Data-Driven, Non-Parametric Inference of Multiple Structures in N-D using Tensor Voting

Gérard Medioni
Philippos Mordohai

Institute for Robotics and Intelligent Systems
University of Southern California

Motivation

- Manifold inference in high-D spaces has applications in:
  - Machine learning
  - Data mining
  - Function approximation
- Express manifold inference as perceptual organization problem
- The tensor voting framework allows the inference of structures (including *non-manifolds*) in N-D

Overview

- Introduction
- The Tensor Voting Framework
- Structure Inference in N-D
- Preliminary Results in N-D
- Conclusions

Problem Statement

- Inference of structures in high dimensional spaces
- Training data:
  - Observations of multivariate data
  - Observations of commands and responses of systems with many degrees of freedom
- Properties:
  - Outlier rejection
  - Local dimensionality estimate
  - Local structure orientation → ability to interpolate and extrapolate

Machine Learning

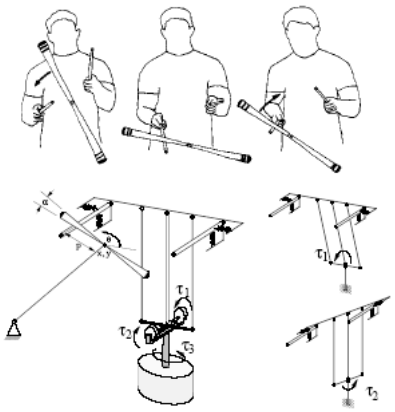
- Learning as function approximation from samples
- Interpolation
- Classification with respect to structures inferred from training data

Data Mining

- Extract features from data
  - Filter responses
  - Statistics, moments
  - Presence of certain attributes
- Objects of same class form clusters (structures) in feature space surrounded by noise (data points not in cluster)

Example: Inverse Kinematics

- Schaal *et al.* 2000
- Robotic arm learns "devil-sticking"
- State is 5-D vector: impact position, angle, velocities of center and angular velocity
- Command is 10-D vector
- Robot learns mapping between current state and command and next state

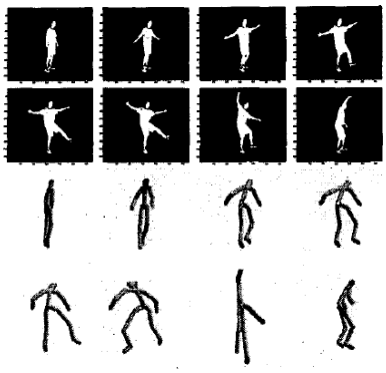


Example: Humanoid Robots

- Schaal *et al.* 2001
- Humanoid robot learns tennis forehand from teacher
- Teacher wears exoskeleton
- Motion model learned as function approximation

Example: Computer Vision

- Rosales *et al.* CVPR 03
- Estimation of joint positions from silhouettes
  - Compute 7 moments from binary silhouette images
  - Label junctions in training images (40 d.o.f.)
  - Estimate junction positions using moments of query images as input to a neural network



Example: Data Mining in Videos

- Sivic and Zisserman 2004
- Extract viewpoint invariant features from video frames
- Cluster spatial configurations of features (before classification)
- Most frequent configuration should correspond to principal actors or objects

Observation

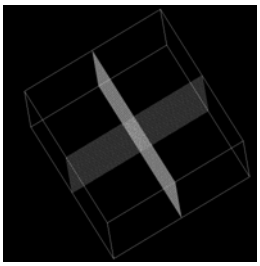
- In all cases: inputs are vectors or points in an N-D space
- Observations explicitly represent a manifold (smooth function)
- New states given commands, or commands to achieve desired state also lie on manifold
- Can be found by estimating manifold's tangent or normal subspace locally

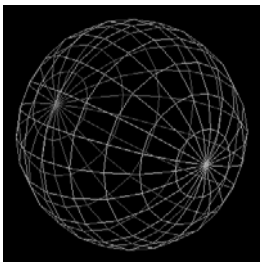
Dimensionality Estimation

- Data in N-D spaces lie on structures of dimension less than N
- Estimate local dimensionality of data
- Allows:
  - Classification of probe data
  - Interpolation based on existing data

Dimensionality Reduction

- Embed data into lower dimensional space
- Decrease storage and computation requirements
- *Not always possible*





Open and Closed Structures

Boundaries and holes of structures are often meaningful

- Represent illegal configurations
- Hard to handle with some models

Local vs. Global Models

- Global models:
  - Single model to fit entire data set
- Local models:
  - Linear
  - Splines
  - K-nearest neighbors
  - Weighted average
  - Locally weighted regression

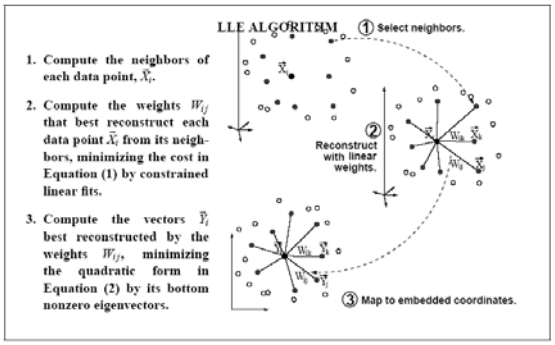
Global Models

- Principal Component Analysis (PCA)
  - Compute projection of maximum variance
- Assumes entire dataset is single *linear* manifold
- Local variations also exist
- Multidimensional Scaling (MDS) can operate in non-Euclidean spaces

Local Models

- Fit simple local models to data
- Operate in fixed neighborhoods or until a fixed number of neighbors is reached
- Suitable for incremental implementations
- Easy cross-validation

Local Models: Locally Linear Embedding (LLE)



- Roweis and Saul, 2000
- Unsupervised learning
- Neighborhood preserving, low dimensional embeddings
- Linear reconstruction based on neighbors

Local Models: Isomap

- Tenenbaum *et al.* 2000
- Construct graph
- Estimate geodesic distance by graph distance
- Preserves manifold's intrinsic geometry
- Unlike LLE:
  - Can handle holes
  - Fails for non-convex manifolds

Local Models: Locally Weighted Learning (LWL)

- Atkenson *et al.* 1999
- Linear fitting within region of validity
- One learning module for each degree of freedom
  - Estimated by locally weighted regression

Why Tensor Voting

- Simultaneous inference of all structure types and their dimensionality
- Non-parametric structures, non-manifolds
- Open or closed structures
- Structures of varying dimensionality
- Robustness to noise
- Local processing

Issues in N-D

- Inadequate data

→ investigate performance wrt amount of data

- Evaluation of prediction quality

→ cross-validation

- Noise removal

→ major strength of Tensor Voting

Issues in N-D

- Automatic tuning of parameters

→ one important free parameter: scale

- Selection of distance metric

→ use statistical methods

- Efficiency

→ time complexity is *O(n logn)*

Overview

- Introduction
- The Tensor Voting Framework
- Structure Inference in N-D
- Preliminary Results in N-D
- Conclusions

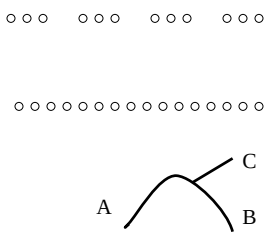
Motivation

- Computational framework to address a wide range of computer vision problems
- Computer Vision attempts to infer scene descriptions from one or more images
  - Primitives and constraints might vary from problem to problem
  - Many problems can be formulated as *perceptual organization* problems in an appropriate space

Perceptual Organization

Gestalt principles:

- Proximity
- Similarity
- Good continuation

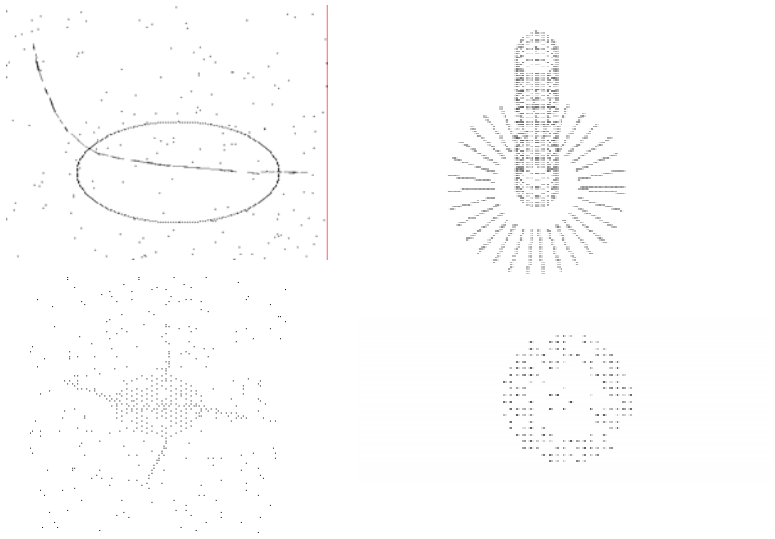


The Smoothness Constraint

Matter is cohesive ⇒ Smoothness

Difficult to implement, as true “almost everywhere” only

Examples



The Tensor Voting Framework

- Data Representation: Tensors
- Constraint Representation: Voting fields
  - enforce smoothness
- Communication: Voting
  - non-iterative
  - no initialization required

Our Approach in a Nutshell

- Each input site propagates its information in a neighborhood
- Each site collects the information cast there
- Salient features correspond to local extrema of saliency

Properties of Tensor Voting

- Non-Iterative
- Can extract all features simultaneously
- One parameter (scale)
- Non-critical thresholds
- Efficient

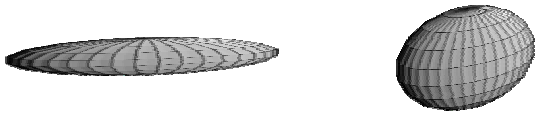
Second Order Symmetric Tensors

- Equivalent to:
  - Ellipse
    - Special cases: “ball” and “stick” tensors
  - 2x2 matrix

Second Order Symmetric Tensors

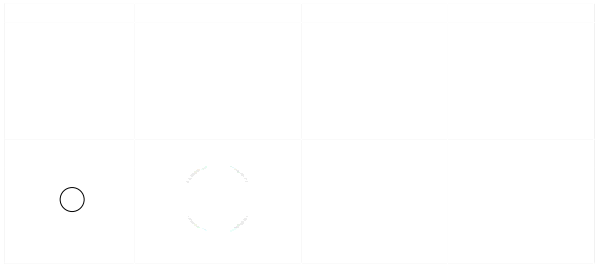
Properties captured by second order symmetric Tensor

– shape: orientation certainty

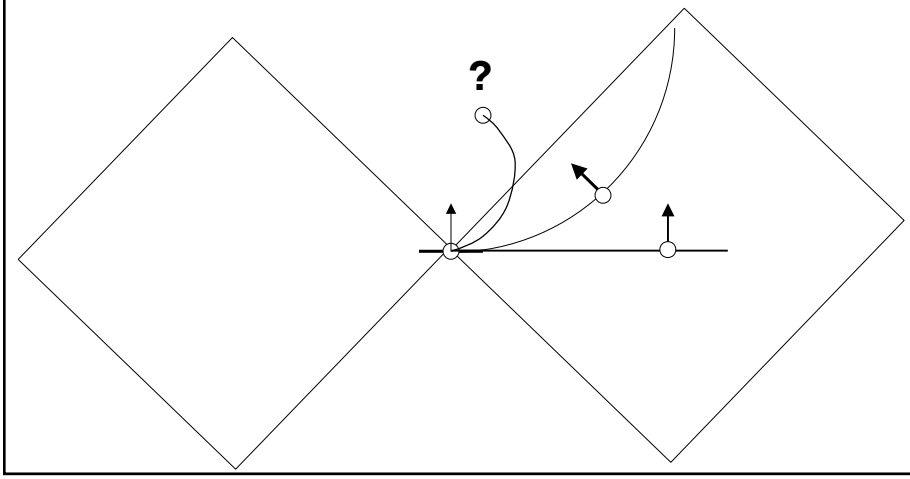


– size**:** feature saliency

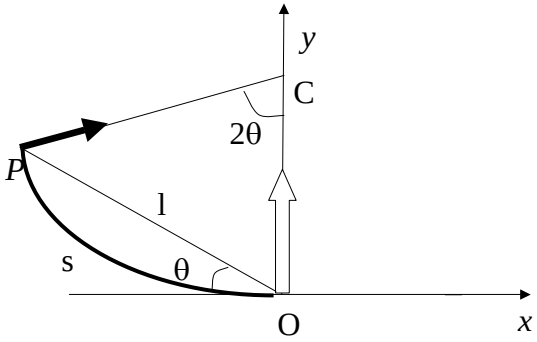
Representation with Second Order Symmetric Tensors



Design of the Voting Field

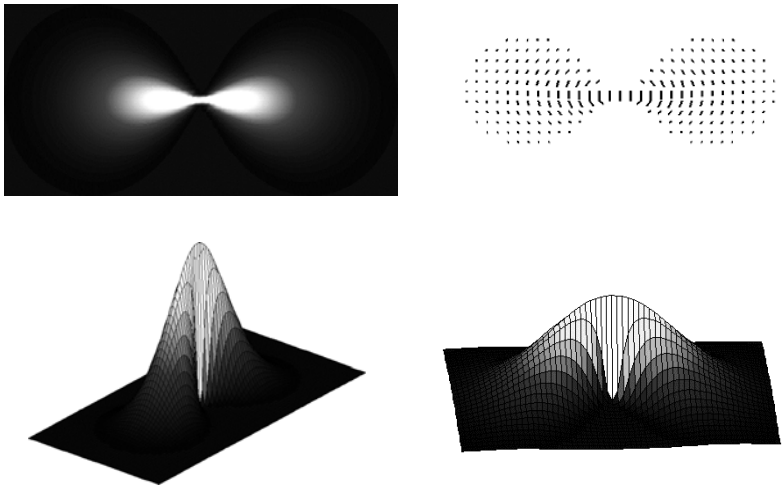


Saliency Decay Function

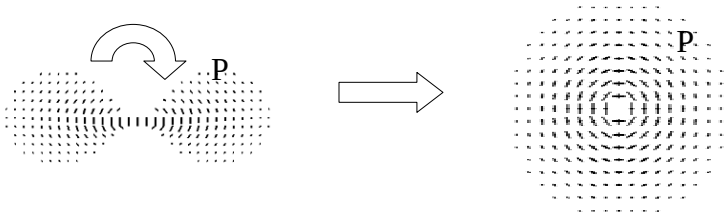


- Votes attenuate with length of smoothest path
- Straight continuation is favored over curved

Fundamental Stick Voting Field



2-D Ball Field



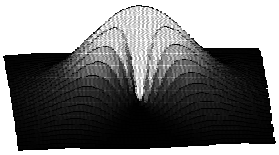
S(P) **B**(P)

Ball field computed by integrating the contributions of rotating stick

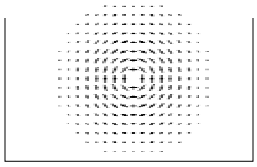
2-D Voting Fields

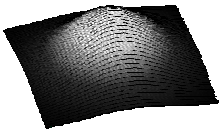
Each input site **propagates** its **information** in a **neighborhood**

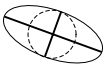
— votes with



○ votes with





 votes with  + 

Vote Accumulation

Each site accumulates second order votes by tensor addition:

○ + ○ = 

○ + — = 

— + — =

 +  = 

Results of accumulation are usually *generic tensors*

Second Order Vote Interpretation

Salient features correspond to local extrema of saliency

At each site

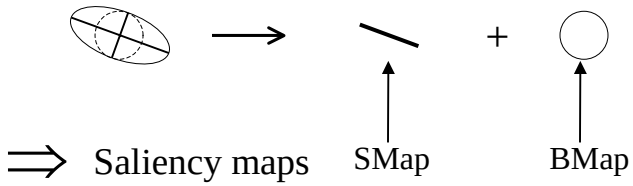
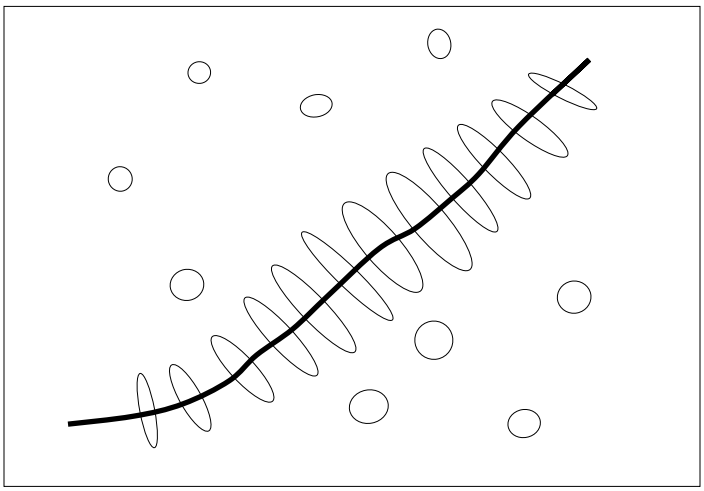
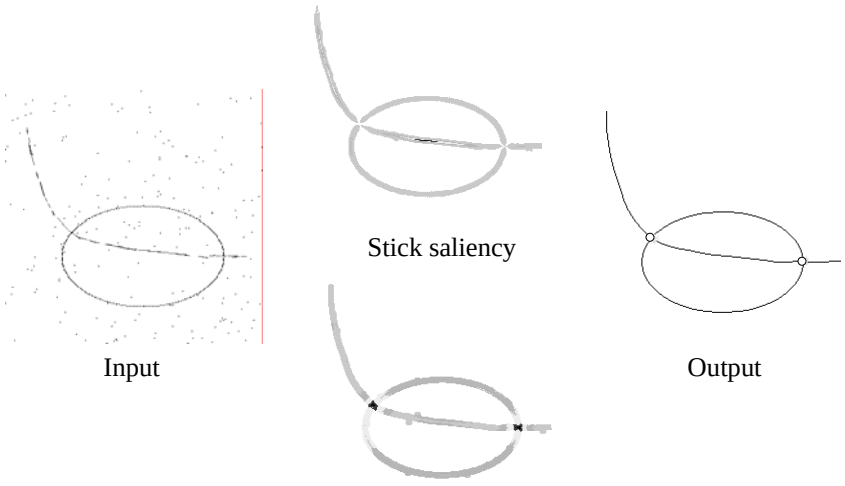


Illustration of Tensor Voting



Example in 2-D



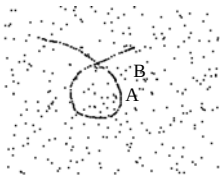
Scale of Voting

- The Scale of Voting is the single critical parameter in the framework
- Essentially defines size of voting neighborhood
  - Gaussian decay has infinite extend, but it is cropped to where votes remain meaningful (e.g. 1% of voter saliency)

Scale of Voting

- The Scale is a measure of the degree of *smoothness*
- Smaller scales correspond to small voting neighborhoods, fewer votes
  - Preserve details
  - More susceptible to outlier corruption
- Larger scales correspond to large voting neighborhoods, more votes
  - Bridge gaps
  - Smooth perturbations
  - Robust to noise

Sensitivity to Scale



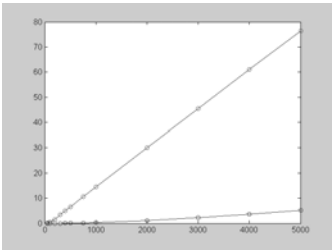
Input

$\sigma = 50$

$\sigma = 500$

$\sigma = 5000$

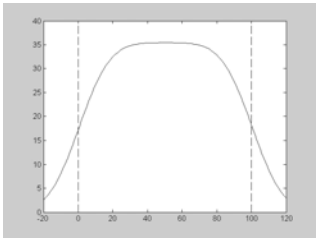
Input: 166 un-oriented inliers, 300 outliers
Dimensions: 960x720
Scale $\in$ [50, 5000]
Voting neighborhood $\in$ [12, 114]



Curve saliency as a function of scale
Blue: curve saliency at A
Red: curve saliency at B

Boundaries?

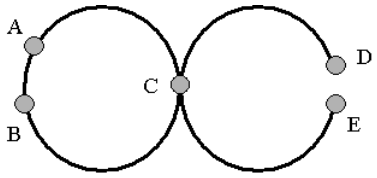
Input



Saliency

No clear way to detect the endpoints of the curve with second order Tensor Voting

Need for First Order Information



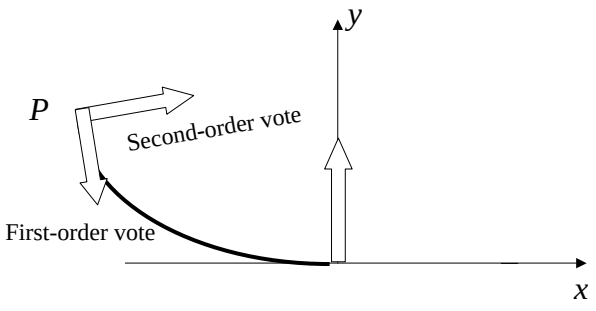
- Second order Tensor Voting can infer curves and junctions
- Second order tensors at A, B, D and E are very similar, but A and B are *very different* from D and E
- Key property of endpoints: all neighbors are on same side

Polarity Vectors

- Representation augmented with Polarity Vectors
- Vectors are first order tensors
- Sensitive to direction from which votes are received
- Exploit property of boundaries to have all their neighbors on the same side of the half-space

First Order Voting

- Votes are cast along the tangent of the smoothest path
- Vector votes instead of tensor votes
- Accumulated by vector addition



First Order Voting Fields

- Magnitude is the same as in the second order case

- First-order Ball field can be derived from the first-order Stick Field after integration

Endpoint Inference

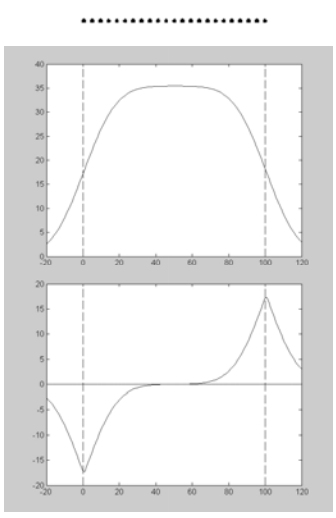
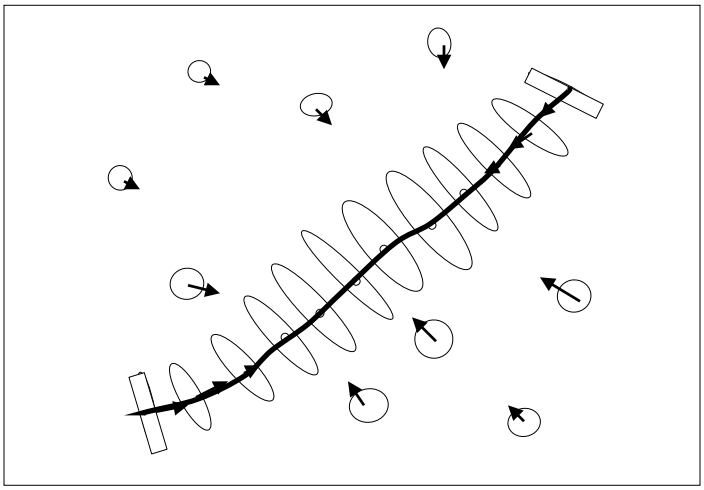
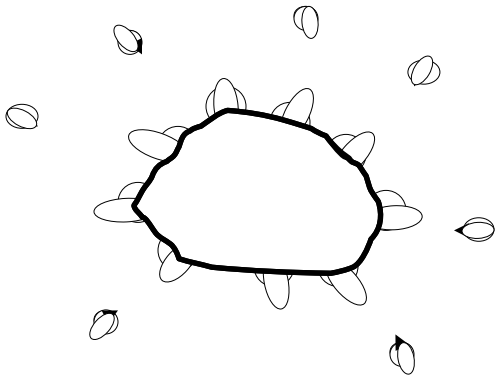


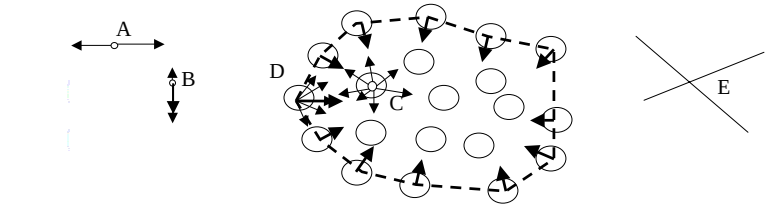
Illustration of First Order Voting



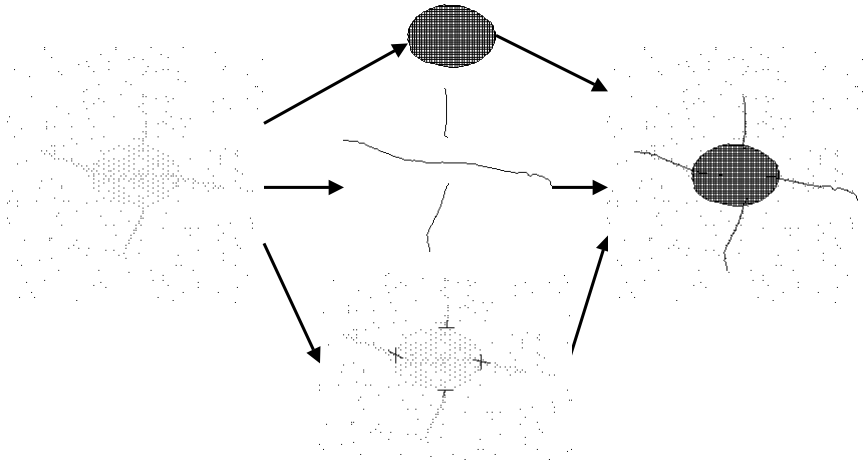
Region Inference



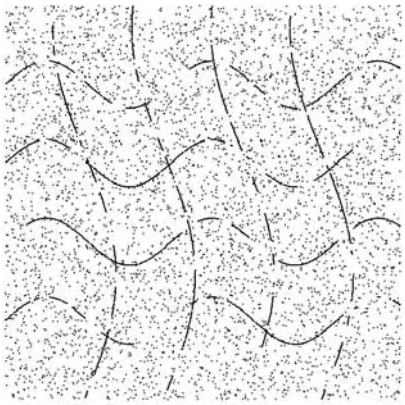
Structure Inference in 2-D



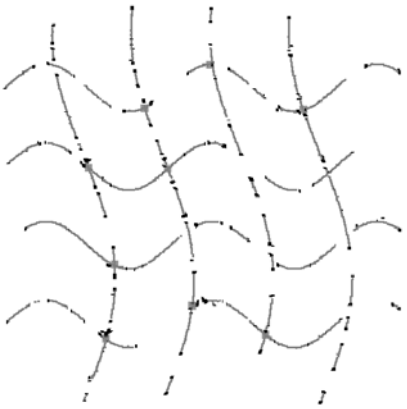
Example in 2-D



Results

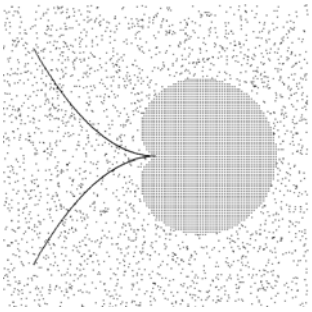


Input



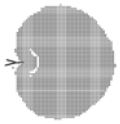
Gray: curve inliers
Black: curve endpoints
Squares: junctions

Results



Input

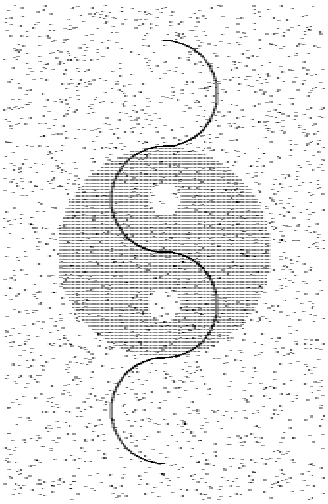
Curve inliers



Region inliers

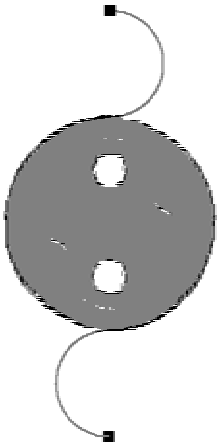
Region boundaries

Results



Input

Curves and endpoints only



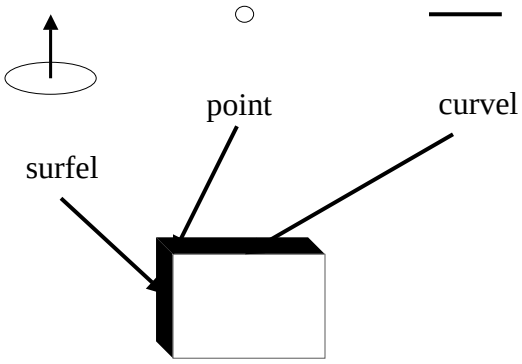
Curves, endpoints and regions

3-D Tensor Voting

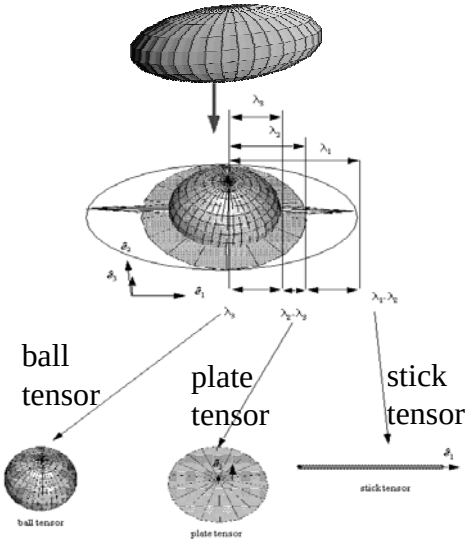
- Representation: **3-D Tensors**
- Constraints: **3-D Voting Fields**
- Data communication: **Voting**

3-D Tensors

The input may consist of



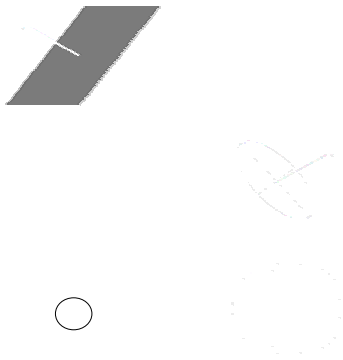
3-D Tensor Decomposition



3 eigenvalues
(λ_{max} λ_{mid} λ_{min})

3 eigenvectors
($\mathbf{V}_{max}$ $\mathbf{V}_{mid}$ $\mathbf{V}_{min}$)

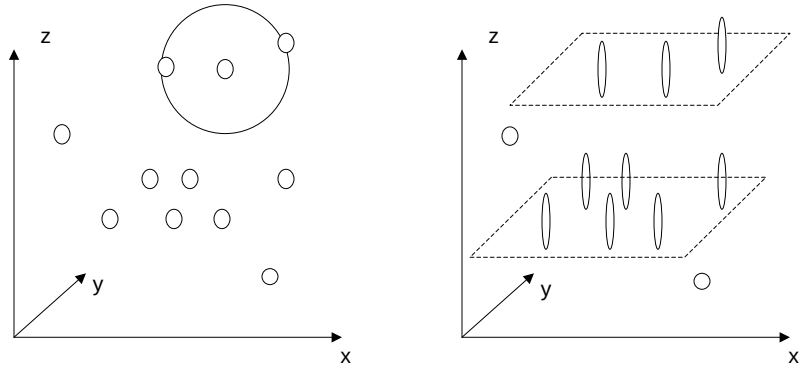
Representation



Tensor Voting in 3-D

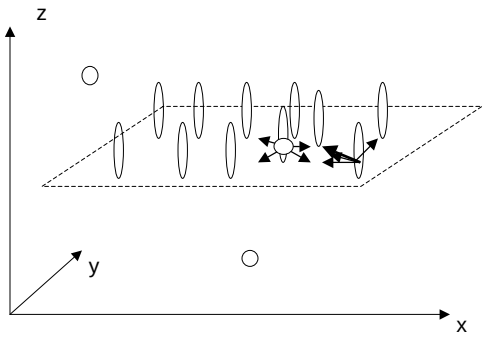
- 2-D stick fields are cuts of the 3-D ones
  - 3-D first and second order stick fields derived by rotating the *fundamental 2-D stick field*
- Plate and Ball fields derived by integrating contributions of rotating stick voter

Second Order Voting



- Tokens in the same structure reinforce each other
- Isolated tokens receive little or contradicting support

First Order Voting

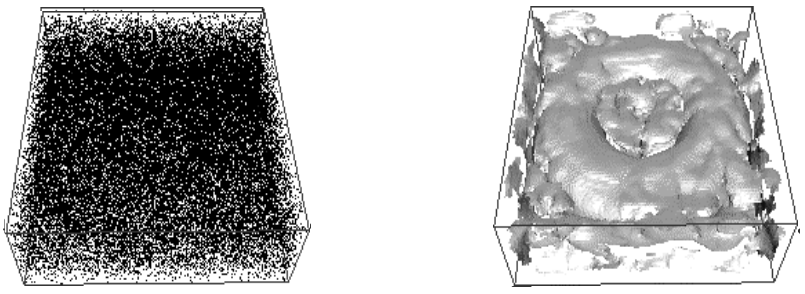


- Tokens in the interior of a structure receive first order votes from all directions
- Tokens at boundaries receive first order votes from one side of a half-space

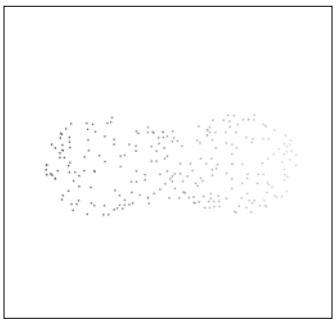
Interpretation of Resulting Tensors

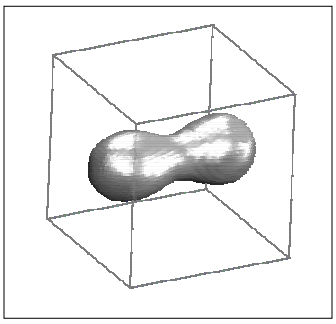
Graceful Degradation with Noise

1200% noise

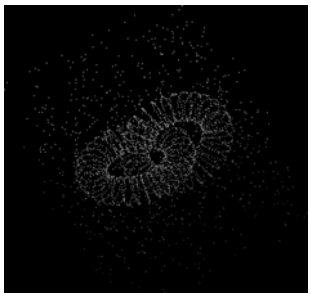


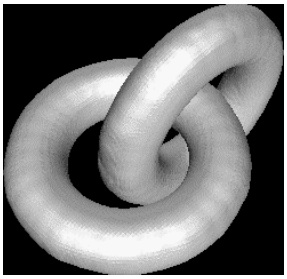
Examples



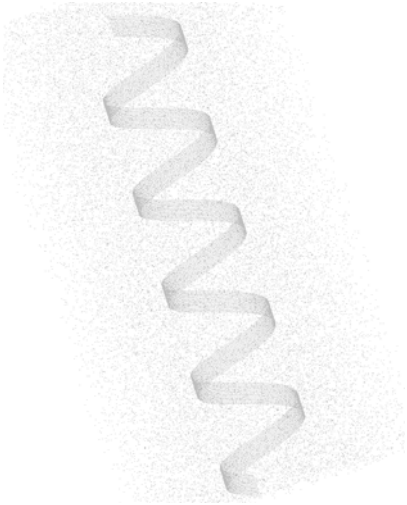


Examples

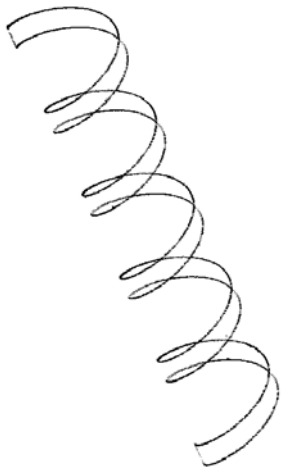




Examples

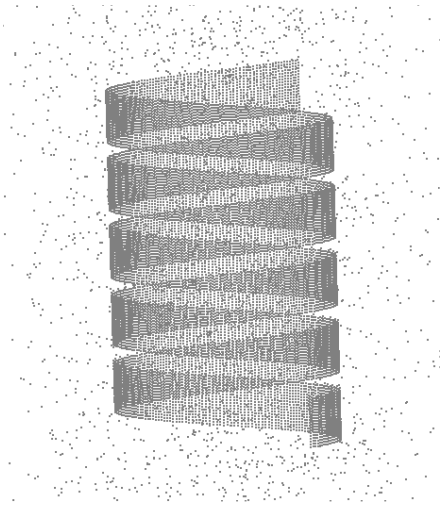


Input

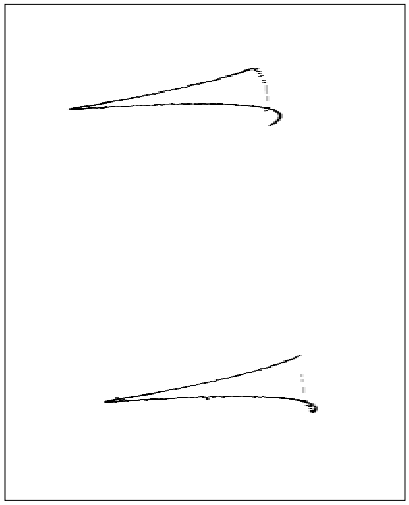


Surface Boundaries

Examples



Input



Surface Boundaries

Overview

- Introduction
- The Tensor Voting Framework
- Structure Inference in N-D
- Preliminary Results in N-D
- Conclusions

Tensor Voting in N-D

Direct generalization from 2-D and 3-D cases

- Tensors become second order, N-dimensional, symmetric, non-negative definite
- Polarity vectors become N-D vectors
- There are N+1 structure types (0-D junction to N-D hyper-volume)
- N second order and N first order fields are required

Issues in N-D

- Space must be Euclidean
  Distances in voting space must be meaningful
- Data structures
  Efficient search for neighbors: use Approximate Nearest Neighbor k-d Trees
- Voting fields
  Pre-computation becomes inefficient when grid positions are comparable to number of tokens

Voting Fields in N-D

- Arbitrary tensors decomposed in N basic tensors
- Vote generation from unit stick is the same
  – Voter, receiver and voting stick define a 2-D plane in any dimension
- Other fields can be derived as shown in previous sections
  – Large time and space requirements

Voting Fields in N-D

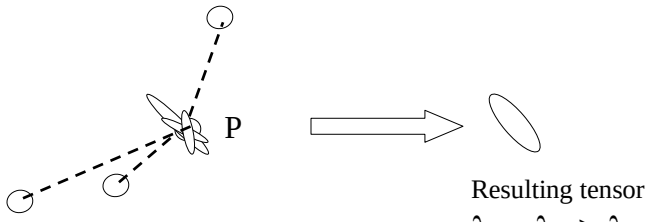
- Easy to handle hyper-surfaces and un-oriented data
  - Stick field is 2-D in any dimension
  - Ball field is always 1-D (function of distance)
- Other fields may be impractical to pre-compute if N is too high
  - N∗N elements required at each position
  - Huge storage requirements
- Exact votes cannot be computed in closed form
  - Computation by integration is time consuming
  - Many computed votes may not be used

Practical Tensor Voting in N-D

- Cast votes without uncertainty component
- Inaccurate but directly computable given voting tensor and receiver position
  - E.g. a 3-D plate tensor (curve) could cast purely plate votes
- Consider “lazy voting”
  - Cast votes only at query points as needed

Vote Accumulation

- By tensor addition
- Even with un-oriented inputs, dominant orientations emerge



Vote Analysis

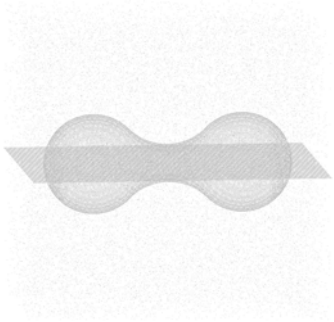
- Compute eigensystem of tensors
- Saliency of (N-1)-D manifold is: $\lambda_1 - \lambda_2$
  - Hyper-surface: $\lambda_1 - \lambda_2$
  - Curve: $\lambda_N - \lambda_{N-1}$
- Maximum saliency is estimate of local dimensionality
- Normals and tangents are given by eigenvectors
  - Hyper-surface: 1 normal, N-1 tangents
  - Curve: N-1 normals, 1 tangent

Learning by Vote Analysis

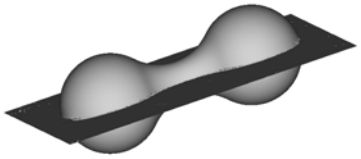
- At each point:
  - Dimensionality estimate
  - Normal subspace
  - Tangent subspace
- Linear constraints provided by local normal and tangent subspaces

  Derivatives can be estimated

Example: Dimensionality Detection



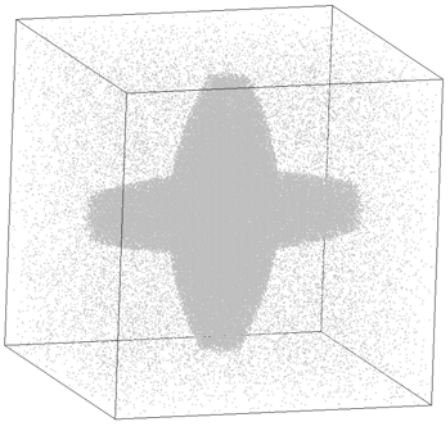
Input



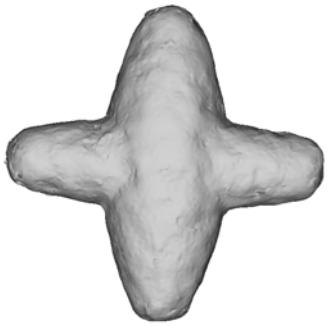
Surfaces

Surface Intersections

Example: Volume Boundaries

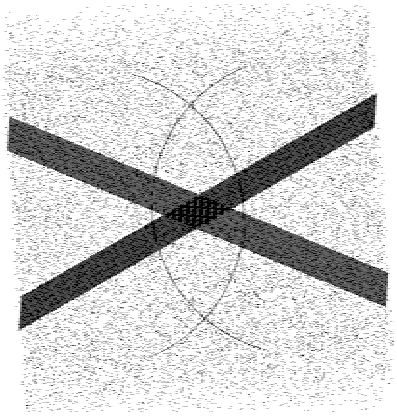


Input

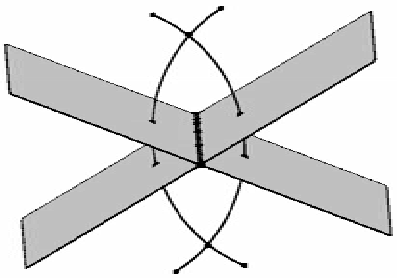


Volume Boundaries

Example: Varying Type of Structures in Clutter



Input



Surfaces - Surface Boundaries – Surface Intersections
Curves – Endpoints - Junctions

Tensor Voting in N-D for Computer Vision Applications

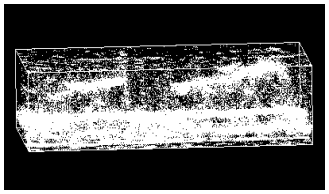
- Motion segmentation in 4-D space (x, y, v_x, v_y)
- Epipolar geometry estimation in 4-D Joint Image Space
- Affine motion parameter estimation in 4-D space
- Epipolar geometry estimation in 8-D space

Motion Segmentation

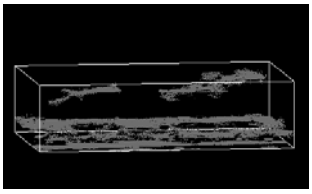
- Problem: Inference of segments of coherent motion
- Pixel correspondences encoded in 4-D (x, y, v_x, v_y) space
  - Correct correspondences form salient 2-D manifolds
  - Wrong ones are outliers
- Salient 2-D manifolds inferred after Tensor Voting

Motion Segmentation Example

Input



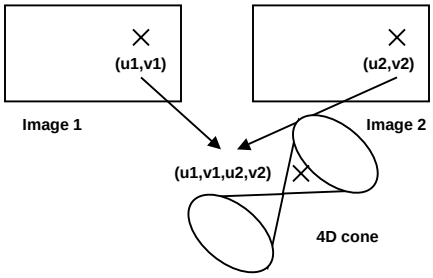
Candidate matches



Inliers

Motion boundaries

Epipolar Geometry Estimation in Joint Image Space



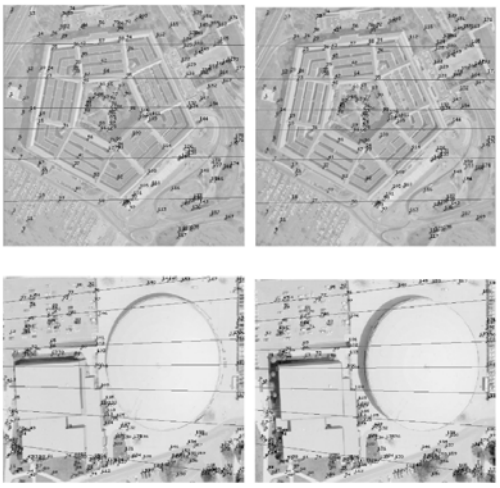
- The epipolar geometry defines a 2-D point cone in the 4-D joint image space (Anandan 2000)

Epipolar Geometry Estimation

Epipolar Geometry Estimation in 8-D

- Fundamental matrix (**F**) describes geometry of two images of a scene taken from different cameras
- **F** is 3*3 and homogeneous
- Corresponding pixels satisfy: $\mathbf{x}^T\mathbf{F}\mathbf{x}$
- 8 linear constraints per pixel correspondence
- Correct correspondences lie on hyper-plane in 8-D

Epipolar Geometry Estimation in 8-D: Results

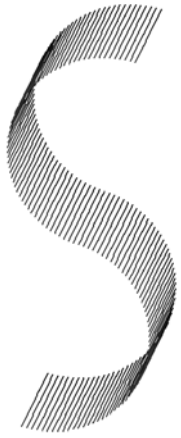


Overview

- Introduction
- The Tensor Voting Framework
- Structure Inference in N-D
- Preliminary Results in N-D
- Conclusions

Synthetic ***S*** in 3-D

- 3-D manifold
- 4960 points
- Voting scales: σ^2 ranges from 50 to 5000
- Field reach: 14 to 136
- Processing time: 47sec to 2min 37sec



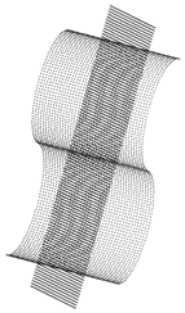
Orientation Estimation Accuracy

- ***S*** consists of two cylindrical parts
- All points classified as surfaces ($\lambda_1 - \lambda_2 > \lambda_2 - \lambda_3$ AND $\lambda_1 - \lambda_2 > \lambda_3$)
- Analytic solution for surface normal is easy to find

Scale (σ^2)	Average error (deg)
50	1.92
100	1.92
200	1.60
300	1.45
400	1.35
500	1.28
750	1.17
1000	1.14
2000	1.24
3000	1.47
4000	1.72
5000	1.99

S and Plane

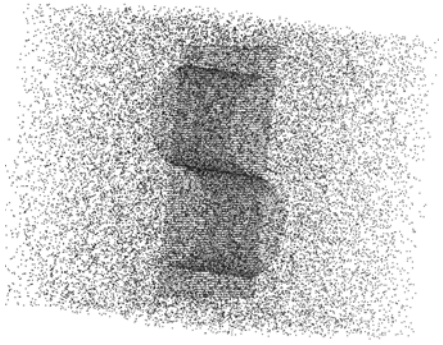
- Added plane splitting ***S***
- Angular error 0.49 degrees for 10343 points at σ^2=1000



Detected 761 curvels

S and plane with Noise

Add 3 random points for each inlier

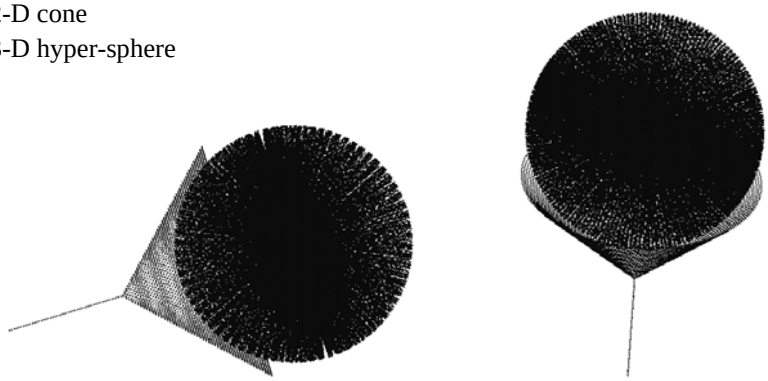


Salient points

Varying Dimensionality: Input

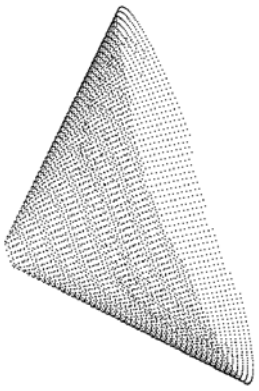
Input: un-oriented points in 4-D

- 1-D line
- 2-D cone
- 3-D hyper-sphere



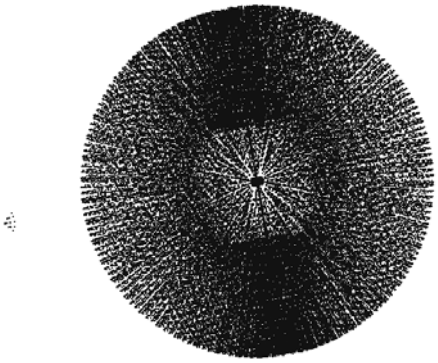
Varying Dimensionality: Results

$\lambda_3 - \lambda_4$ is max



$\lambda_2 - \lambda_3$ is max

Varying Dimensionality: Results



λ_1- λ_2 is max

Overview

- Introduction
- The Tensor Voting Framework
- Structure Inference in N-D
- Preliminary Results in N-D
- Conclusions

Comparison with Local Methods

Tensor Voting can handle:

- Non-linear manifolds
- Non-convex manifolds
- Holes
- Non-manifolds
- Multiple structures of different dimensionality
- Large numbers of observations (up to millions)

Comparison with LLE and Isomap

- Reconstruction properties of LLE comparable to Tensor Voting with no curvature attenuation
- Geodesic distances approximated by graph distances in Isomap and by circular arcs in Tensor Voting
- No need to construct graph

Remaining Issues

- Storage of data in very high dimensional spaces
- Distance function
- Scale selection / inhomogeneous density
- Testing
- Domain