The Technion - Israel Institute of Technology
Department of Electrical Engineering
Signal and Image Processing Laboratory

3D Object Description and Classification by Implicit Polynomials

Hilla Ben-Yaacov

Supervisors:

Prof. David Malah and Dr. Meir Barzohar

June 2008

Outline

- Implicit Polynomials Fitting Algorithms
- Rotation Invariant Fitting
- Rotation Invariants Derivation
- 3D Objects/Faces Classification
- Experimental Results
- Summary and Future Work

2D Implicit Polynomials (IP)

$$P^n(x,y) = a_{00} +$$
$$+ a_{10}x + a_{01}y +$$
$$+ a_{20}x^2 + a_{11}xy + a_{02}y^2 +$$
$$+ \ldots + a_{n0}x^n + a_{0n}y^n$$

$$\underline{a} = \left[a_{00} \;\; a_{10} \;\; a_{01} \;\; a_{20} \;\; a_{11} \;\; a_{02} \;\; \ldots \;\; a_{n0} \;\; a_{0n} \right]^T$$

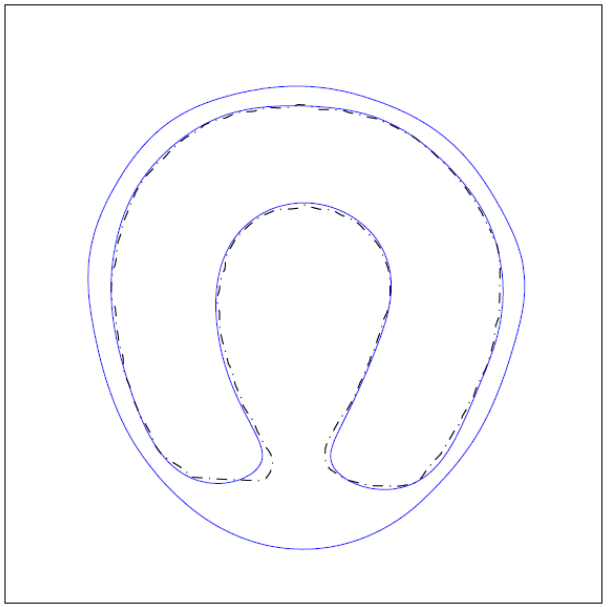
$$\underline{p}^n(x,y) = \left[1 \;\; x \;\; y \;\; x^2 \;\; xy \;\; y^2 \;\; \ldots \;\; x^n \;\; y^n \right]^T$$

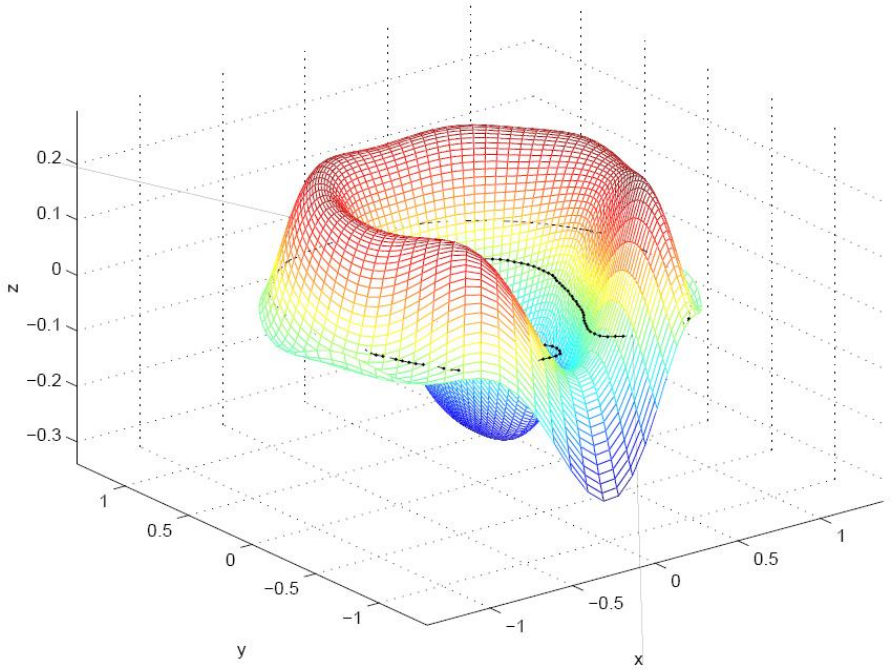
$$P^n(x,y) = \left[\underline{p}^n(x,y) \right]^T \underline{a}$$

2D Implicit Polynomials (IP)

- The IP zero set represents a 2D contour

$$P^n(x, y) = 0$$





3D Implicit Polynomials (IP)

$$P^n(x,y,z) = a_{000} +$$
$$+ a_{100}x + a_{010}y + a_{001}z +$$
$$+ a_{200}x^2 + a_{110}xy + a_{101}xz + a_{020}y^2 + a_{011}yz + a_{002}z^2 +$$
$$+ \ldots + a_{n00}x^n + a_{0n0}y^n + a_{00n}z^n$$

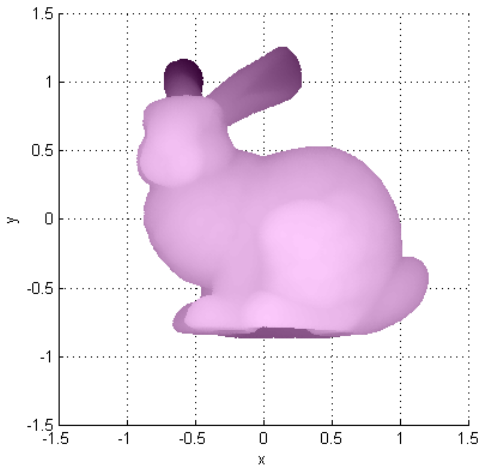
$$\underline{a} = \left[a_{000} \;\; a_{100} \;\; a_{010} \;\; a_{001} \;\; a_{200} \;\; a_{110} \;\; a_{101} \;\; a_{020} \;\; a_{011} \;\; a_{002} \;\; \ldots \;\; a_{n00} \;\; a_{0n0} \;\; a_{00n} \right]^T$$

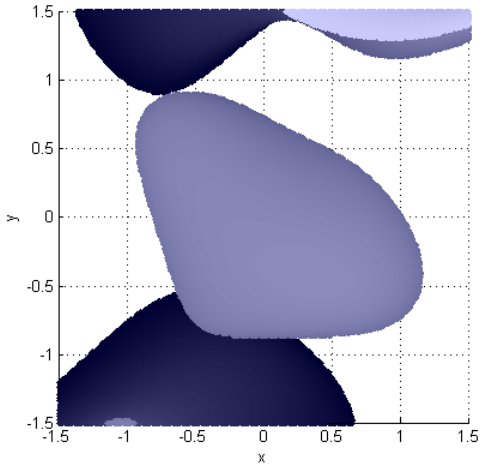
$$\underline{p}^n(x,y,z) = \left[1 \;\; x \;\; y \;\; z \;\; x^2 \;\; xy \;\; xz \;\; y^2 \;\; yz \;\; z^2 \;\; \ldots \;\; x^n \;\; y^n \;\; z^n \right]^T$$

$$P^n(x,y,z) = \left[\underline{p}^n(x,y,z) \right]^T \underline{a}$$

3D IP Fitting

- Input: $\left\{\left(x_i, y_i, z_i\right)\right\}\Big|_{i=1,\ldots,N}$
- Output: $P^n(x, y, z)$





4th degree
IP fit

Fitting Algorithms

- Minimizing the IP values at the data-points

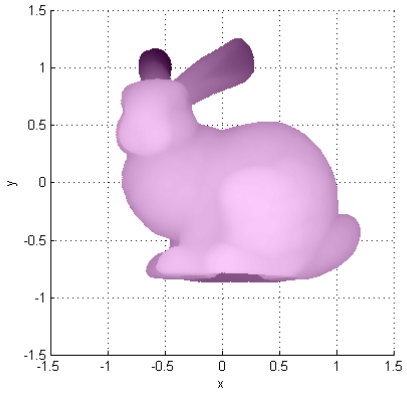
$$\min_{\underline{a}} \sum_{i=1}^{N} \left(P_{\underline{a}}^{n}(x_i, y_i, z_i) \right)^2$$

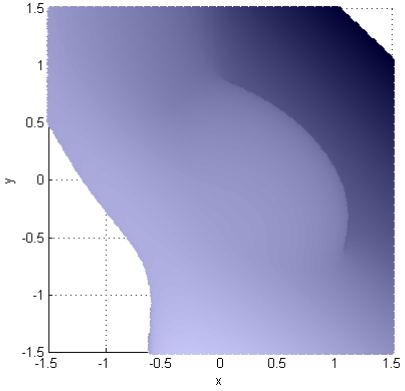
  - Least Squares problem
  - Unstable when the data-points are perturbed
  - The surface may intersect itself

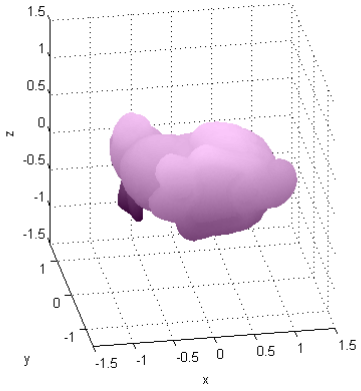
Fitting Algorithms

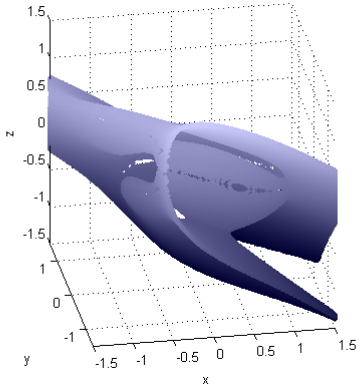
- Minimizing the IP values at the data-points

4^{th} degree IP fit



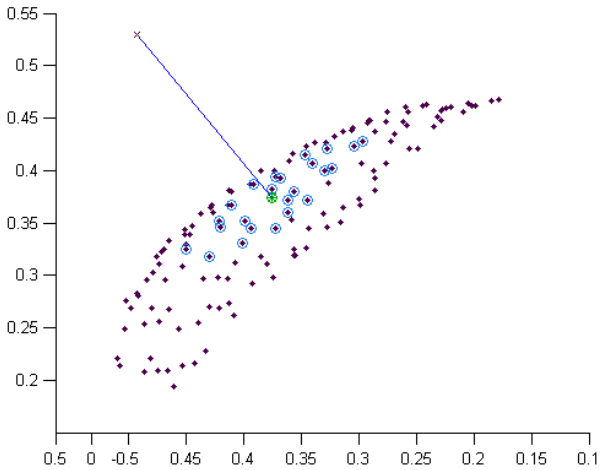






Fitting Algorithms

- Gradient1:
  - Fitting requirements:
    - The IP value at the data-points is 0.
    - The IP gradient direction at each data-point equals the local normal direction.
    - The gradient value is the same (e.g. equal to 1) at every data-point.



(Tarel et al., 2000)

Fitting Algorithms

- Gradient1:

$$P^n(x,y,z) = \left[\underline{p}^n(x,y,z)\right]^T \underline{a}$$

  - Define the matrices:

$$M_0 = \left[\underline{p}^n(x_1,y_1,z_1) \quad \underline{p}^n(x_2,y_2,z_2) \quad \ldots \quad \underline{p}^n(x_N,y_N,z_N)\right]^T$$

$$M_x = \left[\underline{p}_x^n(x_1,y_1,z_1) \quad \underline{p}_x^n(x_2,y_2,z_2) \quad \ldots \quad \underline{p}_x^n(x_N,y_N,z_N)\right]^T$$

$$M_y = \ldots$$

$$M_z = \ldots$$

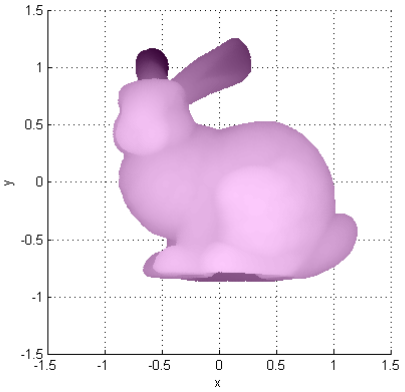
  - Least Squares problem:

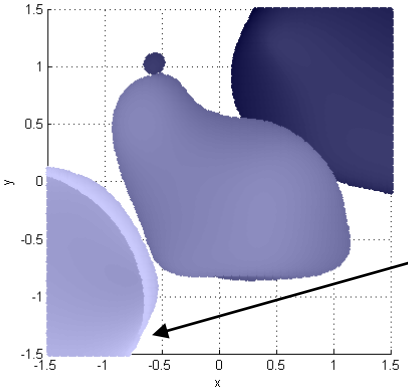
$$M = \begin{bmatrix} M_0 \\ M_x \\ M_y \\ M_z \end{bmatrix} \qquad M\underline{a} = \underline{b}$$

Fitting Algorithms

- Gradient1:
  - Improved stability when the data-points are perturbed

6th degree IP fit





Spurious zero-sets

Fitting Algorithms

- Min-Max/Min-Var:
  - Fitting requirements:
    - The IP value at the data-points is 0.
    - The IP gradient direction at each data-point equals the local normal direction.
    - Min-Max: Minimize the maximum error of the zero-set points
    - Min-Var: Minimize the error variance of the zero-set points

(Helzer et al., 2004)

Fitting Algorithms

- Min-Max:
  - Least Squares problem

$$M_0 = \left[\underline{p}^n(x_1, y_1, z_1) \quad \underline{p}^n(x_2, y_2, z_2) \quad \dots \quad \underline{p}^n(x_N, y_N, z_N) \right]^T$$

  - Modify M_x, M_y and M_z:

$$M_x = \left[\frac{\underline{p}_x^n(x_1, y_1, z_1)}{\left\| \underline{p}^n(x_1, y_1, z_1) \right\|_1} \quad \frac{\underline{p}_x^n(x_2, y_2, z_2)}{\left\| \underline{p}^n(x_2, y_2, z_2) \right\|_1} \quad \dots \quad \frac{\underline{p}_x^n(x_N, y_N, z_N)}{\left\| \underline{p}^n(x_N, y_N, z_N) \right\|_1} \right]^T$$

Fitting Algorithms

- Min-Var:
  - Least Squares problem

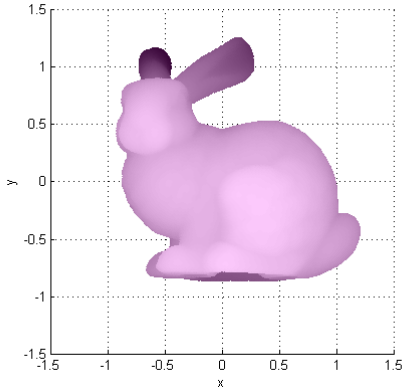
$$M_0 = \left[\underline{p}^n(x_1, y_1, z_1) \quad \underline{p}^n(x_2, y_2, z_2) \quad \dots \quad \underline{p}^n(x_N, y_N, z_N) \right]^T$$

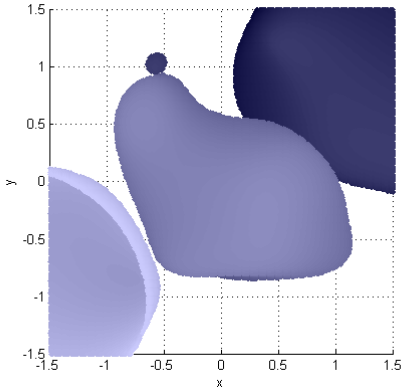
  - Modify M_x, M_y and M_z:

$$M_x = \left[\frac{\underline{p}_x^n(x_1, y_1, z_1)}{\left\| \underline{p}^n(x_1, y_1, z_1) \right\|_2} \quad \frac{\underline{p}_x^n(x_2, y_2, z_2)}{\left\| \underline{p}^n(x_2, y_2, z_2) \right\|_2} \quad \dots \quad \frac{\underline{p}_x^n(x_N, y_N, z_N)}{\left\| \underline{p}^n(x_N, y_N, z_N) \right\|_2} \right]^T$$

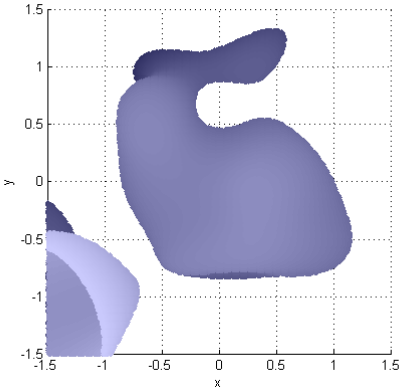
Fitting Algorithms

6^{th} degree IP fit

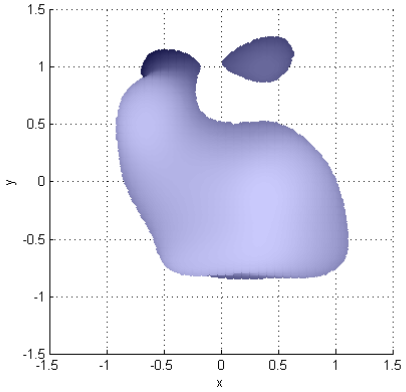




Gradient1



Min-Max

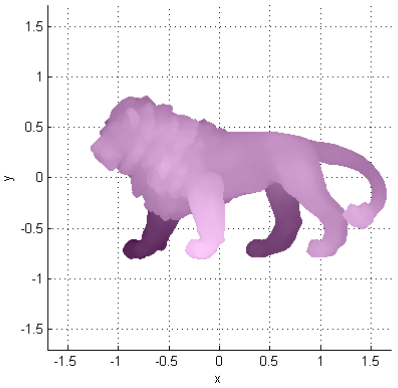


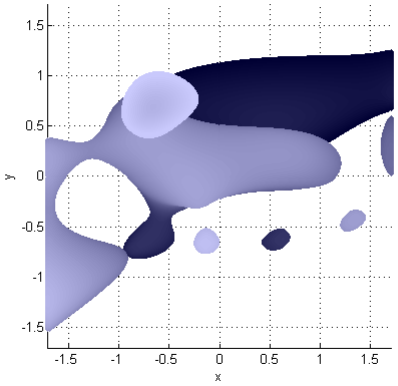
Min-Var

spurious zero-sets appear farther away from the desired zero-set

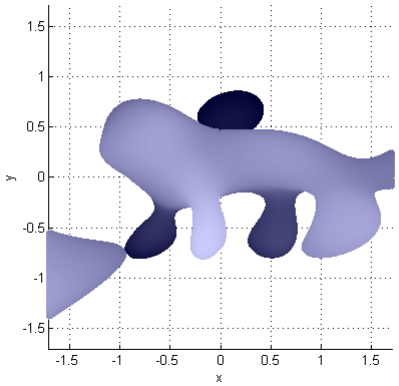
Fitting Algorithms

6^{th} degree IP fit

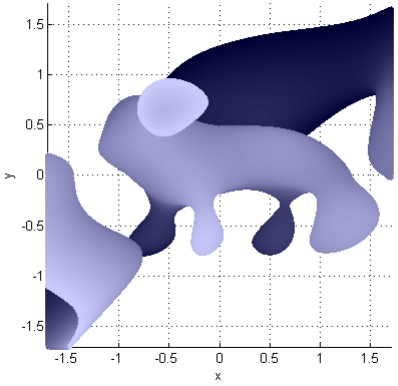




Gradient1



Min-Max



Min-Var

spurious zero-sets appear farther away from the desired zero-set

Rotation Invariant Fitting

- Gradient1 fitting is rotation invariant
- Min-Max fitting depends on: $\left\| \underline{p}^n(x_i, y_i, z_i) \right\|_1$

  $\Rightarrow$ not rotation invariant
- Min-Var fitting depends on: $\left\| \underline{p}^n(x_i, y_i, z_i) \right\|_2$

  $\Rightarrow$ not rotation invariant

Rotation Invariant Min-Max/Min-Var

- Min-Max: We replace the expressions: $\left\| \underline{p}^n(x_i, y_i, z_i) \right\|_1$

  by: $E_{\theta,\varphi}\left\{ \left\| \underline{p}^n(x_i, y_i, z_i) \right\|_1 \right\} \qquad 0 \le \theta < 2\pi \ , \ 0 \le \varphi < \pi$

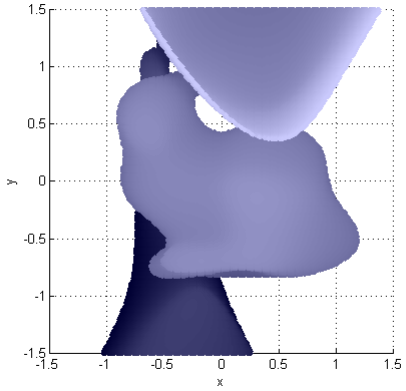
- Min-Var: We replace each monomial $x^k y^l z^m$

  by: $\sqrt{\frac{n!}{k!l!m!}} x^k y^l z^m$

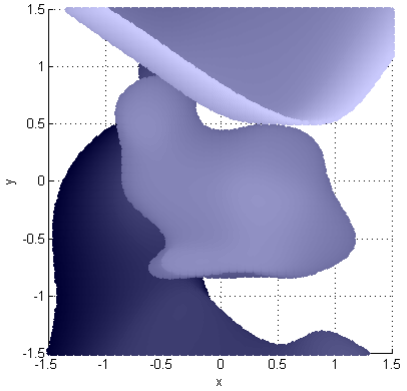
- We get: $\left\| \underline{p}^n(x_i, y_i, z_i) \right\|_2 = \sqrt{\sum_{p=0}^{n} d^{2p}}$

Rotation Invariant Fitting Algorithms

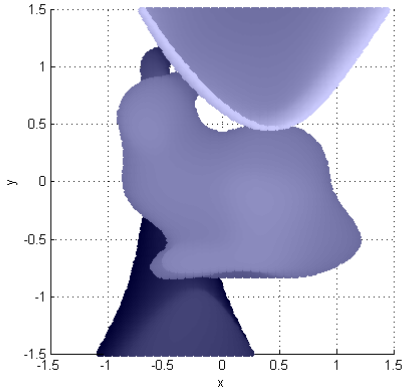
8^{th} degree IP fit



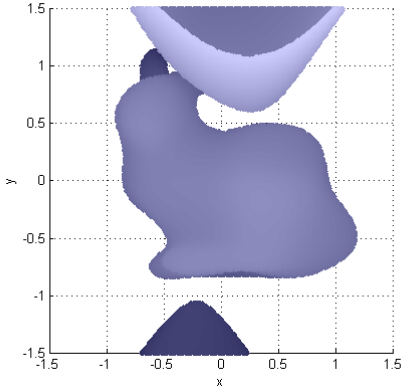
Min-Max



Min-Var



RI-Min-Max



RI-Min-Var

The rotation invariant modification hardly affects the results

IP Based Classification

- The IP coefficients are not rotation invariant.
- Rotation invariant expressions for classification:
  - Exist for 2D polynomials
    - Analytically (Tarel et al., 2000)
    - Using symbolic computation (Keren, 1994)
  - For 3D polynomials:
    - Exist only using symbolic computation (Keren et al., 1996)

Derivation of 3D Rotation Invariant Parameters

- Tensor is a generalized linear representation (a multi-dimensional array)
- The rank of a tensor is the number of array indices required to describe it
- Tensors are used for IP based pose estimation

Tensor Representation of Forms

$$P^n(x_1,x_2,x_3)=\sum_{0\le k,l,m,k+l+m\le n} a_{klm}x_1^k x_2^l x_3^m =$$

$$= \underbrace{a_{000}}_{H_0} + \underbrace{a_{100}x_1+a_{010}x_2+a_{001}x_3}_{H_1(x_1,x_2,x_3)} + \ldots + \underbrace{a_{n00}x_1^n+\ldots+a_{0n0}x_2^n+\ldots+a_{00n}x_3^n}_{H_n(x_1,x_2,x_3)} =$$

$$=\sum_{r=0}^{n} H_r(x_1,x_2,x_3)$$

- $H_r(x_1,x_2,x_3)$ is a homogeneous ternary polynomial of degree r (a 'form')

$$H_r(x_1,x_2,x_3)=\sum_{k+l+m=r} a_{klm}x_1^k x_2^l x_3^m$$

(Tarel et al., 1998)

Tensor Representation (Example)

- A 2nd degree form (3D):

$$H_2(x_1, x_2, x_3) = a_{200}x_1^2 + a_{020}x_2^2 + a_{002}x_3^2 + a_{110}x_1x_2 + a_{101}x_1x_3 + a_{011}x_2x_3$$

- Tensor (rank 2) representation:

$$S_2 = s^{i_1 i_2} = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 \end{matrix} \\ \begin{pmatrix} a_{200} & \frac{a_{110}}{2} & \frac{a_{101}}{2} \\ \frac{a_{110}}{2} & a_{020} & \frac{a_{011}}{2} \\ \frac{a_{101}}{2} & \frac{a_{011}}{2} & a_{002} \end{pmatrix} \end{array} \begin{matrix} \\ x_1 \\ x_2 \\ x_3 \end{matrix}$$

Tensor Representation of Forms

- Using n-dimensional array representation:

$$S_n = (s^{i_1 i_2 \ldots i_n})_{1 \le i_1, i_2, \ldots, i_n \le 3}$$

- We can represent a form of order n:

$$H_n(x_1, x_2, x_3) = \sum_{k+l+m=n} a_{klm} x_1^k x_2^l x_3^m$$

$$H_n(x_1, x_2, x_3) = \sum_{i_1=1}^{3} \sum_{i_2=1}^{3} \ldots \sum_{i_n=1}^{3} s^{i_1 i_2 \ldots i_n} x_{i_1} x_{i_2} \ldots x_{i_n} \quad , \quad s^{i_1 i_2 \ldots i_n} = \frac{a_{klm}}{\frac{n!}{k!l!m!}}$$

Tensor Contraction

- Contraction with respect to 2 indices: summation over the components in which these 2 indices have the same value

$$S_n = (s^{i_1 i_2 \ldots i_n})_{1 \le i_1, i_2, \ldots, i_n \le 3}$$

$$S_{n-2} = s^{i_3 i_4 \ldots i_n} = \sum_{i_1=1}^{3} s^{i_1 i_1 i_3 \ldots i_n}$$

- A total contraction of a tensor gives a zero order tensor which is an invariant.
- Example: the trace of a matrix (tensor, rank 2)

Tensor Representation (Example)

- A 4th degree form (3D):

$$H_4(x_1, x_2, x_3) = a_{400}x_1^4 + a_{040}x_2^4 + a_{004}x_3^4 +$$
$$+a_{310}x_1^3x_2 + a_{301}x_1^3x_3 + a_{130}x_1x_2^3 + a_{031}x_2^3x_3 + a_{103}x_1x_3^3 + a_{013}x_2x_3^3 +$$
$$+a_{220}x_1^2x_2^2 + a_{202}x_1^2x_3^2 + a_{022}x_2^2x_3^2 +$$
$$+a_{211}x_1^2x_2x_3 + a_{121}x_1x_2^2x_3 + a_{112}x_1x_2x_3^2$$

- In this case n=4 $\longrightarrow$ the tensor representation is a 4-dim. array

$$S_4 = (s^{i_1 i_2 i_3 i_4})_{1 \le i_1, i_2, i_3, i_4 \le 3}$$

Tensor Representation (Example)

$$s^{11ij} = \begin{bmatrix} a_{400} & \frac{a_{310}}{4} & \frac{a_{301}}{4} \\ \frac{a_{310}}{4} & \frac{a_{220}}{6} & \frac{a_{211}}{12} \\ \frac{a_{301}}{4} & \frac{a_{211}}{12} & \frac{a_{202}}{6} \end{bmatrix} \quad s^{12ij} = \begin{bmatrix} \frac{a_{310}}{4} & \frac{a_{220}}{6} & \frac{a_{211}}{12} \\ \frac{a_{220}}{6} & \frac{a_{130}}{4} & \frac{a_{121}}{12} \\ \frac{a_{211}}{12} & \frac{a_{121}}{12} & \frac{a_{112}}{12} \end{bmatrix} \quad s^{13ij} = \begin{bmatrix} \frac{a_{301}}{4} & \frac{a_{211}}{12} & \frac{a_{202}}{6} \\ \frac{a_{211}}{12} & \frac{a_{121}}{12} & \frac{a_{112}}{12} \\ \frac{a_{202}}{6} & \frac{a_{112}}{12} & \frac{a_{103}}{4} \end{bmatrix}$$

$$s^{21ij} = \begin{bmatrix} \frac{a_{310}}{4} & \frac{a_{220}}{6} & \frac{a_{211}}{12} \\ \frac{a_{220}}{6} & \frac{a_{130}}{4} & \frac{a_{121}}{12} \\ \frac{a_{211}}{12} & \frac{a_{121}}{12} & \frac{a_{112}}{12} \end{bmatrix} \quad s^{22ij} = \begin{bmatrix} \frac{a_{220}}{6} & \frac{a_{130}}{4} & \frac{a_{121}}{12} \\ \frac{a_{130}}{4} & a_{040} & \frac{a_{031}}{4} \\ \frac{a_{121}}{12} & \frac{a_{031}}{4} & \frac{a_{022}}{6} \end{bmatrix} \quad s^{23ij} = \begin{bmatrix} \frac{a_{211}}{12} & \frac{a_{121}}{12} & \frac{a_{112}}{12} \\ \frac{a_{121}}{12} & \frac{a_{031}}{4} & \frac{a_{022}}{6} \\ \frac{a_{112}}{12} & \frac{a_{022}}{6} & \frac{a_{013}}{4} \end{bmatrix}$$

$$s^{31ij} = \begin{bmatrix} \frac{a_{301}}{4} & \frac{a_{211}}{12} & \frac{a_{202}}{6} \\ \frac{a_{211}}{12} & \frac{a_{121}}{12} & \frac{a_{112}}{12} \\ \frac{a_{202}}{6} & \frac{a_{112}}{12} & \frac{a_{103}}{4} \end{bmatrix} \quad s^{32ij} = \begin{bmatrix} \frac{a_{211}}{12} & \frac{a_{121}}{12} & \frac{a_{112}}{12} \\ \frac{a_{121}}{12} & \frac{a_{031}}{4} & \frac{a_{022}}{6} \\ \frac{a_{112}}{12} & \frac{a_{022}}{6} & \frac{a_{013}}{4} \end{bmatrix} \quad s^{33ij} = \begin{bmatrix} \frac{a_{202}}{6} & \frac{a_{112}}{12} & \frac{a_{103}}{4} \\ \frac{a_{112}}{12} & \frac{a_{022}}{6} & \frac{a_{013}}{4} \\ \frac{a_{103}}{4} & \frac{a_{013}}{4} & a_{004} \end{bmatrix}$$

Tensor Contraction (Example)

- First contraction:

$$S_2 = s^{i_1 i_2} = \sum_{i_3=1}^{3} s^{i_1 i_2 i_3 i_3}$$

- Second contraction:

$$S_0 = \sum_{i_1=1}^{3} \sum_{i_3=1}^{3} s^{i_1 i_1 i_3 i_3}$$

- The result:

$$S_0 = a_{400} + a_{040} + a_{004} + \frac{1}{3}\left(a_{220} + a_{202} + a_{022}\right)$$

Linear Rotation Invariants (3D)

From each even degree form, we derive 1 linear invariant:

$$L_{3D,n} = \sum_{i_1=1}^{3} \cdots \sum_{i_3=1}^{3} \sum_{i_{n-1}=1}^{3} s^{i_1 i_1 i_3 i_3 \ldots i_{n-1} i_{n-1}}$$

$$L_{3D,n} = \sum_{\substack{k,l,m \text{ even} \\ k+l+m=n}} \frac{a_{klm}}{\dfrac{n!}{k!l!m!}} \cdot \frac{(n/2)!}{(k/2)!(l/2)!(m/2)!}$$

$\Rightarrow \left\lfloor \frac{n}{2} \right\rfloor + 1$ linear invariants from an IP of degree n

Linear Rotation Invariants (2D)

- Using the tensor representation we get explicit rotation invariants for the 2D case as well:

$$L_{2D,n} = \sum_{\substack{k,l \text{ even} \\ k+l=n}} \frac{a_{kl}}{\frac{n!}{k!l!}} \cdot \frac{(n/2)!}{(k/2)!(l/2)!}$$

- Faster computation than the existing recursive scheme (Tarel et al., 1998)

Rotation Invariants and Quaternions

- In 2D, quadratic invariants were derived using a complex representation (Tarel et al., 1998) :

$$z = x + iy \qquad P^n(x, y) \Rightarrow P^n(z, \bar{z})$$

- In 3D, the extension of the regular complex representation is quaternions:

$$q = q_0 + q_1 i + q_2 j + q_3 k$$

$$i^2 = -1 \quad j^2 = -1 \quad k^2 = -1$$

$$ij = k \quad jk = i \quad ki = j$$

$$ji = -k \quad kj = -i \quad ik = -j$$

- multiplication is non-commutative: $q_1 q_2 \neq q_2 q_1$

$\Rightarrow$ calculations can't be simplified

Quadratic Rotation Invariants

Instead of quaternions, we used trigonometric identities and derived the following invariants:

$$Q_{3D,1} = a_{100}^2 + a_{010}^2 + a_{001}^2$$

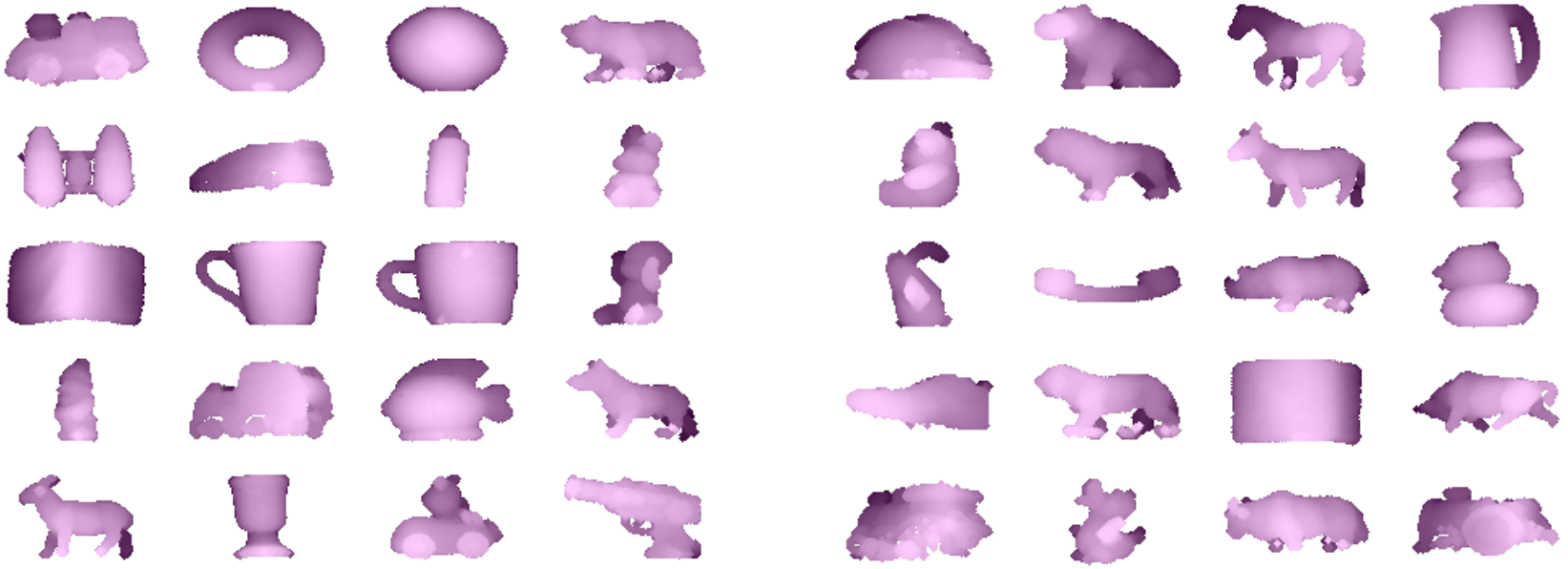
$$Q_{3D,2} = a_{200}^2 + a_{020}^2 + a_{002}^2 -$$
$$-2a_{200}a_{020} - 2a_{200}a_{002} - 2a_{020}a_{002} +$$
$$+a_{110}^2 + a_{101}^2 + a_{011}^2$$

3D Objects/Faces Classification

- Important in various fields, such as robotics and automatic security systems
- We have a dictionary of L different 3D objects (each has several representations)
- Given a new representation of an object, we would like to recognize it as one of the L objects in the dictionary.

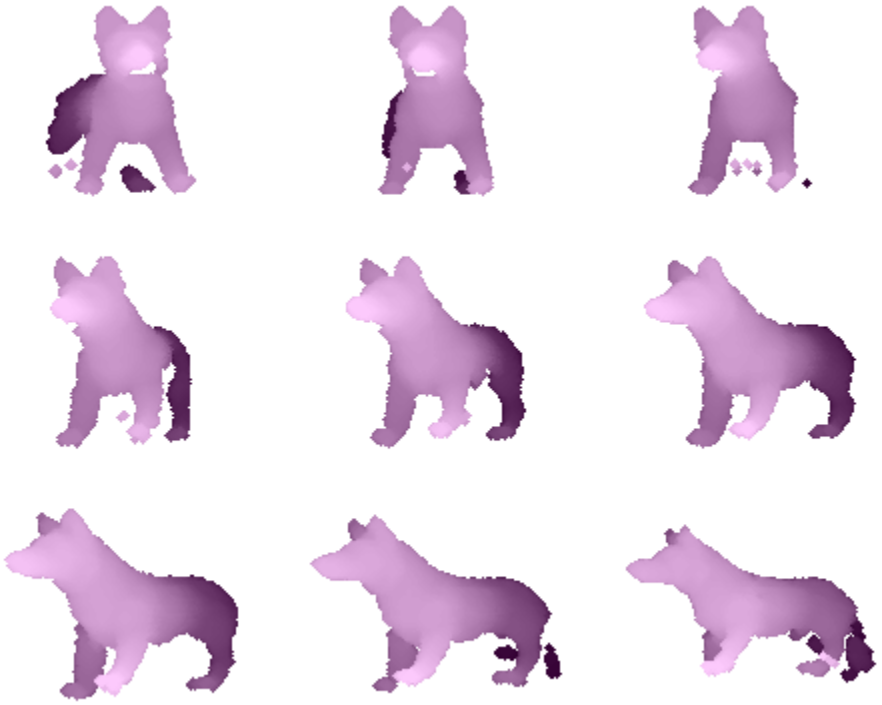
3D Objects Database

- 40 rigid objects – acquired using the equipment of the Geometric Image Processing (GIP) lab at the CS Faculty of the Technion.



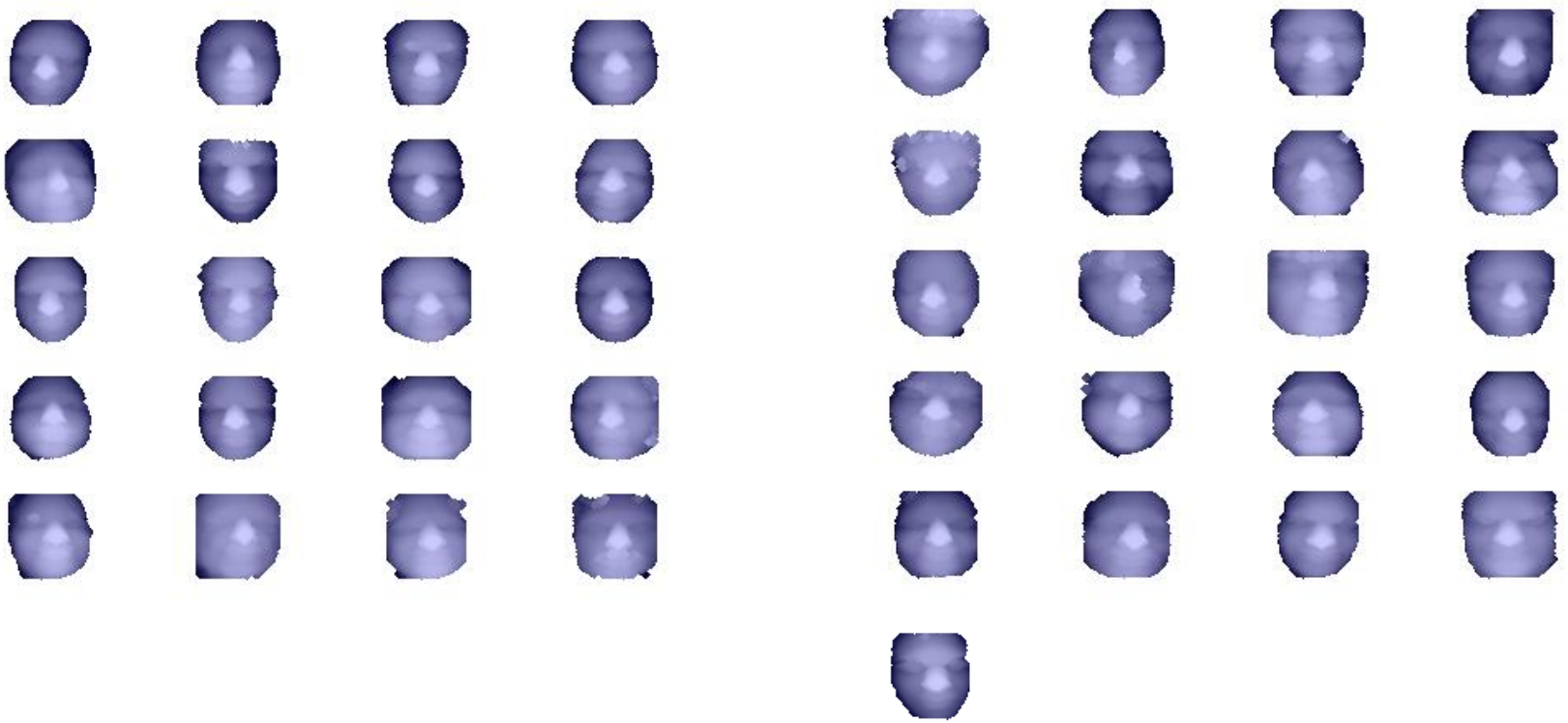
3D Objects Database

- Each object has more than one description:



3D Faces Database

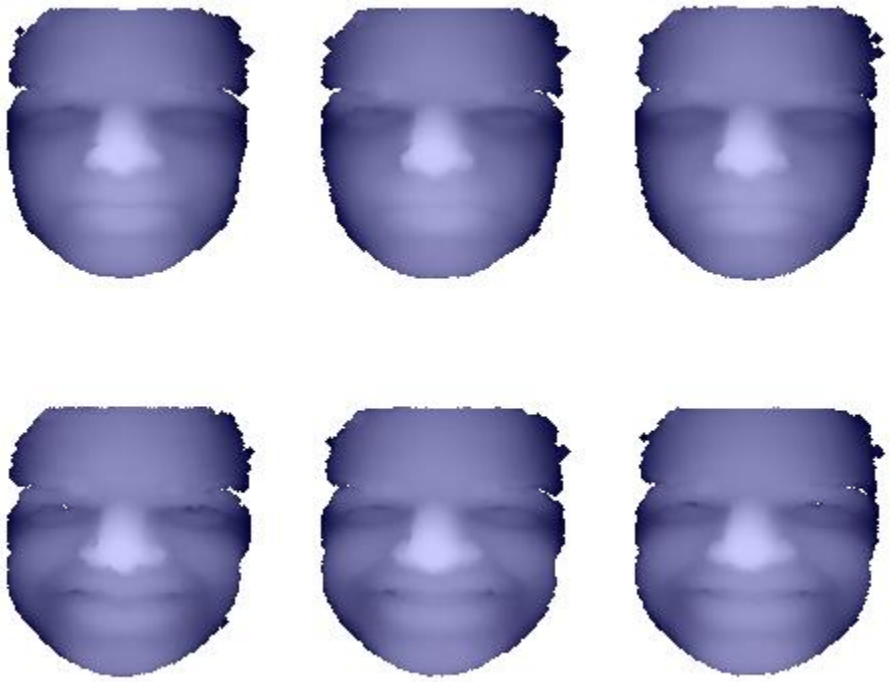
- 41 faces* – gathered by the Geometric Image Processing (GIP) lab at the CS Faculty of the Technion.



*cooperative situation

3D Faces Database

- each face has a few acquisitions:



Pre-Processing

- Translation
- Scaling
- Mirroring
- 2D Projections

Translation

- We locate the center of mass at the origin:

$$x_{i,translated} = x_i - \frac{i}{N}\sum_{k=1}^{N} x_k$$

$$y_{i,translated} = y_i - \frac{i}{N}\sum_{k=1}^{N} y_k$$

$$z_{i,translated} = z_i - \frac{i}{N}\sum_{k=1}^{N} z_k$$

- Center of mass is invariant to rotation and scaling

Scaling

- For IP stability reasons, the data should be scaled so that it would be close to a sphere of radius 1

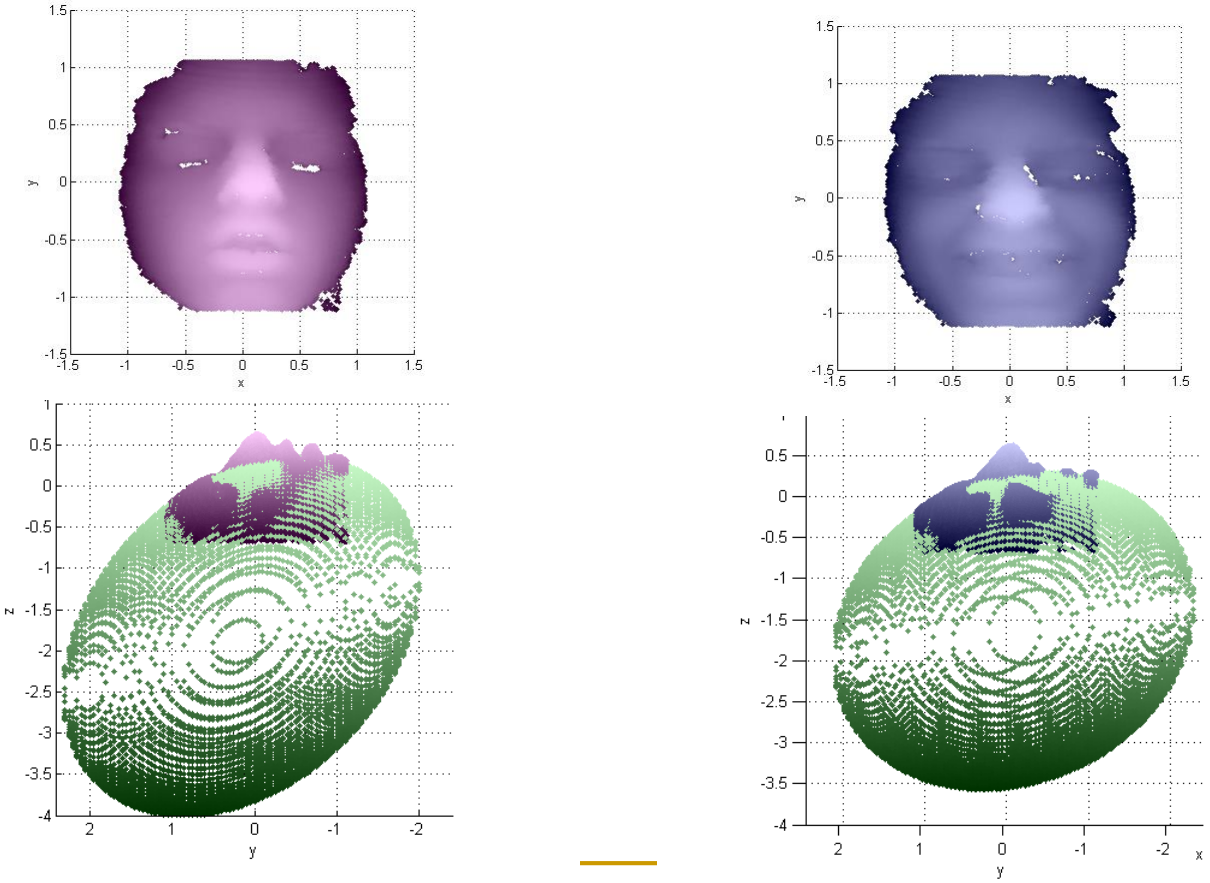
- We choose our scaling factor to be the 75th percentile of the distances of the data-points from the origin:

$$x_{i,scaled} = x_i / S_{75\%}$$
$$y_{i,scaled} = y_i / S_{75\%}$$
$$z_{i,scaled} = z_i / S_{75\%}$$

(Landa, 2006)

Mirroring of Faces

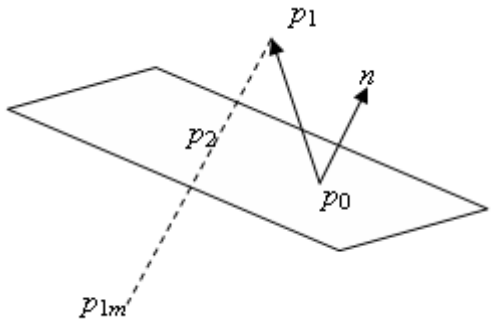
- IP fitting stability problems:



Mirroring of Faces

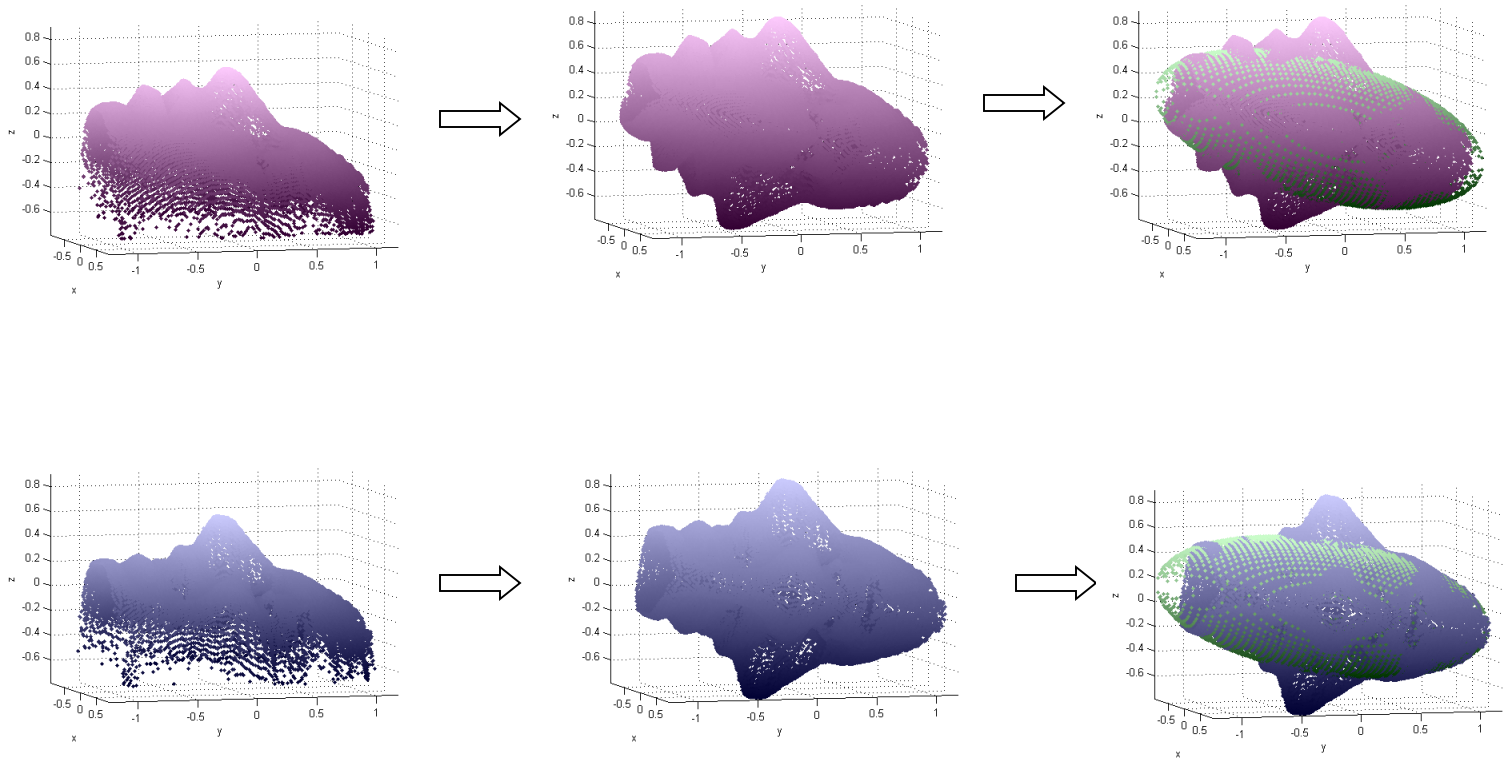
Mirroring scheme:

- Fit a plane to the entire face
- Slide it an empiric distance from the center of mass
- Dispose of points below the plane
- Mirror the rest of the points with respect to the plane



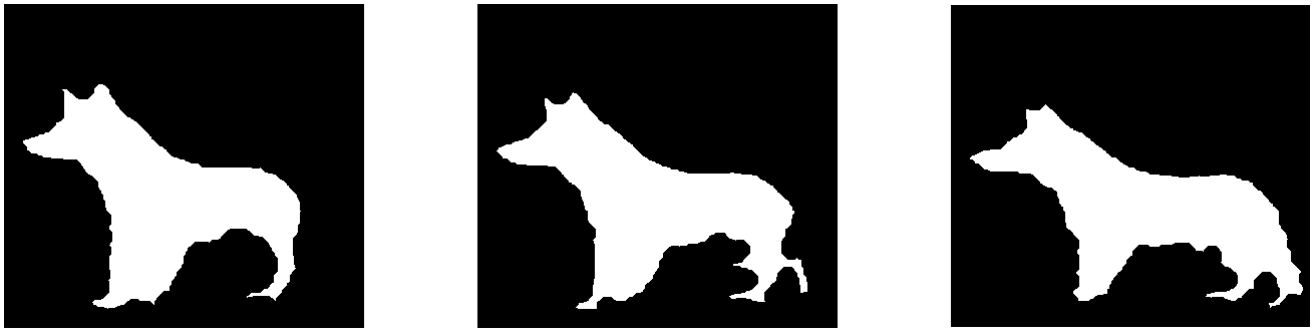
Mirroring of Faces

- Mirroring result:

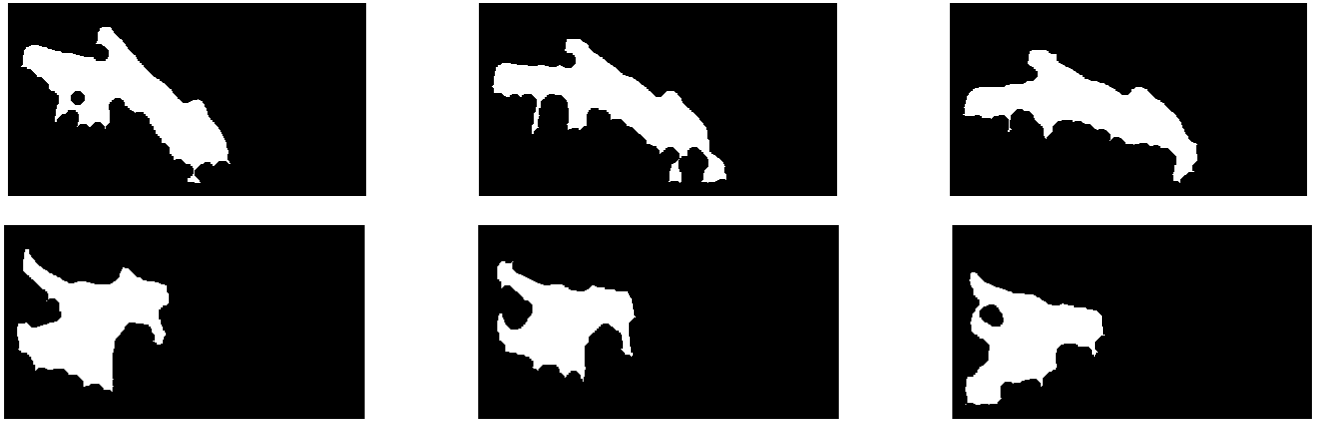


2D Projections – Objects

- *xy* projection:

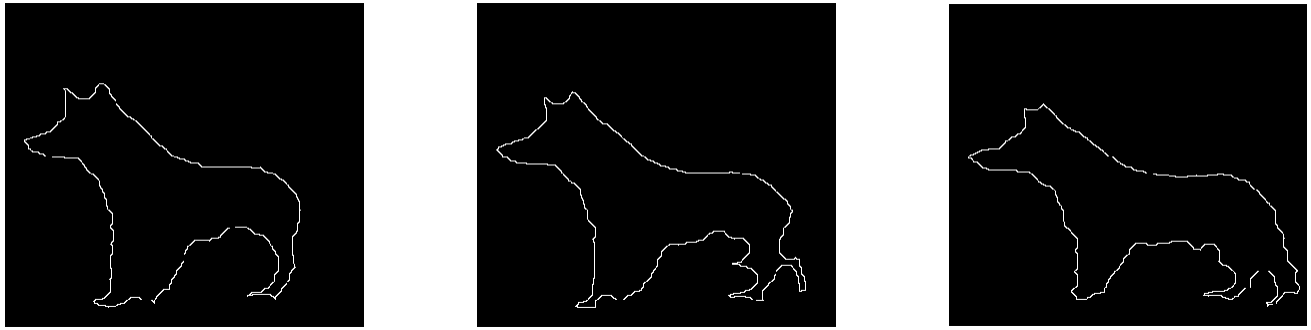


- Without mirroring, the *xz* and *yz* projections have noisy contours:



2D Projections – Objects

- Extract contours using morphological operators:



2D Projections – Faces

- After mirroring:

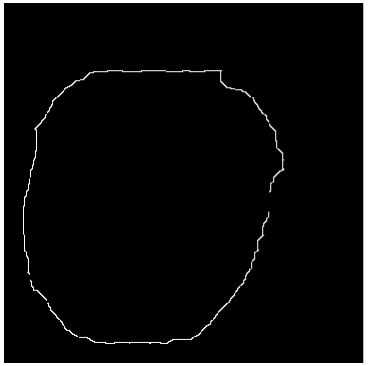
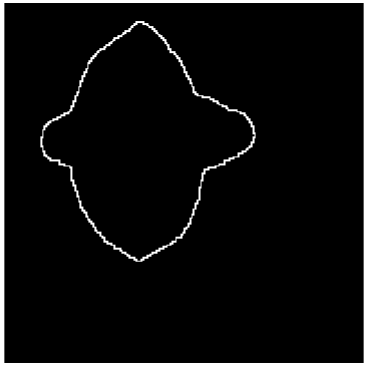
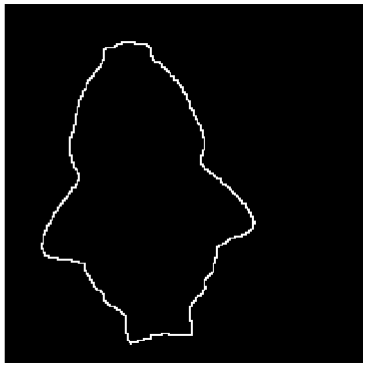
  xy projection *xz* projection *yz* projection

- *xy* projection is very similar for different faces

2D Projections – Faces

- Extract contours using morphological operators:

xy projection *xz* projection *yz* projection

Feature Extraction

- We fit various degrees of IPs (2,4,6,8)*:
  - to the 3D surface
  - to the most descriptive 2D contours
- For each 3D/2D IP – separate the forms and derive linear rotation invariants from each even degree form.
- For each 3D IP – derive 2 quadratic invariants.

*Multi-Order and Fitting-Error Technique (MOFET, Landa, 2006)

Additional Rotation Invariant Features

- IP Fitting Error (2D/3D) – the 75th percentile of the IP fitting errors at each data-point

- Eigen Values of the data scatter matrix (PCA)

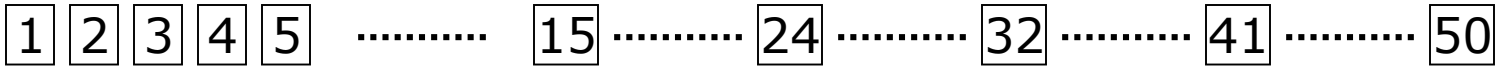
$$S\left(\underline{x},\underline{y},\underline{z}\right)=\frac{1}{N}\begin{bmatrix}\underline{x}^T\\ \underline{y}^T\\ \underline{z}^T\end{bmatrix}\begin{bmatrix}\underline{x} & \underline{y} & \underline{z}\end{bmatrix}=\frac{1}{N}\begin{bmatrix}\sum_i x_i^2 & \sum_i x_i y_i & \sum_i x_i z_i\\ \sum_i y_i x_i & \sum_i y_i^2 & \sum_i y_i z_i\\ \sum_i z_i x_i & \sum_i z_i y_i & \sum_i z_i^2\end{bmatrix}$$

The Features Vector

	Objects Recognition		Faces Recognition	
3D linear invariants	IP degrees 2,4,6	9	IP degrees 2,4,6,8	14
3D IP fitting error	IP degrees 2,4,6	3	---	0
3D quadratic invariants	IP degrees 2,4	4	IP degrees 2,4	4
2D linear invariants	*xy* - IP degrees 2,4,6	9	*xz* - IP degrees 2,4,6 *yz* - IP degrees 2,4	9 5
2D IP fitting error	*xy* - IP degrees 2,4,6	3	---	0
3D PCA eigenvalues	eigenvalues	3	eigenvalues	3
Total	**31 features**		**35 features**	

Learning and Testing

- Rigid Objects:
  - 9 different positions for each object (~12^{o} difference), 5 consecutive frames each
  - Even positions are used as learning (2,4,6,8)
  - Odd positions are used as test (1,3,5,7,9)
- Faces:
  - ~50 different frames for each face (2.5 frame/sec)
  - First 5 frames are used as learning
  - 5 other frames are used as test (linearly spaced along the rest of the movie)



Classifier Design – Objects

- For each learning position:
  - Each of the 5 frames is perturbed 10 times (colored Gaussian noise, std=0.01)
  - For each of the 50 perturbed instances the features vector is calculated
  - Combine pairs of learning positions (2&4,4&6,6&8)
  - Calculate mean and covariance of parameters and create a multi-Gaussian PDF for each pair:

$$P\left(\underline{v}/O_k, V_n\right) = \frac{1}{\left(2\pi\right)^{d/2}\left|\Sigma\right|^{1/2}} \exp\left\{-\frac{1}{2}\left(\underline{v}-\underline{\mu}\right)^T \Sigma^{-1}\left(\underline{v}-\underline{\mu}\right)\right\}$$

(d is the parameters vector dimension)

O_k , $k = 1,2,...,40$

V_n , $n = 2\&4, 4\&6, 6\&8$

Classifier Design – Objects

- For each test position:
  - For each of the 5 frames, the features vector $\underline{v}$ is calculated
  - The probabilities that the object was O_k in position V_n, given that we observed $\underline{v}$:

    $P(O_k, V_n / \underline{v})$, $k = 1,...,40$, $n = 2\&4, 4\&6, 6\&8$
  - Bayes rule:

$$P(O_k, V_n / \underline{v}) = \frac{P(\underline{v} / O_k, V_n) P(O_k, V_n)}{P(\underline{v})} = C \cdot P(\underline{v} / O_k, V_n)$$

Experimental Results (Objects)

Comparison between different fitting algorithms:

	Gradient1	RI-Min-Max	RI-Min-Var
entire features vector (d=31)	**98.8%**	98.1%	98.0%
3D IP features only (d=16)	**96.1%**	94.8%	93.2%

Comparison between our method and other approaches:

Our method (Gradient1, d=31)	Pose estimation and IP coef. as features (d=35)	Shape Spectrum Descriptor (MPEG-7) (d=27)
98.8%	91.6%	90.1%

Experimental Results (Faces)

Comparison between different fitting algorithms:

	Gradient1	RI-Min-Max	RI-Min-Var
entire features vector (d=35)	**97.1%**	96.6%	93.7%
3D IP features only (d=18)	89.3%	**91.7%**	**91.7%**

Comparison between our method and other **general objects** classification approaches:

Our method (Gradient1, d=35)	Pose estimation and IP coef. as features (d=35)	Shape Spectrum Descriptor (MPEG-7) (d=27)
97.1%	93.2%	72.2%

* Classifier design was the same as the objects' classifier

(without the different views)

Running Times

- Matlab implementation
- Average classification time of a single object/face:

Our method (Gradient1, d=31)	Pose estimation and IP coef. as features (d=35)	Shape Spectrum Descriptor (MPEG-7) (d=27)
13 [Sec]	15 [Sec]	55 [Sec]

Future Work

- Dimensionality reduction for automatic features selection
- 3D objects retrieval (classification into categories)
- Derivation of additional quadratic invariants using quaternions

Summary

- Rotation invariant Min-Max and Min-Var fitting algorithms
- Developing linear and quadratic 3D IP based rotation invariants
- Stabilization of IP coefficients using faces mirroring
- Developing 3D Objects classification system, applicable also to faces in a cooperative situation
- Integrating both 2D and 3D features for better classification performance