2012 IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance

Real-Time Pedestrian Tracking with Bacterial Foraging Optimization

Hoang Thanh Nguyen and Bir Bhanu
University of California, Riverside
Riverside, CA 92521 USA
nthoang@cs.ucr.edu, bhanu@cris.ucr.edu

Abstract

In this paper, we present swarm intelligence algorithms for pedestrian tracking. In particular, we present a modified Bacterial Foraging Optimization (BFO) algorithm and show that it outperforms PSO in a number of important metrics for pedestrian tracking. In our experiments, we show that BFO's search strategy is inherently more efficient than PSO under a range of variables with regard to the number of fitness evaluations which need to be performed when tracking. We also compare the proposed BFO approach with other commonly-used trackers and present experimental results on the CAVIAR dataset as well as on the difficult PETS2010 S2.L3 crowd video.

1. Introduction

Human tracking systems generally consist of three main components: 1) detection, 2) tracking, 3) track association, and analysis or the fusion of multiple trackers. Since much of the computational effort is spent on tracking, improving the speed and effectiveness of the tracking component can greatly benefit many surveillance and security applications. A number of challenges make pedestrian tracking difficult:
1) Change in appearance: The visual appearance of pedestrians may change gradually or suddenly between frames, e.g., a person may turn (changing his/her silhouette) or move away/closer from the camera.
2) Non-uniform lighting and shadows: Light may not be uniform across a scene, may change across frames, and pedestrians will generally cast shadows. Non-uniform lighting results in the changed appearance of the same pedestrian depending on the time and location in the scene. Further, shadows complicate a pedestrian's appearance by altering color and size information which may not carry over into other environments (e.g., walking from a outdoor hall with concrete floors to a room with red carpet).
3) Uncalibrated cameras: Uncalibrated cameras provide no definite ground plane or distance information for a tracking algorithm to utilize; regardless of whether the cameras are fixed or non-static, e.g., movable pan/tilt/zoom (PTZ) cameras. Manually calibrating cameras is a labor-intensive task which is not always easy or feasible. Since additional calibration data can only help a tracking algorithm (e.g., by providing depth data or constraints which can be taken into account into a tracker's fitness function), we focus on tracking in the more common uncalibrated camera environment.

This paper focuses on the object tracking component of a human tracking system which must handle the bulk of the above challenges.

2. Related Work and Contributions

The most successful approaches for pedestrian tracking usually focus around Particle Filters [4, 8], Mean Shift [1], or detection-based tracking. An alternative approach considers a family of biologically-inspired evolutionary computational algorithms known as swarm intelligence, a subset of kernel-based approaches. In this category, the most popular approach is Particle Swarm Optimization (PSO) [15]. These trackers are often used to generate short-term "tracklets" which are then used in methods such as Data Association Tracking (DAT) to produce long-term inter or intra-camera tracks.

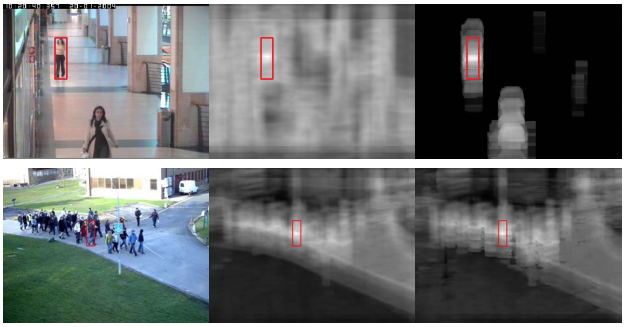
2.1. Contributions

In this paper, we make the following contributions:

1. We adapt the Bacterial Foraging Optimization algorithm for real-time tracking with changes which improve its speed and accuracy. We refer to the modified algorithm as m-BFO.

2. We provide system-level performance measurements of both tracking accuracy and computational efficiency of swarm intelligence algorithms PSO [9] and BFO [11] as well as on commonly-used CamShift and particle filter trackers on the difficult CAVIAR dataset and the PETS2010 *S2.L3* crowd scene. Note that of the numerous approaches to pedestrian tracking, *particle filter* is the most commonly used low-level tracker in

978-0-7695-4797-8/12 $26.00 © 2012 IEEE
DOI 10.1109/AVSS.2012.60



(a) (b) (c)

Figure 1. How pedestrians appear to a tracker. (a) Detected pedestrians, (b) fitness space for the pedestrian using color similarity in the YUV YIQ color space (brighter = higher fitness), (c) fitness space after applying the foreground mask to the input image to reduce background noise.

practice. From the results of [16], however, it has been shown that the evolutionary computation algorithm Particle Swarm Optimization (PSO) is an improved and specific variation of particle filter which outperforms the traditional particle filter implementation. We show that tracking with m-BFO is better than PSO. Therefore, we claim that the proposed m-BFO is better than the particle filter.

3. Technical Approach

The tracking system works as follows. Background subtraction is first performed on input frames. Detected blobs from the subtraction are processed with a pedestrian head-and-shoulders detector, e.g., [10, 14]. The ROI is then extended to encompass the whole body and an appearance signature is created on this initialized silhouette. A fitness function which measures the similarity between two signatures is then be used by the tracker.

3.1. Pedestrian Detection

In order to create an online tracking system, initialization of target locations must be completely automated. The modified Gaussian mixture model (GMM) background subtractor proposed in [17] is used to dynamically learn the background as the input frames are received (with the additional benefit of removing shadows). We remove shadows in order to not confuse the the appearance signature.

In order to automate pedestrian initialization, a Viola-Jones detector [3] trained to detect heads and shoulders and the ROIs of positive detections are extended downward to encompass an estimate of the entire body:

$$height_{body} = height_{head\_shoulders} \times R$$

where R is a fixed ratio that is dependent on the detector used ($R = 3.1$ in this paper).

3.2. Tracking Using Swarm Intelligence

Swarm intelligence is a family of evolutionary stochastic optimization algorithms modeled after biological systems. A swarm consists of a number of particles which independently follow a strategy which allows the swarm to accomplish the common goal of finding an area of optimal fitness.

3.2.1 Bacterial Foraging Optimization

Bacterial Foraging Optimization (BFO) [11] is a stochastic evolutionary swarm intelligence search algorithm designed to model the movement and feeding behavior of *E. coli* bacteria. A swarm consists of a number of particles or "agents" which move or "swim" and "tumble" through an environment searching for concentrations of food (or regions of high fitness from a feature space point of view). Given an image, a swarm of agents is first randomly initialized on the image. The algorithm consists of R "reproduction" loops which execute a number of C "chemotaxis" or movement loops. In each chemotaxis loop, all agents "tumble" (choose a random direction) and are allowed to "swim" (or sample) up to S times in steps of size $Step$ in a gradient hill-climbing manner. At the end of each reproduction step, the bottom T agents with the worst fitness scores die off and an equal number of agents are born at the locations of the T best agents. In this manner, resources are quickly allocated to regions of higher fitness. The agents finally undergo a dispersal step which randomly relocates agents with probability P. This step helps to simulate a changing environment such that the swarm does not fully converge and cease to track in succeeding frames. Figure 2 shows the behavior of a BFO swarm in a fitness space.

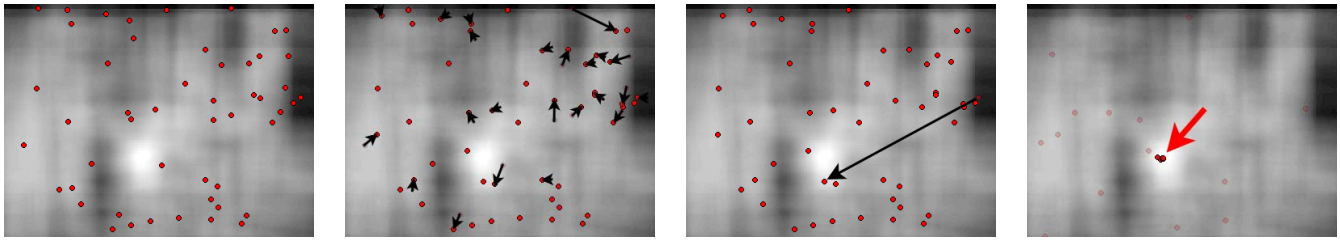
BFO has never been used previously for pedestrian tracking, yet possesses traits which make it suitable to the problem. The near-uniform coverage of the search space is useful for overcoming occlusion (whereas many other approaches lose track once they converge). In addition, the fast propagation of agents to regions of high fitness reduces overhead of having the agents gradually making their way toward global-best fitness regions. This paper utilizes BFO with the following modifications:

Early Termination allows the algorithm to terminate early if positions of adequate fitness are discovered early on.

Lookahead allows the algorithm to accept or reject fitness samples during gradient hill climbing to improve local optimality.

Elitism allows the search agents of highest fitness to stop searching after each round and to select the final location based on a consensus of these agents as opposed to a single highest-fitness sample.

Algorithm 1 summarizes the full procedure.



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

Figure 2. Behavior of a single Bacterial Foraging Optimization swarm searching for a pedestrian. (a) Random initialization, (b) gradient-hill climbing in random directions, (c) death/rebirth of agents with poor fitness to location of agents with best fitness, (d) target location based on consensus of the best agents.

Figure 3. Sample frames of the *S2.L3* crowd scene from the PETS2010 dataset [5] (768 × 576 pixels, 240 frames, and 470 pedestrian ROIs). The objective is the track the two pedestrians labeled A and B as they join in walking with a large incoming crowd.

3.2.2 Particle Swarm Optimization

Particle Swarm Optimization (PSO) [9] is the most commonly-used swarm intelligence approach for tracking and is modeled after the social behavior of schools of fish and flocks of birds. PSO's standard search strategy moves particles according to a linear combination of their current speed and direction, the vector to the position of their local best fitness, and the vector to the position of the swarm's best fitness (see Algorithm 2).

3.2.3 Fitness function

The fitness or objective function used by the two optimization algorithms is computed as follows. For each pedestrian, a color histogram is extracted for all pixels in the YUV YIQ color space using $N = 32$ bins for each of the Y, I, and Q components and normalized to the sum of pixels in the silhouette. YIQ was selected after testing the discrimination power of various colorspaces on the VIPeR dataset [7]. The similarity between two histograms H_0 and H_1 is computed as the histogram intersection:

$$intersect(H_0, H_1) = \sum_{i}^{N} min(H_0^i, H_1^i)$$

Figure 1 shows exhaustively-generated fitness spaces for sample pedestrians by computing the fitness at every pixel against an initialized signature.

CAVIAR Video	# Objs	# ROIs	# Frames
OneStopMoveEnter1cor	20	13,691	1,590
ShopAssistant2cor	20	11,942	3,700
ThreePastShop1cor	20	9,642	1,650
TwoEnterShop3cor	20	6,856	1,149
WalkByShop1cor	20	11,348	2,360

Table 1. Statistics for the five most crowded corridor videos of the CAVIAR dataset.

In order to address tracking of similarly-dressed pedestrians, an additional trajectory smoothness component [13] is useful to give preference to areas of fitness which more closely resemble the current trajectory. Trajectory smoothness at a point can be defined by both smoothness in velocity and smoothness in direction:

$$V_t = P_{t+1} - P_t$$

$$smoothness = W_d \times \frac{V_{t-1} \cdot V_t}{|V_{t-1}||V_t|} + W_v \times \frac{2\sqrt{|V_{T-1}||V_t|}}{|V_{t-1}| + |V_t|}$$

where V_t is the vector between the previous point and a proposed point and W_d, W_v are weights which sum to 1.0 and control emphasis on either direction or velocity. The final fitness function is:

$$fitness = W_s \times smoothness + (1 - W_s) similarity$$

where W_s controls the influence of smoothness.

Algorithm 1 Modified BFO (m-BFO) algorithm

$I \leftarrow$ image to search
$Target \leftarrow$ hist and prev. location of object to search
$R \leftarrow$ number of reproduction steps
$C \leftarrow$ number of chemotaxis steps per reproduction
$S \leftarrow$ max number of swims per chemotaxis step
$Step \leftarrow$ swim step size in pixels
$T \leftarrow$ number of agents to relocate per reproduction
$P \leftarrow$ probability a non-immune agent gets relocated
$Thresh \leftarrow$ min fitness to trigger early termination

procedure BACTERIALFORAGING
  if I is first frame of target **then** ▷ Init first frame
    Initialize agent locations on I
  end if ▷ Early termination?
  if $fitness(Target_{loc}, I, Target_{hist} \geq Thresh$ **then**
    return $Target_{loc}$
  end if
  for R reproduction steps **do** ▷ Begin search
    for C chemotaxis steps **do**
      for all agents A **do**
        $d \leftarrow$ random direction
        for up to S swims **do**
          $l \leftarrow$ new location $Step$ px from A toward direction d
          $f \leftarrow fitness(l, I, Target_{hist})$ ▷ Lookahead
          if $f > A_{current\_fitness}$ **then**
            $A_{current\_fitness} \leftarrow f$
            $A_{current\_location} \leftarrow l$
          else
            Break
          end if
        end for
      end for
    end for
    for all top T agents A with best fitness **do**
      $A_{immunity} \leftarrow true$ ▷ Elitism
    end for ▷ Death/rebirth
    Move the T agents with worst fitness to locations of the T agents with best fitness
  end for ▷ Elimination/dispersal
  for all agents A where $A_{immunity} \neq true$ **do**
    Relocate A to random position with probability P
  end for ▷ Return updated location
  $Target_{loc} \leftarrow$ best location based on all agents A where $A_{immunity} = true$
  return $Target_{loc}$
end procedure

Algorithm 2 Particle Swarm Optimization (PSO) algorithm

$Image \leftarrow$ image to search
$Target \leftarrow$ hist and prev. location of object to search for
$P \leftarrow$ number of agents
$I \leftarrow$ number of iterations
$W_a \leftarrow$ weight of momentum
$W_b \leftarrow$ weight of particle best location
$W_c \leftarrow$ weight of global best location

procedure PARTICLESWARM
  if $Image$ is first frame of target **then**
    Randomly initialize P swarm particles
    $Global_{fitness} \leftarrow 0$
  end if
  for I iterations **do**
    for all particles P **do**
      $fit \leftarrow fitness(P, Image, Target_{hist})$
      if $fit > Localbest_{P,fitness}$ **then**
        $Localbest_{P,fitness} \leftarrow fit$
        $Localbest_{P,location} \leftarrow P_{location}$
      end if
      if $fit > Global_{fitness}$ **then**
        $Global_{fitness} \leftarrow fit$
        $Global_{location} \leftarrow P_{location}$
      end if
      $r_0 \leftarrow rand(0, 1)$
      $r_1 \leftarrow 1 - r_0$
      $V_{t+1} = W_a V_t + W_b r_0 Localbest_{P,location} + W_c r_1 Global_{location}$
      $P_{location} = P_{location} + V_{t+1}$
    end for
  end for
  return $Global_{location}$
end procedure

4. Experimental Results

All trackers are implemented in C++ and experiments are performed using a single Intel Xeon E5345 2.33GHz quad-core CPU.

4.1. Datasets

Experiments are performed on the five most crowed corridor videos of the CAVIAR dataset [6] (see Table 1) as well as the difficult *S2.L3* crowd scenario of the PETS2010 dataset [5].

4.2. Parameter Selection

Parameters are manually optimized for each individual tracking algorithm and then fixed for all experiments. To optimize the parameters of the PSO tracker, for instance, the

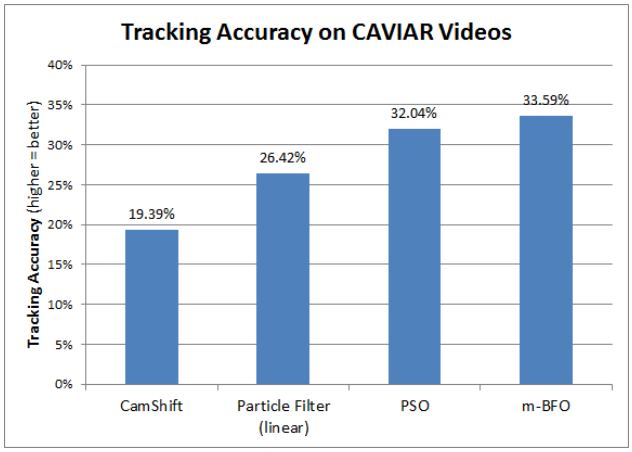


Figure 4. Tracking accuracy comparison of CamShift, particle filter, PSO, and m-BFO on the five CAVIAR videos, averaged over 30 runs for each video.

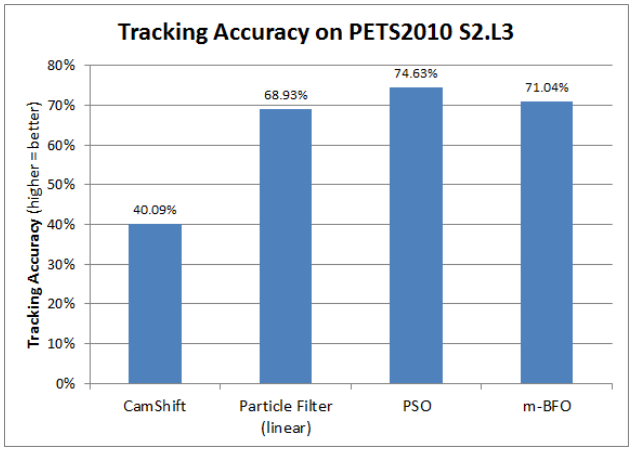


Figure 6. The swarm intelligence approaches perform on par with the particle filter. Results are averaged over 30 runs on the video in Figure 3.

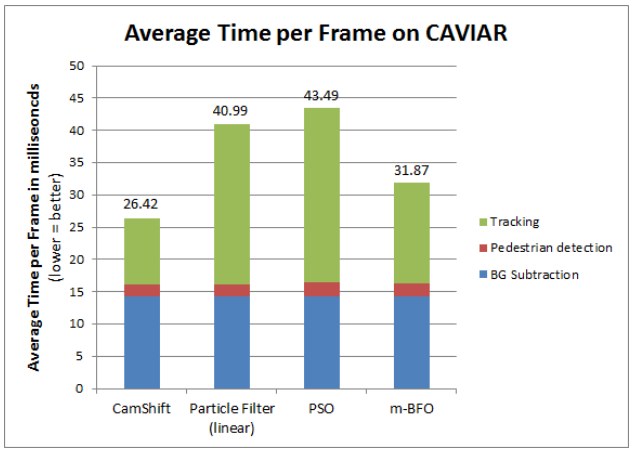


Figure 5. The entire system can be run in real-time on modest hardware (Intel Xeon E5345 2.33GHz). Run times are averaged over the over 30 runs per video.

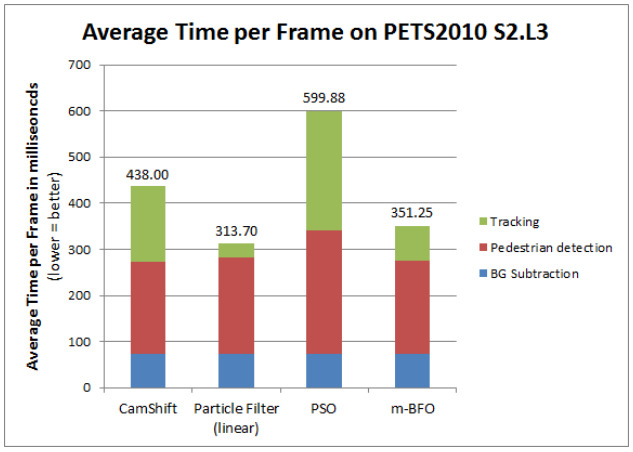


Figure 7. Even with multi-threaded tracking of each pedestrian, the number of pedestrians in the scene (over 40) saturates all available CPU resources (numbers are averaged over 30 runs on an Intel Xeon E5345 2.33GHz quad-core CPU).

tracker was run on a random subset of the CAVIAR videos using various parameter configurations. The parameters with the best average performance were then selected for the full tests. This was done individually for each tracker.

For PSO, $P = 30$, $I = 10$, $W_a = 1.0$, $W_b = 0.04$, and $W_c = 0.04$. For BFO, $P = 10$, $E = 1$, $R = 10$, $C = 1$, and $S = 5$. We use the CamShift available in OpenCV and the particle filter from OpenCVX [12] using 30 particles. Weights for smoothness are $W_d = 0.5$, $W_v = 0.5$, and $W_s = 0.01$.

4.3. Performance Metrics

Tracking accuracy is defined as the percentage of groundtruth ROIs covered by the tracker initialized on that pedestrian. A query ROI $Query$ is considered to be tracking a target if its intersection with the groundtruth ROI GT exceeds at least 50% of their union:

$$is\_tracked(Query, GT) = \frac{Query \cap GT}{Query \cup GT} > 0.50$$

An accuracy of "40%" on CAVIAR means that on average 21,000 of the 53,479 groundtruth ROIs are tracked.

Processing speed is evaluated in time spent per frame (in milliseconds).

4.4. Tracking Results

Figure 4 shows that the swarm intelligence approaches perform on par with particle filter and m-BFO even achieves this performance using fewer resources (Figure 5).

The PETS2010 *S2.L3* video shows that the number of pedestrians (over 40) significantly impacts the speed of the

system, bring the same BFO tracker down to 2.85 FPS (Figure 7). However, the swarm intelligence approaches continue to perform on par with PF accuracy-wise (Figure 6).

4.5. Effect of Parameters

Bacterial Foraging Optimization. The number of agents A, reproduction steps R, chemotaxis steps C, and swims S control the runtime of the BFO algorithm: $O(A*R*C*S)$. The step size $Step$ controls how fast agents swim (higher for bigger steps, lower for finer local search). The number of agents T to relocate controls how much the algorithm balances its search; higher values increase exploitation of areas of higher fitness while lower values increase exploration of the entire search space.

Particle Swarm Optimization. The number of particles P and number of iterations I are the two primary parameters which control the runtime of the PSO algorithm: $O(P*I)$. Setting $W_a < 1.0$ makes particles tend to slow down, > 1.0 makes particles tend to speed up, and $= 1.0$ preserves the current momentum of a particle. The balance of W_b and W_c affect the influence of a particle's local best location vs. the swarm's global best location so far; setting $W_b > W_c$ leads to higher results and increased local search while setting $W_b < W_c$ makes the swarm collapse sooner on a location (though at the risk of converging on a local minima).

4.6. Discussion of Results

Figure 4 shows that the BFO and PSO swarm intelligence approaches achieve comparable results to the more-often used particle filter [8] while Figures 5 and 7 show that such performance is achievable at faster speeds than with particle filter.

5. Conclusions

We provided in-depth results of the tracking performance of PSO and BFO and several other commonly-used trackers on the CAVIAR dataset and the *S2.L3* scenario of the PETS2010 dataset. Since most previous work relies on blob-based similarity, this work can easily be integrated to improve tracking performance of both low-level and higher-level Data Association Trackers (DATs) [2].

Acknowledgements

This work was supported in part by NSF grants 0727129 and 0905671 and ONR grant N00014-09-C-0388.

References

[1] G. R. Bradski. Computer vision face tracking for use in a perceptual user interface. *Intel Technology Journal*, 1998.

[2] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Online multiperson tracking-by-detection from a single, uncalibrated camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 33(9):1820 – 1833, September 2011.

[3] M. Castrillón, O. Déniz, M. Hernández, and C. Guerra. ENCARA2: Real-time detection of multiple faces at different resolutions in video streams. *Journal of Visual Communication and Image Representation (JVCIR)*, pages 130–140, 2007.

[4] X. Fen and G. Ming. Pedestrian tracking using particle filter algorithm. In *IEEE International Conference on Electrical and Control Engineering (ICECE)*, pages 1478 –1481, 2010.

[5] J. Ferryman and A. Ellis. PETS2010: Dataset and challenge. *Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 143–150, 2010. `http://pets2010.net/`.

[6] R. B. Fisher. The PETS04 surveillance ground-truth data sets. In *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)*, 2004. `http://homepages.inf.ed.ac.uk/rbf/CAVIAR`.

[7] D. Gray, S. Brennan, and H. Tao. Evaluating appearance models for recognition, reacquisition, and tracking. In *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)*, 2007.

[8] M. Isard and A. Blake. CONDENSATION: Conditional density propagation for visual tracking. *International Journal of Computer Vision (IJCV)*, 29:5–28, 1998.

[9] J. Kennedy and R. Eberhart. Particle swarm optimization. *International Conference on Neural Networks (ICNN)*, 4:1942–1948 vol.4, 1995.

[10] M. Li, Z. Zhang, K. Huang, and T. Tan. Rapid and robust human detection and tracking based on omega-shape features. In *Sixteenth IEEE International Conference on Image Processing (ICIP)*, pages 2545 –2548, November 2009.

[11] K. M. Passino. Biomimicry of bacterial foraging for distributed optimization and control. *IEEE Control Systems Magazine (CSM)*, Vol. 22, No. 3:52–67, 2002.

[12] N. Seo. OpenCVX: Yet another OpenCV eXtension. `http://code.google.com/p/opencvx/`.

[13] I. K. Sethi and R. Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 9(1):56 – 73, January 1987.

[14] P. Viola and M. Jones. Robust real-time object detection. *Second International Workshop on Statistical and Computational Theories of Vision (SCTV)*, 2001.

[15] X. Zhang, W. Hu, W. Li, W. Qu, and S. Maybank. Multi-object tracking via species based particle swarm optimization. In *12th IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1105 –1112, October 2009.

[16] X. Zhang, W. Hu, S. Maybank, X. Li, and M. Zhu. Sequential particle swarm optimization for visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1 –8, June 2008.

[17] Z. Zivkovic and F. van der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters (PRL)*, 27:773–780, 2006.