Reconstruction de formes 3D

Julie Digne

LIRIS - Équipe GeoMod - CNRS

29 Septembre 2017
Introduction

- Each of these blocks is a challenge!
- Sampling of the existing methods

Thanks to Pierre Alliez and Misha Kazhdan for providing some of the slides.
Introduction: Acquisition of point clouds
3d surfaces typical challenges:
Cleaning the physical measure
3d surfaces typical challenges:
Registering and merging scans
3d surfaces typical challenges:
Orienting the point set
3d surfaces typical challenges: Building a mesh from a set of points

Shape courtesy of blender
Results of the acquisition process
Outline

1. Rigid Registration of partial depth maps
   - Registration from a close to optimal solution
   - Registration from an arbitrary position

2. Surface reconstruction
Point set registration

- Multiple scans from a single object
- Aggregate all the scans into a single model
- Rigid shape registration for scan alignment

Images from the "david laser scanner" website.
ICP: Iterative Closest Point [Chen, Medioni 91], [Besl, McKay 92]

- Optimize the pose of two scans
- Local optimization method
- Need a “close enough” initial position
- Inventory and comparison of all the variants in [Rusinkiewicz, Levoy 2001]
Hypothesis for the naive ICP

- Two “scans” $P$ and $Q$
- $\text{card}P = \text{card}Q$
- Each point $p_i$ of $P$ corresponds to a point $q_i$ of $Q$
- Find the transformation $T$ such that $\forall i T(p_i) = q_i$

Never the case for the problem of registering two partial scans in practice!
Transformation = Rotation + Translation (possibly a scaling)
- 4 parameters for the rotation (quaternion form, 3 by euler angles) et 3 for the translation
- 3 matching pairs (non colinear) give an estimation of the transform
- We never have access to points that are perfectly matched!
Given a set of matching pairs \((p_i)\) and \((q_i)\), find \(T\) minimizing the matching error:

\[
\sum_i \|p_i - Tq_i\|^2
\]
Can be seen as a generalization of the complex numbers to higher dimension.

\[ \dot{q} = q_0 + q_1i + q_2j + q_3k \]

Conjugate of a quaternion \( \dot{q}^* = q_0 - q_1i - q_2j - q_3k \)

Unitary quaternion \( \|\dot{q}\|^2 = \dot{q} \cdot \dot{q}^* = 1 \)

A rotation of axis \((w_x, w_y, w_z)\) and angle \(\theta\) can be seen as the quaternion:

\[
\cos \frac{\theta}{2} + \sin \frac{\theta}{2}(w_xi + w_yj + w_zk)
\]
Manipulating quaternions as matrices

- Vector in space corresponds to an imaginary quaternion \((q_0 = 0)\)
- Advantage: easier to work with than rotation matrices.
- The translation can be deduced [Horn 87]
Better: rotation estimation through SVD

Let \( P = (p_i)_{i=1}^{n} \) and \( Q = (q_i)_{i=1}^{n} \) such that \((p_i, q_i)\) is a matched pair.

Goal: Find \( R, t \) minimizing

\[
F(R, T) = \sum_{i=1}^{n} \|Rp_i + T - q_i\|^2.
\]

1. Centering \( \tilde{p}_i = p_i - \frac{1}{n} \sum_{i=1}^{n} p_i; \quad \tilde{q}_i = q_i - \frac{1}{n} \sum_{i=1}^{n} q_i. \)
2. Compute \( M = P \cdot Q^T \) and its svd \( M = USV^T \)
3. Compute

\[
R = V \begin{pmatrix}
1 & \vdots \\
1 & \ddots \\
& & 1 \\
& & & \text{det}(VU^T)
\end{pmatrix} U^T
\]

4. ... and

\[
T = \frac{1}{n} \sum_{i=1}^{n} q_i - R \left( \frac{1}{n} \sum_{i=1}^{n} p_i \right)
\]
The “standard” ICP algorithm

1. Start with two set of points $\mathcal{P}$ and $\mathcal{Q}$
2. Iterate
   1. For every point $p \in \mathcal{P}$ find the closest point $q \in \mathcal{Q}$
   2. Compute the transformation $T$ minimizing
      \[ \sum_i \| p_i - Tq_i \|^2 \]
3. $\mathcal{Q} \leftarrow T \mathcal{Q}$
Registration from an arbitrary position

- Two (or more) scans in arbitrary positions
- Hypothesis: the scans overlap
- Detect features on the scans and match them (similar as SIFT, FAST...)
- What is a good descriptor for the shape?
Scan matching through local descriptors

- Describe the local neighborhood of the surface
- Spin Images [Johnson 97], [Johnson-Hebert 99]; Snapshot descriptor [Malassiotis 2007]; Shape contexts [Belongie et al. 2002], Unique signature of histograms [Tombari et al. 2010]; Integral Invariants [Pottmann et al. 2007]
- RANSAC: select a subset of “agreeing” pairs
- Estimate a rigid transform from the set of matches.
What are the quality required from a good descriptor?

- Invariance by the transformation

\[ p \in S \quad D(p) = D(T(p)) \]

- Discrimination: two different points must have two different descriptors
- Locality of the description
- Resilience to sampling change
Why is it difficult?

- Choosing an intrinsic parameterization of the tangent plane
- Two umbilical points: same description
- If too local, the descriptor will not be discriminative
- If too wide, the descriptor might contain points not in the overlapping zone!

An example of unstable parameterization
A descriptor example: Spin Images [Johnson-Hebert 99]

- For each point \( p \in \mathcal{P} \) a spin map \( Sl_p \) associated
- \( Sl_p(q) = (\alpha_p(q), \beta_p(q)) \)
  - \( \alpha_p(q) \): distance from \( q \) to the tangent plane at \( p \)
  - \( \beta_p(q) \): distance from \( q \) to the line \( (p, \vec{n}(p)) \)
- **Spin Image**: resampling of the values on a grid
- Spin images comparison by linear correlation computation
RANSAC for transformation estimation

- RANdom SAmple Consensus
- A set of $N$ pairs $(p_i, q_i)$ with possible false matches
- Repeat $k$ times:
  - Select 3 pairs and estimate $T$
  - Compute the number of pairs agreeing with $T$
  - If the error is the lowest store $T$, the error and the consensus set.

With large $k$ the outliers can be well handled
Some guarantees about RANSAC

- $w = \frac{\text{#inliers}}{\text{#points}}$ probability of choosing an inlier pair among the $N$ pairs
- Probability of choosing 3 outliers $1 - w^3$
- Probability of always choosing 3 outliers: $(1 - w^3)^k$
- Probability of success $p$, we want it close to 1!

$$1 - p = (1 - w^3)^k$$

- $p = 0.99, w \approx 0.99$ yields $k \approx 2$
- $p = 0.99, w \approx 0.7$ yields $k \approx 11$
- $p = 0.99, w \approx 0.6$ yields $k \approx 19$
Global shape matching: 4—*points congruent point sets*

[Aiger Mitra Cohen-Or 2008]

- Three pairs of correspondences between $\mathcal{P}$ and $\mathcal{Q}$ uniquely define a rigid transform.
- *A special set* of 4-points simplifies the problem.
- Method for extracting all sets of coplanar 4-points from a 3D point set that are approximately congruent (related by rigid transforms) to a given planar 4-points.
- *Approximate congruence*: two 4-points sets can be aligned up to some allowed tolerance $\delta$. 

---

**Diagram:**

![Diagram of coplanar 4-points set](image)
Idea: Let $X = \{a, b, c, d\}$ be a set of 4 coplanar points (non colinear). Let $e$ be the intersection point between $ab$ and $cd$. Then the ratios

$$r_1 = \frac{\|a - e\|}{\|a - b\|} \quad \text{and} \quad r_2 = \frac{\|c - e\|}{\|c - d\|}$$

are affine invariants.
Global shape matching: 4—points congruent point sets
[Aiger Mitra Cohen-Or 2008]

- Input: Point sets $\mathcal{P}$ and $\mathcal{Q}$, a tolerance $\delta$, $f$ estimated overlap
- Repeat $N$ times
  1. Pick a base $B$ in $\mathcal{P}$
  2. Find all 4—points subsets $(U_i)_{i=1\ldots n}$ of $\mathcal{Q}$ congruent with $B$
  3. Compute the transforms $T_i$ aligning $B$ to $U_i$
  4. Rate the transforms $T_i$ and keep the best one
Global shape matching: 4—points congruent point sets
[Aiger Mitra Cohen-Or 2008]

Finding a coplanar base $B$
- Select 3 points randomly in $\mathcal{P}$
- Find a fourth point so that the four points are coplanar and form a wide base
- *wide* base result in stabler alignments.
- Compute the affine invariants $r_1$ and $r_2$
Global shape matching: *4–points congruent point sets*

[Aiger Mitra Cohen-Or 2008]

Finding all $4–points (U_i)_{i=1 \ldots n}$ of $Q$ congruent with $B$

- For each point pairs $q_1$ and $q_2$ of $Q$
- Compute the intermediate points:

  $$
  e_1 = q_1 + r_1(q_2 - q_1)
  $$
  $$
  e_2 = q_1 + r_2(q_2 - q_1)
  $$

- Any two pairs of points whose intermediate points coincide (one with $r_1$ and one with $r_2$) are congruent to $B$
- Place all points into a range tree (built in $O(n \log n)$), query for all points associated with $r_1$ in the $\delta$-neighborhood of a point associated with $r_2$. 
Global shape matching: 4—*points congruent point sets*
[Aiger Mitra Cohen-Or 2008]

Compute and rate the transform associated to $B$ and $(U_i)_{i=1\ldots n}$

- For each $U_i$ compute the best transform $T_i$ matching $B$ to $U_i$
- Compute $T_i(\mathcal{P})$ and find
  \[
  N_i = \#\{p \in T(\mathcal{P}) | \exists q \in \mathcal{Q}, dist(p, q) < \delta\}
  \]
- If $N_i <$ predefined threshold return $T_i$
Global shape matching: 4—*points congruent point sets* [Aiger Mitra Cohen-Or 2008]

- Robust to noise and outliers

- Handles partial matching even with small overlap between scans

- Fast!
Global shape matching: 4—points congruent point sets
[Aiger Mitra Cohen-Or 2008]
Is the rigid registration enough?

- Case where the scanner “deviates” (large objects...)
- Non-Rigid Range-Scan Alignment Using Thin-Plate Splines [Brown, Rusinkiewicz 2004]
- Global non-rigid alignment of 3-D scans [Brown, Rusinkiewicz 2007]
Outline

1. Rigid Registration of partial depth maps
   - Registration from a close to optimal solution
   - Registration from an arbitrary position

2. Surface reconstruction
For what purpose?

- Interpolating/Approximating?
- Closed surface reconstruction? Boundary preserving surface reconstruction?
- Smooth/piecewise smooth surface?
- Detail preservation/representation sparsity?

Different reconstruction methods depending on the application
Methods coming from computational geometry

- Convex Hulls...
- Crust, Eigencrust, powercrust
- Delaunay filtering
- $\alpha$-shapes
- Ball Pivoting Algorithm
A Delaunay Triangulation of $S$ is the set of all triangles with vertices in $S$ whose circumscribing circle contains no other points in $S^*$.

*Compactness Property:* this is a triangulation that maximizes the min angle.
The Voronoi Diagram of $S$ is a partition of space into regions $V(p)$ ($p \in S$) such that all points in $V(p)$ are closer to $p$ than any other point in $S$.

For a vertex, we can draw an empty circle that just touches the three points in $S$ around the vertex.

Each Voronoi vertex is in one-to-one correspondence with a Delaunay triangle.
Space partitioning

- Given a set of points, we can construct the Delaunay triangulation
- Label each triangle/tetrahedron as inside/outside
- Reconstruction = set of edges/facets that lie between inside and outside triangles/tetrahedra
- Different ways of assigning the labels [Boissonat 84], tight cocoone [Dey Goswami 2003], Powercrust [Amenta et al. 2001] Eigencrust [Kolluri et al. 2004]
The Crust Algorithm [Amenta et al. 1998]

- If we consider the Delaunay Triangulation of a point set, the shape boundary can be described as a subset of the Delaunay edges.
- How do we determine which edges to keep?
- Two types of edges:
  - Those connecting adjacent points on the boundary
  - Those traversing the shape
- Discard those that traverse the shape
The Crust Algorithm [Amenta et al. 1998]

In 2D:
- Given a point set $S$ compute its Voronoi diagram and Voronoi vertices $V$
- Compute the Delaunay triangulation of $S \cup V$
- Keep only edges that connects points in $S$ (eq. to Keeping all edges for which there is a circle that contains the edge but no Voronoi vertices)

In 3D: Not all Voronoi Vertices are added to the set. Only the poles (furthest points of the Voronoi cell) are considered.
Ball Pivoting Algorithm

- BPA computes a triangle mesh interpolating a given point cloud
- Three points form a triangle if a ball of a user-specified radius $\rho$ touches them without containing any other point
- Start with a seed triangle
- The ball pivots around an edge until it touches another point, forming another triangle
- Expand the triangulation over all edges then start with a new seed
Algorithm

- Advancing front triangulation
- Front is a set of edges
Different types of expansion

(f) Expansion case

(g) Gluing case

(h) Hole filling case

(i) Ear filling case
Rotating the sphere
Finding the $R$-circumsphere

The circumcenter $H$ of triangle $ABC$ has barycentric coordinates:

$$(a^2(b^2 + c^2 - a^2), b^2(a^2 + c^2 - b^2), c^2(a^2 + b^2 - c^2))$$

The square circumradius is

$$R^2 = \frac{a^2 \cdot b^2 \cdot c^2}{(a + b + c) \cdot (b + c - a) \cdot (c + a - b) \cdot (a + b - c)}.$$
Finding the $R$-circumsphere

Such a sphere exists only if $R_b^2 - R^2 \geq 0$.

Let us denote by $\mathbf{n}$ the normal to the triangle plane, oriented such that it has a nonnegative scalar product with the vertices normals. Provided $R_b^2 - R^2 \geq 0$ (hence the sphere existence), the center $O$ of the sphere can be found as:

$$O = H + \sqrt{R_b^2 - R^2} \cdot \mathbf{n}.$$
Properties and Guarantees of the resulting mesh

- The surface is guaranteed to be self-intersection free (no triangle will intersect each other except at an edge or vertex, and at most two triangles can be adjacent to an edge).
- Normal coherence on a facet.
- For each triangle there exists an empty ball incident to the three vertices with empty interior.
Detailed area
Detailed area
Detailed area
Detailed area
Detailed area
Detailed area
Detailed area
Detailed area
Detailed area
Smaller ball radius
Smaller ball radius
Smaller ball radius
Smaller ball radius
Smaller ball radius
Smaller ball radius
Smaller ball radius
Smaller ball radius
Smaller ball radius
Smaller ball radius
Smaller ball radius
Smaller ball radius
Figure: Radius too small: areas with lower density are not triangulated. Large radius: higher computation times + detail loss.
**Figure:** Reconstructing the Stanford Bunny point cloud, with a single radius (0.0003), two radii (0.0003; 0.0005) and three radii (0.0003; 0.0005; 0.002).
<table>
<thead>
<tr>
<th>Radius</th>
<th>Time(s)</th>
<th>vertices</th>
<th>facets</th>
<th>boundary edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0003</td>
<td>10s</td>
<td>318032</td>
<td>391898</td>
<td>272832</td>
</tr>
<tr>
<td>0.0003; 0.0005</td>
<td>21s</td>
<td>356252</td>
<td>698963</td>
<td>22727</td>
</tr>
<tr>
<td>0.0003; 0.0005; 0.002</td>
<td>29s</td>
<td>361443</td>
<td>713892</td>
<td>7897</td>
</tr>
</tbody>
</table>

Surface reconstruction
Figure: Detail loss and hole creation due to a too large radius (left) and a too small one (middle). A possible solution is to use multiple radii (right).
Figure: Applying the ball pivoting to a noisy sphere: $r = 0.05$ (left) and $r = 0.02; 0.03; 0.05$ (right). A single radius does not allow to interpolate the input data and applying multiple radii is not a solution in addition to being difficult to tune.
Figure: Bunny and Dragon reconstruction
Problems and solutions

- The larger the ball radius the slower the computation
- The larger the ball radius the more details will be lost
- The smaller the ball radius the more dependent on the sampling
- Varying ball radius $\leftarrow$ slow down the process
- Use of a *scale space*: a multiscale representation of the point cloud.
Summary: Advantages/Drawbacks of the ball pivoting

**Drawbacks**
- Size of the ball?
- No suppression of redundant points
- No hole closure

**Advantages**
- Control on the size of the triangles created
- Radius of the ball determines what is a hole
- Surface boundary preservation

Modification through the use of a *scale space* for better detail preservation.
Implicit surface reconstruction - Level set methods

- See the surface as an isolevel of a given function
- Extract the surface by some contouring algorithm: Marching cubes [Lorensen Cline 87], Particle Systems [Levet et al. 06]
Surface reconstruction from unorganized points
[Hoppe et al. 92]

- **Input:** a set of 3D points
- **Idea:** for points on the surface the signed distance transform has a gradient equal to the normal
  \[ F(p) = \pm \min_{q \in S} \| p - q \| \]
- 0 is a regular value for \( F \) and thus the isolevel extraction will give a manifold
- Compute an associated tangent plane \((o_i, n_i)\) for each point \( p_i \) of the point set
- Orientation of the tangent planes as explained before.
Surface reconstruction from unorganized points

[Hoppe et al. 92]

- Once the points are oriented
- For each point $p$, find the closest centroid $o_i$
- Estimated signed distance function: $\hat{f}(p) = n_i \cdot (p - o_i)$
Poisson Surface Reconstruction [Kazhdan et al. 2006]

- Input: a set of oriented samples
- Reconstructs the indicator function of the surface and then extracts the boundary.
- Trick: Normals sample the function’s gradients
Poisson Surface Reconstruction [Kazhdan et al. 2006]

1. Transform samples into a vector field
2. Fit a scalar-field to the gradients
3. Extract the isosurface
Poisson Surface Reconstruction [Kazhdan et al. 2006]

- To fit a scalar field $\chi$ to gradients $\vec{V}$, solve:

$$\min_{\chi} \|\nabla \chi - \vec{V}\|$$

- Eq to:

$$\nabla \cdot (\nabla \chi) - \nabla \cdot \vec{V} = 0 \Leftrightarrow \Delta \chi = \nabla \cdot \vec{V}$$
Gradient Function of an indicator function = unbounded values on the surface boundaries

We use a smoothed indicator function

**Lemma**

The gradient of the smoothed indicator function is equal to the smoothed normal surface field.

\[
\nabla \cdot (\chi \ast \tilde{F})(q_0) = \int_{\partial M} \tilde{F}(q_0 - p) \cdot \tilde{N}_{\partial M}(p) dp
\]

Chicken and Egg problem: to compute the gradient one must be able to compute an integral over the surface!!
Approximate the integral by a discrete summation

Surface partition in patches $\mathcal{P}(s)$:

$$\nabla \cdot (\chi \ast \tilde{F})(q_0) = \sum_s \int_{\mathcal{P}(s)} \tilde{F}(q_0 - p) \cdot \vec{N}_{\partial M}(p) dp$$

Approximation on each patch:

$$\nabla \cdot (\chi \ast \tilde{F})(q_0) = \sum_s |\mathcal{P}(s)| \tilde{F}(q_0 - s) \cdot \vec{N}(s)$$

Let us define $V(q_0) = \sum_s |\mathcal{P}(s)| \tilde{F}(q_0 - s) \cdot \vec{N}(s)$
Problem Discretization

- Build an adaptive octree $O$
- Associate a function $F_o$ to each node $o$ of $O$ so that: $F_o(q) = F\left(\frac{q-o.c}{o.w}\right)\frac{1}{o.w^3}$ ($o.c$ and $o.w$ are the center and width of node $o$). ⇒ multiresolution structure
- The base function $F$ is the $n$th convolution of a box filter with itself

$$\tilde{V}(q) = \sum_{s \in S} \sum_{o \in N(s)} \alpha_{o,s} F_o(q)s \cdot \vec{N}$$

- Look for $\chi$ such that its projection on $\text{span}(F_o)$ is closest to $\nabla V$:
- Minimize $\sum_{o \in O} \langle \Delta \chi - \nabla \cdot V, F_o \rangle^2$
- Extracted isovalue: mean value of $\chi$ at the sample positions
Varying octree depth
Varying octree depth
Varying octree depth
Resilience to bad normals

Image from Mullen et al. Signing the unsigned, 2010
detail preservation

Poisson  BPA  Scale Space + BPA
Advantages and drawbacks of the Implicit surface reconstruction methods

<table>
<thead>
<tr>
<th>Drawbacks</th>
<th>Advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only semi-sharp, loss of details</td>
<td>Noise robustness</td>
</tr>
<tr>
<td>Final mesh not interpolating the initial pointset</td>
<td>Watertight surface, hole closure</td>
</tr>
<tr>
<td>Marching cubes introduces artefacts</td>
<td>Most standard way of reconstructing a surface</td>
</tr>
<tr>
<td>Watertight surface, very bad with open boundaries</td>
<td></td>
</tr>
</tbody>
</table>
From the signed distance function to the mesh

- At each point in $\mathbb{R}^3$, the signed distance function to the surface can be estimated.
- Extract the 0 levelset of this function: points where this function is 0.

Approximation

Evaluate the function at the vertices of a grid and deduce the local geometry of the surface in each grid cube.
Example in 2D

Images by Ben Anderson
Example in 2D

Images by Ben Anderson
Example in 2D

Images by Ben Anderson
Example in 2D

Images by Ben Anderson
Example in 2D

Images by Ben Anderson
Algorithm in 2D

- Build a grid in the object bounding box
- Evaluate the implicit function $f$ at each grid vertices
- If an edge has its two endvertices $p_i$, $p_j$ such that $s(p_i)s(p_j) < 0$
- Interpolate the intersection point of the isolevel with the grid edge.
- Link the intersection points by lines
From Marching Squares to Marching Cubes

Drawing lines between intersection points is ambiguous and does not give a surface patch.

*Images by Ben Anderson*
Look-up tables

- There are $2^8 = 256$ possible cases for cube corner values.
- By symmetry + rotation arguments it reduces to 15 cases.
- It is thus possible to build a look-up table giving the grid cell triangulation based on the corner values case.
Ambiguous cases

\( V_a \)

\( V_c \)

\( V_c \)

\( V_b \)
Ambiguous cases

(a)

(b)
Ambiguous cases

- Refine the grid to remove ambiguity
- Switch to marching tetrahedra algorithm