Proposed Method Based on Pose Estimation for Complex Activities

Imran Mumtaz, Jiancheng
School of Computer Science, Sichuan University Chengdu, R.P China

Abstract
in this paper we propose a new algorithm for complex activities our approach accomplished all the requirements of GPLVM. It is suitable for both learning and inference of probabilistic models. The experimental performance has been measured on the basis of Benchmark data. Our propose methodology effectively learn latent spaces of complex multi activities data sets in a computational efficient manner. In addition we also introduce a new procedure for learning latent spaces incrementally. Our proposed model cannot tackle new training set without relearning. Our new method is very flexible and easily applied online settings without extensively repeating relearning method. It has positive impact in applications such as robotics, where domain adaptation plays a vital role for accurate prediction features.

Index Terms
Pose estimation, Human Motion, Complex Activities, Dynamic Motion

1. Introduction

Tracking human 3D articulated motions from video sequences is well known to be a challenging machine vision problem. Estimating the human body's 3D location and orientation of the joints is notoriously difficult because it is a high-dimensional problem and is riddled with ambiguities coming from noise, monocular imagery and occlusions. To reduce the complexity of the task, it has become very popular to use prior models of human pose and dynamics [1, 2, 3, 4, 5, 6, 7].

Linear models (e.g. PCA) are among the simplest priors [8, 9, 10], though linearity also restricts a model's expressiveness and results in inaccuracies when learning complex motions. Priors generated from non-linear dimensionality reduction techniques such as Isomap [11] and LLE [12] have also been used for tracking [13, 14].

These techniques try to preserve the local structure of the manifold but tend to fail when manifold assumptions are violated, e.g., in the presence of noise, or multiple activities. Moreover, LLE and Isomap provide neither a probability distribution over the space of possible poses nor a mapping from the latent space to the high dimensional space. While such a distribution and or mapping can be learned post hoc, learning them separately from the latent space typically results in suboptimal solutions. Probabilistic latent variable models (e.g. probabilistic PCA), have the advantage of taking uncertainties into account when learning latent representations. Taylor et al. [15] introduced the use of Conditional Restricted Boltzmann Machines (CRBM) and implicit mixtures of CRBM (imCRBM), which are composed of large collections of discrete latent variables. Unfortunately, learning this type of model is a highly complex task.

A more commonly used latent variable model is the Gaussian Process Latent Variable Model (GPLVM) [16] which has been applied to animation [18] and tracking [19, 20, 21, 22]. While the GPLVM is very successful at modeling small training sets with single activities, it often struggles to learn latent spaces from larger datasets, especially those with multiple activities.

The main reason is that the GPLVM is a non-parametric model; learning requires the optimization of a non-convex function, for which complexity grows with the number of training samples. As such, having a good initialization is key for success [23], though good initializations are not always available [24], especially with complex data. Additionally, GPLVM learning scales cubicly with the number of training examples, and application to large datasets is computationally intractable, making it necessary to use sparsification techniques to approximate learning [25, 26].

As a consequence, the GPLVM has been mainly applied to single activities, e.g., walking or running.

2. Related work

More recent works have focused on handling multiple activities, most often with mixture models [27, 28, 29] or switching models [30, 31, 32]. However, coordinating the different components of the mixture models requires special care to ensure that they are aligned in the latent space [33], thereby complicating the learning process.

In addition, both mixture and switching models require a discrete notion of activity which is not always available, e.g. dancing motions are not a discrete set. Others have tried to couple discriminate action classifiers with action-specific models [34, 35], though accuracy of such systems does not scale well with the number of actions.

A good prior model for tracking should be accurate, expressive enough to capture a wide range of human poses, and easy and tractable for both learning and inference.

Manuscript received March 5, 2016
Manuscript revised March 20, 2016

Unfortunately, none of the aforementioned approaches exhibit all of these properties.

In this paper, we are interested in learning a probabilistic model that fulfill all of these criteria. Towards this end, we propose a stochastic gradient descent algorithm for the GPLVM which can learn latent spaces from random initializations. We draw inspiration for our work from two main sources.

The first, [36], approximates Gaussian process regression for large training sets by doing online predictions based on local neighborhood. The second, [37], maximizes the likelihood function for GPLVM by considering one dimension of the gradient at a time in the context of collaborative filtering. Based on these two works, we propose a similar strategy to approximate the gradient computation within each step of the stochastic gradient descent algorithm.

Local estimation of the gradients allows our approach to efficiently learn models from large and complex training sets while mitigating the problem of local minima. Furthermore, we propose an online algorithm that can effectively learn latent spaces incrementally without extensive relearning. We demonstrate the effectiveness of our approach on the task of monocular and multi-view tracking and show that our approach outperforms the state-of-the-art on the standard benchmark HumanEva [38].

3. Overview of methodology

We first review the GPLVM, the basis of our work, and then introduce our optimization method for learning with stochastic local updates. Finally, we derive an extension of the algorithm which can be applied to the online setting.

The rest of paper we describe this article in such way. Pose Estimation of Complex Activities, we take an aside to discuss an alternative method for learning low-dimensional embedding's. Learned low-dimensional manifolds are commonly used to simplify the pose-estimation problem but can be poor at generalization; models which are more expressive are more difficult or inefficient to learn. In this chapter, we explored an alternative method for embedding poses in a low-dimensional manifold to serve as a pose prior. Existing approaches for establishing pose priors tend to be either too simplistic (linear), too complex to learn, or can only learn latent spaces from "simple data", i.e. single activities such as walking or running. We presented an efficient stochastic gradient descent algorithm for learning probabilistic non-linear latent spaces composed of multiple complex activities. We further extend this method and derive an incremental learning algorithm for an online setting which can update the latent space without extensive relearning

4. Introduction to GPLVM (Gaussian process latent variable model)

The GPLVM assumes that the observed data has been generated by some unobserved latent random variables. More formally, let $\mathbf{Y} = [\mathbf{y}_1, \cdots, \mathbf{y}_N]^T$ be the set of observations $\mathbf{y}_i \in \Re^D$ and $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]^T$ be the set of latent variables $\in \Re^Q, Q \ll D.$. The GPLVM relates the latent variables and the observations via the probabilistic mapping $y^{(d)} = f(\mathbf{x}) + \eta$, with η being i.i.d. Gaussian noise, and y(d) the d-th coordinate of the observations. In particular, the GPLVM places a Gaussian process prior over the mapping f such that marginalization of the mapping can be done in closed form. The resulting conditional distribution becomes

$$p(\mathbf{Y}|\mathbf{X}, \beta) = \frac{1}{\sqrt{(2\pi)^{N.D}|\boldsymbol{K}|^D}} \exp\left(-\frac{1}{2} tr\,(\boldsymbol{K}^{-1}\mathbf{Y}\mathbf{Y}^T)\right) \quad (1)$$

Where K is the kernel matrix with elements $\mathbf{K}_{ij} = \mathrm{K}(\mathbf{x}_i, \mathbf{x}_j)$ and the kernel k has parameters β. Here, we follow existing approaches [131] and use a kernel compounded from an RBF, a bias, and Gaussian noise, i.e.,

$$\mathrm{K}(\mathrm{x}, \mathrm{x}') = \beta_1 \exp\left(-\frac{\|\mathrm{x} - \mathrm{x}'\|^2}{\beta_2}\right) + \beta_3 + \frac{\delta_{x,x'}}{\beta_4}$$

The GPLVM is usually learned by maximum likelihood estimation of the latent coordinates X and the kernel hyper parameters $\beta = \{\beta_1, \cdots, \beta_4\}$. This is equivalent to minimizing the negative log likelihood $\mathcal{L}$:

$$\mathcal{L} = -\ln p(\mathbf{Y}|\mathbf{X}, \beta) = -\frac{DN}{2}\ln 2\pi - \frac{D}{2}\ln|\mathbf{K}| - \frac{1}{2} tr\,(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) \quad (2)$$

Typically a gradient descent algorithm is used for the minimization. The gradient of $\mathcal{L}$ with respect to X can be obtained via the chain rule, where

$$\frac{\partial \mathcal{L}}{\partial \mathbf{X}} = \frac{\partial \mathcal{L}}{\partial \mathbf{K}} \cdot \frac{\partial \mathbf{K}}{\partial \mathrm{X}} = -(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T\mathbf{K}^{-1} - D\mathbf{K}^{-1}) \cdot \frac{\partial \mathbf{K}}{\boldsymbol{\alpha}\mathbf{X}} \quad (3)$$

Similarly, the gradient of $\mathcal{L}$ with respect to β can be found by substituting $\frac{\partial \boldsymbol{K}}{\partial \boldsymbol{X}}$ with $\frac{\partial \mathrm{K}}{\partial \boldsymbol{\beta}}$ in Eq. (3) (see [70] for the exact derivation). As N gets large, however, computing the gradients becomes computationally expensive, because inverting K is of O(N3), with N the number of training examples. More importantly, as the negative log likelihood $\mathcal{L}$ is highly non-convex, especially with respect to X, standard gradient descent approaches tend to get stuck in local minima, and rely on having good initializations for success..

We now demonstrate how a stochastic gradient descent approach can be used to reduce computational complexity as well as decrease the chances of getting trapped in local minima. In particular, as shown in our experiments (Section 3.3), we are able to obtain smooth and accurate manifolds (see Figure 3.1) from random initialization.

A. Stochastic Gradient Descent

In standard gradient descent, all points are taken into account at the same time when computing the gradient; stochastic gradient descent approaches, on the other hand, approximate the gradient at each point individually. Typically, a loop goes over the points in a series or by randomly sampling from the training set. Note that after iterating overall the points, the gradient is exact. As the GPLVM is a non-parametric approach, the gradient computation at each point does not decompose, making it necessary to invert K, an $O(N^3)$ operation at every iteration. We propose, however, to approximate the gradient computation within each step of the stochastic gradient descent algorithm. Therefore, the gradient of $\mathcal{L}$ can be estimated locally for some neighbourhood of points $\mathbf{X}_R$, centered at a reference point $\mathbf{x}_r$, rather than over all of X. Eq. (3.3) can then be evaluated only for the points within the neighborhood, i.e.,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{X}_R} \approx -(\mathbf{K}_R^{-1}\mathbf{Y}_R\mathbf{Y}_R^T\mathbf{K}_R^{-1} - \mathbf{D}\mathbf{K}_R^{-1}) \cdot \frac{\partial \mathbf{K}_R}{\partial \mathbf{X}_R}, \quad (4)$$

Where $\mathbf{K}_R$ is the kernel matrix for $\mathbf{X}_R$ and $\mathbf{Y}_R$ is the corresponding neighbourhood data points.

We employ a random strategy for choosing the reference point $\mathbf{x}_r$. The neighbourhood $\mathcal{R}$ can be determined by any type of distance measure, such as Euclidean distance in the latent space and/or data space, or temporal neighbours when working with time series. More critical than the specific type of distance measure, however, is allowing sufficient coverage of the latent space so that each neighbourhood is not restricted too locally. To keep the complexity low, it is beneficial to sample randomly from a larger set of neighbors (see Section 3.3.1).

The use of stochastic gradient descent has several desirable traits that correct for the aforementioned drawbacks of GPLVMs. First, computational complexity is greatly reduced, making it feasible to learn latent spaces with much larger amounts of data. Secondly, estimating the gradients stochastically and locally improves robustness of the learning process against local minima, making it possible to have a random initialization. An algorithmic summary of stochastic gradient descent learning for GPLVMs is given in Algorithm 1.

Algorithm 1 Stochastic GPLVM

Randomly initialize X
Set β with an initial guess
for $t = 1$ to T do
  Randomly selecte $\mathbf{x}_r$
  find R neighbours around $\mathbf{x}_r$: $\mathbf{X}_R = \mathbf{X} \in \mathcal{R}$
  Compute $\frac{\partial L}{\partial \mathbf{X}_R}$ and $\frac{\partial L}{\partial \boldsymbol{\beta}_R}$ (see Eq. (4.3))
  Update X and β:
    $\Delta \mathbf{X}_t = \boldsymbol{\mu}\mathbf{x} \cdot \Delta \mathbf{X}_{t-1} + \boldsymbol{\eta}\mathbf{x} \cdot \frac{\partial L}{\partial \mathbf{X}_R}$
    $\mathbf{X}_t \leftarrow \mathbf{X}_{t-1} + \Delta \mathbf{X}_t$
    $\Delta \boldsymbol{\beta}_t = \boldsymbol{\mu}_\beta \cdot \Delta \boldsymbol{\beta}_{t-1} + \boldsymbol{\eta}_\beta \cdot \frac{\partial L}{\partial \boldsymbol{\beta}_R}$
    $\boldsymbol{\beta}_t \leftarrow \boldsymbol{\beta}_{t-1} + \Delta \boldsymbol{\beta}_t$
end for

Algorithm 3-1: Algorithm 1 Stochastic GPLVM

B. Incremental Learning

In this section, we derive an incremental learning algorithm based on the stochastic gradient descent approach of the previous section. In this setting, we have an initial model which we would like to update as new data comes in on the fly. More formally, let $\mathbf{Y}_{orig}$ be the initial training data, and $\mathbf{X}_{orig}$ and $\boldsymbol{\beta}_{orig}$ be a model learned from $\mathbf{Y}_{orig}$ using stochastic GPLVM. For every step in the online learning, let Yincr be new data, which can be as little as a single point or an entire set of training points.

Let $\mathbf{Y} = [\mathbf{Y}_{orig}, \mathbf{Y}_{incr}] \in \mathbb{R}^{(N+M)\times D}$ be the set of training points containing both the already trained data $\mathbf{Y}_{orig}$, and the new incoming data $\mathbf{Y}_{incr}$, and let $\mathbf{X} = [\mathbf{X}_{orig}, \mathbf{X}_{incr}] \in \mathbb{R}^{(N+M)\times Q}$ be the corresponding latent coordinates, where M is the number of newly added training examples. Let $\widehat{\mathbf{X}}_{orig}$ be the estimate of the latent coordinates that has already been learned.

A possible strategy is to update only the incoming points; however, we would like to exploit the new data for improving the estimate of the entire manifold, therefore we propose to learn the full X. To prevent the already-learned manifold from diverging and also to speed up learning, we add a regularizer to the log-likelihood to encourage original points to not deviate too far from their initial estimate. To this end, we use the Frobenius norm of the deviation from the estimate $\widehat{\mathbf{X}}_{orig}$. Learning is then done by minimizing the regularized negative log-likelihood

$$\mathcal{L}_{incr} = \mathcal{L} + \lambda \cdot \frac{1}{N} \left\| \boldsymbol{X}_{1:N,:} - \hat{\boldsymbol{X}}_{orig} \right\|_F^2 \quad (5)$$

Here, X1:N,: indicates the first N rows of X, while λ is a weighting on the regularization term. The gradient of $\mathcal{L}$ with respect to $\mathbf{X}_R$ can then be computed as

$$\frac{\partial \mathcal{L}_{incr}}{\partial \mathbf{X}_R} = \frac{\partial \mathcal{L}}{\partial \mathbf{X}_R} + \lambda \cdot \frac{2}{N} \cdot (X_{1:N,:} - \hat{X}_{orig}) \frac{\partial \mathbf{X}_{1:N,:}}{\partial \mathbf{X}_R} \quad (6)$$

We employ a stochastic gradient descent approach for our incremental learning, where the points are sampled randomly from Xincr. Note that while xr is only sampled from Xincr in the subsequent learning step, this does not exclude points in Xorig from being a part of the neighbourhood $\mathcal{R}$, and thus from being updated. We have chosen a nearest neighbor approach by comparing Yincr to Yorig for estimating an initial Xincr, though other possibilities include performing a grid search in the latent space and selecting locations with the highest global log-likelihood (Eq. (3.2)) or training a regressor from Yorig to Xorig to be applied to Yincr. An algorithmic summary of the incremental method is provided in Algorithm 2

Algorithm 2 Incremental Stochastic GPLVM

for $t = 1$ **to** T_1 **do**
  Learn $\mathbf{X}_{orig}$ *and* β_{orig} **as per Algorithm 1.**
end for
Initialize $\mathbf{X}_{incr}$ **using nearest neighbours.**
Set $\beta = \beta_{orig}$
Group data:
  $\mathbf{Y} = [\mathbf{Y}_{orig}, \mathbf{Y}_{incr}]$
  $\mathbf{X} = [\mathbf{X}_{orig}, \mathbf{X}_{incr}]$
for $t = T_1$ **to** T_2 **do**
  randomly select $\mathbf{x}_r \in \mathbf{X}_{incr}$
  find R **neighbours around** $\mathbf{x}_r : \mathbf{X}_R = \mathbf{X} \in \mathcal{R}$
  Compute $\frac{\partial L_{incr}}{\partial X_R}$ *and* $\frac{\partial L_{incr}}{\partial \beta_R}$ **(see Eq. (3.6))**
  Update X and β:
    $\Delta \mathbf{X}_t = \mu \mathbf{x} \,.\, \Delta \mathbf{X}_{t-1} + \eta \mathbf{x} . \frac{\partial L_{incr}}{\partial \mathbf{X}_R}$
    $\mathbf{X}_t \leftarrow \mathbf{X}_{t-1} + \Delta \mathbf{X}_t$
    $\Delta \boldsymbol{\beta}_t = \mu \boldsymbol{\beta} \,.\, \Delta \boldsymbol{\beta}_{t-1} + \eta_{\beta} . \frac{\partial L_{incr}}{\partial \beta_R}$
    $\boldsymbol{\beta}_t \leftarrow \boldsymbol{\beta}_{t-1} + \Delta \boldsymbol{\beta}_t$
  end for

Algorithm 3-2: Algorithm 2 Incremental Stochastic GPLVM

the highest global log-likelihood (Eq. (3.2)) or training a regressor from $\mathbf{Y}_{orig}$ to $\mathbf{X}_{\mathbf{orig}}$ to be applied to $\mathbf{Y}_{incr}$. An algorithmic summary of the incremental method is provided in Algorithm 2

A. ***Tracking Framework***

During training, a latent variable model $\mathcal{M}$ is learned from $\mathbf{Y}_M$, where $\mathbf{Y}_M$ are relative joint locations with respect to a root node. We designate the learned latent points as $\mathbf{X}_M$. During inference, tracking is performed in the latent space using a particle filter. The corresponding pose is computed by projecting back to the data space via the Gaussian process mapping learned in the GPLVM.

We model the state $\mathbf{s}$ at time t as $\mathbf{s}_t = (\mathbf{x}_t, \mathbf{g}_t, \mathbf{r}_t)$ where $\mathbf{x}_t$ denotes position in the latent space, while $\mathbf{g}_t$ and $\mathbf{r}_t$ are the global position and rotation of the root node.

Particles are initialized in the latent space by a nearest neighbour search between the observed 2D image pose in the first frame of the sequence and the projected 2D poses of $\mathbf{Y}_M$. Particles are then propagated from frame to frame using a first-order Markov model

$$\mathbf{x}_t^i = \mathbf{x}_{t-1}^i + \mathbf{x}_t^i, \qquad \mathbf{g}_t^i = \mathbf{g}_{t-1}^i + \mathbf{g}_t^i, \qquad \mathbf{r}_t^i = \mathbf{r}_{t-1}^i + \mathbf{r}_t^i \quad (7)$$

We approximate the derivative xi with the difference between temporally sequential points of the nearest neighbors in $\mathbf{X}_M$, while gi and ri are drawn from individual Gaussians with means and standard deviations estimated from the training data. The tracked latent position ^xt at time t is then approximated as the mode over all particles in the latent space while ^yt is estimated via the mean Gaussian process estimate

$$\hat{\mathbf{y}}_t = \mu_M + \mathbf{Y}_M^T \mathbf{K}^{-1} \mathrm{k}(\hat{\mathbf{x}}_t, \mathbf{X}_M) \quad (8)$$

With μ_M the mean of $\mathbf{Y}_M$ and $\mathrm{k}(\hat{\mathbf{x}}_t, \mathbf{X}_M)$ the vector with elements $\mathrm{k}(\hat{\mathbf{x}}_t, \mathbf{x}_m)$ for all $\mathbf{x}_m$ in $\mathbf{X}_M$. Note that the computation of $\mathbf{K}^{-1}$ needs to be performed only once and can be stored.

5. Experimental results

We demonstrate the effectiveness of our model when applied to tracking in both monocular and multi-view scenarios. In all cases, the latent models were learned with $\mu_x = 0.8\,,\ \mu_\beta = 0.5\,, \eta_x = 10e-4\,, \eta_\beta = 10e-8;$ we annealed these parameters over the iterations. To further smooth the learned models, we incorporate a Gaussian Process prior over the dynamics of the training data in the latent space [138] for the GPLVM and the stochastic GPLVM.

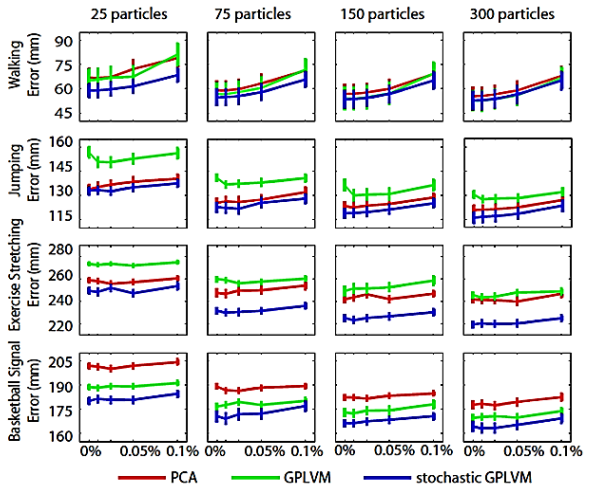
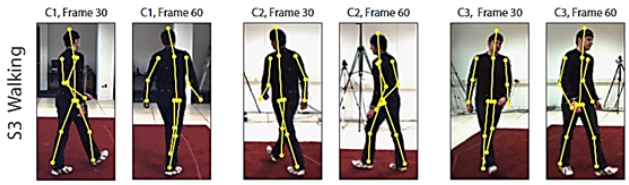


Fig 1: Cross-subject 3D tracking errors for each type of activity sequence with respect to amount of additive noise for different number of particles, where error bars represent the standard deviation from repetitions runs.



Train	Test	GPLVM	CRBM	imCRBM	Ours
S3	S3	71.6 ± 10.0	49.8 ± 2.2	51.4 ± 0.9	$\mathbf{45.4 \pm 1.1}$
S1,2,3	S3	123.8 ± 16.7	70.9 ± 2.1	$\mathbf{43.4 \pm 4.1}$	46.5 ± 1.4

Figure 2: Experimental result of complex activities and comparison of 3D tracking errors.

6. Conclusion & further extension

in this article we provide a GPLVM model which is accurate for learning and inference it based on latent spaces of complex activities, data set and easily computational efficient. We implement our model on humanEva benchmark. Further we use learning latent spaces incrementally they cannot handel new training samples without re-learning. Our proposed method can easily applicable online setting without extensive relearning. In future we further investigate the incorporation of dynamics into the stochastic model specific for multiple activities and changing the paramenters of mathematics formulas to improve the efficiency and accuracy of our model.

Acknowledgment

Author is greatful for Prof. Jiancheng Lv. For his contribution and review able commit make the words more valuable.

References

[1] Sidenbladh, H., Black, M., & Fleet, D. (2000). Stochastic tracking of 3D human figures using 2D image motion. In Proceedings European Conference on Computer Vision (ECCV).

[2] Urtasun, R., Fleet, D., & Fua, P. (2006). 3D people tracking with gaussian process dynamical models. In Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[3] Wang, J., Fleet, D., & Hertzmann, A. (2008). Gaussian process dynamical models for human motion. PAMI, 30(2), 283–298.

[4] Xu, X., & Li, B. (2007). Learning motion correlation for tracking articulated human body with a rao-blackwellised particle filter. In Proceedings International Conference on Computer Vision (ICCV).

[5] J¨aggli, T., Koller-Meier, E., & Van Gool, L. (2009). Learning generative models for multi activity body pose estimation. International Journal of Computer Vision (IJCV), 83(2), 121–134.

[6] Li, R., Tian, T.-P., Sclaroff, S., & Yang, M.-H. (2010). 3D human motion tracking with a coordinated mixture of factor analyzers. International Journal of Computer Vision (IJCV), 87, 170–190. 2.7,

[7] Taylor, G., Sigal, L., Fleet, D., & Hinton, G. (2010). Dynamical binary latent variable models for 3D human pose tracking. In Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[8] Ormoneit, D., Lemieux, C., & Fleet, D. (2001). Lattice particle filters. In Proceedings Conference on Uncertainty in Artificial Intelligence.

[9] Urtasun, R., Fleet, D., Hertzman, A., & Fua, P. (2005). Priors for people tracking from small training sets. In Proceedings International Conference on Computer Vision (ICCV).

[10] Tenenbaum, J., de Silva, V., & Langford, J. (2000). A Global Geometric Framework for Nonlinear Dimensionality Reduction. Science.

[11] Roweis, S., & Saul, L. (2000). Nonlinear dimensionality reduction by locally linear embedding. Science, 290(5500), 2323–232.

[12] Gall, J., Yao, A., & Van Gool, L. (2010b). 2D action recognition serves 3D human pose estimation. In Proceedings European Conference on Computer Vision (ECCV).

[13] Taylor, G., Sigal, L., Fleet, D., & Hinton, G. (2010). Dynamical binary latent variable models for 3D human pose tracking. In Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR).Geiger, A., Urtasun, R., & Darrell, T. (2009). Rank priors for continuous non-linear dimensionality reduction. In Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[14] Lawrence, N. (2005). Probabilistic non-linear principal component analysis with Gaussian process latent variable

models. Journal of Machine Learning Research, 6, 1783–1816.
[15] Quinonero-Candela, J., & Rasmussen, C. (2006). A unifying view of sparse approximate gaussian process regression. Journal of Machine Learning Research, (p. 2005).
[16] Lawrence, N. (2007). Learning for larger datasets with the gaussian process latent variable model. In International Conference on Artificial Intelligence and Statistics (AISTATS).
[17] Lin, R., Liu, C., Yang, M., Ahja, N., & Levinson, S. (2006). Learning nonlinear manifolds from time series. In Proceedings European Conference on Computer Vision (ECCV).
[18] Choi,W., Shahid, K., & Savarese, S. (2009). What are they doing? collective activity classification using spatio-temporal relationship among people. In International Workshop on Visual Surveillance (VSWS).
[19] Roweis, S., Saul, L., & Hinton, G. (2002). Global coordination of local linear models. In Neural Information Processing Systems (NIPS).
[20] Gall, J., Yao, A., & Van Gool, L. (2010b). 2D action recognition serves 3D human pose estimation. In Proceedings European Conference on Computer Vision (ECCV).
[21] Urtasun, R., & Darrell, T. (2008). Sparse probabilistic regression for activity-independent human pose inference. In Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
[22] Sigal, L., Balan, A., & Black, M. (2010). Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. International Journal of Computer Vision (IJCV), 87(1-2), 4–27.