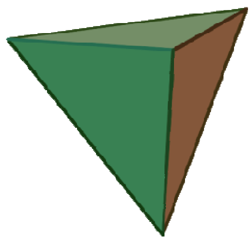
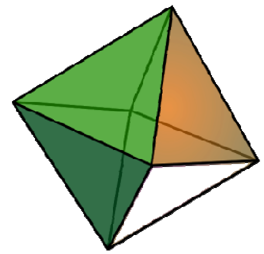
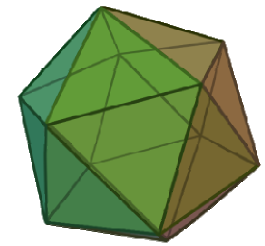
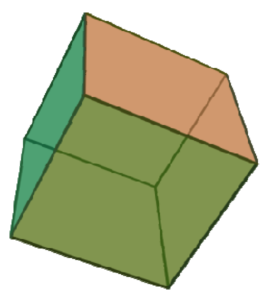
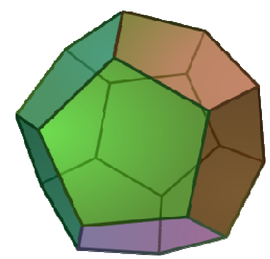






Correspondence-Free Alignment of 3D Object Models



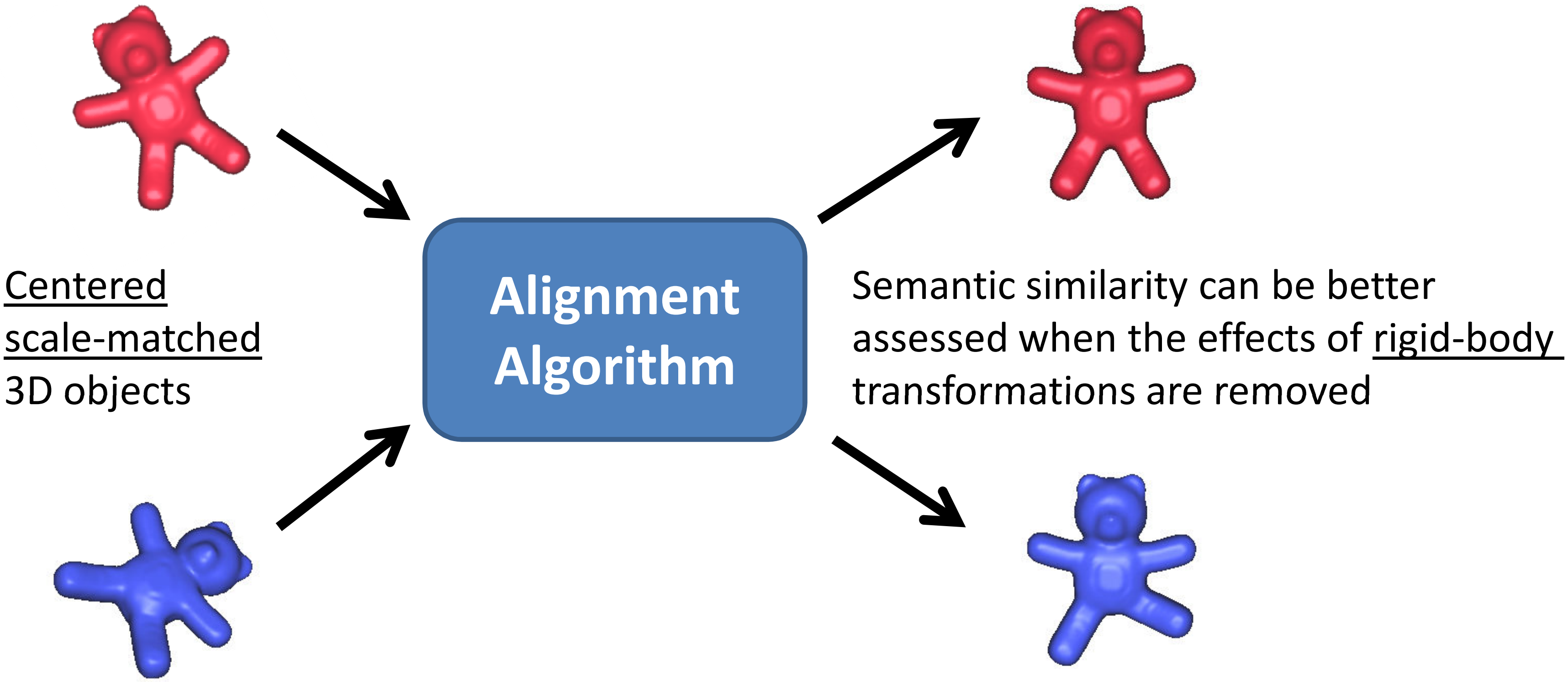


Ceyhun Burak Akgül, Boğaziçi University EE Dept., Istanbul, Turkey
Bülent Sankur, Boğaziçi University EE Dept., Istanbul, Turkey
Yücel Yemez, Koç University Computer Eng. Dept., Istanbul, Turkey

Outline

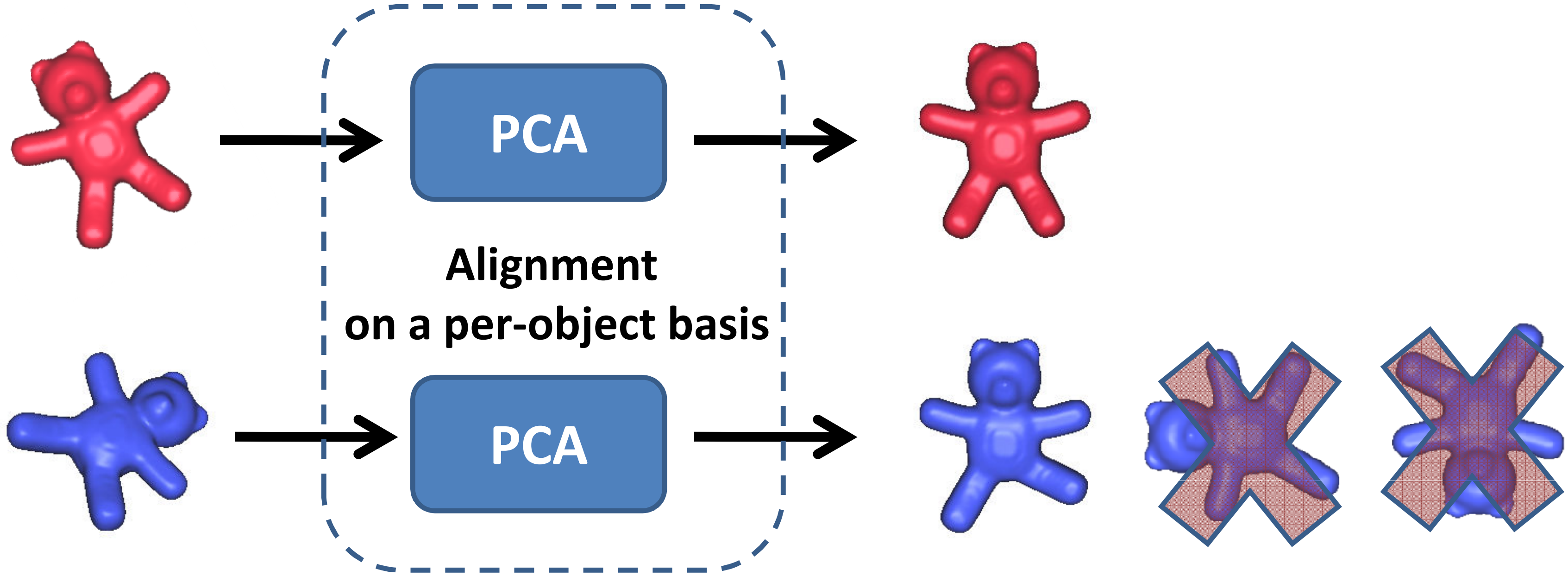
- The 3D Shape Alignment Problem
- Density-Based Shape Description
- Symmetry Properties of Regular Polyhedra
- Alignment Algorithms
- Experiments
- Concluding Remarks

The Alignment Problem

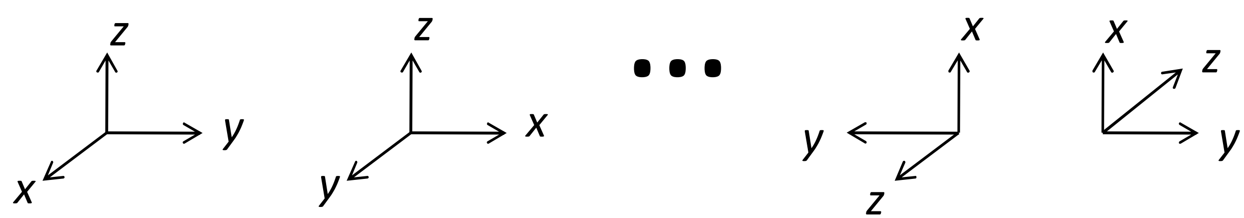


Rigid-body alignment is a fundamental step in shape matching tasks: e.g., in 3D object retrieval

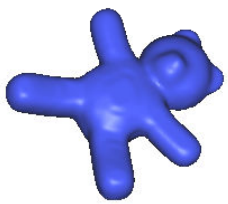
The Alignment Problem



- Usually principal axes are correctly found
- But, <u>labeling the axes</u> and <u>assigning polarities</u> are problematic



The Alignment Problem



Centered
scale-matched
3D objects

In this work

Alignment by Distance Minimization

over 3D Rotations and 3D Reflections

$$\Gamma^{*} = \arg\min_{\Gamma \in \mathcal{G}} dist\left(\Gamma\left[A\right], B\right)$$

Minimize the distance between two objects *A* and *B* over a finite set $\mathcal{G}$ of 3D rotations and reflections Γ

The Alignment Problem

$$\Gamma^* = \arg\min_{\Gamma \in \mathcal{G}} dist\left(\Gamma\left[A\right], B\right)$$

$$\Gamma^* = \arg\min_{\Gamma \in \mathcal{G}} dist\left(\Gamma\left[\mathbf{f}_A\right], \mathbf{f}_B\right)$$

- Minimizing the distance between A and B: <u>correspondence problem</u>
- Instead, "register" the objects on a common mathematical domain via shape descriptors $\mathbf{f}_A$ and $\mathbf{f}_B$
- Then minimize the distance between the shape descriptors $\mathbf{f}_A$ and $\mathbf{f}_B$

The Alignment Problem

$$\Gamma^* = \arg\min_{\Gamma \in \mathcal{G}} dist\left(\Gamma\left[A\right], B\right)$$

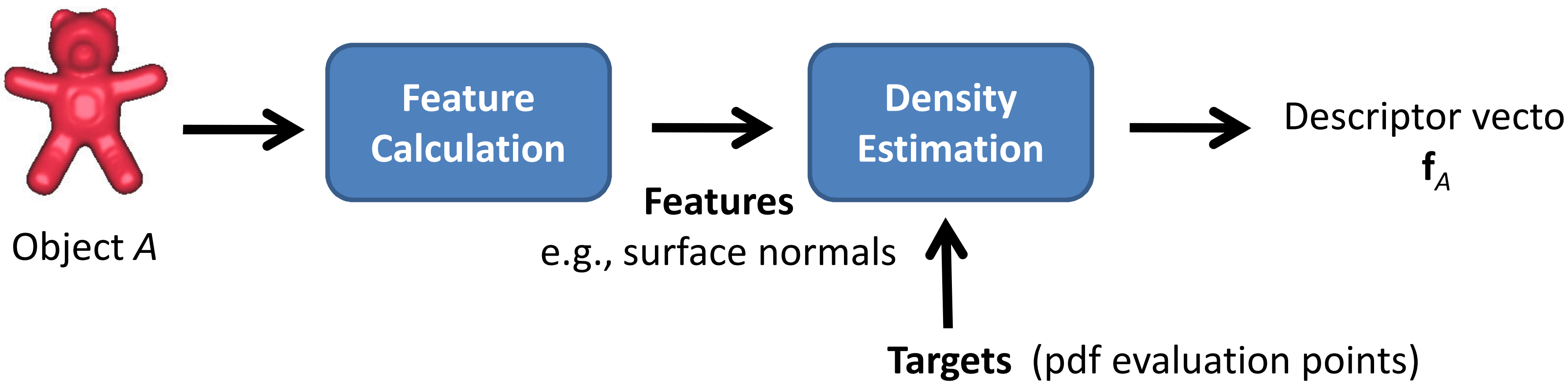
$$\Gamma^* = \arg\min_{\Gamma \in \mathcal{G}} dist\left(\Gamma\left[\mathbf{f}_A\right], \mathbf{f}_B\right)$$

We should be able to compute $\Gamma[\mathbf{f}_A]$ very fast without explicitly transforming the object via $\Gamma[A]$

Density-Based Shape Description

C. B. Akgül *et al.* IEEE Trans on PAMI 31(6), June 2009.

- A density-based shape descriptor is the sampled pdf of a surface feature



- When the object is rotated, pose-dependent features rotate exactly the same way.
- Pose-dependent features (e.g., normal vector, radial direction) are defined on the unit-sphere

  → targets should be <u>selected from the unit-sphere</u>

Density-Based Shape Description

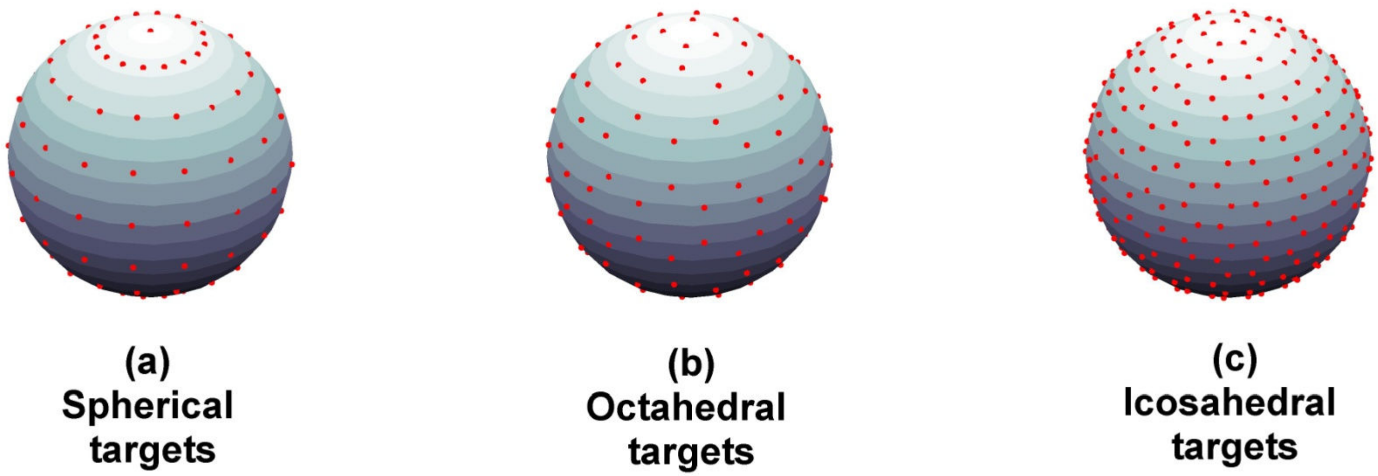
C. B. Akgül *et al.* IEEE Trans on PAMI 31(6), June 2009.

Target Selection by Polyhedron Subdivision:

1. Take a regular polyhedron, say an octahedron, circumscribed by the unit-sphere
2. Subdivide in four each of the eight faces of the octahedron
3. Iterate recursively over the new faces
4. Radially project the barycenters of the resulting faces back to the unit-sphere to obtain targets for pose-dependent features

Density-Based Shape Description

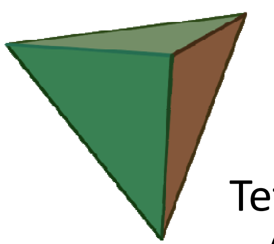
C. B. Akgül *et al.* IEEE Trans on PAMI 31(6), June 2009.



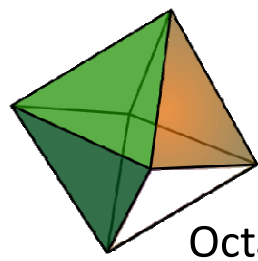
- Targets selected by polyhedron subdivision are more uniformly spaced than spherical targets
- They also inherit symmetry properties of regular polyhedra
- These symmetry properties enable fast and exact alignment in the case of certain 3D rotations and reflections

Symmetry Properties of Polyhedra

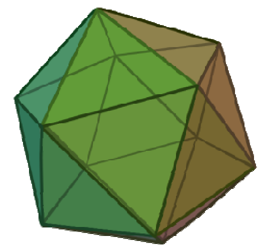
A regular polyhedron (a Platonic solid) enjoys certain symmetry properties in the sense that it is possible to perform certain transformations that change the position of individual faces but leave the polyhedron in a position that is indistinguishable from its original position.



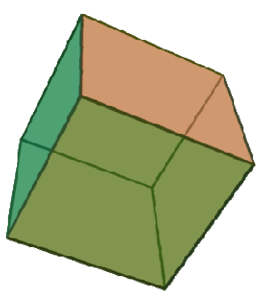
Tetrahedron
(4 faces)



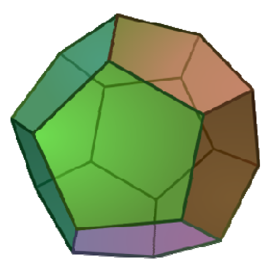
Octahedron
(8 faces)



Icosahedron
(20 faces)



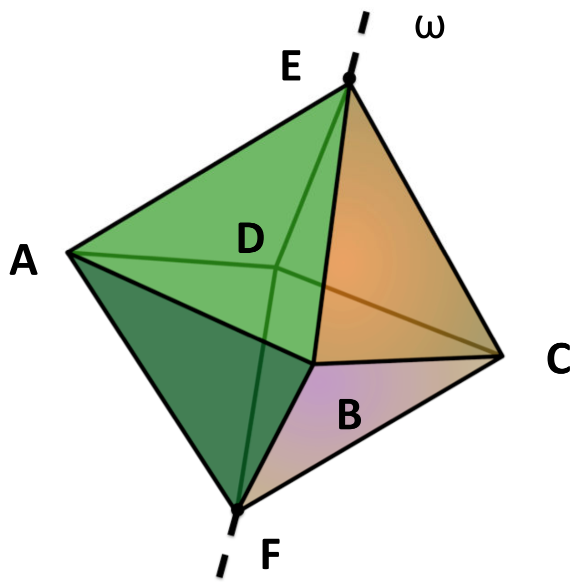
Cube (4 faces)
dual of octahedron



Dodecahedron (12 faces)
dual of icosahedron

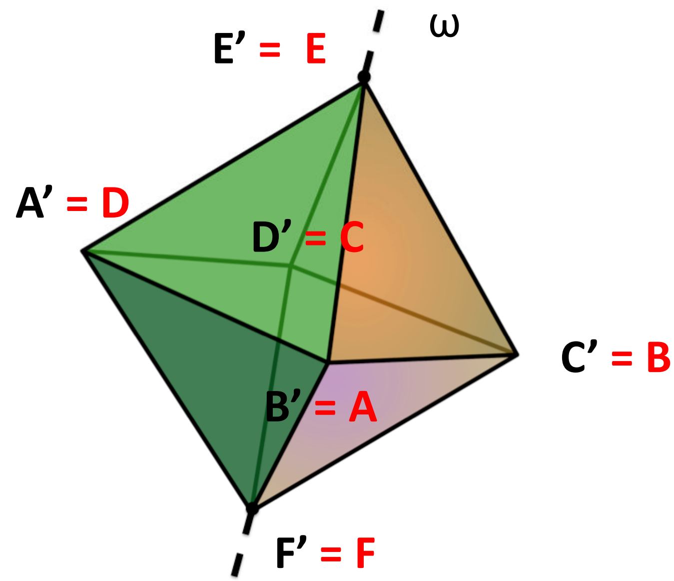
Symmetry Properties of Polyhedra

When a regular polyhedron is rotated around one of its symmetry axes by a certain amount, it looks exactly the same from a geometrical consideration. <u>The only change is a relabeling of the vertices (and faces)</u>.



Rotation around ω by π/2 radians

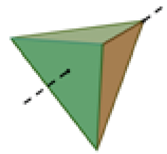
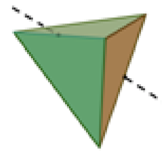
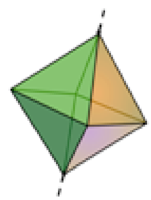
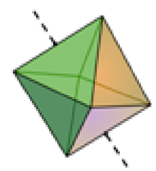
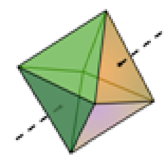
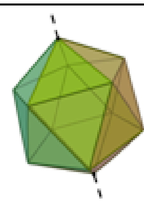
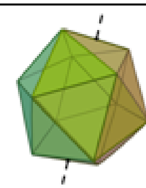
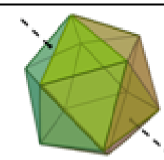
A → B
B → C
C → D
D → A
E → E
F → F



1. A polyhedral symmetry operation induces a permutation of the vertex labels

2. This also holds for the vertices obtained by polyhedron subdivision

Symmetry Properties of Polyhedra

	Symmetry Axis I	Symmetry Axis II	Symmetry Axis III	No. of Symmetries
Tetrahedron 4 vertices 4 faces 6 edges	 **Type:** vertex-to-face	 **Type:** edge-to-edge	- No axis of Type III-	**12 rotations** **12 rotoreflections** **24 in total**
Octahedron 6 vertices 8 faces 12 edges	 **Type:** vertex-to-vertex	 **Type:** edge-to-edge	 **Type:** face-to-face	**24 rotations** **24 rotoreflections** **48 in total**
Icosahedron 12 vertices 20 faces 30 edges	 **Type:** vertex-to-vertex	 **Type:** edge-to-edge	 **Type:** face-to-face	**60 rotations** **60 rotoreflections** **120 in total**

Exact Alignment

The Problem: $\Gamma^* = \arg\min_{\Gamma \in \mathcal{G}} dist\left(\Gamma\left[\mathbf{f}_A\right], \mathbf{f}_B\right)$

The Algorithm:

Input: Descriptors $\mathbf{f}_A$ and $\mathbf{f}_B$, a finite set of rigid-body transformations $\mathcal{G} = \{\Gamma_n\}_{n=1}^N$

Initialize: $MINVAL = \infty$

For each $\Gamma_n \in \mathcal{G}$

(1) $\mathbf{f} \leftarrow \Gamma_n\left[\mathbf{f}_A\right]$

(2) $temp \leftarrow dist\left(\mathbf{f}, \mathbf{f}_B\right)$

(3) If $temp < MINVAL$ then $MINVAL \leftarrow temp$ and $\Gamma^* \leftarrow \Gamma_n$

End

Output: Aligning transformation $\Gamma^* \in \mathcal{G}$

The Critical Step: (1) $\mathbf{f} \leftarrow \Gamma[\mathbf{f}_A]$

Exact Alignment

Fact 1. *The density-based descriptor corresponding to a pose-dependent feature consists of pdf values evaluated at target points selected on the unit sphere*

Fact 2. *A symmetry of a polyhedron induces a permutation of its vertex labels. Subdivisions of the polyhedron inherit these symmetry properties.*

Consequence 1. *If the targets points are selected by polyhedron subdivision and the transformation* Γ *corresponds to one of the polyhedral symmetries, then the step* (1) $\mathbf{f} \leftarrow \Gamma[\mathbf{f}_A]$ *is just* <u>*a permutation of the entries in the descriptor vector*</u> $\mathbf{f}_A$, <u>*which can be performed almost instantaneously*</u>*.*

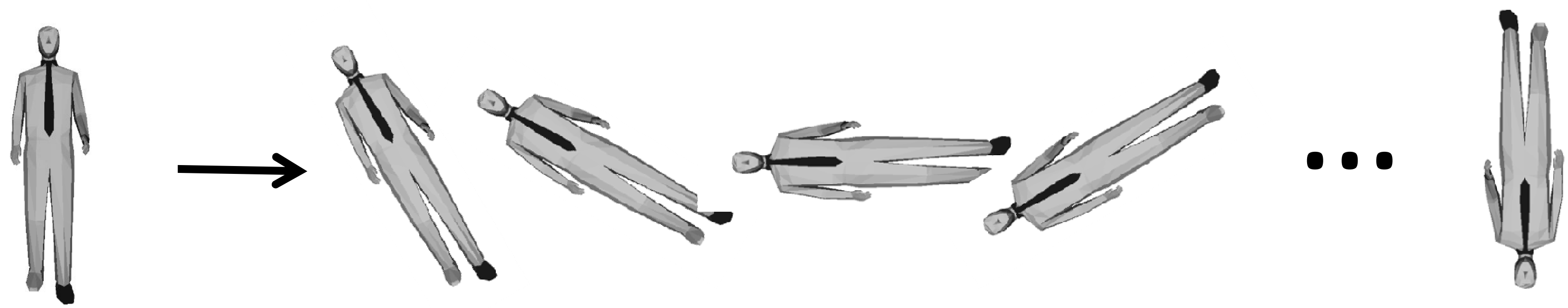
Consequence 2. *If the minimization is carried out* <u>*over the set of polyhedral symmetries*</u>*, then the solution found is exact.*

Approximate Alignment

- For arbitrary 3D rotations (other than polyhedral), the permutation property does not strictly hold.
- To extend the procedure to arbitrary 3D rotations:
  - Discretize the infinite set of 3D rotations by a suitable parametrization
  - Generate target permutations by a nearest-neighbor procedure
  - Each permutation will "approximately" correspond to a transformation from the discrete set of 3D rotations

Experiments

A self-alignment test



Number of Strict Rotation Estimation Errors

Octahedral rotations (48 rotoreflections)	Icosahedral Rotations (120 rotoreflections)	512 arbitrary rotations*
0/48	**0/120**	**14/512 (2.7%)**

* Obtained by discretizing the Rodrigues parametrization of 3D rotations:

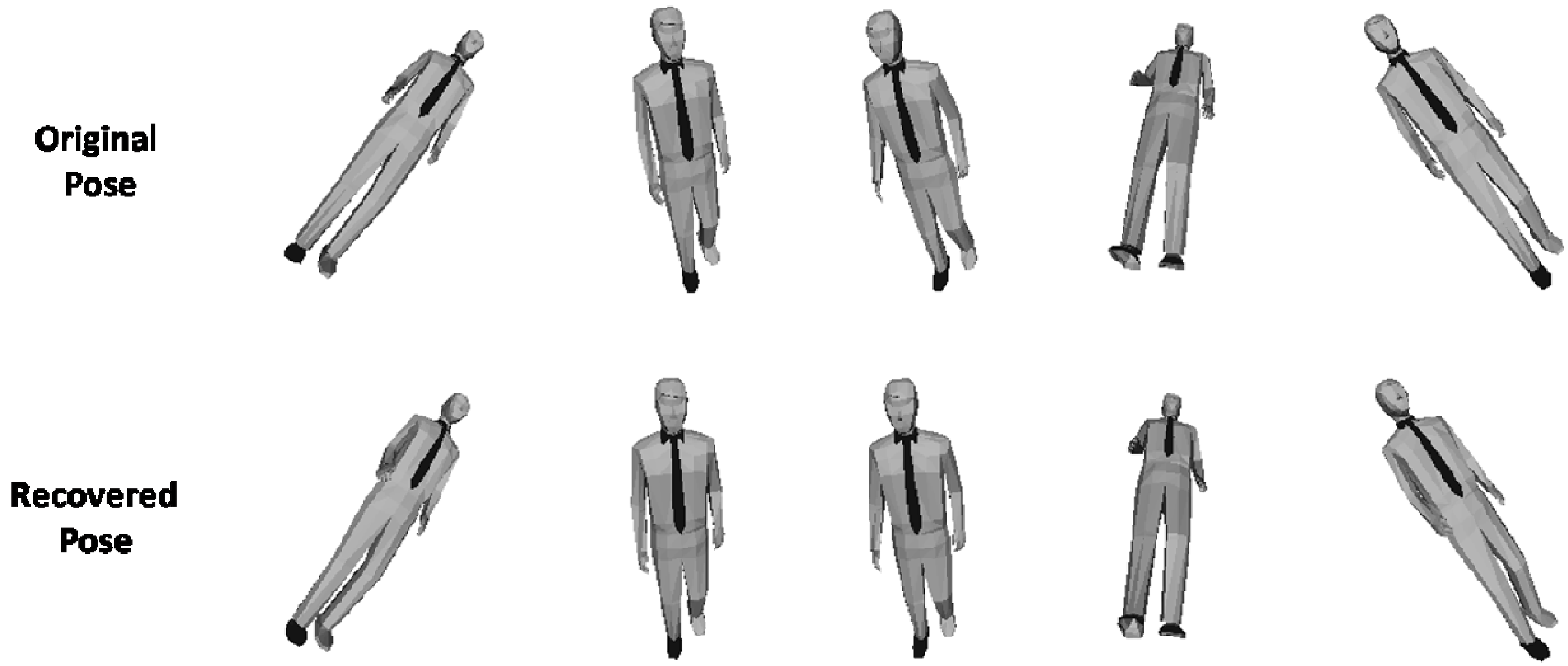
$$\Gamma = uu^T - \cos\theta\left(I - uu^T\right) + \sin\theta\left[u\right]_\times$$

The algorithm accurately recovers the pose of an object with respect to its original pose when the applied transformation coincides with a transformation from the predetermined set over which the distance minimization is carried out

Experiments

A self-alignment test

Pose Estimation Errors for the case of 512 Arbitrary Rotations



In the few cases where the recovered rotation was not correct, the estimated poses were nevertheless very close to the pose corresponding to the applied rotation.

α

Experiments

Alignment between two different models of the same class:

Rotate model *A* with respect to model B using each of the 512 arbitrary 3D rotations, A and B belong to the same class

Performance over the set of 512 Arbitrary Rotations*

	Percentage of Correct Alignments				Axis Alignment Measure α			
	Mean	Median	Min	Max	Mean	Median	Min	Max
Human	81.2	82.7	60.8	90.6	0.72	0.84	0.06	0.99
Dog	71.9	89.8	21.3	96.7	0.89	0.98	0.30	0.99
Plane	75.8	90.2	21.4	99.8	0.61	0.67	0.11	0.89
Head	52.8	63.0	0.0	99.4	0.71	0.89	0.02	0.99
Wine glass	45.9	51.9	13.8	65.3	0.66	0.72	0.02	1.00

☑

*** Each class contains 5 models → 10 alignment comparisons/class**
Statistics are computed over these 10 comparisons for each class

Concluding Remarks

- A computationally efficient correspondence-free shape alignment algorithm
- Minimizing the distance between shape descriptors solves the correspondence problem
- The permutation property enables fast look-up table based implementation:

  ~ 1 msec for a single alignment on a standard PC

- Extension to arbitrary 3D rotations has limited resolving power
- More involved optimization procedures can be pursued to recover finer 3D rotations.