All you want to know about GPs: Applications and Extensions of GPLVM

Raquel Urtasun and Neil Lawrence

TTI Chicago, University of Sheffield

June 16, 2012

Applications of GPLVM

We will concentrate on a few successful applications in computer vision

- Pose priors for character animation
- Pose priors for human pose estimation and tracking
- Deformation priors for shape estimation
- Shape priors for Segmentation

GPLVM for Character Animation

- Learn a GPLVM from a small mocap sequence
- Pose synthesis by solving an optimization problem

$$\operatorname*{argmin}_{\mathbf{x},\mathbf{y}} -\log p(\mathbf{y}|\mathbf{x})$$
$$\text{such that} \quad C(\mathbf{y}) = 0$$

- These handle constraints may come from a user in an interactive session, or from a mocap system.

GPLVM for Character Animation

- Learn a GPLVM from a small mocap sequence
- Pose synthesis by solving an optimization problem

$$\operatorname*{argmin}_{\mathbf{x},\mathbf{y}} -\log p(\mathbf{y}|\mathbf{x})$$
$$\text{such that} \quad C(\mathbf{y}) = 0$$

- These handle constraints may come from a user in an interactive session, or from a mocap system.
- Smooth the latent space by adding noise in order to reduce the number of local minima.
- Optimization in an annealed fashion over different anneal version of the latent space.

GPLVM for Character Animation

- Learn a GPLVM from a small mocap sequence
- Pose synthesis by solving an optimization problem

$$\underset{\mathbf{x},\mathbf{y}}{\operatorname{argmin}} - \log p(\mathbf{y}|\mathbf{x})$$

$$\text{such that} \quad C(\mathbf{y}) = 0$$

- These handle constraints may come from a user in an interactive session, or from a mocap system.
- Smooth the latent space by adding noise in order to reduce the number of local minima.
- Optimization in an annealed fashion over different anneal version of the latent space.

Application: Replay same motion

[K. Grochow, S. Martin, A. Hertzmann and Z. Popovic, Siggraph 2004]

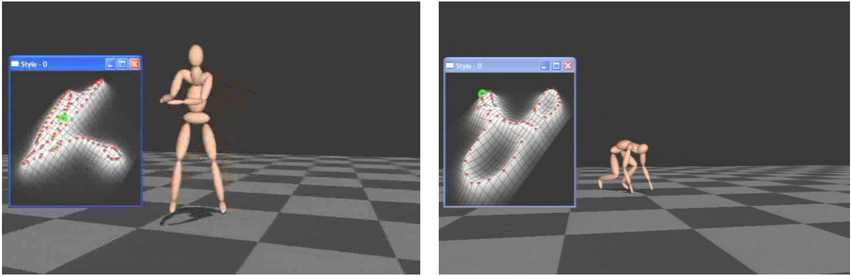


Figure: Syle-IK

Application: Keyframing joint trajectories

[K. Grochow, S. Martin, A. Hertzmann and Z. Popovic, Siggraph 2004]

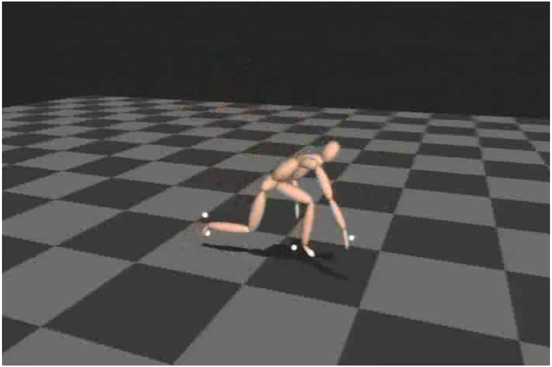


Figure: Syle-IK

Application: Deal with missing data in mocap

[K. Grochow, S. Martin, A. Hertzmann and Z. Popovic, Siggraph 2004]

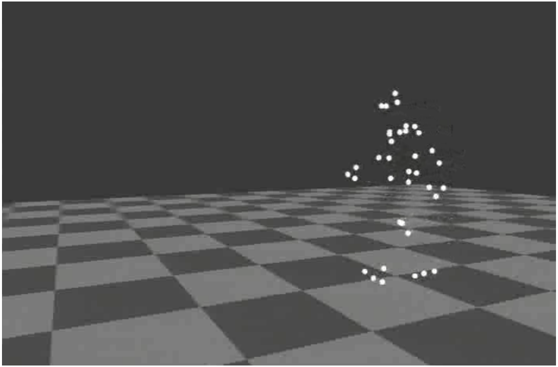


Figure: Syle-IK

Application: Style Interpolation

[K. Grochow, S. Martin, A. Hertzmann and Z. Popovic, Siggraph 2004]

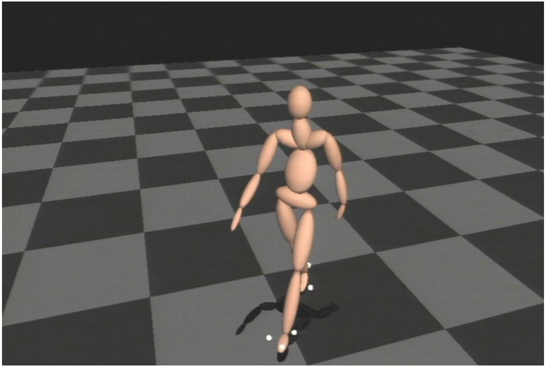


Figure: Syle-IK

Applications: Animation from Images

[K. Grochow, S. Martin, A. Hertzmann and Z. Popovic, Siggraph 2004]

- Requires manual interaction
- Next we will see how to do this automatically with these models

Figure: Syle-IK

The problem of human pose estimation

- The goal is given an image $\mathbf{I}$ to estimate the 3D location and orientation of the body parts $\mathbf{y}$.

Notation

ϕ — the state to be estimated

$\mathbf{I}$ – the image

$\mathbf{x}$ – the latent representation

n — number of training samples

$\mathbf{I}_{t:0}$ — image observations up to time t

$\mathbf{y}_{t:0}$ — poses up to time t

Pose estimation

- **Generative approaches:** focus on modeling

$$p(\phi|\mathbf{I}) = \frac{p(\mathbf{I}|\phi)p(\phi)}{p(\mathbf{I})}$$

- **Discriminative approaches:** focus on modeling directly

$$p(\phi|\mathbf{I})$$

We saw how to directly model $p(\phi, \mathbf{x})$ with a GP before, where $\phi = \mathbf{y}$. Let's now focus on generative approaches.

Generative approaches

Generative approach models

$$p(\phi|\mathbf{I}) = \frac{p(\mathbf{I}|\phi)p(\phi)}{p(\mathbf{I})}$$

Types of generative approaches:

- **Bayesian approaches:** focus on approximating $p(\phi|\mathbf{I})$, usually via sampling (e.g., particle filter).
- **Optimization or energy-based techniques:** focus on computing the MAP or ML estimate of $p(\phi|\mathbf{I})$.

Generative approaches

Generative approach models

$$p(\phi|\mathbf{I}) = \frac{p(\mathbf{I}|\phi)p(\phi)}{p(\mathbf{I})}$$

Types of generative approaches:

- **Bayesian approaches:** focus on approximating $p(\phi|\mathbf{I})$, usually via sampling (e.g., particle filter).
- **Optimization or energy-based techniques:** focus on computing the MAP or ML estimate of $p(\phi|\mathbf{I})$.

Common to all of them is the need to model

- **Image likelihood:** $p(\mathbf{I}|\phi)$
- **Priors:** $p(\phi)$

Generative approaches

Generative approach models

$$p(\phi|\mathbf{I}) = \frac{p(\mathbf{I}|\phi)p(\phi)}{p(\mathbf{I})}$$

Types of generative approaches:

- **Bayesian approaches:** focus on approximating $p(\phi|\mathbf{I})$, usually via sampling (e.g., particle filter).
- **Optimization or energy-based techniques:** focus on computing the MAP or ML estimate of $p(\phi|\mathbf{I})$.

Common to all of them is the need to model

- **Image likelihood:** $p(\mathbf{I}|\phi)$
- **Priors:** $p(\phi)$

In general $p(\mathbf{I})$ is assumed constant and ignored. The different trackers then depend on the different modeling choices and optimization procedures.

Generative approaches

Generative approach models

$$p(\phi|\mathbf{I}) = \frac{p(\mathbf{I}|\phi)p(\phi)}{p(\mathbf{I})}$$

Types of generative approaches:

- **Bayesian approaches:** focus on approximating $p(\phi|\mathbf{I})$, usually via sampling (e.g., particle filter).
- **Optimization or energy-based techniques:** focus on computing the MAP or ML estimate of $p(\phi|\mathbf{I})$.

Common to all of them is the need to model

- **Image likelihood:** $p(\mathbf{I}|\phi)$
- **Priors:** $p(\phi)$

In general $p(\mathbf{I})$ is assumed constant and ignored. The different trackers then depend on the different modeling choices and optimization procedures.

Particle filter revisited

The posterior density is described with three terms

$$p(\phi_t|\mathbf{I}_{t:0}) = \frac{t(\mathbf{I}_t|\phi_t)p(\phi_t|\mathbf{I}_{t-1:0})}{p(\mathbf{I}_t|\mathbf{I}_{t-1:0})}$$

- **Prior:** defines the knowledge of the model

$$p(\phi_t|\mathbf{I}_{t-1:0}) = \int p(\phi_t|\phi_{t-1})p(\phi_{t-1}|\mathbf{I}_{t-1:0})d\phi_{t-1}$$

Particle filter revisited

The posterior density is described with three terms

$$p(\phi_t|\mathbf{I}_{t:0}) = \frac{t(\mathbf{I}_t|\phi_t)p(\phi_t|\mathbf{I}_{t-1:0})}{p(\mathbf{I}_t|\mathbf{I}_{t-1:0})}$$

- **Prior:** defines the knowledge of the model

$$p(\phi_t|\mathbf{I}_{t-1:0}) = \int p(\phi_t|\phi_{t-1})p(\phi_{t-1}|\mathbf{I}_{t-1:0})d\phi_{t-1}$$

- **Likelihood:** $p(\mathbf{I}_t|\phi_t)$ determines the measurement noise model
- **Evidence:** which involves

$$p(\mathbf{I}_n|\mathbf{I}_{t-1:0}) = \int p(\mathbf{I}_t|\phi_t)p(\phi_t|\mathbf{I}_{t-1:0})d\phi_t$$

Particle filter revisited

The posterior density is described with three terms

$$p(\phi_t|\mathbf{I}_{t:0}) = \frac{t(\mathbf{I}_t|\phi_t)p(\phi_t|\mathbf{I}_{t-1:0})}{p(\mathbf{I}_t|\mathbf{I}_{t-1:0})}$$

- **Prior:** defines the knowledge of the model

$$p(\phi_t|\mathbf{I}_{t-1:0}) = \int p(\phi_t|\phi_{t-1})p(\phi_{t-1}|\mathbf{I}_{t-1:0})d\phi_{t-1}$$

- **Likelihood:** $p(\mathbf{I}_t|\phi_t)$ determines the measurement noise model
- **Evidence:** which involves

$$p(\mathbf{I}_n|\mathbf{I}_{t-1:0}) = \int p(\mathbf{I}_t|\phi_t)p(\phi_t|\mathbf{I}_{t-1:0})d\phi_t$$

Optimization techniques

It is defined as minimizing the following programs:

$$\phi^*_{ML} = \underset{\phi}{\operatorname{argmin}} - \log p(\mathbf{I}|\phi)$$

$$\phi^*_{MAP} = \underset{\phi}{\operatorname{argmin}} - \log p(\mathbf{I}|\phi) - \log p(\phi)$$

It suffers from the following problems:

- Local minima: usually $-\log p(\mathbf{I}|\phi)$ is a non-convex function of ϕ.
- Initialization: usually hand initialized or use discriminative approaches.
- Drift: As times goes, the estimate gets worst.
- Difficult to define a good general $-\log p(\mathbf{I}|\phi)$.

Optimization techniques

It is defined as minimizing the following programs:

$$
\begin{aligned}
\phi^*_{ML} &= \underset{\phi}{\operatorname{argmin}} \; -\log p(\mathbf{I}|\phi) \\
\phi^*_{MAP} &= \underset{\phi}{\operatorname{argmin}} \; -\log p(\mathbf{I}|\phi) - \log p(\phi)
\end{aligned}
$$

It suffers from the following problems:

- Local minima: usually $-\log p(\mathbf{I}|\phi)$ is a non-convex function of ϕ.
- Initialization: usually hand initialized or use discriminative approaches.
- Drift: As times goes, the estimate gets worst.
- Difficult to define a good general $-\log p(\mathbf{I}|\phi)$.

Usually computationally more efficient than particle filter (if not use GPU).

This can be combined with particle filter to create hybrid monte-carlo.

Optimization techniques

It is defined as minimizing the following programs:

$$\begin{aligned}
\phi^*_{ML} &= \operatorname*{argmin}_{\phi} -\log p(\mathbf{I}|\phi) \\
\phi^*_{MAP} &= \operatorname*{argmin}_{\phi} -\log p(\mathbf{I}|\phi) - \log p(\phi)
\end{aligned}$$

It suffers from the following problems:

- Local minima: usually $-\log p(\mathbf{I}|\phi)$ is a non-convex function of ϕ.
- Initialization: usually hand initialized or use discriminative approaches.
- Drift: As times goes, the estimate gets worst.
- Difficult to define a good general $-\log p(\mathbf{I}|\phi)$.

Usually computationally more efficient than particle filter (if not use GPU).

This can be combined with particle filter to create hybrid monte-carlo.

GPLVM as a prior for Tracking

Likelihood models: $p(\mathbf{I}|\phi)$

- Monocular tracking: 2D-3D correspondences, silhouettes, edges, template matching, etc.
- Multi-view tracking: stereo, visual hull, etc.

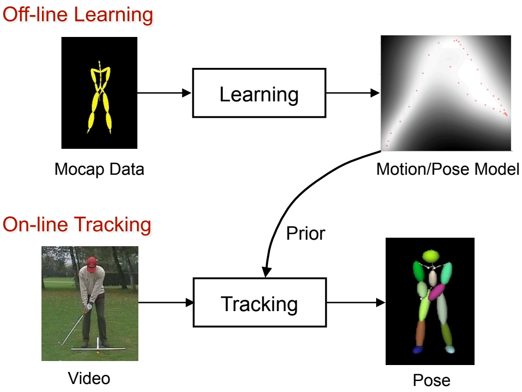
Priors: $p(\phi)$

- Pose priors
- Dynamical priors
- Shape priors

Note that I have defined ϕ as a general quantity, not just the pose, e.g., it includes the latent coordinates.

Generative tracking: Priors for 3D people tracking

- Learn off-line prior models from Mocap: GPLVM
- Use then online to constrain the tracking.



Tracking formulation

- For each image I_t we have to estimate the state $\phi_t = (\mathbf{y}_t, \mathbf{x}_t)$.
- Bayesian formulation of the tracking

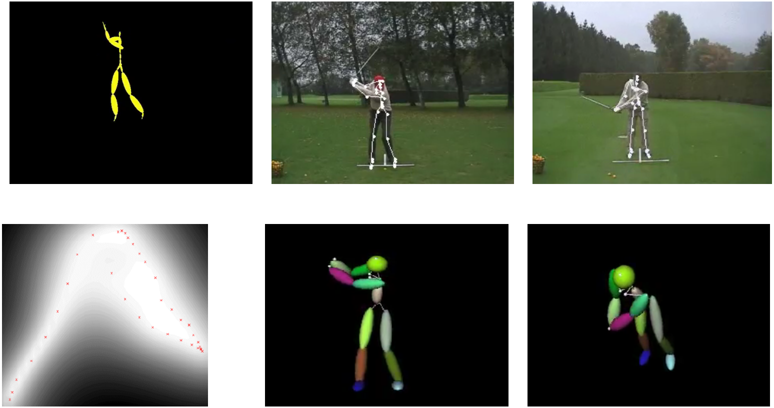
$$p(\phi_{t:t+\tau}|\mathbf{I}_{t:t+\tau}, \mathbf{X}, \mathbf{Y}) \propto \prod_i p(\mathbf{I}_{t+i}|\phi_{t+i}) \prod_i p(\mathbf{y}_{t+i}|\mathbf{x}_{t+i}, \mathbf{X}, \mathbf{Y})$$

- The image likelihood is composed of the distance to 2D joints automatically tracked using WSL (Jepson et al. 03).
- Tracking by minimizing

$$-\log p(\phi_{t:t+\tau}|\mathbf{I}_{t:t+\tau}, \mathbf{X}, \mathbf{Y}) = \mathcal{L}_{images} + \mathcal{L}_{prior}$$

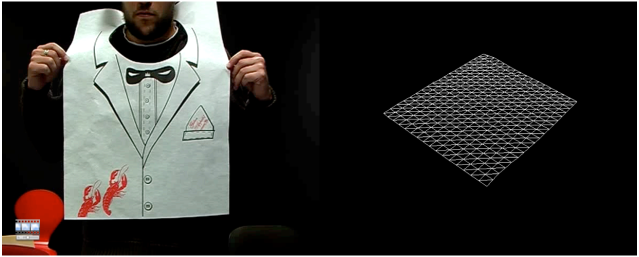
Tracking from a single example!

[R. Urtasun, D. J. Fleet, A. Hertzmann and P. Fua, ICCV 2005]



- Feature or bug?

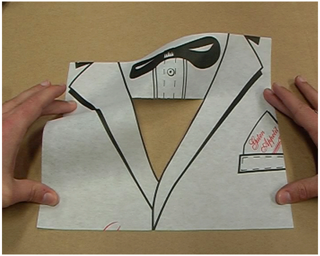
Non-rigid shape deformation



Monocular 3D shape recovery is severely under-constrained:

- Complex deformations and low-texture objects.
- Deformation models are required to disambiguate.
- Building realistic physics-based models is very complex.
- Learning the models is a popular alternative.

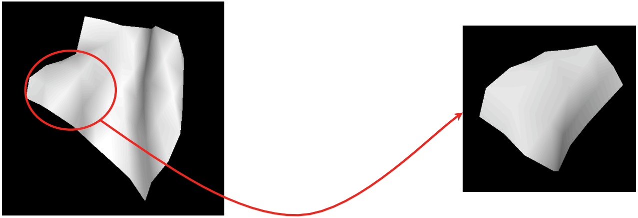
Global deformation models



State-of-the-art techniques learn global models that

- require large amounts of training data,
- must be learned for each new object.

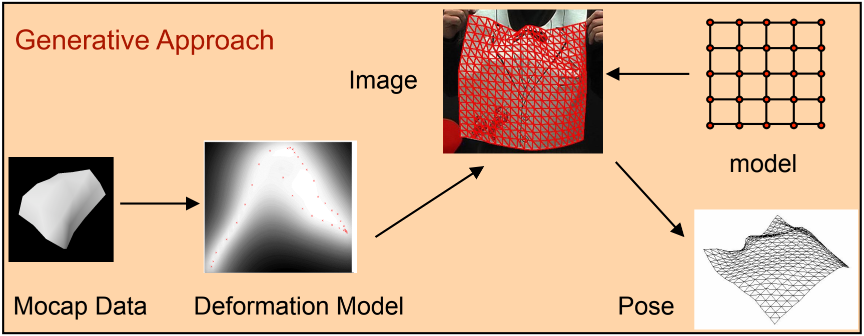
Key observations



1. Locally, all parts of a physically homogeneous surface obey the same deformation rules.
2. Deformations of small patches are much simpler than those of a global surface, and thus can be learned from fewer examples.

→ Learn Local Deformation Models and combine them into a global one representing the particular shape of the object of interest.

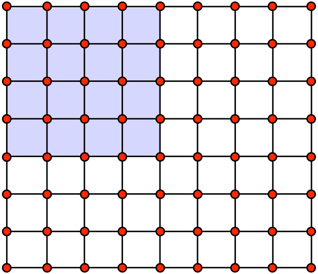
Overview of the method



Combining the deformations

Use a Product of Experts (POE) paradigm (Hinton 99):

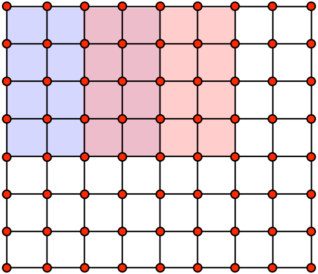
- High dimensional data subject to low dimensional constraints.
- A global deformation should be composed of highly probable local ones.
- For homogeneous materials, all local patches follow the same deformation rules.
- Learn a single local model, and replicate it to cover the whole object.



Combining the deformations

Use a Product of Experts (POE) paradigm (Hinton 99):

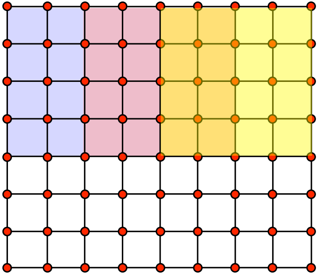
- High dimensional data subject to low dimensional constraints.
- A global deformation should be composed of highly probable local ones.
- For homogeneous materials, all local patches follow the same deformation rules.
- Learn a single local model, and replicate it to cover the whole object.



Combining the deformations

Use a Product of Experts (POE) paradigm (Hinton 99):

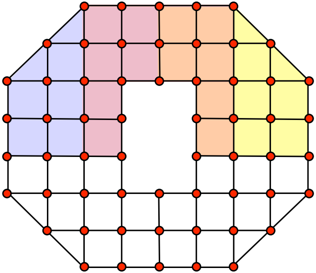
- High dimensional data subject to low dimensional constraints.
- A global deformation should be composed of highly probable local ones.
- For homogeneous materials, all local patches follow the same deformation rules.
- Learn a single local model, and replicate it to cover the whole object.



Combining the deformations

Use a Product of Experts (POE) paradigm (Hinton 99):

- High dimensional data subject to low dimensional constraints.
- A global deformation should be composed of highly probable local ones.
- For homogeneous materials, all local patches follow the same deformation rules.
- Learn a single local model, and replicate it to cover the whole object.



→ Same deformation model represents arbitrary shapes and topologies.

Tracking

- For each image I_t we have to estimate the state $\phi_t = (\mathbf{y}_t, \mathbf{x}_t)$.
- Bayesian formulation of the tracking

$$p(\phi_t | \mathbf{I}_t, \mathbf{X}, \mathbf{Y}) \propto p(\mathbf{I}_t | \phi_t) p(\mathbf{y}_t | \mathbf{x}_t, \mathbf{X}, \mathbf{Y}) p(\mathbf{x}_t)$$

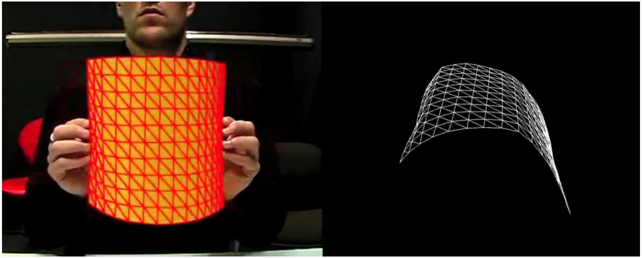
- The image likelihood is composed of texture (template matching) and edge information

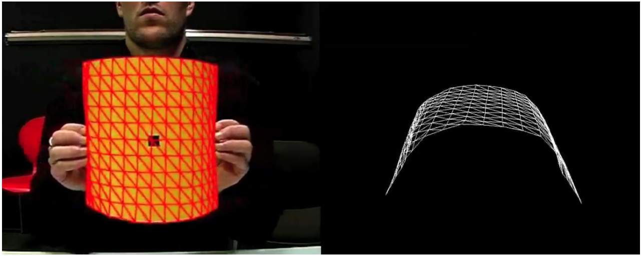
$$p(\mathbf{I}_t | \phi_t) = p(\mathbf{T}_t | \phi_t) p(\mathbf{E}_t | \phi_t)$$

- Tracking by minimizing the posterior

Tracking poorly-textured surfaces

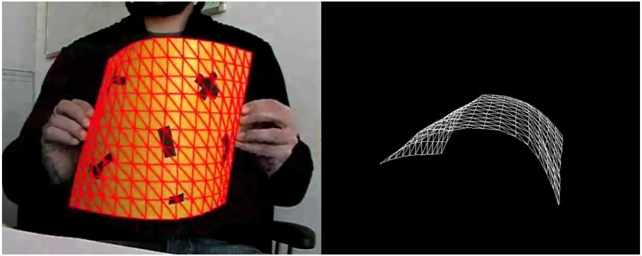
[M. Salzmann, R. Urtasun and P. Fua, CVPR 2008]

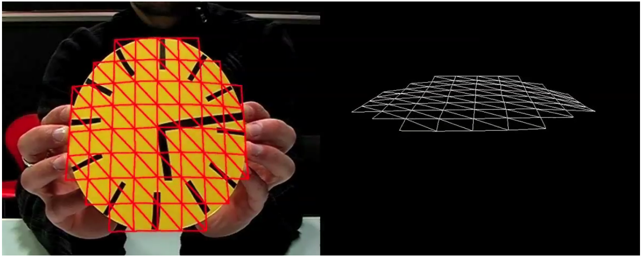




Same prior model for different shapes

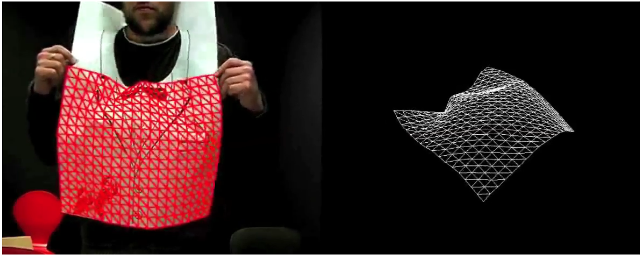
[M. Salzmann, R. Urtasun and P. Fua, CVPR 2008]

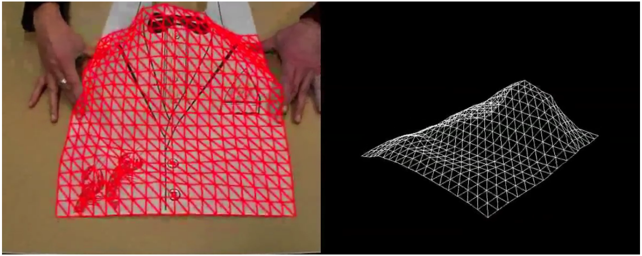




More complex materials

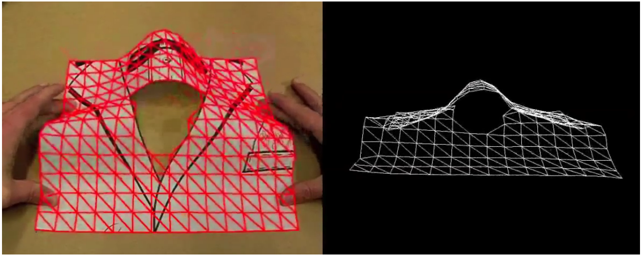
[M. Salzmann, R. Urtasun and P. Fua, CVPR 2008]

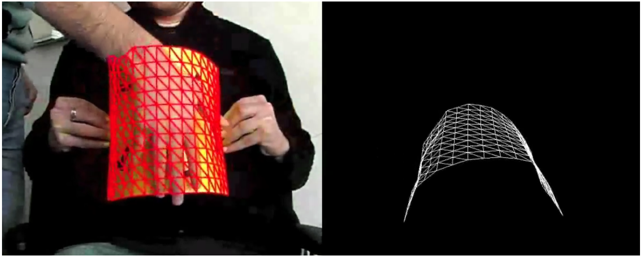




Different topology and Occlusions

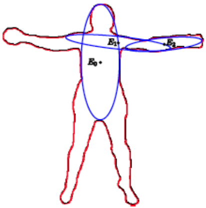
[M. Salzmann, R. Urtasun and P. Fua, CVPR 2008]





Shape Priors in Level Set Segmentation

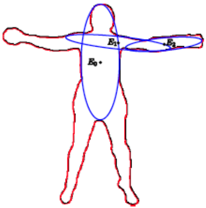
- Represent contours with elliptic Fourier descriptors



- Learn a GPLVM on the parameters of those descriptors

Shape Priors in Level Set Segmentation

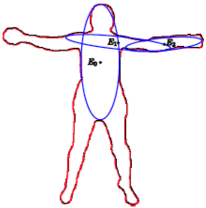
- Represent contours with elliptic Fourier descriptors



- Learn a GPLVM on the parameters of those descriptors
- We can now generate close contours from the latent space

Shape Priors in Level Set Segmentation

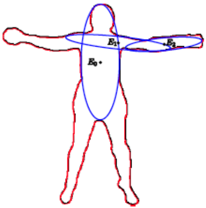
- Represent contours with elliptic Fourier descriptors



- Learn a GPLVM on the parameters of those descriptors
- We can now generate close contours from the latent space
- Segmentation is done by non-linear minimization of an image-driven energy which is a function of the latent space

Shape Priors in Level Set Segmentation

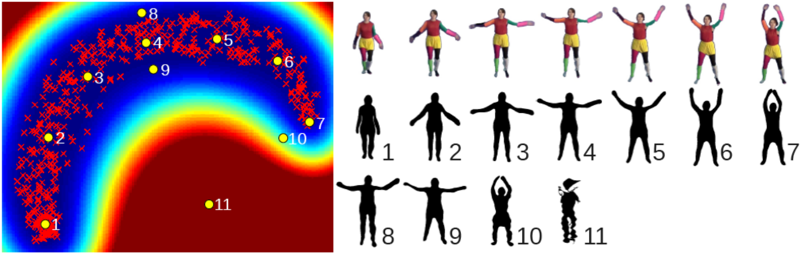
- Represent contours with elliptic Fourier descriptors

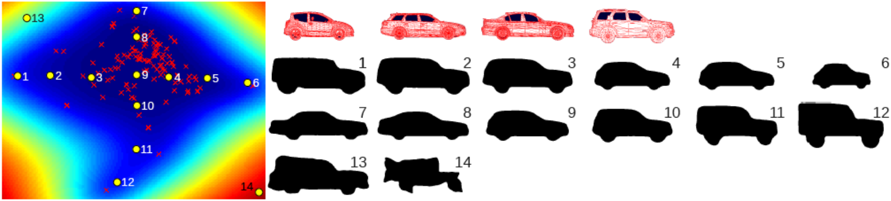


- Learn a GPLVM on the parameters of those descriptors
- We can now generate close contours from the latent space
- Segmentation is done by non-linear minimization of an image-driven energy which is a function of the latent space

GPLVM on Contours

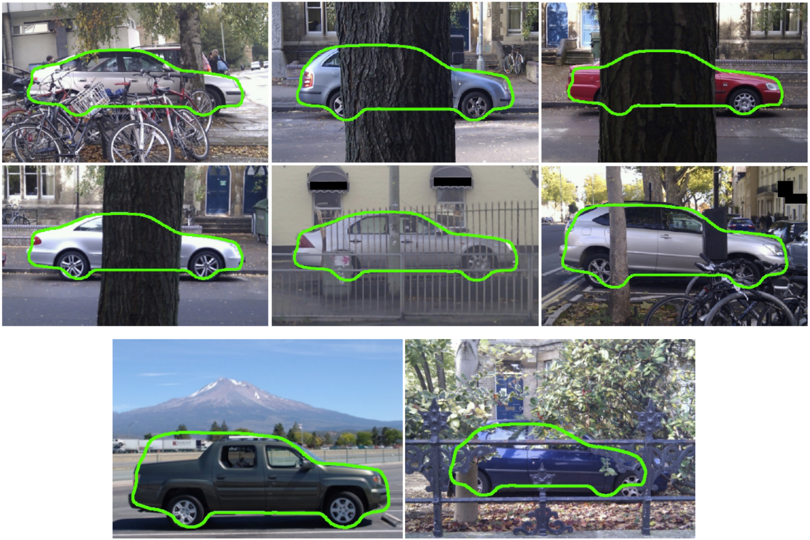
[V. Prisacariu and I. Reid, ICCV 2011]





Segmentation Results

[V. Prisacariu and I. Reid, ICCV 2011]



Does it work all the time?

Is training with so little data a bug or a feature?

Problems with the GPLVM

- It relies on the optimization of a non-convex function

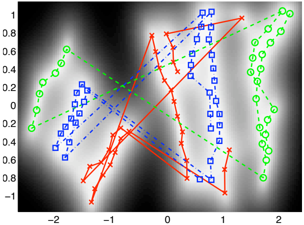
$$\mathcal{L} = \frac{p}{2} \ln |\mathbf{K}| + \frac{p}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) \ .$$

Problems with the GPLVM

- It relies on the optimization of a non-convex function

$$\mathcal{L} = \frac{p}{2} \ln |\mathbf{K}| + \frac{p}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) \ .$$

- Even with the right dimensionality, they can result in poor representations if initialized far from the optimum.

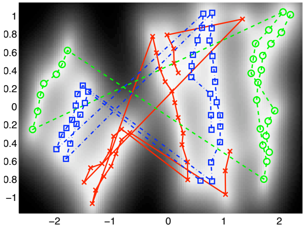


Problems with the GPLVM

- It relies on the optimization of a non-convex function

$$\mathcal{L} = \frac{p}{2} \ln |\mathbf{K}| + \frac{p}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) \ .$$

- Even with the right dimensionality, they can result in poor representations if initialized far from the optimum.



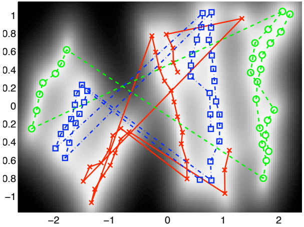
- This is even worst if the dimensionality of the latent space is small.

Problems with the GPLVM

- It relies on the optimization of a non-convex function

$$\mathcal{L} = \frac{p}{2} \ln |\mathbf{K}| + \frac{p}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) \ .$$

- Even with the right dimensionality, they can result in poor representations if initialized far from the optimum.



- This is even worst if the dimensionality of the latent space is small.
- As a consequence this models have only been applied to small databases of a single activity.

Solutions that have been proposed

1. Constrain the back-mapping
2. Incorporate dynamics when learning the latent space
3. Rank priors for continuous dimensionality reduction
4. Incorporate prior knowledge
5. Stochastic gradient descent optimization

1) Back Constraints

Local Distance Preservation (Lawrence et al. 06)

- Most dimensional reduction techniques preserve local distances.
- The GP-LVM does not.
- GP-LVM maps smoothly from latent to data space.
  - Points close in latent space are close in data space.
  - This does not imply points close in data space are close in latent space.
- Kernel PCA maps smoothly from data to latent space.
  - Points close in data space are close in latent space.
  - This does not imply points close in latent space are close in data space.

Back Constraints in the GP-LVM

Back Constraints

- The Neuroscale (Lowe, 96) made latent positions a function of the data.

$$x_{i,j} = f_j\left(\mathbf{y}_{i,:}; \mathbf{v}\right)$$

- We can use the same idea to force the GP-LVM to respect local distances.
  - By constraining each $\mathbf{x}_i$ to be a 'smooth' mapping from $\mathbf{y}_i$ local distances can be respected.
- This works because in the GP-LVM we maximise wrt latent variables, we don't integrate out.
- Can use any 'smooth' function:
  1. Neural network.
  2. RBF Network.
  3. Kernel based mapping.

Optimising BC-GPLVM

Computing Gradients

- GP-LVM normally proceeds by optimising

$$L(\mathbf{X}) = \log p(\mathbf{Y}|\mathbf{X})$$

  with respect to $\mathbf{X}$ using $\frac{\mathrm{d}L}{\mathrm{d}\mathbf{X}}$.
- The back constraints are of the form

$$x_{i,j} = f_j(\mathbf{y}_{i,:}; \mathbf{v})$$

  where $\mathbf{v}$ are parameters.
- We can compute $\frac{\mathrm{d}L}{\mathrm{d}\mathbf{v}}$ via chain rule and optimise parameters of mapping.

Motion Capture Results

[N. Lawrence and J. Quinonero-Candela, ICML 2006]

`demStick1` **and** `demStick3`

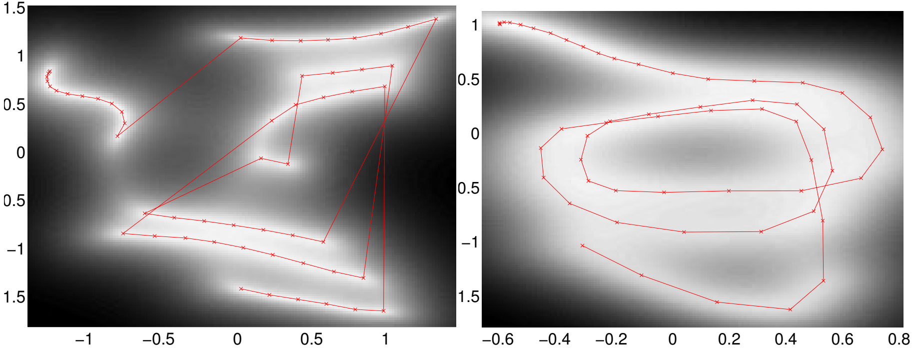
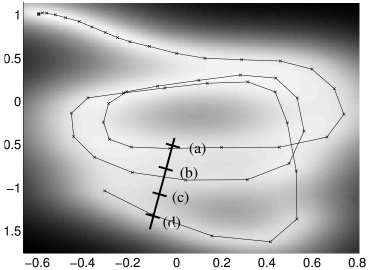


Figure: The latent space for the motion capture data with (*right*) and without (*left*) back constraints.

Stick Man Results

[N. Lawrence and J. Quinonero-Candela, ICML 2006]

`demStickResults`



Projection into data space from four points in the latent space. The inclination of the runner changes becoming more upright.

2) Adding Dynamics

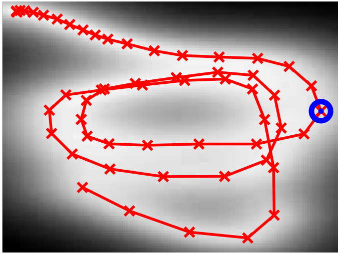
MAP Solutions for Dynamics Models

- Data often has a temporal ordering.
- Markov-based dynamics are often used.
- For the GP-LVM
  - Marginalising such dynamics is intractable.
  - But: MAP solutions are trivial to implement.
- Many choices: Kalman filter, Markov chains *etc.*.
- (Wang et al. 05) suggest using a Gaussian Process.

Gaussian Process Dynamics

GP-LVM with Dynamics

- Autoregressive Gaussian process mapping in latent space between time points.

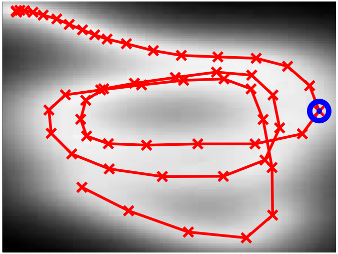


t

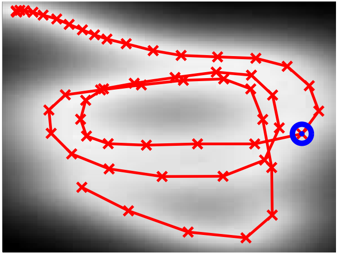
Gaussian Process Dynamics

GP-LVM with Dynamics

- Autoregressive Gaussian process mapping in latent space between time points.



t

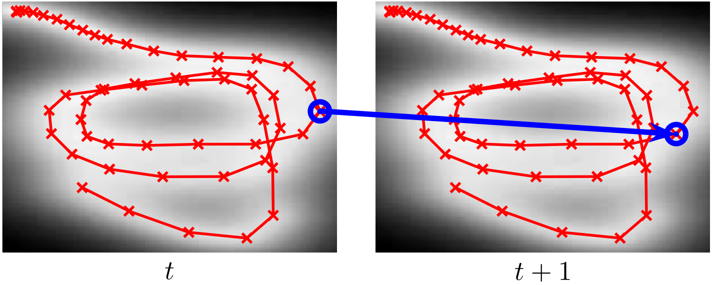


$t+1$

Gaussian Process Dynamics

GP-LVM with Dynamics

- Autoregressive Gaussian process mapping in latent space between time points.



Motion Capture Results

`demStick1` **and** `demStick2`

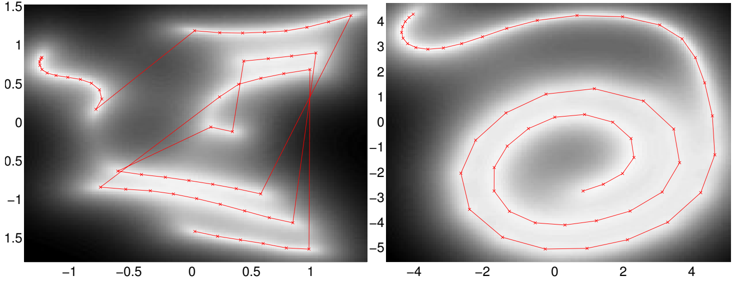
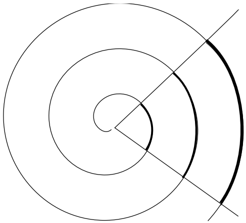


Figure: The latent space for the motion capture data without dynamics (*left*), with auto-regressive dynamics (*right*) based on an exponentiated quadratic kernel.

Regressive Dynamics

Inner Groove Distortion

- Autoregressive unimodal dynamics, $p(\mathbf{x}_t|\mathbf{x}_{t-1})$.
- Forces spiral visualisation.
- Poorer model due to inner groove distortion.



Regressive Dynamics

Direct use of Time Variable

- Instead of auto-regressive dynamics, consider regressive dynamics.
- Take $\mathbf{t}$ as an input, use a prior $p(\mathbf{X}|\mathbf{t})$.
- User a Gaussian process prior for $p(\mathbf{X}|\mathbf{t})$.
- Also allows us to consider variable sample rate data.
- **Problem**: The notion of time might not be appropiate.

Motion Capture Results

[N. Lawrence and A. Moore, ICML 2007]

`demStick1`, `demStick2` **and** `demStick5`

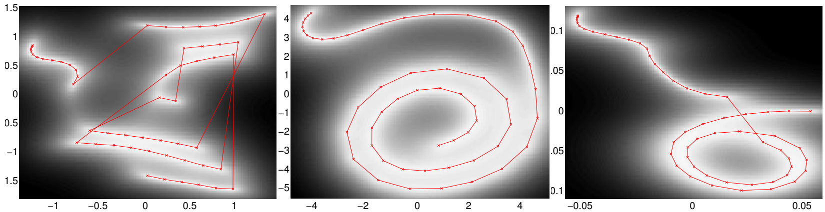


Figure: The latent space for the motion capture data without dynamics (*left*), with auto-regressive dynamics (*middle*) and with regressive dynamics (*right*) based on an exponentiated quadratic kernel.

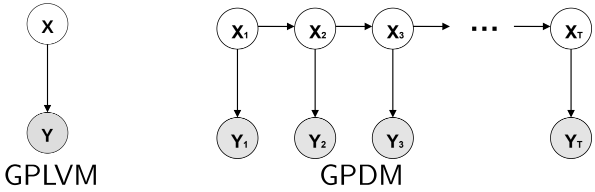
Incorporating dynamics into Tracking

- The mapping from latent space to high dimensional space as

$$\mathbf{y}_{i,:} = \mathbf{W}\psi(\mathbf{x}_{i,:}) + \boldsymbol{\eta}_{i,:}, \quad \text{where} \quad \eta_{i,:} \sim N\left(\mathbf{0}, \sigma^2\mathbf{I}\right).$$

- We can augment the model with ARMA dynamics. This is called Gaussian process dynamical models (GPDM) (Wang et al., 05).

$$\mathbf{x}_{t+1,:} = \mathbf{P}\phi(\mathbf{x}_{t:t-\tau,:}) + \boldsymbol{\gamma}_{i,:}, \quad \text{where} \quad \gamma_{i,:} \sim N\left(\mathbf{0}, \sigma_d^2\mathbf{I}\right).$$



Model Learned for tracking

Model learned from 6 walking subjects,1 gait cycle each, on treadmill at same speed with a 20 DOF joint parameterization (no global pose)

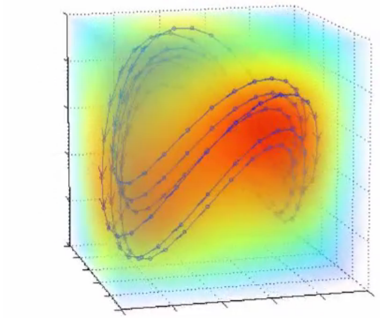


Figure: Density

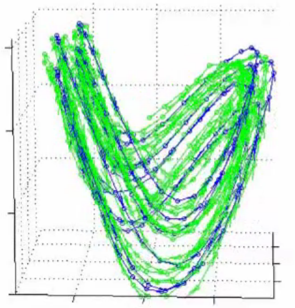
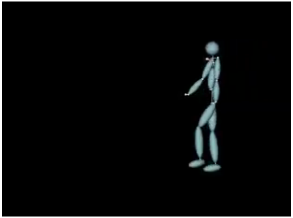
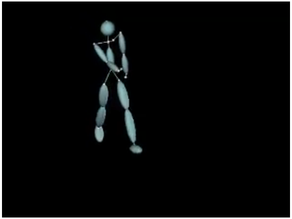


Figure: Randomly generated trajectories

Tracking results

[R. Urtasun, D. Fleet and P. Fua, CVPR 2006]





Estimated latent trajectories

[R. Urtasun, D. Fleet and P. Fua, CVPR 2006]

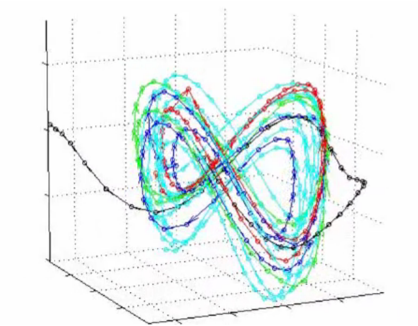
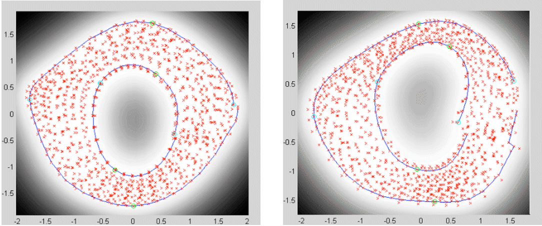


Figure: Estimated latent trajectories. (cian) - training data, (black) - exaggerated walk, (blue) - occlusion.

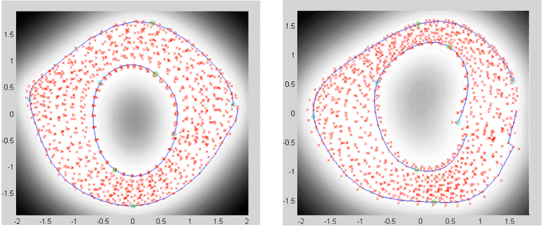
Visualization of Knee Pathology

Two subjects, four walk gait cycles at each of 9 speeds (3-7 km/hr)

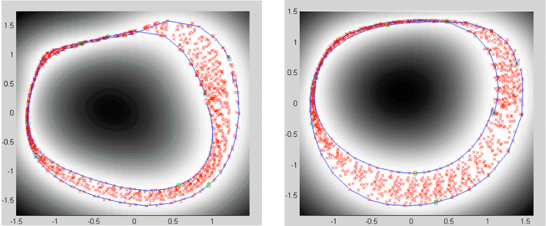


Visualization of Knee Pathology

Two subjects, four walk gait cycles at each of 9 speeds (3-7 km/hr)



Two subjects with a knee pathology.



3) Rank Priors for Dimensionality Reduction

- No distortion is introduced by an initialization step; the latent coordinates are initialized to be the original observations

$$\mathbf{X}_{init} = \mathbf{Y}$$

- We introduce a prior over the latent space that encourages latent spaces to be low dimensional.
- Our method is able to estimate the latent space and its dimensionality.

Continuous dimensionality reduction

- We want to encourage latent space that are low-dimensional.
- Dimensionality can be measure by the rank of $\mathbf{X}\mathbf{X}^T$.

Continuous dimensionality reduction

- We want to encourage latent space that are low-dimensional.
- Dimensionality can be measure by the rank of $\mathbf{X}\mathbf{X}^T$.
- We would like to penalize the rank, but the rank is a discrete function. The optimization would have to solve a complex combinatorial problem.

Continuous dimensionality reduction

- We want to encourage latent space that are low-dimensional.
- Dimensionality can be measure by the rank of $\mathbf{X}\mathbf{X}^T$.
- We would like to penalize the rank, but the rank is a discrete function. The optimization would have to solve a complex combinatorial problem.
- We relax the rank minimization and define a prior that encourages sparsity of the eigenvalues, such that:

$$\mathcal{L} = \frac{p}{2} \ln |\mathbf{K}| + \frac{p}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) + \alpha \sum_{i=1}^{p} \phi(s_i)$$

with s_i the eigenvalues of $\bar{\mathbf{X}}\bar{\mathbf{X}}^T$, $\bar{\mathbf{X}}$ the zero-mean $\mathbf{X}$, and ϕ is a function that encourages sparsity.

Continuous dimensionality reduction

- We want to encourage latent space that are low-dimensional.
- Dimensionality can be measure by the rank of $\mathbf{X}\mathbf{X}^T$.
- We would like to penalize the rank, but the rank is a discrete function. The optimization would have to solve a complex combinatorial problem.
- We relax the rank minimization and define a prior that encourages sparsity of the eigenvalues, such that:

$$\mathcal{L} = \frac{p}{2} \ln |\mathbf{K}| + \frac{p}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) + \alpha \sum_{i=1}^{p} \phi(s_i)$$

with s_i the eigenvalues of $\bar{\mathbf{X}}\bar{\mathbf{X}}^T$, $\bar{\mathbf{X}}$ the zero-mean $\mathbf{X}$, and ϕ is a function that encourages sparsity.

Choice of the penalty function

- Common choice for sparseness is the power family

$$\phi(s_i, r) = |s_i|^r$$

  $r = 1$ is a Laplace prior (i.e., L1 norm), which is linear.

- However, our objective function is non-convex. We use a penalty that drives faster to zero the small singular values

$$\phi(s_i) = \log(1 + \beta s_i) \ .$$

Choice of the penalty function

- Common choice for sparseness is the power family

$$\phi(s_i, r) = |s_i|^r$$

  $r = 1$ is a Laplace prior (i.e., L1 norm), which is linear.
- However, our objective function is non-convex. We use a penalty that drives faster to zero the small singular values

$$\phi(s_i) = \log(1 + \beta s_i) \ .$$

Estimating the dimensionality

- Minimizing the negative log posterior results in a reduction of the energy of the spectrum. We prevent this by optimizing instead

$$\min_{\mathbf{y},\boldsymbol{\theta}} p(\mathbf{Y}|\mathbf{X},\boldsymbol{\theta})$$

$$\text{s. t.} \forall i \; s_i \geq 0, \quad E(\mathbf{Y}) - E(\mathbf{X}) = 0$$

  with the energy $E(\mathbf{X}) = \sum_i s_i^2$.

- Finally, we choose the dimensionality to be

$$Q = \text{argmax}_i \frac{s_i}{s_{i+1} + \epsilon}$$

  where $\epsilon \ll 1$, and $s_1 \geq s_2 \geq \cdots \geq s_D$

Estimating the dimensionality

- Minimizing the negative log posterior results in a reduction of the energy of the spectrum. We prevent this by optimizing instead

$$\min_{\mathbf{y},\boldsymbol{\theta}} p(\mathbf{Y}|\mathbf{X},\boldsymbol{\theta})$$
$$\text{s. t.} \forall i \; s_i \geq 0, \quad E(\mathbf{Y}) - E(\mathbf{X}) = 0$$

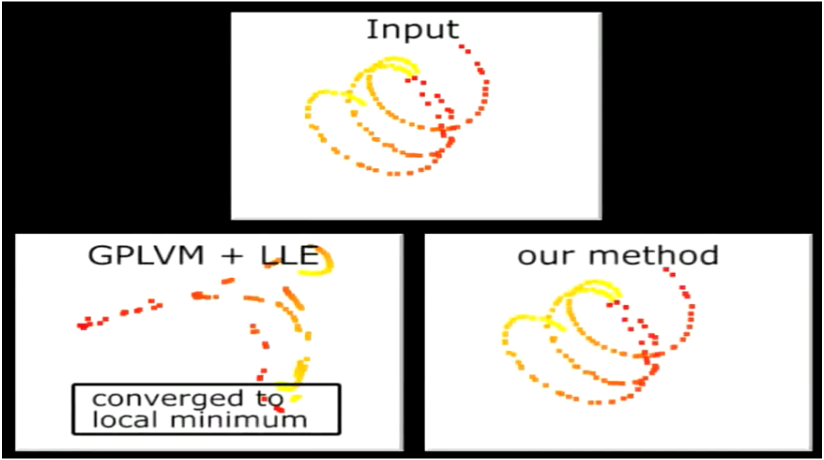
  with the energy $E(\mathbf{X}) = \sum_i s_i^2$.
- Finally, we choose the dimensionality to be

$$Q = \text{argmax}_i \frac{s_i}{s_{i+1} + \epsilon}$$

  where $\epsilon \ll 1$, and $s_1 \geq s_2 \geq \cdots \geq s_D$

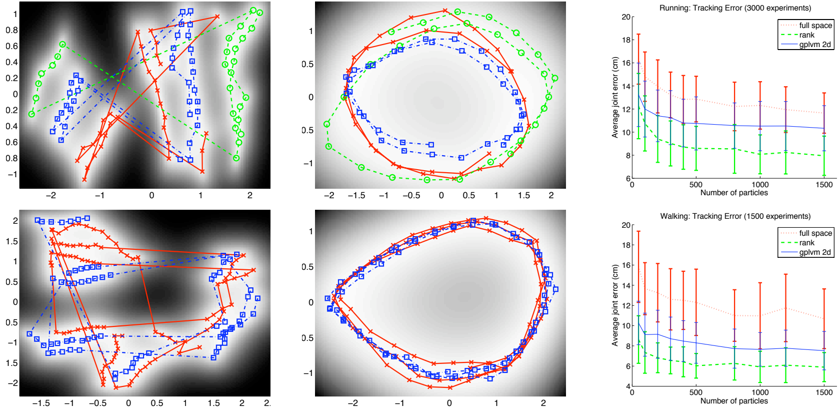
Dimensionality Estimation Results

[A. Geiger, R. Urtasun and T. Darrell, CVPR 2009]



Tracking from Mocap

[A. Geiger, R. Urtasun and T. Darrell, CVPR 2009]



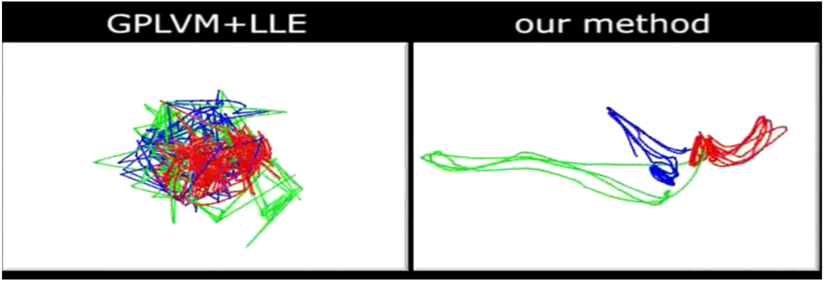
(GPLVM init PCA) (our method) (Error comparison)

Figure: Tracking running (top) and walking (bottom) motions from 2D mocap data. Results are averaged over 10 splits.

Tracking and classifying in the kitchen domain

[A. Geiger, R. Urtasun and T. Darrell, CVPR 2009]

You can learn for the first time latent spaces that are composed of multiple motions.



4) Incorporating prior knowledge

- It is useful to use prior knowledge when additional information is available.
- We design priors over the latent space that incorporate the prior knowledge.
- Prior is based on the Locally Linear Embedding (LLE) [Roweiss, 01] cost function

$$\mathcal{L} = \frac{p}{2} \ln |\mathbf{K}| + \frac{p}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) + \lambda \sum_{i=1}^{N} \sum_{q=1}^{d} ||\mathbf{x}_{i,q} - \sum_{j \in \eta_i} w_{ij,q}\mathbf{x}_{j,q}||^2$$

  with $\mathbf{x}_{i,q}$ the q-th dimension of $\mathbf{x}_i$.
- We define the weights to reflect the prior knowledge.

4) Incorporating prior knowledge

- It is useful to use prior knowledge when additional information is available.
- We design priors over the latent space that incorporate the prior knowledge.
- Prior is based on the Locally Linear Embedding (LLE) [Roweiss, 01] cost function

$$\mathcal{L} = \frac{p}{2} \ln |\mathbf{K}| + \frac{p}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) + \lambda \sum_{i=1}^{N} \sum_{q=1}^{d} ||\mathbf{x}_{i,q} - \sum_{j \in \eta_i} w_{ij,q}\mathbf{x}_{j,q}||^2$$

  with $\mathbf{x}_{i,q}$ the q-th dimension of $\mathbf{x}_i$.
- We define the weights to reflect the prior knowledge.

Example 1: generate animations by sampling

[R. Urtasun, D. J Fleet, A. Geiger, J. Popovic, T. Darrell and N. Lawrence, ICML 2008]

- We learn style-content separation models using the following sources of prior knowledge
  - smoothness: points close in observation space should be close in latent space.
  - cyclic structure: points with similar phase should be close.
  - transitions: points where a transition could happen should be close in the latent space.

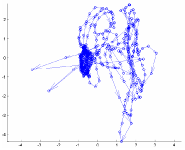


Figure: GPLVM

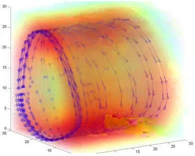


Figure: Topologies

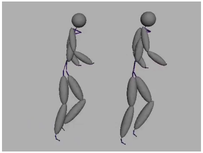
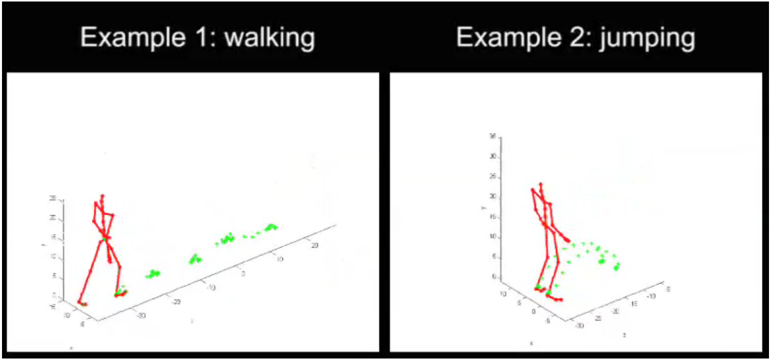


Figure: Sampling

Example 2: generate animations from user constrains

[R. Urtasun, D. J Fleet, A. Geiger, J. Popovic, T. Darrell and N. Lawrence, ICML 2008]

- This problem can be formulated very similarly to tracking.
- Minimize the distance to the user constrains given the motion priors.



5) Stochastic Gradient Descent

[N. Lawrence and R. Urtasun, ICML 2009]

- Learning: maximize likelihood wrt $\mathbf{X}$ and $\boldsymbol{\theta}$.
- This typically get's stuck close to initialization
- We suggest stochastic gradient descent.
  - Do local updates, by selecting points at random
  - Compute gradients in the local neighborhood of the selected points.
- The complexity of each iteration is only $\mathcal{O}(R^3)$, with $R \ll N$, with R the size of the neighborhood
- If the matrix has missing data (e.g., netflix challenge) this is exact, otherwise it's an approximation.

Stochastic Algorithm

Algorithm 1: Stochastic GPLVM
Randomly initialize $\mathbf{X}$
Set $\boldsymbol{\theta}$ with an initial guess
for t **= 1:T**
randomly select $\mathbf{x}_r$
find R neighbors around $\mathbf{x}_r$: $\mathbf{X}_R = \mathbf{X} \in \mathcal{R}$
Compute $\frac{\partial L}{\partial \mathbf{X}_R}$ and $\frac{\partial L}{\partial \boldsymbol{\theta}_R}$
Update $\mathbf{X}$ and $\boldsymbol{\theta}$:
$\Delta \mathbf{X}_t = \mu_X \cdot \Delta \mathbf{X}_{t-1} + \eta_X \cdot \frac{\partial L}{\partial \mathbf{X}_R}$
$\mathbf{X}_t \leftarrow \mathbf{X}_{t-1} + \Delta \mathbf{X}_t$
$\Delta \boldsymbol{\theta}_t = \mu_{\boldsymbol{\theta}} \cdot \Delta \boldsymbol{\theta}_{t-1} + \eta_{\boldsymbol{\theta}} \cdot \frac{\partial L}{\partial \boldsymbol{\theta}_R}$
$\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} + \Delta \boldsymbol{\theta}_t$

Figure: **Stochastic gradient descent and incremental learning** for the GPLVM; $\mu_{(\cdot)}$ is a momentum parameter and $\eta_{(\cdot)}$ is the learning rate.

Results on MOCAP

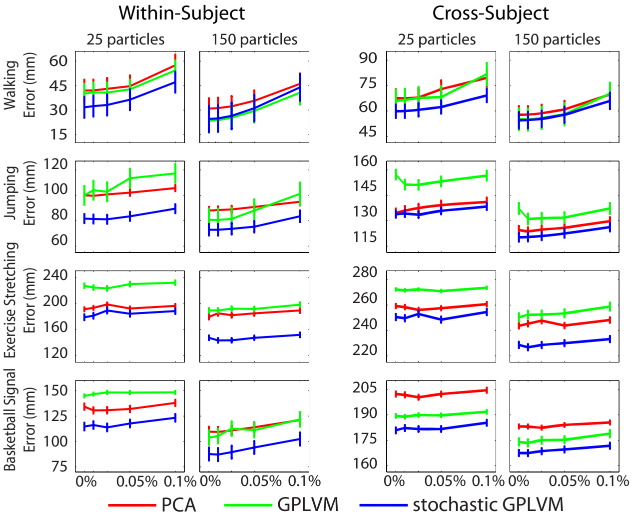
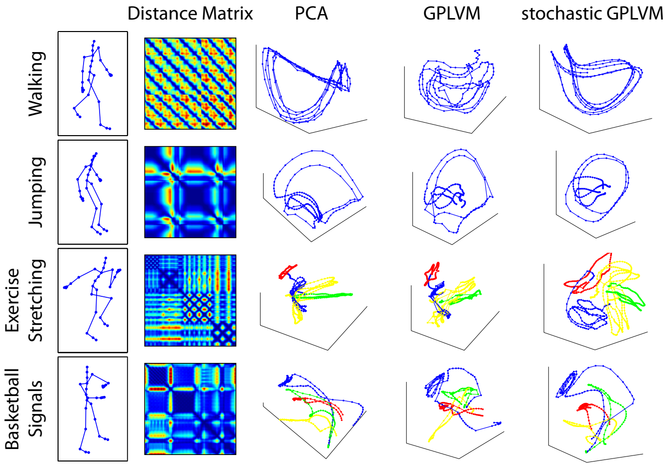


Figure: **Within- and cross-subject 3D tracking errors** for each type of activity sequence with respect to amount of additive noise for different number of particles

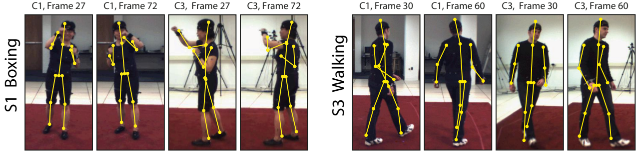
Smooth Latent Space Learning

[A. Yao, J. Gall, L. Van Gool and R. Urtasun, NIPS 2011]



Humaneva Results

[A. Yao, J. Gall, L. Van Gool and R. Urtasun, NIPS 2011]



Train	Test	[Xu07]	[Li10]	GPLVM	CRBM	imCRBM	Ours
S1	S1	-	-	57.6 $\pm$ 11.6	48.8 $\pm$ 3.7	58.6 $\pm$ 3.9	**44.0 $\pm$ 1.8**
S1,2,3	S1	140.3	-	64.3 $\pm$ 19.2	55.4 $\pm$ 0.8	54.3 $\pm$ 0.5	**41.6 $\pm$ 0.8**
S2	S2	-	68.7 $\pm$ 24.7	98.2 $\pm$ 15.8	**47.4 $\pm$ 2.9**	67.0 $\pm$ 0.7	54.4 $\pm$ 1.8
S1,2,3	S2	149.4	-	155.9 $\pm$ 48.8	99.1 $\pm$ 23.0	69.3 $\pm$ 3.3	**64.0 $\pm$ 2.9**
S3	S3	-	69.6 $\pm$ 22.2	71.6 $\pm$ 10.0	49.8 $\pm$ 2.2	51.4 $\pm$ 0.9	**45.4 $\pm$ 1.1**
S1,2,3	S3	156.3	-	123.8. $\pm$ 16.7	70.9 $\pm$ 2.1	**43.4 $\pm$ 4.1**	46.5 $\pm$ 1.4

Model	Tracking Error
[Pavlovic00] as reported in [Li07]	569.90 $\pm$ 209.18
[Lin06] as reported in [Li07]	380.02 $\pm$ 74.97
GPLVM	121.44 $\pm$ 30.7
[Li07]	117.0 $\pm$ 5.5
Best CRBM [Taylor10]	75.4 $\pm$ 9.7
Ours	**74.1 $\pm$ 3.3**

Is that all?

Other Extensions

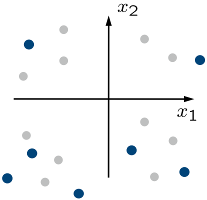
1. Discriminative GPLVMs
2. Hierarchical GPLVMs
3. Multi-output GPLVM
4. Deformation transfer
5. Style-content separation
6. Connectivity priors for animation

1) Priors for supervised learning

- We introduce a prior that is based on the Fisher criteria

$$p(\mathbf{X}) \propto \exp\left\{-\frac{1}{\sigma_d^2} tr\left(\mathbf{S}_w^{-1}\mathbf{S}_b\right)\right\} \ ,$$

with $\mathbf{S}_b$ the between class matrix and $\mathbf{S}_w$ the within class matrix

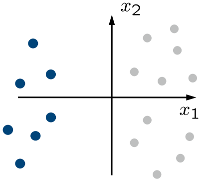


1) Priors for supervised learning

- We introduce a prior that is based on the Fisher criteria

$$p(\mathbf{X}) \propto \exp\left\{-\frac{1}{\sigma_d^2} tr\left(\mathbf{S}_w^{-1}\mathbf{S}_b\right)\right\} \ ,$$

with $\mathbf{S}_b$ the between class matrix and $\mathbf{S}_w$ the within class matrix



$$\mathbf{S}_b = \sum_{i=1}^{L} \frac{n_i}{N}(\mathbf{M}_i - \mathbf{M}_0)(\mathbf{M}_i - \mathbf{M}_0)^T$$

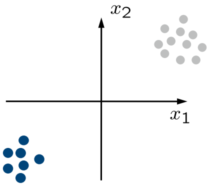
where $\mathbf{X}^{(i)} = [\mathbf{x}_1^{(i)}, \cdots, \mathbf{x}_{n_i}^{(i)}]$ are the n_i training points of class i, $\mathbf{M}_i$ is the mean of the elements of class i, and $\mathbf{M}_0$ is the mean of all the training points of all classes.

1) Priors for supervised learning

- We introduce a prior that is based on the Fisher criteria

$$p(\mathbf{X}) \propto \exp\left\{ -\frac{1}{\sigma_d^2} tr\left(\mathbf{S}_w^{-1}\mathbf{S}_b\right)\right\} \ ,$$

with $\mathbf{S}_b$ the between class matrix and $\mathbf{S}_w$ the within class matrix



$$\mathbf{S}_b = \sum_{i=1}^{L} \frac{n_i}{N}(\mathbf{M}_i - \mathbf{M}_0)(\mathbf{M}_i - \mathbf{M}_0)^T$$

$$\mathbf{S}_w = \sum_{i=1}^{L} \frac{n_i}{n}\left[\frac{1}{n_i}\sum_{k=1}^{N_i}(\mathbf{x}_k^{(i)} - \mathbf{M}_i)(\mathbf{x}_k^{(i)} - \mathbf{M}_i)^T\right]$$

where $\mathbf{X}^{(i)} = [\mathbf{x}_1^{(i)}, \cdots, \mathbf{x}_{n_i}^{(i)}]$ are the n_i training points of class i, $\mathbf{M}_i$ is the mean of the elements of class i, and $\mathbf{M}_0$ is the mean of all the training points of all classes.

- As before the model is learned by maximizing $p(\mathbf{Y}|\mathbf{X})p(\mathbf{X})$.

1) Priors for supervised learning

- We introduce a prior that is based on the Fisher criteria

$$p(\mathbf{X}) \propto \exp\left\{-\frac{1}{\sigma_d^2} tr\left(\mathbf{S}_w^{-1}\mathbf{S}_b\right)\right\} ,$$

with $\mathbf{S}_b$ the between class matrix and $\mathbf{S}_w$ the within class matrix

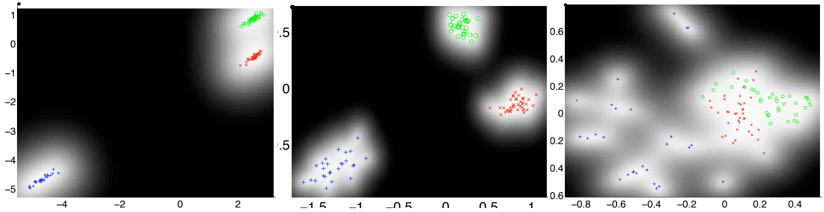


Figure: 2D latent spaces learned by D-GPLVM on the oil dataset are shown, with 100 training examples and different values of σ_d. Note that as $1/\sigma_d^2$ increases the model becomes more discriminative but has worse generalization.

Experimental evaluation

[R. Urtasun and T. Darrell, ICML 2007]

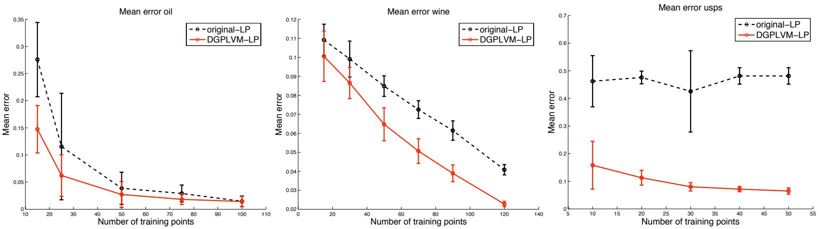
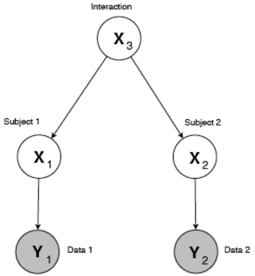


Figure: Mean classification error for the (left) oil (middle) UCI-Wine and (right) USPS datasets. The oil datasets has 3 classes and $D = 12$. The UCI-Wine database has 2 classes with $D = 13$. The USPS dataset consist on discriminating 3's and 5's, $D = 256$.

Hierarchical GP-LVM

2) Stacking Gaussian Processes

- Regressive dynamics provides a simple hierarchy.
  - The input space of the GP is governed by another GP.



- By stacking GPs we can consider more complex hierarchies.
- Ideally we should marginalise latent spaces
  - In practice we seek MAP solutions.

Two Correlated Subjects

[N. Lawrence and A. Moore, ICML 2007]

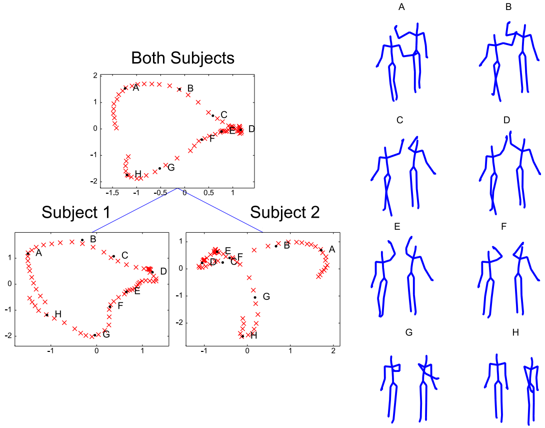


Figure: Hierarchical model of a 'high five'.

Within Subject Hierarchy

Decomposition of Body

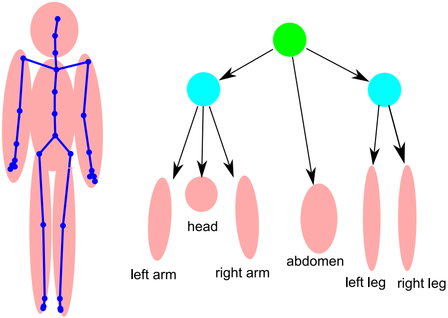


Figure: Decomposition of a subject.

Single Subject Run/Walk

[N. Lawrence and A. Moore, ICML 2007]

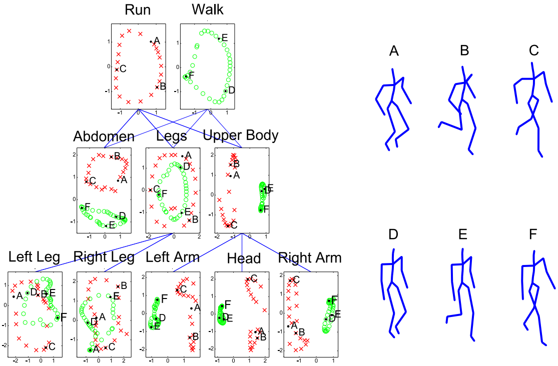
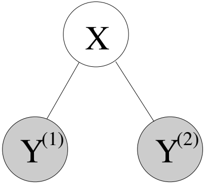


Figure: Hierarchical model of a walk and a run.

3) Modeling Multiple Outputs with GPLVM

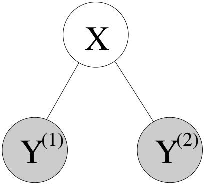
- Single space to model correlations between two different data sources, e.g., images & text, image & pose.
- Shared latent spaces: (Shon et al. NIPS'06, Ek et al. MLMI'07, Navaratnam et al. ICCV'07).



- Effective when the views are correlated.
- But not all information is shared between both views.

3) Modeling Multiple Outputs with GPLVM

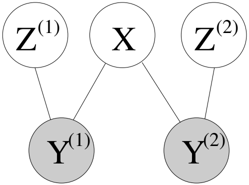
- Single space to model correlations between two different data sources, e.g., images & text, image & pose.
- Shared latent spaces: (Shon et al. NIPS'06, Ek et al. MLMI'07, Navaratnam et al. ICCV'07).



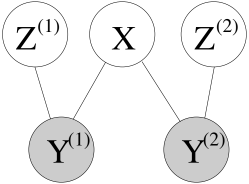
- Effective when the views are correlated.
- But not all information is shared between both views.

Shared-Private Factorization

- In real scenarios, the views are neither fully independent, nor fully correlated.
- Shared models
  - either allow information relevant to a single view to be mixed in the shared signal,
  - or are unable to model such private information.
- Solution: Model shared and private information (Ek et al. MLMI'08, Leen 2008)



Factorized Orthogonal Latent Spaces (FOLS)



A FOLS model can be learned by minimizing (Salzmann et al. 10)

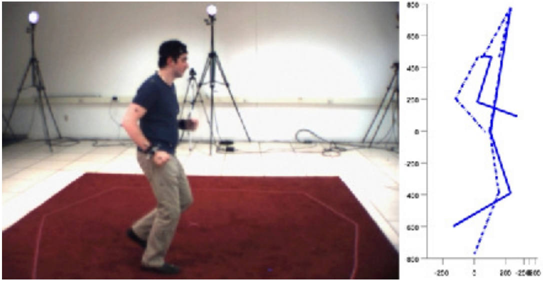
$$\mathcal{L} = L_{data} + L_{ortho} + L_{dim} + L_{energy} \; .$$

- It does continuous dimensionality reduction
- Orthogonality prior to encourage the different latent spaces to be non-redundant.

$$L_{ortho} = \alpha \sum_i \left(||\mathbf{X}^T \cdot \mathbf{Z}^{(i)}||_F^2 + \sum_{j>i} ||(\mathbf{Z}^{(i)})^T \cdot \mathbf{Z}^{(j)}||_F^2 \right) \; .$$

Experiments: discriminative pose estimation

We seek to recover the 3D pose from image features

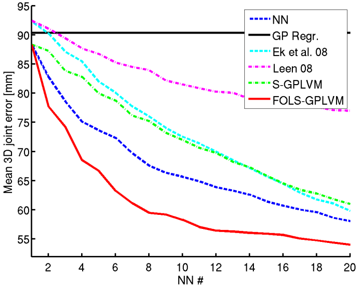


- $\mathbf{Y}^{(1)}$ is image representation
- $\mathbf{Y}^{(2)}$ pose (i.e., 3D angles for each joint)

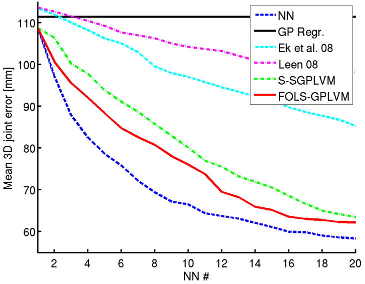
Humaneva: Jog and Walk

[M. Salzmann, C. Ek, R. Urtasun and T. Darrell, AISTATS 2010]

Discriminative Pose Estimation: hopeless?



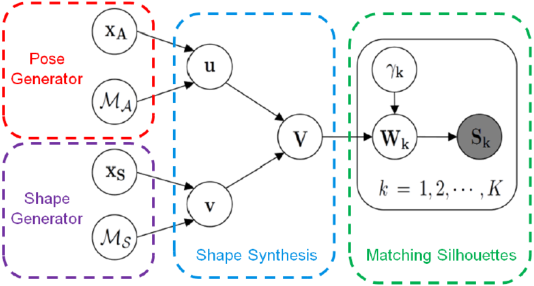
(Jog)



(Walk)

4) Modeling Pose and Shape

- Model two types of variation: **phenotype** variation and **pose**
- They model each variation with an independent GPLVM

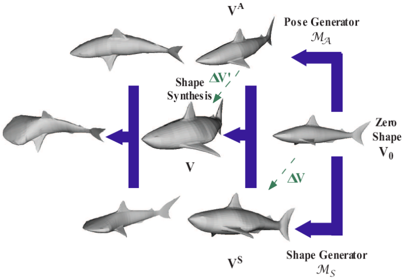


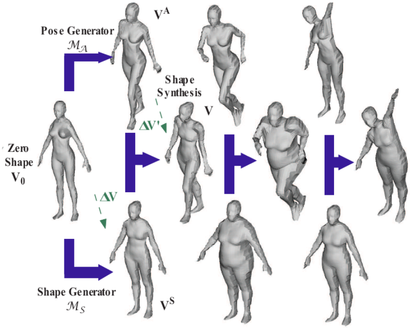
- Models have to be registered!
- Combine both at inference by "deformation transfer" [Sumner et al., 04]

$$\mathbf{V} = \mathbf{V}^A + \mathbf{J}(\mathbf{V}^S - \mathbf{V}^0) + \mathbf{n}_V$$

Generating 3D Shapes

- For shape synthesis the posterior is non-Gaussian, thus it requires approximations





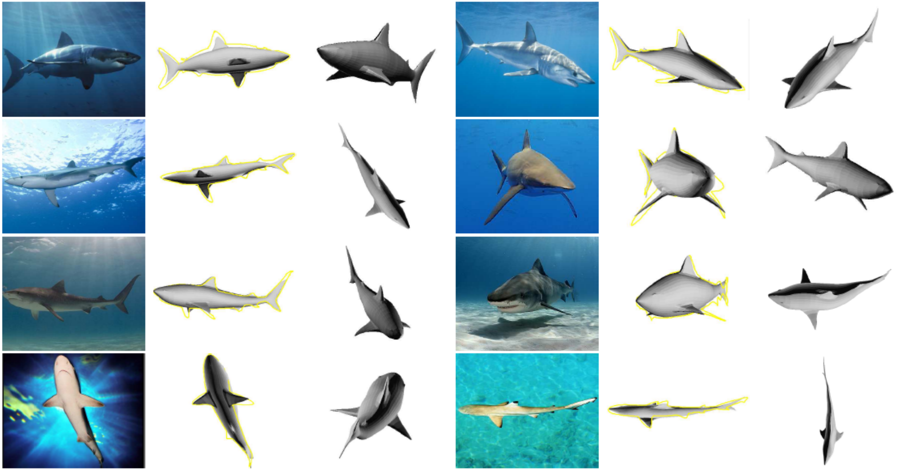
Matching Silhouettes

Silhouette matching is a two-stage process

- Initial segmentation using Grabcuts
- Project the 3D shape to the 2D image plane
- Chamfer matching of 2D silhouettes

Results: Sharks

[Y. Chen, T. Kim and R. Cipolla, ECCV 2010]

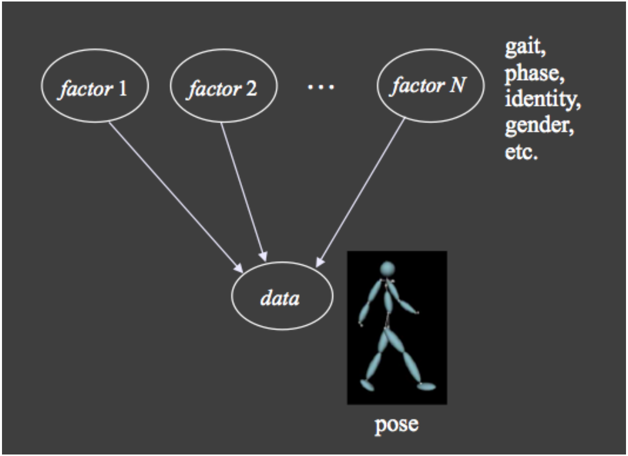


Results: Humans

[Y. Chen, T. Kim and R. Cipolla, ECCV 2010]

5) Style Content Separation and Multi-linear models

Multiple aspects that affect the input signal, interesting to factorize them



Multilinear models

- Style-Content Separation (Tenenbaum & Freeman 00)

$$\mathbf{y} = \sum_{ij} w_{ij} a_i b_j + \epsilon$$

- Multi-linear analysis (Vasilescu & Terzopoulous 02)

$$\mathbf{y} = \sum_{ijk\cdots} w_{ijk\ldots} a_i b_j c_k \cdots + \epsilon$$

- Non-linear basis functions (Elgammal & Lee, 2004)

$$\mathbf{y} = \sum_{ij} w_{ij} a_i \phi_j(b) + \epsilon$$

Multi (non)-linear models with GPs

- In the GPLVM
$$\mathbf{y} = \sum_j w_j \phi_j(\mathbf{x}) + \epsilon = \mathbf{w}^T \Phi(\mathbf{x}) + \epsilon$$
with
$$E[\mathbf{y}, \mathbf{y}'] = \Phi(\mathbf{x})^T \Phi(\mathbf{y}) + \beta^{-1}\delta = k(\mathbf{x}, \mathbf{x}') + \beta^{-1}\delta$$

- Multifactor Gaussian process
$$\mathbf{y} = \sum_{i,j,k,\cdots} w_{ijk\cdots} \phi_i^{(1)} \phi_j^{(1)} \phi_k^{(1)} \cdots + \epsilon$$
with
$$E[\mathbf{y}, \mathbf{y}'] = \prod_i \Phi^{(i)}{}^T \Phi^{(i)} + \beta^{-1}\delta = \prod_i k_i(\mathbf{x}^{(i)}, \mathbf{x}^{(i)}{}') + \beta^{-1}\delta$$

Multi (non)-linear models with GPs

- In the GPLVM

$$\mathbf{y} = \sum_j w_j \phi_j(\mathbf{x}) + \epsilon = \mathbf{w}^T \Phi(\mathbf{x}) + \epsilon$$

with

$$E[\mathbf{y}, \mathbf{y}'] = \Phi(\mathbf{x})^T \Phi(\mathbf{y}) + \beta^{-1}\delta = k(\mathbf{x}, \mathbf{x}') + \beta^{-1}\delta$$

- Multifactor Gaussian process

$$\mathbf{y} = \sum_{i,j,k,\cdots} w_{ijk\cdots} \phi_i^{(1)} \phi_j^{(1)} \phi_k^{(1)} \cdots + \epsilon$$

with

$$E[\mathbf{y}, \mathbf{y}'] = \prod_i {\Phi^{(i)}}^T \Phi^{(i)} + \beta^{-1}\delta = \prod_i k_i(\mathbf{x}^{(i)}, {\mathbf{x}^{(i)}}') + \beta^{-1}\delta$$

- Learning in this model is the same, just the kernel changes.

Multi (non)-linear models with GPs

- In the GPLVM

$$\mathbf{y} = \sum_j w_j \phi_j(\mathbf{x}) + \epsilon = \mathbf{w}^T \Phi(\mathbf{x}) + \epsilon$$

with

$$E[\mathbf{y}, \mathbf{y}'] = \Phi(\mathbf{x})^T \Phi(\mathbf{y}) + \beta^{-1}\delta = k(\mathbf{x}, \mathbf{x}') + \beta^{-1}\delta$$

- Multifactor Gaussian process

$$\mathbf{y} = \sum_{i,j,k,\cdots} w_{ijk\ldots} \phi_i^{(1)} \phi_j^{(1)} \phi_k^{(1)} \cdots + \epsilon$$

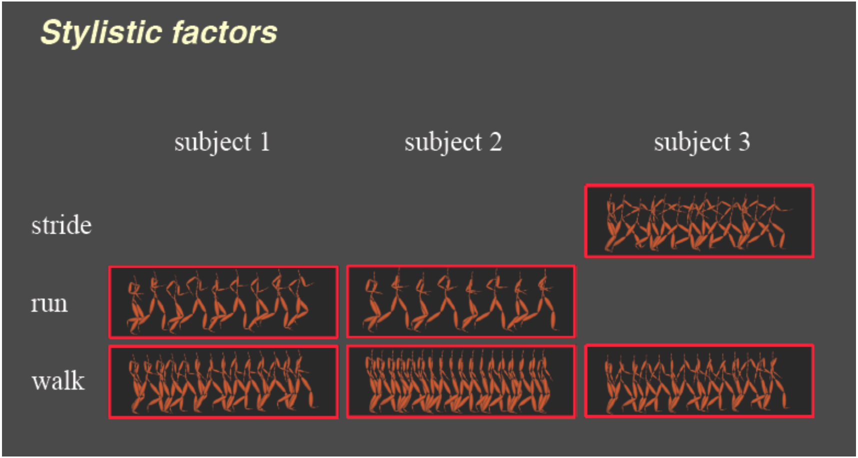
with

$$E[\mathbf{y}, \mathbf{y}'] = \prod_i {\Phi^{(i)}}^T \Phi^{(i)} + \beta^{-1}\delta = \prod_i k_i(\mathbf{x}^{(i)}, {\mathbf{x}^{(i)}}') + \beta^{-1}\delta$$

- Learning in this model is the same, just the kernel changes.

Training Data

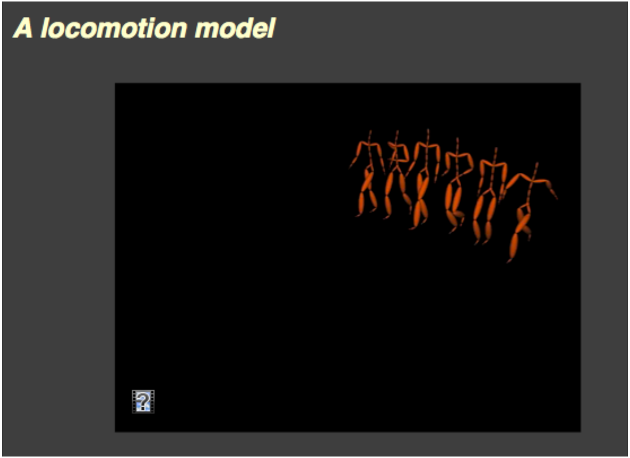
Each training motion is a collection of poses, sharing the same combination of subject (s) and gait (g).



Character Animation

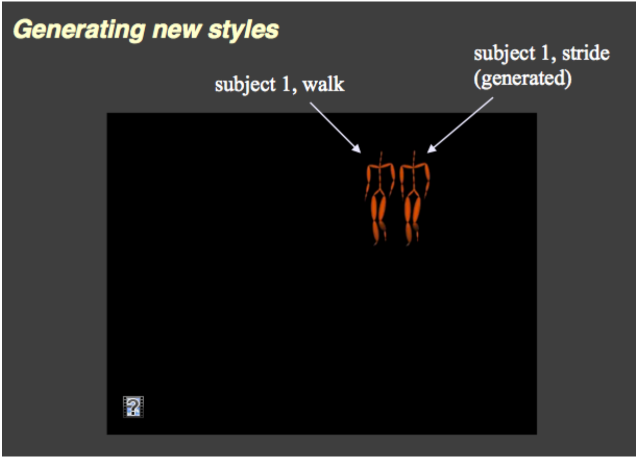
[J. Wang, D. Fleet and A. Hertzmann, ICML 2007]

Training data, 6 sequences, 314 frames in total



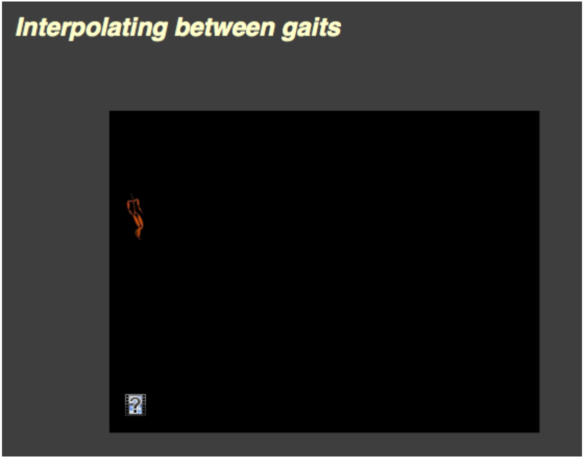
Generating new styles for a subject

[J. Wang, D. Fleet and A. Hertzmann, ICML 2007]



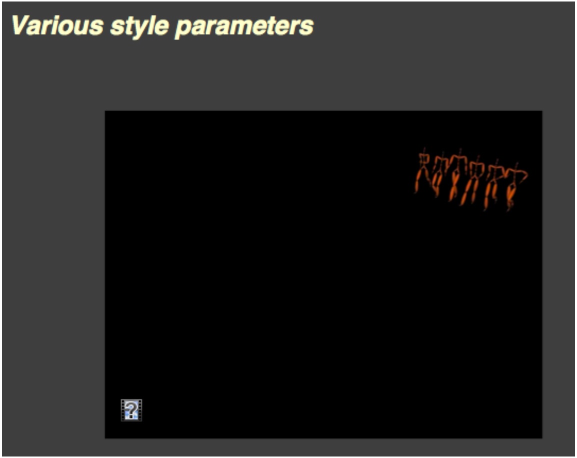
Interpolating Gaits

[J. Wang, D. Fleet and A. Hertzmann, ICML 2007]



Generating Different Styles

[J. Wang, D. Fleet and A. Hertzmann, ICML 2007]



6) Continuous Character Control

- When employing GPLVM, different motions get too far apart
- Difficult to generate animations where we transition between motions
- Back-constraints or topologies are not enough
- New prior that enforces connectivity in the graph

$$\ln p(\mathbf{X}) = w_c \sum_{i,j} \ln K_{ij}^d$$

with the **graph diffusion kernel $\mathbf{K}^d$** obtain from

$$K_{ij}^d = \exp(\beta \mathbf{H}) \qquad \text{with} \qquad \mathbf{H} = -\mathbf{T}^{-1/2}\mathbf{L}\mathbf{T}^{-1/2}$$

the graph Laplacian, and $\mathbf{T}$ is a diagonal matrix with $T_{ii} = \sum_j w(\mathbf{x}_i, \mathbf{x}_j)$,

$$L_{ij} = \begin{cases} \sum_k w(\mathbf{x}_i, \mathbf{x}_k) & \text{if } i = j \\ -w(\mathbf{x}_i, \mathbf{x}_j) & \text{otherwise.} \end{cases}$$

and $w(\mathbf{x}_i, \mathbf{x}_j) = ||\mathbf{x}_i - \mathbf{x}_j||^{-p}$ measures similarity.

Embeddings: Walking

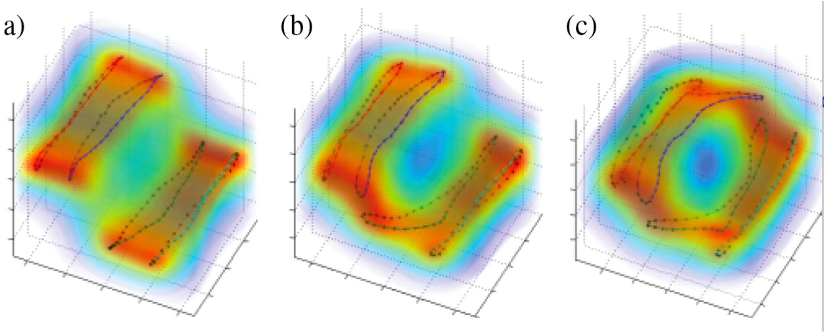


Figure: Walking embeddings learned (a) without the connectivity term, (b) with $w_c = 0{:}1$, and (c) with $w_c = 1{:}0$.

Embeddings: Punching

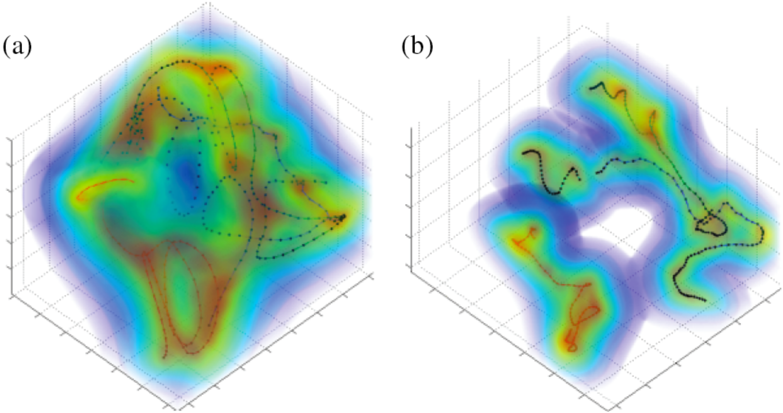


Figure: Embeddings for the punching task (a) with and (b) without the connectivity term.

Video Results

[S. Levine, J. Wang, A. Haraux, Z. Popovic and V. Koltun, Siggraph 2012]