PCL :: Registration

Initial alignment of point clouds

Refining initial alignments

September 25, 2011

pointcloudlibrary

Outline

1. Registration
2. Registration using the Iterative Closest Points (ICP)
3. Feature-Based Initial Alignment
4. Example/Tutorial Code

pointcloudlibrary

Registration

Wanted: transformation that aligns one point cloud to another.

- for initially aligning point clouds (based on features)
- for refining initial alignments
  (using the Iterative Closest Point (ICP) Algorithm)

pointcloudlibrary

ICP (1/6)

Registration using the Iterative Closest Point (ICP) Algorithm

Given an input point cloud and a target point cloud

1. determine pairs of corresponding points,
2. estimate a transformation that minimizes the distances between the correspondences,
3. apply the transformation to align input and target.

pointcloudlibrary

ICP (2/6)

- Given:
  Two n-dimensional sets of points – the *model set* $M = \{\mathbf{m}_i | \mathbf{m}_i \in \mathbb{R}^n, i = 1, \ldots, N_m\}$ and the *data set* $D = \{\mathbf{d}_j | \mathbf{d}_j \in \mathbb{R}^n, j = 1, \ldots N_d\}$
- Wanted:
  A rotation R and a translation Δt that map D onto M.
- handled as an optimization problem $\Longrightarrow$ Minimizing mapping error E

$$E(R, \Delta t) = \sum_{i=1}^{N_m} \sum_{j=1}^{N_d} w_{i,j} \left|\left|\mathbf{m}_i - (R\mathbf{d}_j + \Delta t)\right|\right|^2 \quad (1)$$

- Weighting factor $w_{i,j}$ encodes point correspondences, $w_{i,j} = 1$ for correspondence$(\mathbf{m}_i, \mathbf{d}_i)$, 0 otherwise
- Correspondence from Neighbor Search

pointcloudlibrary

ICP (3/6)

The ICP API

```
pcl::IterativeClosestPoint<InType, OutType> icp;
```

Provide a pointer to the input point cloud

```
icp.setInputCloud (input_cloud);
```

Provide a pointer to the target point cloud

```
icp.setInputTarget (target_cloud);
```

Align input to target to obtain

```
icp.align (aligned_cloud);
Eigen::Matrix4f transformation = icp.
getFinalTransformation ();
```

- the aligned cloud (transformed copy of input cloud),
- and transformation used for alignment.

pointcloudlibrary

ICP (4/6)

The ICP API: Termination Criteria

- Max. number of iteration steps
  → set via `setMaximumIterations(nr_iterations)`
- Convergence: Estimated transformation doesn't change (the sum of differences between current and last transformation is smaller than a user-defined threshold)
  → set via `setTransformationEpsilon(epsilon)`
- A solution was found (the sum of squared errors is smaller than a user-defined threshold)
  → set via `setEuclideanFitnessEpsilon(distance)`

pointcloudlibrary

ICP (5/6)

The ICP API: Problems with ICP

False correspondences negatively affect the alignment (the algorithms gets caught in local minima).

- Maximum distance between correspondences:

```
icp.setMaxCorrespondenceDistance (distance);
```

- Use RANSAC to neglect false correspondences

```
icp.setRANSACOutlierRejectionThreshold (distance);
```

  - Model: Transformation (estimated on 3 samples)
  - Inliers: Points whose distance to the corresponding is below the given threshold

pointcloudlibrary

ICP (6/6)

The ICP API: Problems with ICP

The ICP algorithms needs a rough initial alignment.

- Use Features to get an initial alignment!

→ `pcl::SampleConsensusInitialAlignment`

pointcloudlibrary

Initial Alignment (1/7)

1. Compute sets of keypoints
2. Compute (local) feature descriptors (e.g. FPFH)
3. Use SAC-based approach to find initial alignment
   - 3.1 Take 3 random correspondence pairs
   - 3.2 Compute transformation for these pairs
   - 3.3 Apply transformation to all source points, and determine inliers
4. Use best transformation for initial alignment, and ICP for refinement

*

pointcloudlibrary

Initial Alignment (2/7)

points on similar surfaces

*

pointcloudlibrary

Initial Alignment (3/7)

*

pointcloudlibrary

Initial Alignment (4/7)

Outdoor Example: Non-Linear Optimization

*

pointcloudlibrary

Initial Alignment (5/7)

pointcloudlibrary

Initial Alignment (6/7)

pointcloudlibrary

Initial Alignment (7/7)

The SampleConsensusInitialAlignment API

```
pcl::SampleConsensusInitialAlignment<PointT, PointT,
DescriptorT> sac;
```

Provide a pointer to the input point cloud and features

```
sac.setInputCloud (source_points);
sac.setSourceFeatures (source_descriptors);
```

Provide a pointer to the target point cloud and features

```
sac.setInputTarget (target_points);
sac.setTargetFeatures (target_descriptors);
```

Align input to target to obtain

```
sac.align (aligned_cloud);
Eigen::Matrix4f transformation = sac.
getFinalTransformation ();
```

pointcloudlibrary

Example/Tutorial Code (1/5)

Initial alignment of point clouds

```
pcl::SampleConsensusInitialAlignment<PointT, PointT,
    DescriptorT> sac;
sac.setMinSampleDistance (min_sample_distance);
sac.setMaxCorrespondenceDistance (
    max_correspondence_distance);
sac.setMaximumIterations (nr_iterations);

sac.setInputCloud (source_points);
sac.setSourceFeatures (source_descriptors);

sac.setInputTarget (target_points);
sac.setTargetFeatures (target_descriptors);

PointCloudPtr aligned_source(new PointCloud);
sac.align (*aligned_source);
Eigen::Matrix4f initial_T = sac.getFinalTransformation
    ());
```

pointcloudlibrary

Example Code (2/5)

Refining initial alignments using ICP

```
pcl::IterativeClosestPoint<PointT, PointT> icp;
icp.setMaxCorrespondenceDistance (distance);
icp.setRANSACOutlierRejectionThreshold (distance);
icp.setTransformationEpsilon (transformation_epsilon);
icp.setMaximumIterations (max_iterations);

icp.setInputCloud (aligned_source); // from (1)
icp.setInputTarget (target_points);

PointCloud registration_output;
icp.align (registration_output);

Eigen::Matrix4f refined_T =
     icp.getFinalTransformation () * initial_T);
```

pointcloudlibrary

Example Code (3/5)

Example: Correspondences in initial alignment

```
$ ./correspondence_viewer ../../data/robot/robot0
../../data/robot/robot1 -n 5
```

pointcloudlibrary

Example Code (4/5)

Example: Initial alignment + refinement

```
$ ./test_registration ../../data/robot/robot0 ../../
data/robot/robot1 -i 0.025,0.01,500 -r 0.05,0.05,0,100
```

pointcloudlibrary

Example Code (5/5)

Example: Final model = robot0 + robot1

```
$ ./test_registration ... -s robot_registered.pcd
$ pcd_viewer robot_registered.pcd
```