Nonrigid Structure from Motion

Evan Herbst

Outline

- Problem formulations
- Torresani et al
- Bartoli et al
- Comparison

Last Week

Problem Space

- Single viewpoint image (Monocular)
- Recover 3D Shape

Shape hypothesis

Input image

Types of Shape Model

- Physics-based models
  - cons: requires material properties,
    models smoothly deforming objects
  - pros: computationally efficient,
    infers shape in un-textured regions
- Global models learned from data
  - cons: requires lots of data,
    object-shape-specific,
    linear or quadratic models
- Local models learned from data
  - pros: same model for all parts of a homogeneous surface,
    more constrained than global models

(compiled from multiple original slides)

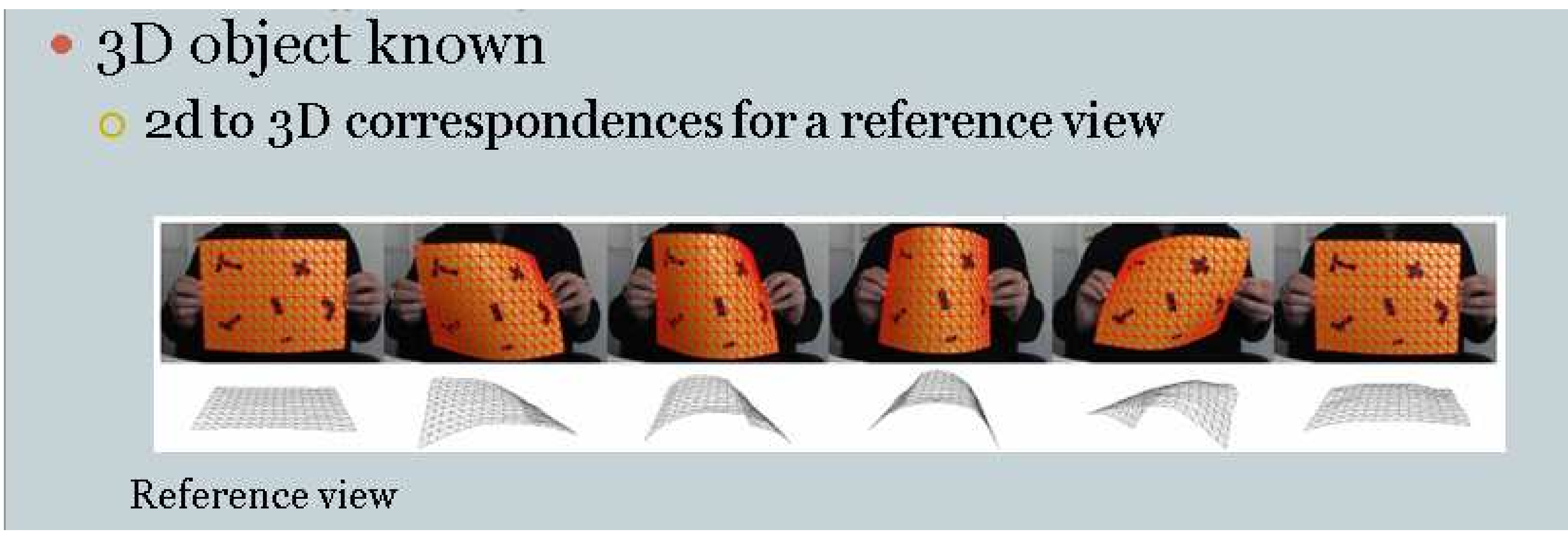
Local Deformations Approach

given

- space of all shapes
- an image at each timestep
- 2-d point tracks for these images

to find

- shape at each timestep



Making the Problem Harder

given

- ~~space of all shapes~~
- an image at each timestep
- 2-d point tracks for these images

to find

- shape at each timestep
- space of all shapes

Outline

- Problem formulations
- Torresani et al
  - Algorithm
  - Results
  - Summary
- Bartoli et al
- Comparison

A Global Shape Model

Bregler/Hertzmann/Biermann, CVPR00: linear subspace model

- "shape": set of n points on an object

$$\vec{s}_i(t) \in \mathbb{R}^3$$

A Global Shape Model

Bregler/Hertzmann/Biermann, CVPR00: linear subspace model

- "shape": set of n points on an object

$$\vec{s}_i(t) \in \mathbb{R}^3$$

- allowable shapes lie in a K-dimensional subspace of $\mathbb{R}^{3n}$

$$\begin{aligned} \text{mean shape } \bar{s}_i &= \frac{1}{m}\sum_{t=1}^{m} \vec{s}_i(t) \\ \vec{s}_i(t) &= \bar{s}_i + V_i \vec{z}(t) \end{aligned}$$

(can concatenate vectors)

A World Model

Torresani/Hertzmann/Bregler, PAMI08

- linear subspace shape model

$$\overrightarrow{s}_i(t) = \bar{s}_i + V_i \overrightarrow{z}(t)$$

A World Model

Torresani/Hertzmann/Bregler, PAMI08

- linear subspace shape model

$$\overrightarrow{s}_i(t) = \bar{s}_i + V_i \overrightarrow{z}(t)$$

- weak perspective projection model

$$\overrightarrow{p}_i(t) = c(t) R_t(\overrightarrow{s}_i(t) + \overrightarrow{t}_t) + \overrightarrow{n}(t)$$

$\overrightarrow{n}(t)$ zero-mean Gaussian noise with stdev σ_{noise}

Smoothness Prior

Torresani/Hertzmann/Bregler, PAMI08

$$\overrightarrow{z}(t) \sim \mathcal{N}(\overrightarrow{0}, I)$$

Smoothness Prior

Torresani/Hertzmann/Bregler, PAMI08

$$\vec{z}\,(t) \sim \mathcal{N}(\vec{0}, I)$$

- similar deformations at different times

Smoothness Prior

Torresani/Hertzmann/Bregler, PAMI08

$$\vec{z}(t) \sim \mathcal{N}(\vec{0}, I)$$

- similar deformations at different times
- force small deformations? no

Smoothness Prior

Torresani/Hertzmann/Bregler, PAMI08

$$\overrightarrow{z}(t) \sim \mathcal{N}(\overrightarrow{0}, I)$$

- similar deformations at different times
- force small deformations? no
- 2-d projection $\overrightarrow{p}_i(t)$ is distributed normally (weak perspective)

Algorithm

Torresani/Hertzmann/Bregler, PAMI08

- probabilistic model $\Longrightarrow$ can use max-likelihood principle

$$L(\text{point } i) = \prod_t p(\overrightarrow{p}_i\,(t)|c(t), R_t, \overrightarrow{t}_t, \bar{s}_i, V_i, \sigma_{noise})$$

Algorithm

Torresani/Hertzmann/Bregler, PAMI08

- probabilistic model $\Longrightarrow$ can use max-likelihood principle

$$L(\text{point } i) = \prod_t p(\vec{p}_i(t) | \underbrace{c(t), R_t, \vec{t}_t, \bar{s}_i, V_i, \sigma_{noise}}_{\text{M step updates these}})$$

- do EM to allow for missing point tracks
  - E step: update posterior over basis weights $\vec{z}(t)\ \forall t$
  - M step: maximize data likelihood $L(\cdot)$ by optimizing mean shape, basis shapes, camera parameters, and noise

EM Initialization

EM needs initial estimates for everything

EM Initialization

EM needs initial estimates for everything

- pick a K (# bases) that's not too small

EM Initialization

EM needs initial estimates for everything

- pick a K (# bases) that's not too small
- use affine rigid SFM (Tomasi-Kanade) to get mean shape and camera transformations

EM Initialization

EM needs initial estimates for everything

- pick a K (# bases) that's not too small
- use affine rigid SFM (Tomasi-Kanade) to get mean shape and camera transformations
- σ_{noise} should be large

EM Initialization

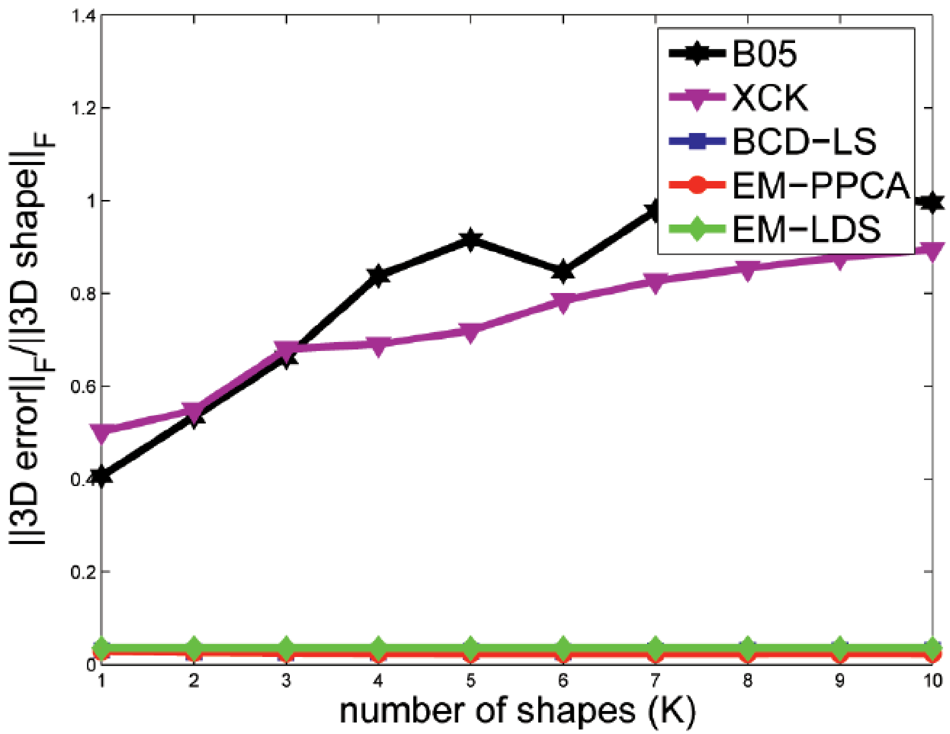
EM needs initial estimates for everything

- pick a K (# bases) that's not too small
- use affine rigid SFM (Tomasi-Kanade) to get mean shape and camera transformations
- σ_{noise} should be large
- iteratively add basis shapes (and basis weights) so as to minimize reprojection error

Outline

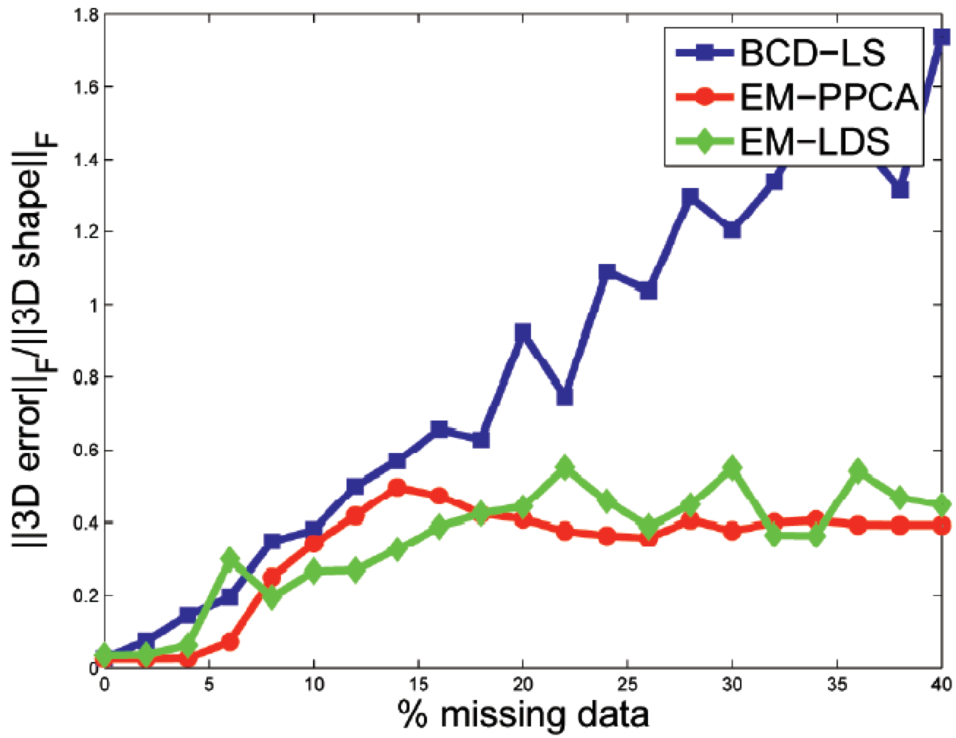
- Problem formulations
- Torresani et al
  - Algorithm
  - Results
  - Summary
- Bartoli et al
- Comparison

Results: Dependence on K



(Torresani face dataset)

Results: Missing Data



(Torresani face dataset)

Outline

- Problem formulations
- Torresani et al
  - Algorithm
  - Results
  - Summary
- Bartoli et al
- Comparison

Things to Remember

- simple extension of linear subspace shape model
- max likelihood optimization minimizes reprojection error
- robust against overfitting

Outline

- Problem formulations
- Torresani et al
- Bartoli et al
  - Algorithm
  - Results
  - Summary
- Comparison

Reminder: Linear Subspace Model

- “shape”: set of n points on an object

$$\overrightarrow{s}_i\,(t) \in \mathbb{R}^3$$

Reminder: Linear Subspace Model

- "shape": set of n points on an object

$$\overrightarrow{s}_i(t) \in \mathbb{R}^3$$

- allowable shapes lie in a K-dimensional subspace of $\mathbb{R}^{3n}$

$$\begin{aligned} \overrightarrow{s}_i(t) &= \bar{s}_i + V_i \overrightarrow{z}(t) \\ \text{mean shape } \bar{s}_i &= \frac{1}{m} \sum_{t=1}^{m} \overrightarrow{s}_i(t) \end{aligned}$$

Another World Model

Bartoli et al, CVPR08

- linear subspace shape model

$$\overrightarrow{s}_i\,(t) = D_t\left(\bar{s}_i + V_i\,\overrightarrow{z}\,(t)\right)$$

Another World Model

Bartoli et al, CVPR08

- linear subspace shape model

$$\overrightarrow{s}_i(t) = D_t \left(\bar{s}_i + V_i \overrightarrow{z}(t) \right)$$

- full perspective projection model

$$\overrightarrow{p}_i(t) = P_t(\overrightarrow{s}_i(t) + \overrightarrow{t}_t) \qquad \text{(up to a scalar)}$$

$$P_t = K_t R_t T_t$$

Another World Model

Bartoli et al, CVPR08

- linear subspace shape model

$$\vec{s}_i(t) = D_t\left(\bar{s}_i + V_i\,\vec{z}(t)\right)$$

- full perspective projection model

$$\vec{p}_i(t) = P_t(\vec{s}_i(t) + \vec{t}_t) \qquad \text{(up to a scalar)}$$

$$P_t = K_t R_t T_t$$

  - makes problem much harder

Algorithm

Bartoli et al, CVPR08

1. rigid SFM (with standard techniques) assuming projection matrices known
   - produces mean shape and camera-world transformations

Algorithm

Bartoli et al, CVPR08

1. rigid SFM (with standard techniques) assuming projection matrices known
    - produces mean shape and camera-world transformations
    - bad (explanation in a bit)

Algorithm

Bartoli et al, CVPR08

1. rigid SFM (with standard techniques) assuming projection matrices known
   - produces mean shape and camera-world transformations
   - bad (explanation in a bit)
2. iteratively
   (a) optimize one additional basis shape to minimize reprojection error

Algorithm

Bartoli et al, CVPR08

1. rigid SFM (with standard techniques) assuming projection matrices known
   - produces mean shape and camera-world transformations
   - bad (explanation in a bit)
2. iteratively
   (a) optimize one additional basis shape to minimize reprojection error
   (b) cross-validate (novel for SFM)

Algorithm

Bartoli et al, CVPR08

1. rigid SFM (with standard techniques) assuming projection matrices known
   - produces mean shape and camera-world transformations
   - **bad**
2. iteratively
   (a) optimize one additional basis shape to minimize reprojection error
   (b) cross-validate

Decomposition of Camera Matrix

- Torresani: $\overrightarrow{p}_i(t) = c(t)R_t(\overrightarrow{s}_i(t) + \overrightarrow{t}_t) + \overrightarrow{n}(t)$
- Bartoli:

$$\overrightarrow{s}_i(t) = D_t\left(\bar{s}_i + V_i\overrightarrow{z}(t)\right)$$

$$\overrightarrow{p}_i(t) = P_t(\overrightarrow{s}_i(t) + \overrightarrow{t}_t)$$ (up to a scalar)

Decomposition of Camera Matrix

- Torresani: $\overrightarrow{p}_i(t) = c(t)R_t(\overrightarrow{s}_i(t) + \overrightarrow{t}_t) + \overrightarrow{n}(t)$
- Bartoli:

$$\overrightarrow{s}_i(t) = D_t\left(\bar{s}_i + V_i\overrightarrow{z}(t)\right)$$

$$\overrightarrow{p}_i(t) = P_t(\overrightarrow{s}_i(t) + \overrightarrow{t}_t) \qquad \text{(up to a scalar)}$$

  - overparameterizes but ignores P_t

Decomposition of Camera Matrix

- Torresani: $\overrightarrow{p}_i(t) = c(t)R_t(\overrightarrow{s}_i(t) + \overrightarrow{t}_t) + \overrightarrow{n}(t)$
- Bartoli:

$$\overrightarrow{s}_i(t) = D_t\left(\bar{s}_i + V_i\overrightarrow{z}(t)\right)$$

$$\overrightarrow{p}_i(t) = P_t(\overrightarrow{s}_i(t) + \overrightarrow{t}_t) \qquad \text{(up to a scalar)}$$

  - overparameterizes but ignores P_t
  - set D_t under rigidity assumption; never refine

Decomposition of Camera Matrix

- Torresani: $\overrightarrow{p}_i(t) = c(t)R_t(\overrightarrow{s}_i(t) + \overrightarrow{t}_t) + \overrightarrow{n}(t)$
- Bartoli:

$$\overrightarrow{s}_i(t) = D_t\left(\bar{s}_i + V_i\overrightarrow{z}(t)\right)$$

$$\overrightarrow{p}_i(t) = P_t(\overrightarrow{s}_i(t) + \overrightarrow{t}_t) \qquad \text{(up to a scalar)}$$

  - overparameterizes but ignores P_t
  - set D_t under rigidity assumption; never refine
    - scene reconstruction can end up completely wrong

Algorithm

Bartoli et al, CVPR08

1. rigid SFM (with standard techniques) assuming projection matrices known
   - produces mean shape and camera-world transformations
   - bad
2. iteratively
   (a) **optimize one additional basis shape to minimize reprojection error**
   (b) cross-validate

Objective Function

- reprojection error plus temporal and spatial smoothness terms

$$O(\{V_i\},\{\vec{z}(t)\}) = \sum_{\text{point } i, \text{image } t} \left(v_{i,t} \left| \vec{p}_i(t) - \vec{p}_i^{\,reproj}(t) \right|^2 \right) + \lambda O_{temporal} + \kappa O_{spatial}$$

($v_{i,t}$ visibility flags)

Objective Function

- reprojection error plus temporal and spatial smoothness terms

$$O(\{V_i\},\{\vec{z}(t)\}) = \sum_{\text{point } i, \text{image } t} \left(v_{i,t} \left| \vec{p}_i(t) - \vec{p}_i^{\,reproj}(t) \right|^2 \right) + \lambda O_{temporal} + \kappa O_{spatial}$$

($v_{i,t}$ visibility flags)

- approximate so each point i presents an independent subproblem
  - still nonlinear
  - very efficient

Temporal Smoothness Term

Bartoli et al, CVPR08

$O($basis shapes, basis weights$) =$ reprojection error $+ O_{temporal} + O_{spatial}$

temporal smoothness term a slight generalization of

$$O_{temporal} = \sum_{t=1}^{m} \left| \vec{z}\,(t) - \vec{z}\,(t-1) \right|^2,$$

Spatial Smoothness Term

Bartoli et al, CVPR08

$$O(\text{basis shapes, basis weights}) = \text{reprojection error} + O_{temporal} + O_{spatial}$$

$$O_{spatial} = \sum_{\text{points } i,j\text{, basis shapes } k} \psi(i,j) \left| \overrightharpoon{v}_{ik} - \overrightharpoon{v}_{jk} \right|^2 ,$$

$\psi(i,j)$ a decreasing function of distance between points i, j on the mean shape,
$\overrightharpoon{v}_{ik}$ the location of point i in shape mode k

Spatial Smoothness Term

Bartoli et al, CVPR08

$$O(\text{basis shapes, basis weights}) = \text{reprojection error} + O_{temporal} + O_{spatial}$$

$$O_{spatial} = \sum_{\text{points } i,j \text{, basis shapes } k} \psi(i,j) \left| \vec{v}_{ik} - \vec{v}_{jk} \right|^2,$$

$\psi(i,j)$ a decreasing function of distance between points i,j on the mean shape,
$\vec{v}_{ik}$ the location of point i in shape mode k

Initialization

actual basis representation:

$$\begin{aligned}\overrightarrow{s}_i(t) &= \bar{s}_i + \overrightarrow{z}(t)\; V_i \\ &= \bar{s}_i + \sum_{\text{basis } k} (a_{kt}\; b_{it}\; \overrightarrow{C}_{it})\end{aligned}$$

Initialization

actual basis representation:

$$
\begin{aligned}
\overrightarrow{s}_i(t) &= \bar{s}_i + \overrightarrow{z}(t)\; V_i \\
&= \bar{s}_i + \sum_{\text{basis } k}(a_{kt}\; b_{it}\, \overrightarrow{C}_{it})
\end{aligned}
$$

(d)

Initialization

actual basis representation:

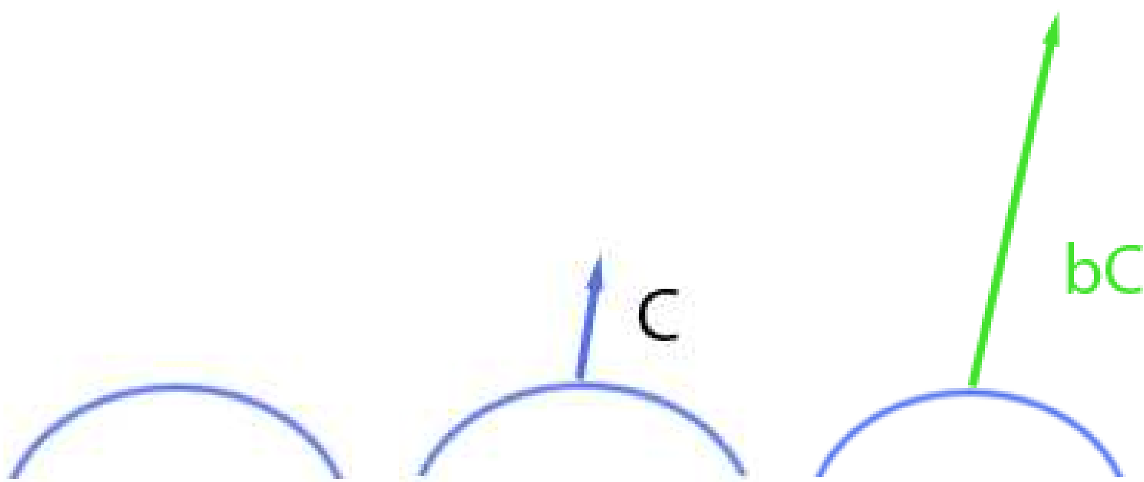
$$\begin{aligned}\vec{s}_i(t) &= \bar{s}_i + \vec{z}(t)\; V_i \\ &= \bar{s}_i + \textstyle\sum_{\text{basis } k}(\; a_{kt}\; b_{it}\, \vec{C}_{it})\end{aligned}$$

(g) (h)

Initialization

actual basis representation:

$$
\begin{aligned}
\overrightarrow{s}_i(t) &= \bar{s}_i + \overrightarrow{z}(t)\ V_i \\
&= \bar{s}_i + \textstyle\sum_{\text{basis } k}(\ a_{kt}\ \ b_{it}\ \overrightarrow{C}_{it})
\end{aligned}
$$

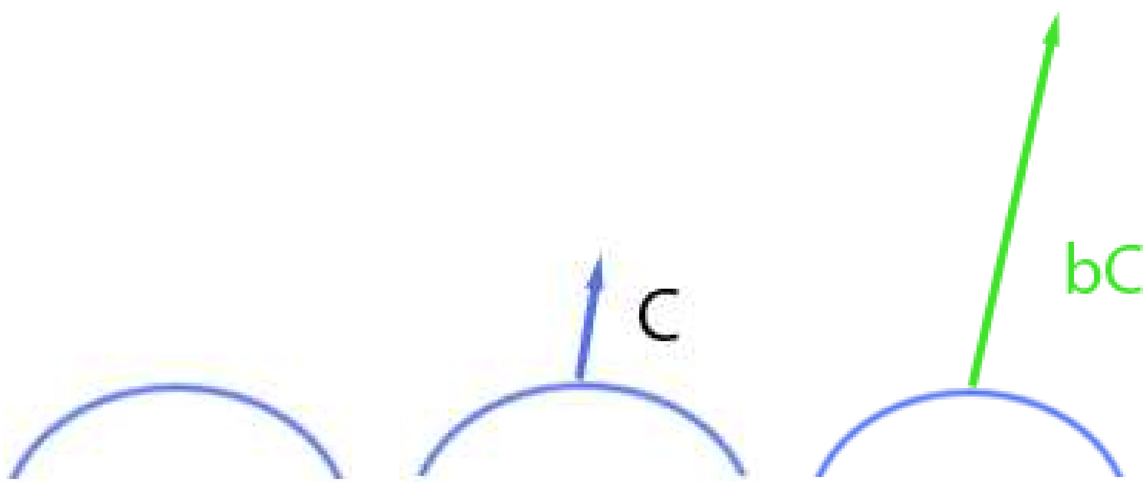


(j) (k) (l)

Initialization

actual basis representation:

$$\begin{aligned}\overrightarrow{s}_i(t) &= \bar{s}_i + \overrightarrow{z}(t)\ V_i \\ &= \bar{s}_i + \sum_{\text{basis } k}(\ a_{kt}\ \ b_{it}\ \overrightarrow{C}_{it})\end{aligned}$$



(m) (n) (o)

initialize directions, then magnitudes

Initialization

- direction vectors $\{\overrightarrow{C}_{it}\}$

Initialization

- direction vectors $\{\overrightarrow{C_{it}}\}$

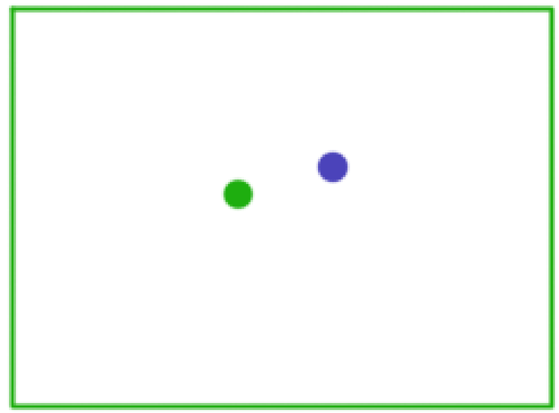


image t, point i

Initialization

- direction vectors $\{\overrightarrow{C_{it}}\}$

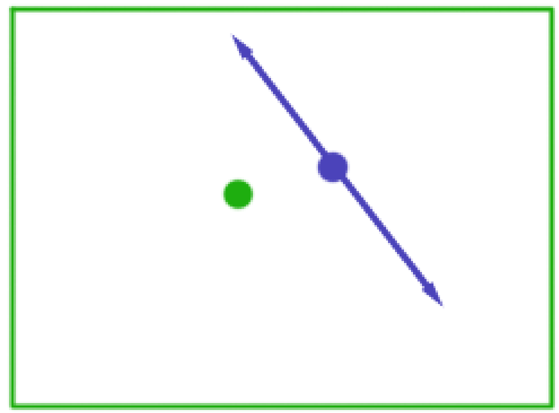


image t, point i

Initialization

- direction vectors $\{\overrightarrow{C_{it}}\}$

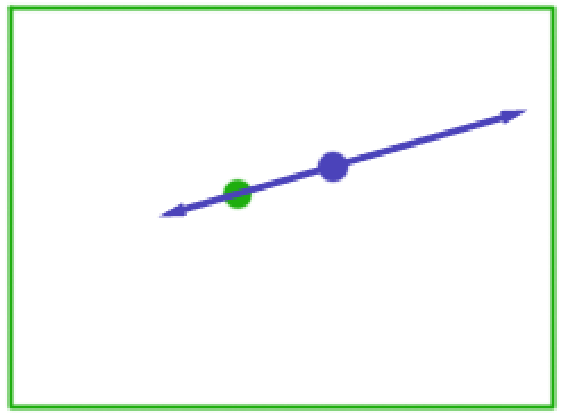
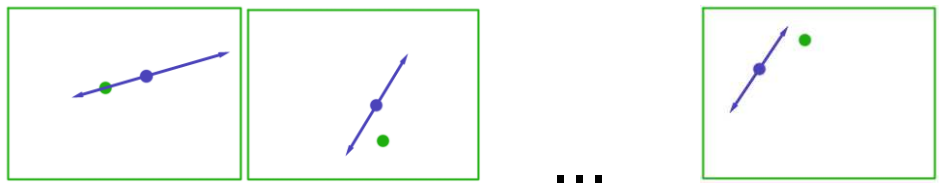


image t, point i

Initialization

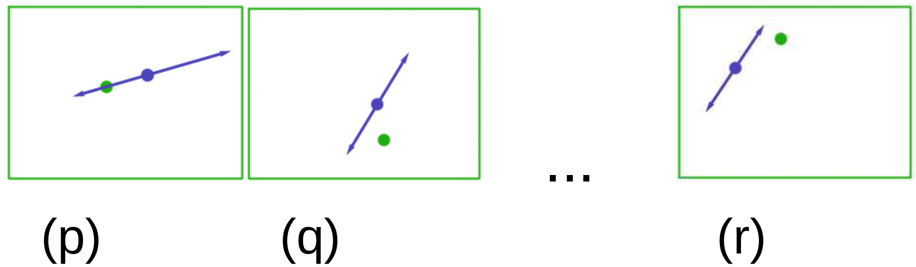
- direction vectors $\{\overrightarrow{C_{it}}\}$



(m) (n) (o)

Initialization

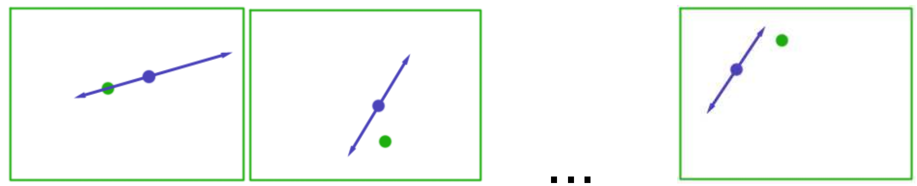
- direction vectors $\{\overrightarrow{C}_{ik}\}$



- magnitudes $\{b_{ik}\}$, basis weights $\{a_{kt}\}$

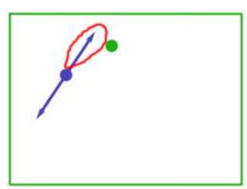
Initialization

- direction vectors $\{\overrightarrow{C}_{ik}\}$



(s) (t) (u)

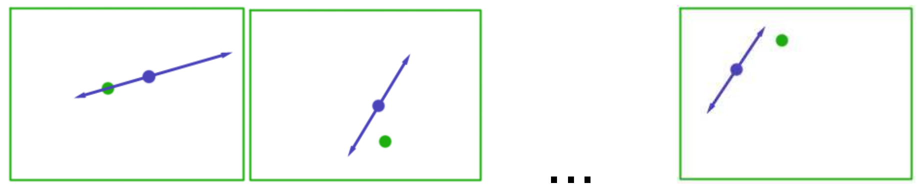
- magnitudes $\{b_{ik}\}$, basis weights $\{a_{kt}\}$
  - each point in each image constrains one a and one b



a function of $a_{kt}b_{ik}$

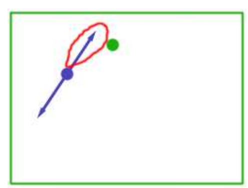
Initialization

- direction vectors $\{\overrightarrow{C}_{ik}\}$



(v) (w) (x)

- magnitudes $\{b_{ik}\}$, basis weights $\{a_{kt}\}$
  - each point in each image constrains one a and one b



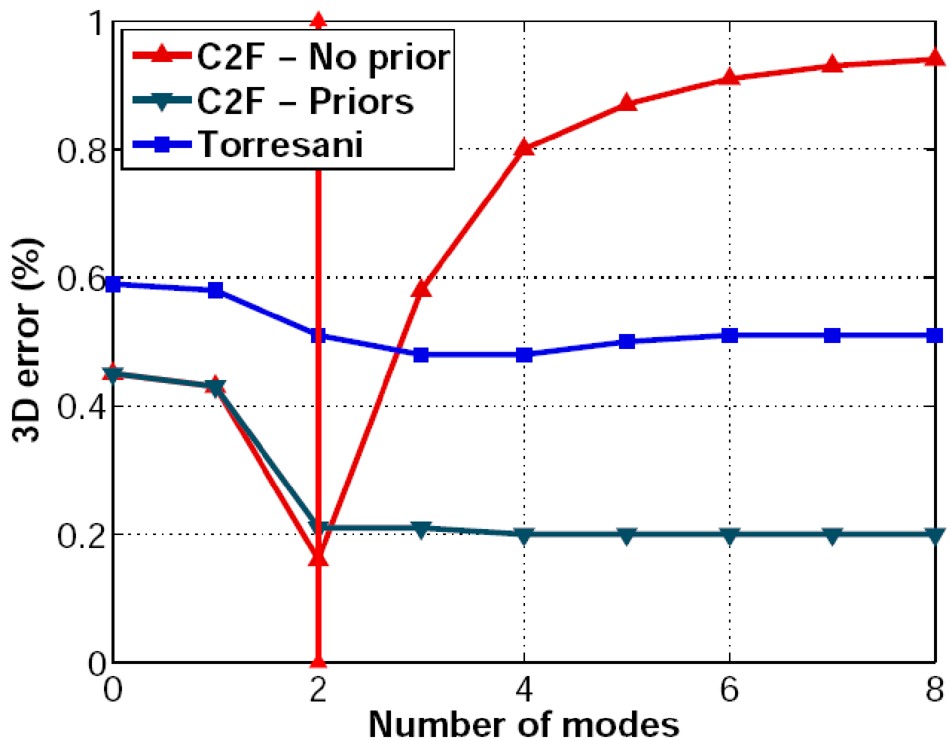
a function of $a_{kt}b_{ik}$

  - need one more constraint for each kth basis; $|\overrightarrow{a}_k| = 1$ will do

Outline

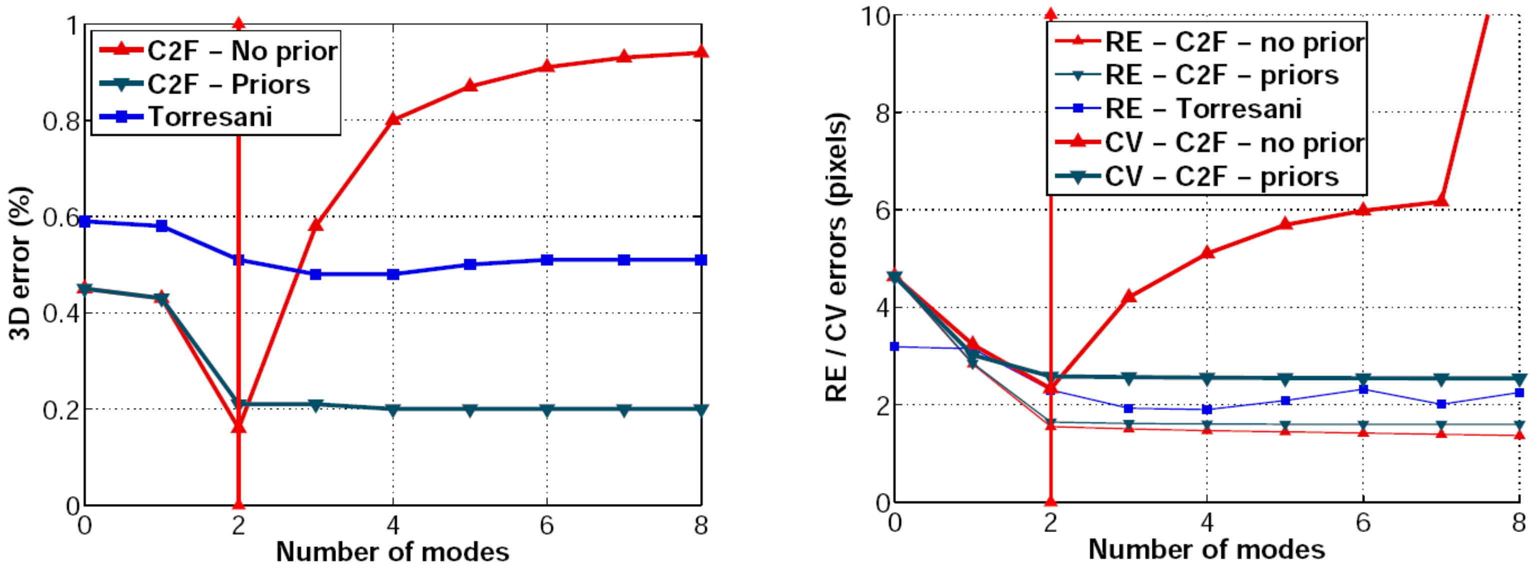
- Problem formulations
- Torresani et al
- Bartoli et al
  - Algorithm
  - Results
  - Summary
- Comparison

Results: Dependence on K



(Candide face dataset)

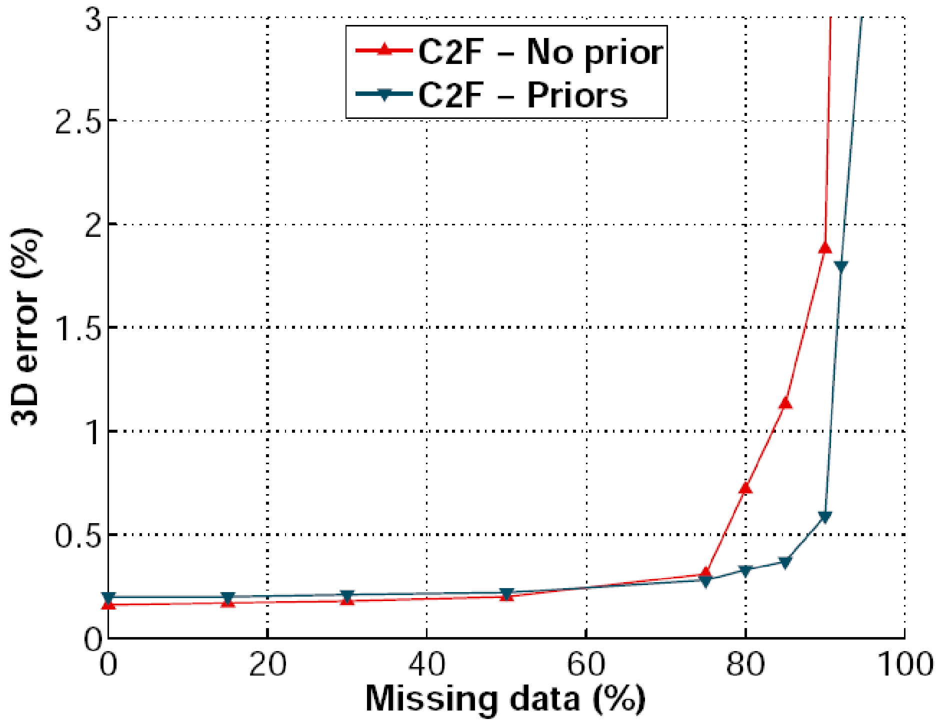
Results: Effect of Cross-Validation



(y) (z)

(Candide face dataset)

Results: Missing Data



(Candide face dataset)

Outline

- Problem formulations
- Torresani et al
- Bartoli et al
  - Algorithm
  - Results
  - Summary
- Comparison

Contributions

Torresani et al (2000, 2001, 2003)

- linear subspace shape model
- max-likelihood algorithm

Contributions

Torresani et al (2000, 2001, 2003)

- linear subspace shape model
- max-likelihood algorithm

Bartoli et al (2008)

- efficient solution
  - no costly refinement (of anything)
  - independence approximation
- cross-validation as a termination criterion

Outline

- Problem formulations
- Torresani et al
- Bartoli et al
- Comparison

Head-to-Head

Torresani	Bartoli
• probabilistic model + max likelihood solution	• approximate solution with regularization; each point independent

Head-to-Head

Torresani	Bartoli
• probabilistic model + max likelihood solution	• approximate solution with regularization; each point independent
• must choose σ_{noise} carefully	• must choose smoothness-term weights λ, κ carefully

Head-to-Head

Torresani	Bartoli
• probabilistic model + max likelihood solution	• approximate solution with regularization; each point independent
• must choose σ_{noise} carefully	• must choose smoothness-term weights λ, κ carefully
• refine camera transformations each iter	• solve for camera transformations once (bad)

Head-to-Head

Torresani	Bartoli
• probabilistic model + max likelihood solution	• approximate solution with regularization; each point independent
• must choose σ_{noise} carefully	• must choose smoothness-term weights λ, κ carefully
• refine camera transformations each iter	• solve for camera transformations once (bad)
• predefine K, let EM make some weights small	• cross-validate after adding each basis shape