Computer Vision and Image Processing for Automated Surveillance

Conrad Sanderson

NICTA

The University of Queensland Australia

- e-mail: conradsand [at] ieee [dot] org
- web: http://conradsanderson.id.au

Part 1: Robust Foreground Detection

Published in:

- V. Reddy, C. Sanderson, B.C. Lovell.
  Improved Foreground Detection via Block-based Classifier Cascade with Probabilistic Decision Integration.
  IEEE Transactions on Circuits and Systems for Video Technology,
  Vol. 23, No. 1, 2013.

- official version: http://dx.doi.org/10.1109/TCSVT.2012.2203199
- arXiv pre-print: http://arxiv.org/pdf/1303.4160v1
- C++ source code: http://arma.sourceforge.net/foreground/

Algorithm integrated into a commercial surveillance product!

- Early approaches to foreground segmentation:
  - obtain clear view of the background
  - given a frame, subtract the background
  - leftover pixels: foreground
- Problems:
  - background changes with time (eg. illumination changes)
  - **noisy** $\therefore$ clean up pixels using ad-hoc post-processing (eg. erosion)
- Common approach:
  - model each background location with a stochastic model
  - pixels with low probability: foreground
  - adapt model to take into account background changes
  - better, but **still noisy**, still requires **ad-hoc post-processing**
- **Core problems:**
  - classification done at the pixel level
  - rich contextual information **not taken into account**
  - foreground segmentation $\neq$ background subtraction

Proposed Method

1
- divide given image into **overlapping** blocks
- generate low-dimensional descriptors for each block

2
- classify each block into foreground/background
- use a **classifier cascade**

3
- for each pixel **integrate** block level decisions
- results in pixel-level foreground/background segmentation
- ad-hoc post-processing **not necessary**

4
- background model re-initialisation
- for scenarios with sudden and significant scene changes

Step 1

- divide given image into **overlapping** blocks
- block size: 8×8
- generate low-dimensional descriptors for each block:

$$\boldsymbol{d}_{(i,j)} = \left[c_0^{[r]}, \cdots, c_3^{[r]}, \quad c_0^{[g]}, \cdots, c_3^{[g]}, \quad c_0^{[b]}, \cdots, c_3^{[b]} \right]$$

$c_n^{[k]} = n$-th 2D DCT coefficient for the k-th colour $\{$r, g, b$\}$

Step 2

- initial classification of each block into foreground/background
- use a classifier cascade
  - as soon as one classifier classifies as background
- each stage stage analyses a block from a unique perspective:

i: texture analysis: $p(\boldsymbol{d}_{(i,j)}) = \mathcal{N}(\boldsymbol{d}_{(i,j)} \mid \boldsymbol{\mu}_{(i,j)}, \boldsymbol{\Sigma}_{(i,j)})$

- background model for each block
- background model trained using a robust method, capable of using a cluttered background
- background model is adapted during execution

if $p(\boldsymbol{d}_{(i,j)}) > T_{(i,j)}$: classify as background

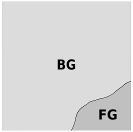
ii: shadow: if $\text{cosdist}(\boldsymbol{d}_{(i,j)}, \boldsymbol{\mu}_{(i,j)}) < C_1$: classify as background

iii: temporal correlation check: classify block as background if:

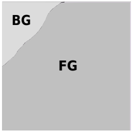
(a) $\boldsymbol{d}_{(i,j)}^{[\text{prev}]}$ was classified as background, AND

(b) $\text{cosdist}(\boldsymbol{d}_{(i,j)}^{[\text{prev}]}, \boldsymbol{d}_{(i,j)}) \leq C_2$

Step 3



block A



block B

- **no overlapping**:
  misclassification inevitable at the pixel level

- **with overlapping**:
  for each pixel, integrate initial classifications of all relevant blocks:

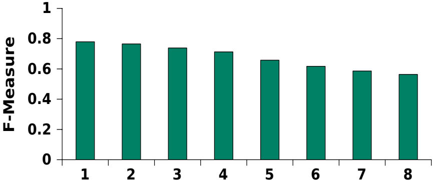
$$P\left(\text{fg} \mid I_{(x,y)}\right) = \frac{B^{\text{fg}}_{(x,y)}}{B^{\text{total}}_{(x,y)}} = \frac{\text{num. of foreground blocks containing pixel } I(x,y)}{\text{total num. of blocks containing pixel } I(x,y)}$$

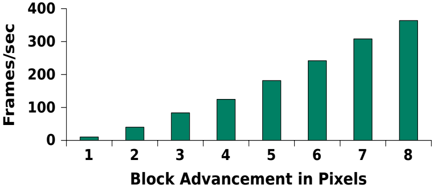
- classify pixel $I(x,y)$ as foreground if $P\left(\text{fg} \mid I_{(x,y)}\right) \geq 0.90$

- **no need for any ad-hoc post-processing!**

Trade-Off: Accuracy vs Speed

- sliding block-by-block analysis
- each block is 8×8
- blocks are overlapping
- 1 pixel advance = max overlap
- 8 pixel advance = no overlap
- $F\text{-}measure = 2\frac{recall \cdot precision}{recall + precision}$
- Achieves real-time processing at the cost of slightly reduced accuracy (2 pixel advance)





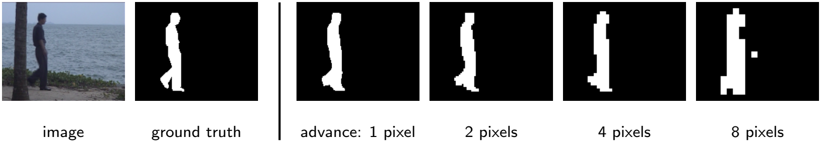


image | ground truth | advance: 1 pixel | 2 pixels | 4 pixels | 8 pixels

Comparative Evaluation

- Datasets:
  - I2R: http://perception.i2r.a-star.edu.sg/bk_model/bk_index.html
  - Wallflower: research.microsoft.com/en-us/um/people/jckrumm/WallFlower/TestImages.htm
- Compare with:
  - GMM based [1] (with morphological post-processing)
  - feature histograms [2]
  - Normalised Vector Distances (NVD) [3] (block based approach)
  - Probabilistic Self-Organizing Maps (SOM) [4]
  - Stochastic Approximation (SA) [5]

---

[1] P. KaewTraKulPong et al.: *An improved adaptive background mixture model for real-time tracking with shadow detection*. In: *Proc. European Workshop Advanced Video Based Surveillance Systems* (2001).

[2] L. Li et al.: *Foreground object detection from videos containing complex background*. In: *Proc. International Conference on Multimedia* (2003).

[3] T. Matsuyama et al.: *Background subtraction under varying illumination*. In: *Systems and Computers in Japan* 37.4 (2006).

[4] Ezequiel López-Rubio et al.: *Foreground detection in video sequences with probabilistic self-organizing maps*. In: *International Journal of Neural Systems* 21.3 (2011).

[5] Ezequiel López-Rubio et al.: *Stochastic approximation for background modelling*. In: *CVIU* 115.6 (2011).

original images

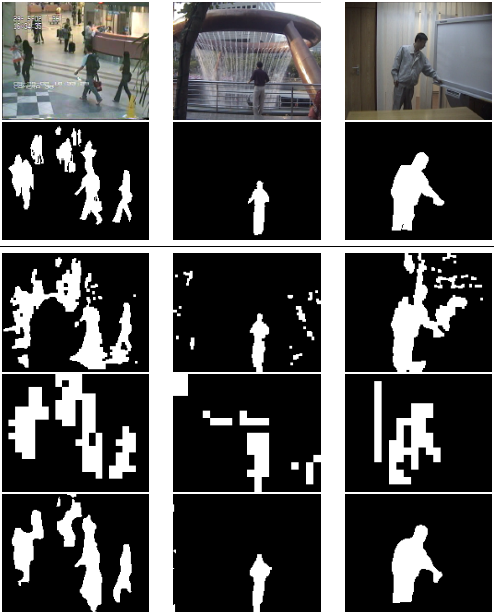
ground truth

GMM

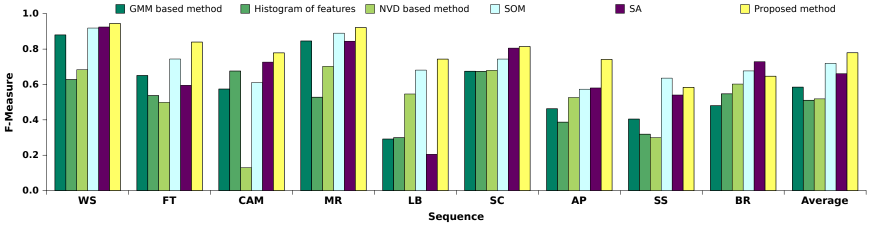
(with post-processing!)

NVD

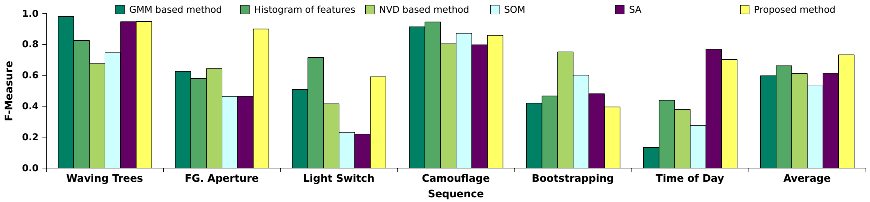
proposed method



Quantitative evaluation on I2R dataset:



Quantitative evaluation on Wallflower dataset:



- On average, the proposed method obtains more accurate foreground detection
- More consistent performance across various environments
- Does not require any ad-hoc post-processing
- Can achieve real-time processing
- C++ implementation available as open source code: http://arma.sourceforge.net/foreground/

Part 2: Person Re-Identification

Published in:

- A. Alavi, Y. Yang, M. Harandi, C. Sanderson.
  Multi-Shot Person Re-Identification via Relational Stein Divergence.
  IEEE International Conference on Image Processing (ICIP), 2013.
- official version: http://dx.doi.org/10.1109/ICIP.2013.6738731
- arXiv pre-print: http://arxiv.org/pdf/1403.0699v1

- Given images of a person from camera view 1, find matching person from camera view 2
- Difficult:
    - imperfect person detection / localisation
    - large pose changes
    - occlusions
    - illumination changes
    - low resolution

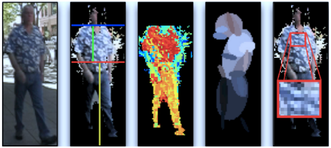
Popular Previous Approaches

Partial Least Squares (PLS) based [6]

- decompose an image into overlapping blocks
- extracts features from each block: textures, edges, colours
- concatenated into one feature vector (high dimensional)
- learn discriminative dimensionality reduction for each person
- classification: projection to each model + Euclidean distance
- **downsides:**
  - concatenation = fixed spatial relations between blocks
  - ∴ **does not allow for movement of blocks**!
  - ∴ **easily affected** by imperfect localisation and pose variations

[6] W.R. Schwartz et al.: *Learning discriminative appearance-based models using partial least squares*. In: *SIBGRAPI* (2009).

Symmetry-Driven Accumulation of Local Features (SDALF)[7]



- foreground detection
- two horizontal axes of asymmetry to isolate: head, torso, legs
- use vertical axes of appearance symmetry for torso and legs
- extract: HSV histogram, stable colour regions, textures
- estimation of symmetry affected by deformations & pose variations:
  - ∴ **noisy features**

[7] M. Farenzena et al.: *Person re-identification by symmetry-driven accumulation of local features*. In: *CVPR* (2010).

Proposed Method

- Aim to obtain a compact & robust representation of an image:
  - allow for imprecise person detection
  - allow for deformations
  - $\therefore$ do not use rigid spatial relations
  - do not use brittle feature extraction based on symmetry
- Steps:
  1. foreground estimation
  2. for each foreground pixel, extract feature vector containing colour and local texture information
  3. represent the set of feature vectors as a covariance matrix
  4. covariance matrix is a point on a Riemannian manifold
  5. map matrix from R. manifold to vector in Euclidean space, **while taking into account curvature of the manifold!**
  6. use standard machine learning for classification

Feature Extraction

- For each foreground pixel, extract feature vector:

$$\boldsymbol{f} = [\ x,\ y,\ HSV_{xy},\ \Lambda_{xy},\ \Theta_{xy}\ ]^T$$

where

  - $HSV_{xy} = [H_{xy}, S_{xy}, \widehat{V}_{xy}]$ = colour values of the HSV channels
  - $\Lambda_{xy} = [\lambda^R_{xy}, \lambda^G_{xy}, \lambda^B_{xy}]$ = gradient magnitudes
  - $\Theta_{xy} = [\theta^R_{xy}, \theta^G_{xy}, \theta^B_{xy}]$ = gradient orientations

- (not limited to above, can certainly use other features)

- Given set $F = \{\boldsymbol{f}_i\}_{i=1}^{N}$, calculate covariance matrix:

$$\boldsymbol{C} = \frac{1}{N-1} \sum_{i=1}^{N} (\boldsymbol{f}_i - \boldsymbol{\mu})(\boldsymbol{f}_i - \boldsymbol{\mu})^T$$

  - low dimensional representation, independent of image size

How to Compare Covariance Matrices?

- Naive method:
  - brute-force vectorisation of matrix
  - use Euclidean distance between resultant vectors
- Naive method kind-of works, BUT:
  - covariance matrix = symmetric positive definite (SPD) matrix
  - space of SPD matrices = interior of a convex cone in $\mathbb{R}^{D^2}$
  - space of SPD matrices = Riemannian manifold[8]
  - $\therefore$ covariance matrix = point on a Riemannian manifold
  - naive method **disregards** curvature of manifold!
  - geodesic distance: shortest path along the manifold (eg. on a sphere)

[8] X. Pennec et al.: *A Riemannian Framework for Tensor Computing*. In: *IJCV* 66.1 (2006).

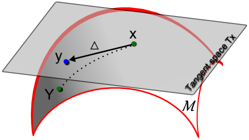
How to Measure Distances on Riemannian Manifolds?

- Use Affine Invariant Riemannian Metric (AIRM) [9]:

$$\delta_R\left(\boldsymbol{A}, \boldsymbol{B}\right) = \left\| \log\left(\boldsymbol{B}^{-\frac{1}{2}} \boldsymbol{A} \boldsymbol{B}^{-\frac{1}{2}} \right) \right\|_F$$

  - intensive use of matrix inverses, square roots, logarithms [10]
  - $\therefore$ **computationally demanding!**

- Choose a tangent pole, and map all points to tangent space



  - tangent space is Euclidean space
  - faster, but less precise
  - **true geodesic distances are only to the tangent pole!**

[9] X. Pennec et al.: *A Riemannian Framework for Tensor Computing*. In: *IJCV* 66.1 (2006).

[10] V. Arsigny et al.: *Log-Euclidean metrics for fast and simple calculus on diffusion tensors*. In: *Magnetic Resonance in Medicine* 56.2 (2006).

Stein Divergence

- Related to AIRM, but much faster [11]

$$\delta_S(\boldsymbol{A}, \boldsymbol{B}) = \log\left(\det\left(\frac{\boldsymbol{A}+\boldsymbol{B}}{2}\right)\right) - \frac{1}{2}\log\left(\det\left(\boldsymbol{A}\boldsymbol{B}\right)\right)$$

- divergence, **not a true distance!**

Proposed: Relational Divergence Classification

- Obtain a set of training covariance matrices $\{\boldsymbol{T}\}_{i=1}^{N}$
- For matrix $\boldsymbol{C}$, calculate its Stein divergence to each training covariance matrix:

$$[\ \delta_S(\boldsymbol{C}, \boldsymbol{T}_1)\ \ \delta_S(\boldsymbol{C}, \boldsymbol{T}_2)\ \ \cdots\ \ \delta_S(\boldsymbol{C}, \boldsymbol{T}_N)\ ]\ \in\ \mathbb{R}^N$$

- In effect, we have **mapped** matrix $\boldsymbol{C}$ from manifold space to Euclidean space, while (approximately) taking into account manifold curvature
- Can now use **standard** machine learning methods

[11] S. Sra: *A new metric on the manifold of kernel matrices with application to matrix geometric means.* In: *NIPS* (2012).

Comparative Evaluation

- After mapping from manifold space to Euclidean space, use LDA based classifier
- Use ETHZ dataset [12]
  - captured from a moving camera
  - occlusions and wide variations in appearance
- Compare with:
  - directly using the Stein divergence
  - Histogram Plus Epitome (HPE) [13]
  - Partial Least Squares (PLS)[14]
  - Symmetry-Driven Accumulation of Local Features (SDALF)[15]

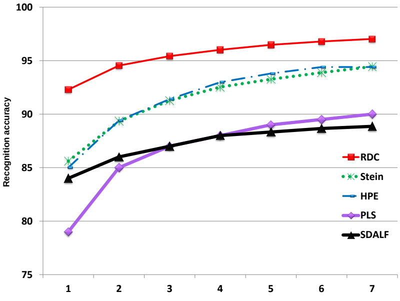
[12] A. Ess et al.: *Depth and Appearance for Mobile Scene Analysis*. In: *ICCV* (2007).

[13] Loris Bazzani et al.: *Multiple-Shot Person Re-identification by HPE Signature*. In: *ICPR* (2010).

[14] W.R. Schwartz et al.: *Learning discriminative appearance-based models using partial least squares*. In: *SIBGRAPI* (2009).

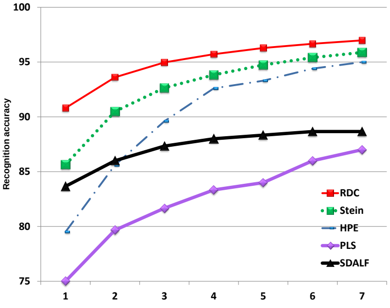
[15] M. Farenzena et al.: *Person re-identification by symmetry-driven accumulation of local features*. In: *CVPR* (2010).

ETHZ sequence 1



rank

ETHZ sequence 2



rank

RDC	=	Relational Divergence Classification (proposed method)
Stein	=	direct use of Stein divergence (no mapping)
HPE	=	Histogram Plus Epitome
PLS	=	Partial Least Squares
SDALF	=	Symmetry-Driven Accumulation of Local Features

Part 3: Object Tracking on Manifolds

Published in:

- S. Shirazi, C. Sanderson, C. McCool, M. Harandi.
  Bags of Affine Subspaces for Robust Object Tracking.
  arXiv:1408.2313, 2014.
- Full paper: http://arxiv.org/pdf/1408.2313v2

Object tracking is hard:

- occlusions
- deformations
- variations in pose
- variations in scale
- variations in illumination
- imposters / similar objects

Tracking algorithms can be categorised into:

1. generative tracking
   - represent object through a particular appearance model
   - search for image area with most similar appearance
   - examples: mean shift tracker [16] and FragTrack [17]
2. discriminative tracking
   - treat tracking as binary classification task
   - discriminative classifier trained to explicitly separate object from non-object areas
   - example: Multiple Instance Learning (MILTrack) [18]
   - example: Tracking-Learning-Detection (TLD) [19]
   - requires larger training dataset than generative tracking

---

[16] Dorin Comaniciu et al.: *Kernel-based object tracking*. In: *IEEE PAMI* 25.5 (2003).

[17] A. Adam et al.: *Robust fragments-based tracking using the integral histogram*. In: *IEEE CVPR* (2006).

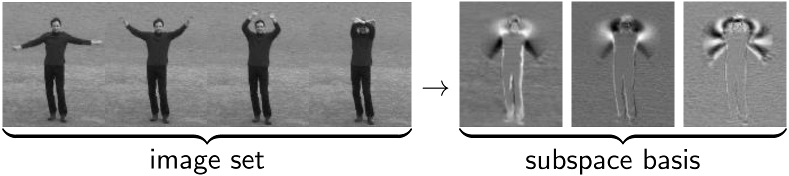
[18] B. Babenko et al.: *Robust object tracking with online multiple instance learning*. In: *IEEE PAMI* 33.8 (2011).

[19] Z. Kalal et al.: *Tracking-learning-detection*. In: *IEEE PAMI* 34.7 (2012).

Promising approach for generative tracking:

→ model object appearance via **subspaces**

- originated with the work of Black and Jepson [20]
- apply eigen decomposition on a set of object images
- resulting eigen vectors define a linear subspace
- subspaces able to capture perturbations of object appearance



[20] Michael J Black et al.: *EigenTracking: Robust matching and tracking of articulated objects using a view-based representation*. In: *IJCV* 26.1 (1998), pp. 63–84.

Many developments to address limitations:

- sequentially update the subspace [21][22]
- more robust update of the subspace [23][24][25]
- online updates using distances to subspaces on Grassmann manifolds [26]

But still not competitive with discriminative methods!

[21] Danijel Skocaj et al.: *Weighted and robust incremental method for subspace learning*. In: *ICCV* (2003).

[22] Yongmin Li: *On incremental and robust subspace learning*. In: *Pattern Recognition* 37.7 (2004).

[23] J. Ho et al.: *Visual tracking using learned linear subspaces*. In: *IEEE CVPR* (2004).

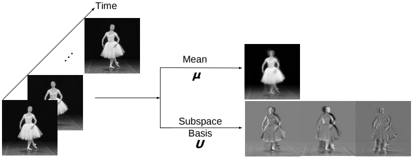
[24] Jongwoo Lim et al.: *Incremental learning for visual tracking*. In: *NIPS* (2004).

[25] D.A. Ross et al.: *Incremental learning for robust visual tracking*. In: *IJCV* 77.1-3 (2008).

[26] T. Wang et al.: *Online subspace learning on Grassmann manifold for moving object tracking in video*. In: *IEEE ICASSP* (2008).

Two major **shortcomings** in all subspace based trackers:

1. **mean** of the image set is not used
   - the mean can hold useful discriminatory information!

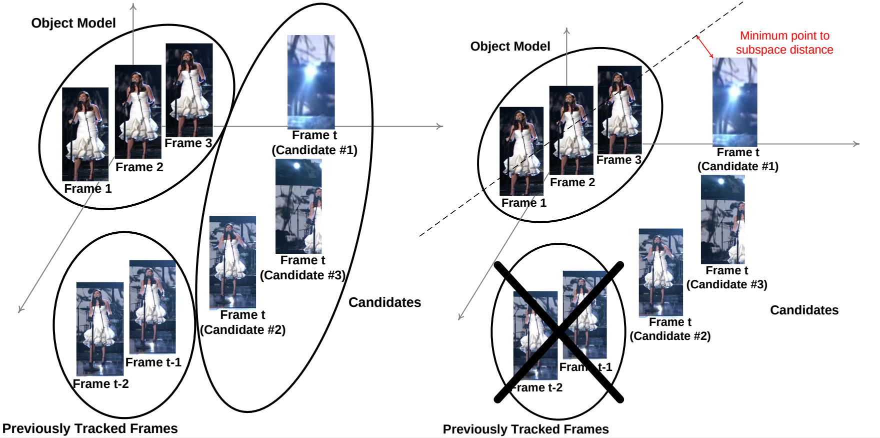


2. search for object location is typically done using **point-to-subspace distance**
   - compare a candidate image area from ONE frame against the model (multiple frames)
   - easily affected by drastic appearance changes (eg. occlusions)

Point-to-subspace distance

- each image is represented as a point
- object model (subspace) is conceptually represented as a line

- previously tracked frames are disregarded when comparing candidate frames to object model
- reduces memory of the system
- can easily lead to incorrect frame selection



Proposed Tracking Approach

Comprised of 4 intertwined components:

1. particle filtering framework (for efficient search)
2. model appearance of each particle as an **affine subspace**
   - takes into account tracking history (longer memory)
   - takes into account the mean
3. object model: **bag of affine subspaces**
   - continuously updated set of affine subspaces
   - longer memory
   - handles drastic appearance changes
4. likelihood of each particle according to object model:
   - **(i)** distance between means
   - **(ii)** distance between bases: **subspace-to-subspace distance**

1. Particle Filtering Framework

- Using standard particle filtering framework [27]
- History of object's location is parameterised as a distribution
  - set of particles represents the distribution
  - each particle represents a location and scale:

$$\boldsymbol{z}_i^{(t)} = [x_i^{(t)}, y_i^{(t)}, s_i^{(t)}]$$

- Use distribution to create a set of candidate object locations in a new frame
- Obtain **appearance** of each particle: $\mathcal{A}_i^{(t)}$
- Choose new location of object as the particle with highest likelihood according to **object model** $\mathcal{B}$:

$$\boldsymbol{z}_*^{(t)} = \boldsymbol{z}_j^{(t)}, \quad \text{where} \quad j = \operatorname*{argmax}_i \; p\left(\mathcal{A}_i^{(t)} | \mathcal{B}\right)$$

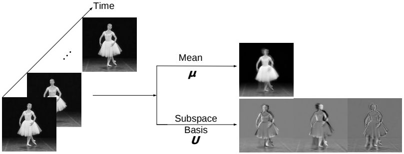
[27] M.S. Arulampalam et al.: *A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking*. In: *IEEE Trans. Signal Processing* 50.2 (2002).

2. Model Appearance of Each Particle as an Affine Subspace

- Affine subspace represented as a 2-tuple:

$$\mathcal{A}_i^{(t)} = \left\{ \boldsymbol{\mu}_i^{(t)}, \boldsymbol{U}_i^{(t)} \right\}$$

$\boldsymbol{\mu}$: mean
$\boldsymbol{U}$: subspace basis



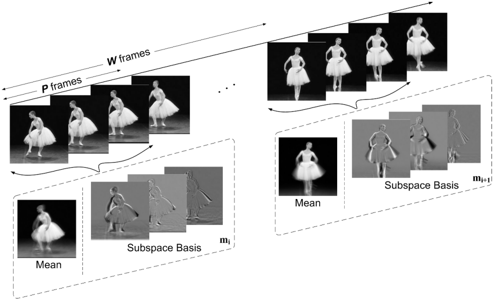
- Appearance includes:
  1. appearance of the i-th candidate location
  2. appearance of tracked object in several preceding frames

3. Object Model: Bag of Affine Subspaces

- Drastic appearance changes (eg. occlusions) adversely affect subspaces
- Instead of modelling the object using only one subspace, use a **bag of subspaces**:

$$\mathcal{B} = \{\mathcal{A}_1, \cdots, \mathcal{A}_K\}$$

- Simple **model update:** the bag is updated every W frames by replacing the oldest affine subspace with the newest



4. Likelihood of Each Particle According to Object Model

- Particle filtering framework requires: $p\left(\mathcal{A}_i^{(t)}|\mathcal{B}\right)$
- Appearance of each candidate area: $\mathcal{A}_i^{(t)} = \left\{\boldsymbol{\mu}_i^{(t)}, \boldsymbol{U}_i^{(t)}\right\}$
- Object model: $\mathcal{B} = \{\mathcal{A}_1, \cdots, \mathcal{A}_K\}$
- Our definition: $p\left(\mathcal{A}_i^{(t)}|\mathcal{B}\right) = \sum_{k=1}^{K} \widehat{p}\left(\mathcal{A}_i^{(t)}|\mathcal{B}[k]\right)$
  - $\mathcal{B}[k]$ is the k-th affine subspace in bag $\mathcal{B}$
  - $\widehat{p}\left(\mathcal{A}_i^{(t)}|\mathcal{B}[k]\right) = \frac{p\left(\mathcal{A}_i^{(t)}|\mathcal{B}[k]\right)}{\sum_{j=1}^{N} p\left(\mathcal{A}_j^{(t)}|\mathcal{B}[k]\right)}$, where $N =$ num. of particles
  - $p\left(\mathcal{A}_i^{(t)}|\mathcal{B}[k]\right) \approx \exp\left\{-\underbrace{\mathrm{dist}(\mathcal{A}_i^{(t)}, \mathcal{B}[k])}_{\textbf{distance between affine subspaces}}\right\}$

- Define the **distance** between two affine subspaces as:

$$\text{dist}(\mathcal{A}_i, \mathcal{A}_j) = \alpha\, \widehat{\mathrm{d}}_o\left(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j\right) + (1-\alpha)\, \widehat{\mathrm{d}}_g\left(\boldsymbol{U}_i, \boldsymbol{U}_j\right)$$

  - $\widehat{\mathrm{d}}_o\left(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j\right)$ = normalised Euclidean distance between means
  - $\widehat{\mathrm{d}}_g\left(\boldsymbol{U}_i, \boldsymbol{U}_j\right)$ = normalised geodesic distance between bases

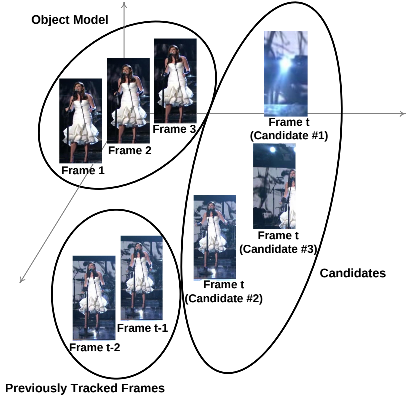
- **Grassmann manifolds:**
  - space of all n-dimensional linear subspaces of $\mathbb{R}^D$ for $0<n<D$
  - a point on Grassmann manifold $\mathcal{G}_{D,n}$ is a $D \times n$ matrix

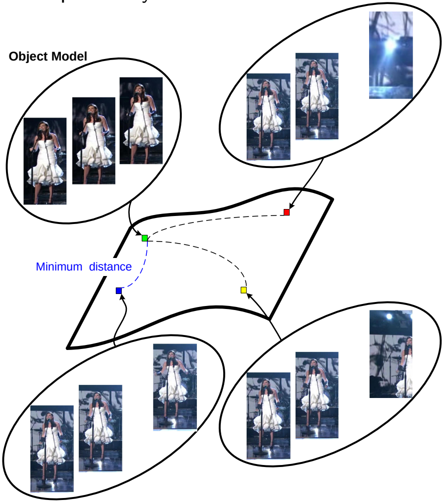
- Geodesic distance between subspaces $\boldsymbol{U}_i$ and $\boldsymbol{U}_j$ is:

$$\mathrm{d}_g\left(\boldsymbol{U}_i, \boldsymbol{U}_j\right) = \|[\theta_1, \theta_2, \cdots, \theta_n]\|$$

  - $[\theta_1, \theta_2, \cdots, \theta_n]$ = vector of principal angles
  - θ_1 = smallest angle btwn. all pairs of unit vectors in $\boldsymbol{U}_i$ and $\boldsymbol{U}_j$
  - principal angles are computed via SVD of $\boldsymbol{U}_i^T \boldsymbol{U}_j$

- each image set is represented as a point on a Grassmann manifold
- explicitly takes into account previously tracked frames





Computational Complexity

- Generation of new affine subspace:
  - patch size: $H_1 \times H_2$
  - represent patch as vector: $D = H_1 \times H_2$
  - use patches from P frames
  - $\therefore$ SVD of $D \times P$ matrix
  - $D >> P$
  - using optimised thin SVD[28]: $\mathcal{O}\left(Dn^2\right)$ operations
  - $n =$ number of basis vectors
- To keep computational requirements relatively low:
  - patch size: 32×32
  - number of frames: 5
  - number of basis vectors: 3

[28] Matthew Brand: *Fast low-rank modifications of the thin singular value decomposition*. In: *Linear Algebra and its Applications* 415.1 (2006).

Comparative Evaluation

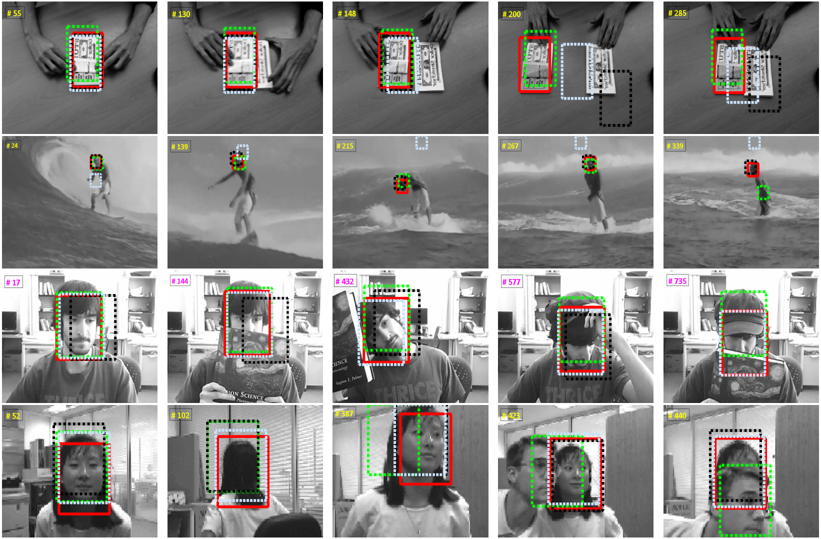
- Evaluation on 8 commonly used videos in the literature
- Compared against recent tracking algorithms:
  - Tracking-Learning-Detection (TLD)[29]
  - Multiple Instance Learning (MILTrack) [30]
  - Sparse Collaborative Model (SCM) [31]
- Qualitative and quantitative evaluation

[29] Z. Kalal et al.: *Tracking-learning-detection*. In: *IEEE PAMI* 34.7 (2012).

[30] B. Babenko et al.: *Robust object tracking with online multiple instance learning*. In: *IEEE PAMI* 33.8 (2011).

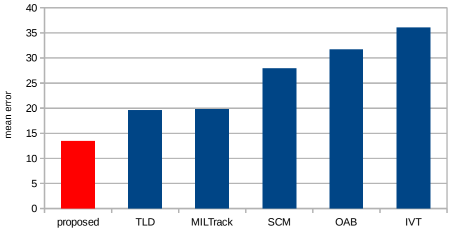
[31] Wei Zhong et al.: *Robust object tracking via sparsity-based collaborative model*. In: *IEEE CVPR* (2012).

proposed method	TLD (PAMI 2012)	MILTrack (PAMI 2011)	SCM (CVPR 2012)



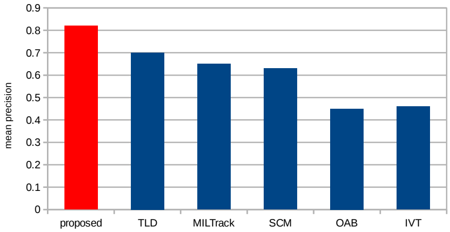
Quantitative Results

- Used two measures:
  1. **centre location error:** distance between the centre of the bounding box and the ground truth object position
  2. **precision:** percentage of frames where the estimated object location is within a pre-defined distance to ground truth



average error

(lower = better)



average precision

(higher = better)

Future Work

- Affected by motion blurring (rapid motion or pose variations)
- Better update scheme by measuring the effectiveness of new affine subspace before adding it to the bag
- Allow bag size and update rate to be dynamic, possibly dependent on tracking difficulty

Part 4: Related Work on Surveillance Technologies

Overview of our papers on:

- face recognition in realistic scenarios
- shadow removal for improved object detection and tracking
- estimation of true background in cluttered surveillance videos
- face selection for improved recognition in surveillance videos

Y. Wong, M. Harandi, C. Sanderson.
On Robust Face Recognition via Sparse Coding: The Good, The Bad and The Ugly.
IET Biometrics, Vol. 3, No. 4, 2014.

- official version: http://dx.doi.org/10.1049/iet-bmt.2013.0033
- arXiv pre-print: http://arxiv.org/pdf/1303.1624v1

Summary:

- Shows that most face recognition systems based on sparse coding:
  - rely on flawed assumptions
  - are inapplicable to realistic scenarios: open-set identification and misalignment (imperfect face detection / localisation)
- Proposes sparse coding on patch-based face representation
  - results in a robust face descriptor
  - robust to face misalignment & environmental variations
  - readily applicable to open set identification and verification

A. Sanin, C. Sanderson, B.C. Lovell.
Shadow Detection: A Survey and Comparative Evaluation of Recent Methods.
Pattern Recognition, Vol. 45, No. 4, 2012.

- official version: http://dx.doi.org/10.1016/j.patcog.2011.10.001
- arXiv pre-print: http://arxiv.org/pdf/1304.1233v1
- C++ source code: http://arma.sourceforge.net/shadows/

Summary:

- Shadow removal is a critical step for improving object detection and object tracking
- Places shadow detection algorithms in a feature-based taxonomy: chromacity, physical, geometry and textures
- Quantitatively compares recent algorithms in terms of shadow detection and discrimination rates, colour desaturation
- Small-region texture based method is especially robust

V. Reddy, C. Sanderson, B.C. Lovell.

A Low-Complexity Algorithm for Static Background Estimation from Cluttered Image Sequences in Surveillance Contexts.

Image and Video Processing, 2011.

- official version: http://dx.doi.org/10.1155/2011/164956
- arXiv pre-print: http://arxiv.org/pdf/1303.2465v1
- C++ source code: http://arma.sourceforge.net/background_est/

Summary:

- True background model is unavailable in many practical circumstances: surveillance videos cluttered with foreground objects
- Propose a sequential technique for estimation of static backgrounds
- Background is reconstructed through a Markov Random Field framework
- Image sequences are analysed on a block-by-block basis; clique potentials are computed based on the combined frequency response of the candidate block and its neighbourhood

Y. Wong, S. Chen, S. Mau, C. Sanderson, B.C. Lovell.
Patch-based Probabilistic Image Quality Assessment for Face Selection and Improved Video-based Face Recognition.
IEEE Conf. Computer Vision and Pattern Recognition Workshops (CVPRW), 2011.

- official version: http://dx.doi.org/10.1109/CVPRW.2011.5981881
- arXiv pre-print: http://arxiv.org/pdf/1304.0869v2
- surveillance database: http://arma.sourceforge.net/chokepoint/

Summary:

- In face recognition from surveillance videos, face images are captured over multiple frames in uncontrolled conditions
- Using all face images (including poor quality images) can degrade face recognition performance!
- Current face selection techniques are incapable of simultaneously handling all relevant environmental factors
- Propose an efficient patch-based face image quality assessment algorithm which quantifies similarity of face images to a probabilistic face model, representing an "ideal" face

Part 5: Rethinking Approaches to Computer Vision Research

- Adapt the main **lessons** learned[32] from **big data**:
  - more data provides more depth
  - some correlations or trends are only visible in large datasets
  - don't sample, use all data: sampling throws out information!
  - sampling is a **leftover** from a bygone age: when we had lack of storage & processing power

- Implications for computer vision algorithms:
  - trade-off between amount of data that can be processed and algorithm complexity
  - better to make a **fast & "imprecise"** algorithm that can go through a **lot of data**, instead of a **slow & "precise"** algorithm
  - design algorithms **from the start** to be **scalable**: **parallelisable** and able to process **chunks of data** at a time

[32] V. Mayer-Schönberger et al.: *Big Data*. John Murray Publishers, 2013.

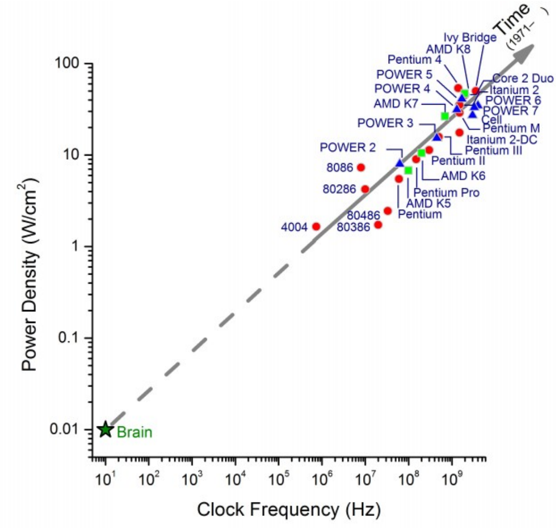
- Algorithms are currently implemented to run on CPUs:
  - stem from **Von Neumann architecture** (1945)
  - read instruction, read data, process data, store data, ...
  - good for fast processing of spreadsheets
  - **inefficient** for computer vision: slow and uses **lots of energy**

- Organic brain:
  - NOT Von Neumann architecture
  - data is encoded and processed in terms of **spikes** (eg. rate of spikes)
  - massively parallel execution
  - easily deals with incomplete data
  - **energy efficient**

CPU $\longrightarrow$

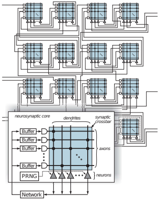
vs

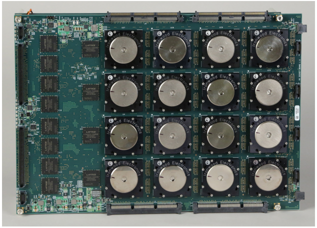
Brain $\longrightarrow$



$\therefore$ CPUs use lots of energy, get hot, and don't accomplish much ...

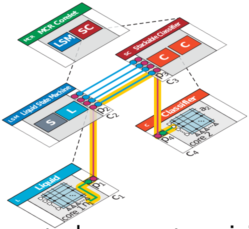
- TrueNorth: new computer architecture from IBM Research[33]:
  - rough approximation of the organic brain
  - NOT simply a hardware implementation of ANNs
  - implements interconnected modules of spiking neurons
  - implemented using existing CMOS hardware building blocks
  - 4096 cores, 1 million neurons, 5.4 billion transistors
  - each core has memory ("synapses"), processors ("neurons"), and communication ("axons")





[33] Andrew S. Cassidy et al.: *Cognitive computing building block: A versatile and efficient digital neuron model for neurosynaptic cores.* In: *IJCNN* (2013).

- Fundamentally different programming approach:
  - composing networks of neurosynaptic cores



- Successfully implemented computer vision algorithms[34]:
  - digit recognition, collision avoidance, optical flow, eye detection, ...
  - 400 billion synaptic operations per second (SOPS) per watt
  - most efficient supercomputer: 4.5 billion FLOPS per watt
  - uses **less energy**: 176,000 times more efficient than a modern CPU running the same brain-like workload

[34] Steve K. Esser et al.: *Cognitive computing systems: Algorithms and applications for networks of neurosynaptic cores*. In: *IJCNN* (2013).

- **Implications:**
  - a paradigm shift is on the horizon
  - nature of computer vision research will need to adapt to make use of the new architecture

- **How deep does the rabbit hole go?**
  - the organic brain already contains excellent vision algorithms, thanks to a few billion years of evolution
  - is the code used by the organic brain **similar** to the code used by TrueNorth?
  - if so, can we **reverse engineer** the pre-existing algorithms in the brain?
  - **re-implement** the reverse engineered algorithms on the TrueNorth architecture?

- **Questions? Comments?**
  e-mail: conradsand [at] ieee [dot] org

- **More papers on computer vision & machine learning:**
  http://conradsanderson.id.au/papers.html