequences as input (the accuracy will be similar since it is the different implementation of the same algorithm). We refer to CWLR implemented in Python on the control station as CWLR&.

We report the mean $F$-measure, mean block-wise $F$-measure and FPS of all the methods on each video in Tables 1–3, respectively. We show some example results in Fig. 3. Among the online methods, the one with the highest $F$-measure is marked red in Tables 1 and 2, and the second highest is marked green. It can be seen that our algorithm achieves the highest overall $F$-measure and block-wise $F$-measure among all the online methods. We note that GOSUS reports a promising performance with tuned parameters for each video as in [20], but when uniform parameter setting (closer to real scenario of a robotic system) is used, its performance varies on each video and the overall performance drops.

Comparing the accuracy between the online and batch methods, we find that in terms of pixel-level accuracy ($F$-measure in Table 1), incPCP achieves a comparable $F$-measure to the original PCP, and GRASTA, GOSUS and our algorithm outperform the original PCP. However, in terms of region-level accuracy (block-wise $F$-measure in Table 2), PCP performs better than all the online methods excepting CWLR. It means that on the benchmark dataset, a batch scheme can better capture the region of moving object, despite of lower accuracy at the pixel level. Such a result shows that on the benchmark data, the existing PCP-based and PCP-like online algorithms without the foreground contiguity prior (GRASTA, GOSUS, incPCP) are preferable to the task requiring a high pixel-level accuracy like motion segmentation, but they are less effective for the region-level detection, compared to the batch algorithms. CWLR addresses this problem. In comparison to the online methods and batch PCP, CWLR achieves better accuracy in both pixel-level and region-level. This result suggests that the foreground contiguity prior is able to improve the performance of background subtraction in both pixel-level and region-level. On the other hand, DECOLOR outperforms all other methods. The reason is the combination of a contiguity foreground prior and the batch scheme.

As for speed, incPCP is the fastest algorithm. Our algorithm is an online algorithm but it is not real time, because additional computational cost is induced by the foreground contiguity prior modeling a pixel-wise neighboring information. However, compared to the algorithms using this prior, our algorithm is approximately 7.5 times faster than batch DECOLOR algorithm and 6 times faster than online GOSUS algorithm on the desktop. Also, we find that CWLR runs slower on the control station than the desktop. We think the reason is that the hardware configuration of our desktop is better than the control station (recall that the hardware of control station: Intel i5-3210M quad-core 2.50 GHz CPU and 6 GB RAM; the hardware of desktop: Intel i5-3470 quad-core 3.20 GHz CPU, 16 GB RAM).

[2] http://perception.i2r.a-star.edu.sg/bk_model/bk_index.html.

[3] http://sites.google.com/site/hejunzz/grasta.

[4] http://pages.cs.wisc.edu/jiaxu/projects/gosus/.

[5] https://sites.google.com/a/istec.net/prodrig/Home/en/pubs/incpcp.

[6] http://perception.csl.illinois.edu/matrix-rank/sample_code.html.

[7] http://fling.seas.upenn.edu/xiaowz/dynamic/wordpress/?p=144.

**Table 1**
F-measure of all methods compared on I2R dataset.

| Video | PCP*[13] | DECOLOR*[14] | MoG#[1] | GRASTA#[19] | GOSUS#[20] | incPCP#[21, 22] | CWLR# |
|---|---|---|---|---|---|---|---|
| Bootstrap | 0.5964 | 0.6248 | 0.4240 | 0.6017 | 0.6599 | 0.5289 | 0.5443 |
| Campus | 0.3405 | 0.7652 | 0.2115 | 0.2152 | 0.1669 | 0.2092 | 0.7911 |
| Curtain | 0.4927 | 0.8342 | 0.4186 | 0.7816 | 0.8700 | 0.6182 | 0.7615 |
| Escalator | 0.6039 | 0.7183 | 0.2744 | 0.4265 | 0.4058 | 0.3747 | 0.6137 |
| Fountain | 0.5226 | 0.8618 | 0.3875 | 0.6620 | 0.6778 | 0.6502 | 0.7958 |
| Hall | 0.4840 | 0.5597 | 0.3603 | 0.5355 | 0.4644 | 0.4744 | 0.4807 |
| Lobby | 0.5833 | 0.5654 | 0.3441 | 0.4059 | 0.1856 | 0.4460 | 0.6375 |
| ShoppingMall | 0.5739 | 0.6800 | 0.5407 | 0.6724 | 0.7152 | 0.6809 | 0.6279 |
| WaterSurface | 0.3392 | 0.8873 | 0.2280 | 0.7725 | 0.7873 | 0.5258 | 0.5612 |
| Mean | 0.5041 | 0.7219 | 0.3543 | 0.5635 | 0.5481 | 0.5009 | 0.6460 |

*batch (off-line) method #online method

**Table 2**
Block-wise F-measure of all methods compared on I2R dataset.

| Video | PCP*[13] | DECOLOR*[14] | MoG#[1] | GRASTA#[19] | GOSUS#[20] | incPCP#[21, 22] | CWLR# |
|---|---|---|---|---|---|---|---|
| Bootstrap | 0.8092 | 0.7754 | 0.7356 | 0.8189 | 0.6224 | 0.8329 | 0.6197 |
| Campus | 0.4340 | 0.8277 | 0.2631 | 0.2516 | 0.1235 | 0.2449 | 0.8376 |
| Curtain | 0.8087 | 0.8777 | 0.7733 | 0.9799 | 0.9600 | 0.8700 | 0.8479 |
| Escalator | 0.7091 | 0.6992 | 0.5207 | 0.6020 | 0.3517 | 0.4649 | 0.6440 |
| Fountain | 0.4954 | 0.8894 | 0.6177 | 0.5630 | 0.9014 | 0.6106 | 0.9111 |
| Hall | 0.8474 | 0.5245 | 0.5081 | 0.8304 | 0.6473 | 0.8292 | 0.5482 |
| Lobby | 0.8852 | 0.9393 | 0.5905 | 0.6550 | 0.5671 | 0.7965 | 0.6868 |
| ShoppingMall | 0.7580 | 0.6849 | 0.7805 | 0.7730 | 0.7005 | 0.7613 | 0.6718 |
| WaterSurface | 0.8668 | 0.8880 | 0.6107 | 0.7813 | 0.9789 | 0.7326 | 0.6464 |
| Mean | 0.6547 | 0.7679 | 0.5168 | 0.6240 | 0.5679 | 0.5961 | 0.7126 |

*batch (off-line) method #online method

**Table 3**
FPS of all methods compared on I2R dataset.

| Video | PCP[a] [13] | DECOLOR[a] [14] | MoG[b] [1] | GRASTA[b] [19] | GOSUS[b] [20] | incPCP[b] [21,22] | CWLR[b] | CWLR&[b] |
|---|---|---|---|---|---|---|---|---|
| Bootstrap | 1.36 | 0.17 | 2.98 | 21.4 | 0.81 | 27.8 | 5.30 | 4.04 |
| Campus | 3.37 | 2.55 | 2.77 | 21.8 | 0.97 | 26.9 | 7.41 | 4.54 |
| Curtain | 1.29 | 0.27 | 2.65 | 21.8 | 1.24 | 28.1 | 5.74 | 6.06 |
| Escalator | 1.12 | 0.26 | 2.83 | 27.0 | 0.83 | 36.2 | 6.00 | 3.22 |
| Fountain | 4.20 | 1.63 | 2.84 | 18.5 | 1.31 | 22.0 | 7.35 | 3.85 |
| Hall | 0.93 | 0.14 | 2.19 | 17.2 | 0.67 | 26.7 | 4.49 | 4.23 |
| Lobby | 1.69 | 0.79 | 2.64 | 19.1 | 1.13 | 26.2 | 6.03 | 5.59 |
| ShoppingMall | 0.73 | 0.14 | 0.71 | 11.8 | 0.18 | 14.4 | 1.65 | 1.21 |
| WaterSurface | 4.65 | 0.72 | 2.83 | 21.3 | 1.02 | 26.5 | 6.40 | 5.28 |
| Mean | 2.15 | 0.74 | 2.49 | 20.0 | 0.91 | 26.1 | 5.60 | 4.22 |

[a] Batch (off-line) method.

[b] Online method.

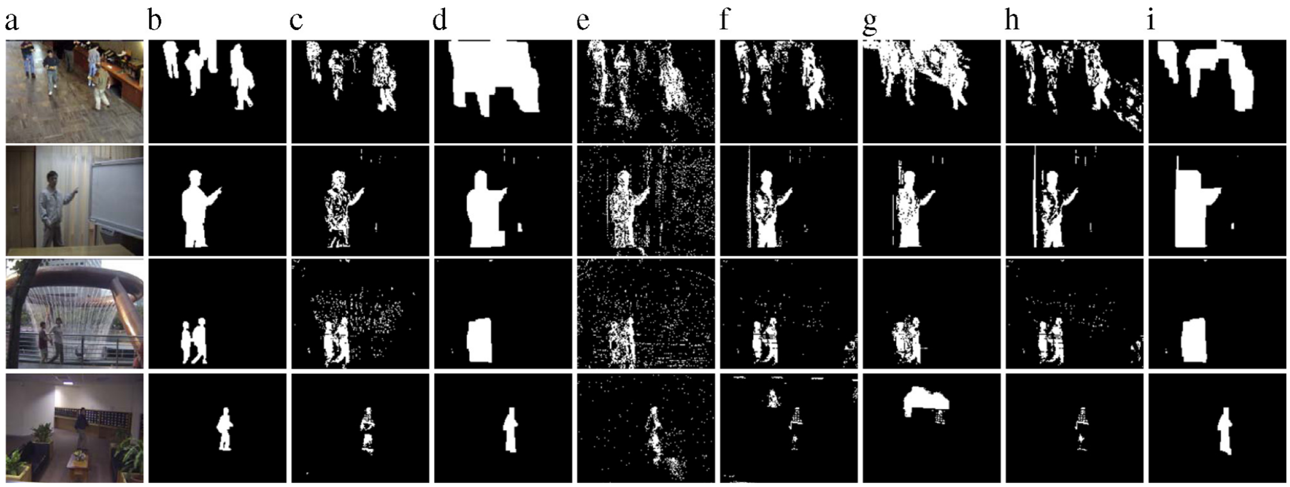


**Fig. 3.** Example results of comparison methods. (a) The original video frame (from top to bottom: Bootstrap, Curtain, Fountain, Lobby); (b) the ground truth; (c) PCP; (d) DECOLOR; (e) MoG; (f) GRASTA; (g) GOSUS; (h) incPCP; (i) CWLR.

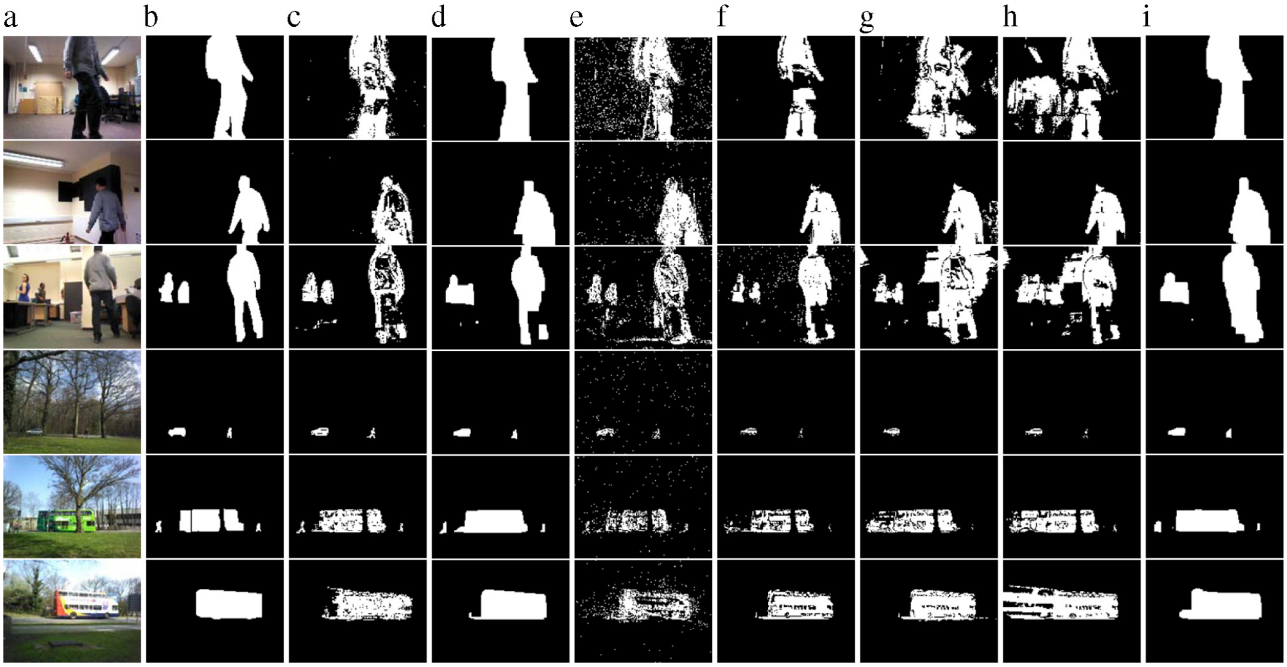


**Fig. 4.** Example results of comparison methods on Nao captured Data. (a) The original video frame (from top to bottom: video1, video2, video3, video4, video5, video6); (b) the ground truth; (c) PCP; (d) DECOLOR; (e) MoG; (f) GRASTA; (g) GOSUS; (h) incPCP; (i) CWLR.

### 5.3. Experiment on Nao data

*Dataset*. To investigate the performance of CWLR in the Nao humanoid robot based monitoring system, we conduct experiment on the image sequences captured by Nao robot.[8] We capture 6 sequences using the monitoring system described in Section 4. 3 of the sequences are captured indoor environment in a lab, and 3 of the sequences are captured outdoor in a campus. The sequences mainly suffer from indoor or outdoor illumination changing. The number of frames in each sequence is between 1000 and 3000. The resolution of all the sequence is set to $120 \times 160$ . For each sequence, we manually segment some ground truth images for performance evaluation. Some example images of each sequence are shown in Fig. 4(a).

*Results and discussions*. We make comparison with the same methods in Section 5.2 using the same evaluations. Similar to the experiment on the benchmark data (Section 5.2), for a fair comparison of speed, we report frames per second (FPS) as the speed of proposed CWLR in the Matlab implementation running on the desktop. Also, to evaluate the speed of CWLR in the proposed monitoring system, we report FPS of CWLR implemented in Python on the control station (referred as CWLR&).

We report the mean $F$-measure, mean block-wise $F$-measure and FPS of all the methods on each video in Tables 4–6, respectively. We show some example results in Fig. 4. Similar to the results on benchmark data, the method with the highest $F$-measure among the online methods is marked red in Tables 4 and 5, and the second highest is marked green. We find that on Nao data, the accuracy of all the algorithm is generally higher than that on benchmark data. We think a possible reason is that the background changing is less dramatic in our working environment of Nao.

Also, it can be seen that different from the results on the benchmark data, on the Nao data, batch PCP outperforms all the online low rank based methods excepting CWLR in terms of both pixel-level and region-level accuracy. CWLR has a better accuracy than PCP in pixel level, and the accuracy is also comparable between CWLR and PCP in region level. On the other hand, similar to the results on benchmark data, batch DECOLOR still achieves the highest accuracy in both pixel level and region level.

In terms of speed, the results on Nao captures are similar to that on the benchmark data. incPCP is the fastest algorithm. CWLR algorithm has the highest speed among the algorithms using foreground contiguity prior (DECOLOR and GOSUS). It can be seen from Table 6 that CWLR is approximately 12.5 times faster than batch DECOLOR algorithm and online GOSUS algorithm on Nao captures.

### 5.4. Influence of the size of background subspace

In the proposed CWLR algorithm, the background subspace $\boldsymbol{B}$ is constructed using the nearest $k$ frames prior to the current frame. In other words, $k$ is a parameter that determines the size of background subspace. Therefore, it is necessary to investigate the influence of $k$ on the performance of proposed algorithm.

To make this investigation, we report the $F$-measure and Block-wise $F$-measure of CWLR on I2R dataset and Nao captures, with $k$ varied from 3 to 70. The results are reported in Fig. 5.

It can be seen that, when $k < 10$, the performance improves with $k$ for most sequences on both datasets. This is because, when $k$ is small, the background subspace includes too little information to represent the background; in this case, increasing $k$ will include more background information into the background subspace, enabling the background subspace to better represent the background in sequences, hence leading to an improved performance.

On the other hand, when $k \geq 10$, the performance remains stable or fluctuates for most sequences on both datasets. The reason is that, when $k$ is sufficiently large, the background subspace contains sufficient information to represent the background in sequences; as a result, adding more background information by increasing $k$ leads to little improvement.

Finally, investigating the influence of $k$ on the mean performance, we find that, for I2R dataset, the mean performance remains stable for all $k$; for Nao dataset, the mean performance improves with $k$ when $k < 10$, and it is stable when $k \geq 10$.

[8] https://www.dropbox.com/s/s67w7ba8vb15546/NaoVideos.zip?dl=0.

**Table 4**
F-measure of all methods compared on Nao captured data.

| Video | PCP* [13] | DECOLOR* [14] | MoG# [1] | GRASTA# [19] | GOSUS# [20] | incPCP# [21, 22] | CWLR# |
|---|---|---|---|---|---|---|---|
| video1 | 0.7387 | 0.8396 | 0.5994 | 0.7772 | 0.6630 | 0.7604 | 0.8176 |
| video2 | 0.7762 | 0.9091 | 0.6593 | 0.7461 | 0.6941 | 0.7187 | 0.8638 |
| video3 | 0.6966 | 0.8980 | 0.6831 | 0.7844 | 0.6829 | 0.6873 | 0.7913 |
| video4 | 0.8181 | 0.8655 | 0.3662 | 0.6113 | 0.6024 | 0.6749 | 0.8615 |
| video5 | 0.7376 | 0.8040 | 0.3341 | 0.6248 | 0.5780 | 0.6387 | 0.7669 |
| video6 | 0.8586 | 0.8817 | 0.4219 | 0.8049 | 0.8453 | 0.7693 | 0.8800 |
| Mean | 0.7710 | 0.8663 | 0.5107 | 0.7278 | 0.6776 | 0.7082 | 0.8302 |

*batch (off-line) method #online method

**Table 5**
Block-wise F-measure of all methods compared on Nao captured data.

| Video | PCP* [13] | DECOLOR* [14] | MoG# [1] | GRASTA# [19] | GOSUS# [20] | incPCP# [21, 22] | CWLR# |
|---|---|---|---|---|---|---|---|
| video1 | 0.8720 | 0.8896 | 0.7083 | 0.8705 | 0.7926 | 0.8375 | 0.8536 |
| video2 | 0.8629 | 0.9369 | 0.7051 | 0.7836 | 0.7652 | 0.7304 | 0.8793 |
| video3 | 0.9189 | 0.9389 | 0.8286 | 0.8529 | 0.7857 | 0.7319 | 0.8261 |
| video4 | 0.8643 | 0.8690 | 0.4087 | 0.7544 | 0.7087 | 0.7947 | 0.8645 |
| video5 | 0.7565 | 0.8148 | 0.4597 | 0.7596 | 0.6801 | 0.7519 | 0.7844 |
| video6 | 0.8499 | 0.8897 | 0.5087 | 0.9016 | 0.8746 | 0.8068 | 0.8838 |
| Mean | 0.8541 | 0.8898 | 0.6152 | 0.8204 | 0.7678 | 0.7755 | 0.8486 |

*batch (off-line) method #online method

**Table 6**
FPS of all methods compared on Nao captured data.

| Video | PCP[a] [13] | DECOLOR[a] [14] | MoG[b] [1] | GRASTA[b] [19] | GOSUS[b] [20] | incPCP[b] [21,22] | CWLR[b] | CWLR&[b] |
|---|---|---|---|---|---|---|---|---|
| video1 | 4.35 | 0.51 | 3.02 | 34.9 | 0.53 | 42.5 | 7.20 | 4.56 |
| video2 | 3.64 | 0.77 | 2.91 | 33.9 | 0.53 | 55.0 | 7.19 | 3.93 |
| video3 | 2.63 | 0.43 | 3.01 | 36.6 | 0.51 | 52.3 | 6.72 | 3.14 |
| video4 | 1.27 | 0.61 | 2.84 | 30.0 | 0.61 | 38.7 | 7.44 | 7.01 |
| video5 | 1.93 | 0.62 | 2.52 | 33.0 | 0.53 | 37.6 | 6.77 | 5.29 |
| video6 | 1.03 | 0.39 | 2.90 | 37.4 | 0.68 | 48.5 | 7.56 | 4.20 |
| Mean | 2.48 | 0.56 | 2.87 | 34.3 | 0.57 | 45.8 | 7.17 | 4.69 |

[a] Batch (off-line) method.

[b] Online method.

Therefore, based on the above observations, we conclude that, (1) when $k$ is small (lower than 10 in our experiment), a larger $k$ leads to generally better performance; (2) when $k$ is sufficiently large (larger than 10 in our experiment), changing this parameter has little influence on the performance of proposed algorithm. Based on the above conclusions, given an arbitrary image sequence, we may vary $k$ around 10 to seek a good setting of this parameter.

### 5.5. Result summarization

Analyzing all the experimental results on both benchmark data and Nao data, we can see a clear trade-off between the performance and speed. Without foreground contiguity prior, some online methods including GRASTA and incPCP are able to achieve a real time speed, but the accuracy is unstable. It can be seen that although on the benchmark data these methods outperform batch PCP in accuracy, PCP is still more accurate on the Nao data.

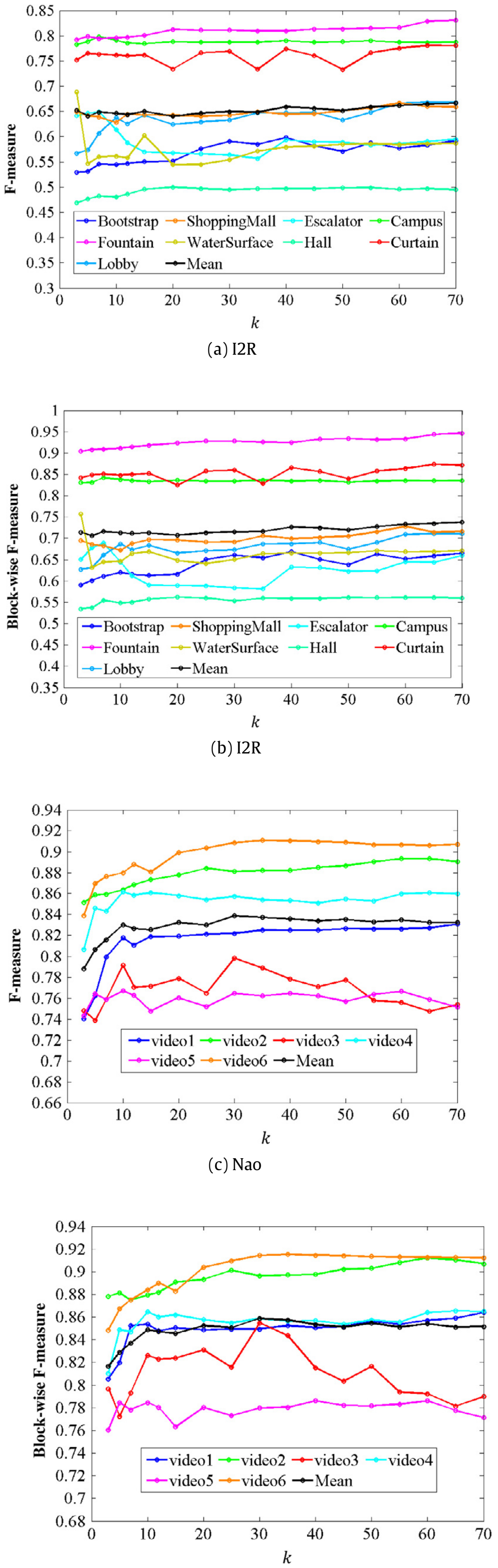
On the other hand, the accuracy can be significantly improved by incorporating the foreground contiguity prior into the low rank based model. It can be seen that on both Nao data and benchmark data, DECOLOR and CWLR have a generally better accuracy than the methods without the foreground contiguity prior. For GOSUS, although its performance is unstable in our experiment due to a fixed parameter setting, it is still able achieve a promising performance with fined-tuned parameters according to the results in [20]. However, such performance is at the cost of more computational burden. Among the methods using a foreground contiguity prior, the fastest one is CWLR, but the speed is still slower than the fastest algorithm without the contiguity prior.

Finally, focusing on the performance of CWLR, we can see that it achieves the highest overall accuracy among the low rank based online background subtraction algorithms on both benchmark data and Nao data. Also, on the proposed Nao robot based monitoring system, CWLR achieves a speed of 4.69 fps. It means that the delay caused by detection using the proposed algorithm is lower than 220 ms on the proposed monitoring system. This speed enables the proposed algorithm to be used for some applications based on the result of background subtraction like scene understanding, human–robot interaction and anomaly detection.

Therefore, we conclude that CWLR achieves a high accuracy and an acceptable speed in a Nao robot based monitoring system. This result demonstrates the effectiveness of CWLR. It also shows that low rank based model and the foreground contiguity prior are effective methods for background subtraction, and it has the potential to be applied in real robotic systems.

## 6. Conclusion

In this paper, we propose a background subtraction algorithm and implement it on a Nao humanoid robot based monitoring system. The proposed algorithm uses a contiguous foreground prior. It includes a regression based low rank background model and a foreground model promoting the foreground contiguity. We formulate the background and foreground model as a contiguously weighted linear regression problem which can be solved



(a) I2R

(b) I2R

(c) Nao

(d) Nao

**Fig. 5.** Influence of *k* on the proposed algorithm.

efficiently. Also, we study the performance of the proposed background subtraction algorithm in a real robotic system by designing a Nao humanoid robot based monitoring system which runs our algorithm for background subtraction. Experimental results show that CWLR achieves an improved accuracy compared to current low rank model based online algorithms, and the accuracy of CWLR is also comparable to the best batch method in our experiment. In terms of speed, CWLR is able to run at 5.60/7.17 fps (benchmark data/nao data) on a desktop and 4.22/4.69 fps on the designed monitoring system. This speed is much faster than the recent low rank model based background subtraction algorithms using similar contiguous foreground prior. Also, this speed means that the delay of detection is lower than 150 ms on our desktop and lower than 220 ms on the proposed monitoring system. This delay is small and it enables the proposed algorithm to be used for some applications based on the result of background subtraction like scene understanding, human–robot interaction and anomaly detection. Thus, we conclude that the proposed background subtraction algorithm is able to improve the accuracy of background subtraction, and the algorithm achieves a fast speed that enables it to be used for applications in a real robotic system. Future work may consider a parallel implementation to further improve the speed, achieving a real time background subtraction system with high accuracy.

## Acknowledgments

This research is financed by the European cross-border cooperation program INTERRG IV A France (Channel) - England, the European Regional Development Fund (ERDF), and Cognitive Assisted Living Ambient System (COALAS) Project (No. 4194), URL: http://coalas-project.eu.

## References

[1] N. Friedman, S. Russell, Image segmentation in video sequences: a probabilistic approach, in: Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence, 1997.

[2] G. Stauffer, W. Grimson, Adaptive background mixture models for real-time tracking, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 1999.

[3] A. Elgammal, D. Harwood, L. Davids, Non-parametric model for background subtraction, in: 6th European Conference on Computer Vision, ECCV, 2000.

[4] A. Monnet, A. Mittal, N. Paragios, V. Ramesh, Background modeling and subtraction of dynamic scenes, in: International Conference on Computer Vision, ICCV, 2003.

[5] J. Zhong, S. sclaroff, Segmenting foreground objects from a dynamic textured background via a robust kalman filter, in: International Conference on Computer Vision, ICCV, 2003.

[6] N. Oliver, B. Rosario, A. Pentland, A bayesian computer vision system for modeling human interactions, IEEE Trans. Pattern Anal. Mach. Intell. 22 (2000) 831–843.

[7] E. Lopez-Rubio, R. Luque-Barna, E. Dominguez, Foreground detection in video sequences with probabilistic self-organizing maps, in: IEEE International Workshop on Haptic Audio Visual Environment and their Applications, 2011.

[8] E. Lopez-Rubio, R. Luque-Barna, Stochastic approximation for background modeling, Comput. Vis. Image Underst. 115 (6) (2011) 735–749.

[9] L. Maddalena, A. Petrosino, A self-organizing approach to background subtraction for visual surnelliance applications, IEEE Trans. Image Process. 17 (7) (2008) 1168–1177.

[10] J. Huang, X. Huang, D. Metaxas, Learning with dynamic group sparsity, in: International Conference on Computer Vision, 2009.

[11] C. Zhao, X. Wang, W. Cham, Background subtraction via robust dictionary learning, in: EURASIP Journal on Image and Video Processing, 2011.

[12] C. Lu, J. Shi, J. Jia, Online robust dictionary learning, in: IEEE Conference on Computer Vision and Pattern Recognition, 2013.

[13] J. Wright, A. Ganesh, S. Gao, Y. Ma, Robust principal component analysis? Journal of the ACM 58 (2011).

[14] X. Zhou, C. Yang, W. Yu, Moving object detection by detecting contiguous outliers in the low-rank representation, IEEE Trans. Pattern Anal. Mach. Intell. 35 (3) (2013) 597–610.

[15] A. Sobral, A. Vacavant, A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos, Comput. Vis. Image Underst. 122 (2014) 4–21.

[16] T. Bouwmans, Traditional and recent approaches in background modeling for foreground detection: an overview, Comput. Sci. Rev. 11–12 (2014) 31–66.

[17] C. Qiu, N. Vaswani, Support predicted modified-cs for recursive robust principal components pursuit, in: IEEE International Symposium on Information Theory, ISIT, 2011.

[18] F. Seidel, C. Hage, M. Kleinsteuber, A smoothed lp-norm robust online subspace tracking method for background subtraction in video, in: Machine Vision and Applications, vol. 25, 2014.

[19] J. He, L. Balzano, L. Szlam, Incremental gradient on the grassmannian for online foreground and background separation in subsampled video, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2012.

[20] J. Xu, V.K. Ithapu, J.R.L. Mukherjee, V. Singh, Grassmannian online subspace updates with structured-sparsity, in: IEEE International Conference on Computer Vision, ICCV, 2013.

[21] P. Rodríguez, B. Wohlberg, A matlab implementation of a fast incremental principal component pursuit algorithm for video background modeling, in: Proceedings of IEEE International Conference on Image Processing, ICIP, 2014.

[22] P. Rodriguez, B. Wohlberg, Incremental principal component pursuit for video background modeling, Math. Imaging Vis. 55 (2016) 1–18.

[23] T. Bouwmans, E. Zahzah, Robust PCA via principal component pursuit: A review for a comparative evaluation in video surveillance, Comput. Vis. Image Underst. 122 (2014) 22–34.

[24] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, B. Maisonnier, Mechatronic design of nao humanoid, in: IEEE International Conference on Robotics and Automation, 2009.

[25] Aldebaran Robotics - SAS (Limited Company), Nao documentation, in: http://doc.aldebaran.com/1-14/family/nao_h25/index_h25.html.

[26] Y. Hu, K. Sirlantzis, G. Howells, N. Ragot, P. Rodriguez, An online background subtraction algorithm using contiguously weighed linear regression, in: European Signal Processing Conference, 2015.

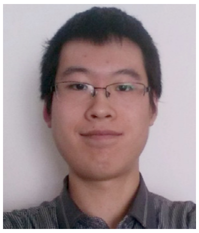
[27] S. Li, Markov Random Field Modeling in Computer Vision, Springer, 1995.

[28] Y. Boykov, O. Veksler, R. Zabih, Fast approximation energy minimization via graph cuts, IEEE Trans. Pattern Anal. Mach. Intell. 23 (11) (2001) 1222–1239.

[29] Aldebaran Robotics - SAS (Limited Company), Naoqi framework, in: http://doc.aldebaran.com/1-14/dev/naoqi/index.html.

[30] E. Carley, K. Sirlantzis, G. Howells, S. Kelly, Non-overlapping dual camera fall detection using the nao humanoid robot, in: International Conference on Emerging Security Technologies, EST, 2014.

[31] L. Li, W. Huang, I. Gu, Q. Tian, Foreground object detection from videos containing complex background, in: ACM International Conference on Multimedia, 2003.



**Yang Hu** received the B.Sc. degree in Electronics and Information Engineering from Dalian Maritime University, China, in 2010, and the M.Sc. degree in Engineering from Dalian University of Technology, China, in 2013. He is currently a Ph.D. candidate in Electronic Engineering at the University of Kent, UK. His research interest focuses on biometrics, computer vision and pattern recognition.

**Dr. Konstantinos Sirlantzis** is Lecturer in Image Processing and Vision at the School of Engineering and Digital Arts, University of Kent. He has a strong track record in image analysis and understanding, artificial intelligence and neural networks for pattern recognition and biometrics-based security applications. Dr. Sirlantzis currently leads a research team working on Social Assistive and Autonomous Robotics (SAR) and health-related Assistive Technologies (AT). He organized and chaired a range of international conferences and workshops such as RASC 2006 and Emerging Security Technologies 2010. Dr. Sirlantzis has published more than 100 peer reviewed papers in journals and conferences.

**Dr. Gareth Howells** is a Reader in Secure Electronic Systems at the University of Kent, UK. He has been involved in research relating to security, biometrics and pattern classification techniques for over twenty five years and has been instrumental in the development of novel ICMetric based security technology deriving secure encryption keys from the operating characteristics of digital systems. He has been awarded, either individually or jointly, several major research grants relating to the pattern classification and security fields, publishing over 180 papers in the technical literature. Recent work has been directed towards the development of secure device authentication systems which has received significant funding from several funding bodies and is currently in the process of being commercially exploited.

**Dr. Nicolas Ragot** received a Master of Engineering from ESIGELEC, France, in 2002 and the Master of Science degree in Electrical Engineering from Université Paris XI in 2003. He earned his Ph.D. in Instrumentation and Vision Sensor Control from Université de Rouen in 2009. Since 2009, he is a lecturer-researcher in the Instrumentation, IT and Systems Lab at ESIGELEC–IRSEEM. His research interests deal with computer vision, omnidirectional vision and their applications to perception for intelligent vehicles.

**Dr. Paul Rodriguez** received the B.Sc. degree in Electrical Engineering from the ''Pontificia Universidad Católica del Perú'' (PUCP), Lima, Peru, in 1997, and the M.Sc. and Ph.D. degrees in Electrical Engineering from the University of New Mexico, USA, in 2003 and 2005 respectively. He spent two years (2005–2007) as a postdoctoral researcher at Los Alamos National Laboratory, and is currently a Full Professor with the Department of Electrical Engineering at PUCP. His research interests include AM–FM models, parallel algorithms, adaptive signal decompositions, and inverse problems in signal and image processing.