agenda

- Introduction
- ParticleFilter
- Evaluation Function
- Parallelization
- Optimization
- Implementation
- Results
Why tracking?

► 30 fps depth sensors
► Helping object recognition in real-time.
Particle Filter

\[
\begin{align*}
X_{t-1|t-1} & \\
x_t^{(i)} & \\
x_{t-1|t-1} & \\
\end{align*}
\]

Prediction:

\[
X_t|t-1
\]

Weighting:

\[
\begin{align*}
w_t^{(i)} &= R_t(y_t|x_t^{(i)}|t-1) \\
y_t & \\
\end{align*}
\]

Resampling:

\[
X_t|t
\]

[T.Higuchi, Particle Filter, '05]
Particle Filter

\[ x = \{ p, r \} \]

Prediction

Weighting

Resampling

Point Cloud from Depth Sensor

Evaluation Function

\[ w_t^{(i)} = R_t(y_t | x_{t|t-1}^{(i)}) \]

\[ X_{t-1|t-1} \quad X_{t|t-1} \quad X_{t|t-1} \quad X_t | t \]
\[ w^{(i)} = \sum_j L_{distance}(p_j, q_j) L_{color}(p_j, q_j) \]

\[ L_{distance}(p_j, q_j) = \frac{1}{1 + \alpha |p_j - q_j|^2} \]

\[ L_{color}(p_j, q_j) = \frac{1}{1 + \beta |p_{jcolor} - q_{jcolor}|^2} \]

\( p_j \): a point of the hypothesis pointcloud

\( q_j \): the nearest point of the input pointcloud to \( p_j \)
Evaluation Function

\[ x^{(i)}_{t-1|t-1} \]
Evaluation Function

\[ x^{(i)}_{t-1|t-1} \]
Evaluation Function

\[ x^{(i)}_{t-1|t-1} \]
Evaluation Function

\[ l_j = L_{distance}(p_j, q_j)L_{color}(p_j, q_j) \]

\[ L_{distance}(p_j, q_j) = \frac{1}{1 + \alpha |p_j - q_j|^2} \]

\[ L_{color}(p_j, q_j) = \frac{1}{1 + \beta |p_{j\text{color}} - q_{j\text{color}}|^2} \]
\[ w_t^{(i)} = \sum l_j \]

\[ x_{t-1|t-1} \]
Evaluation Function

\[ w_t^{(i)} = \sum_{j} l_j \]

\[ p_j \rightarrow q_j \]

\[ x_{t-1|t-1}^{(i)} \]
Parallelization

Benefit of Particle Filter

► Easy to parallelize

- single core
- 8 cores (hyperthreaded)

2~6x faster

1mm downsampled
\( \mathcal{O}(NM \log n) \)

- \( N \): the number of reference points
- \( M \): the number of particles
- \( n \): the number of measured points
Downsampling

$O(NM \log n)$

- Downsampling!

1mm downsampling

2mm downsampling

4~5x faster
$O\left( NM \log n \right)$

- Adaptive Particle Filter
- KLD Sampling [D. Fox, IJRR03]

$$M = \frac{1}{2\epsilon} \chi^2_{k-1,1-\delta}$$

$$= \frac{k-1}{2\epsilon} \left\{ 1 - \frac{2}{9(k-1)} + \sqrt{\frac{2}{9(k-1)}} z_{1-\delta} \right\}^3$$
How can we decrease the number of input points?
Optimizing Octree and Kdtree
Optimizing Octree and KdTree
Optimizing Octree and Kdtree
Optimizing Octree and Kdtree
Optimizing Octree and Kdtree
Optimizing Octree and Kd-tree
Optimizing Octree and Kdtree
Implementation

Tracker
  - ParticleFilterTracker
    - ParticleFilterOMPTracker
  - KLDAdaptiveParticleFilterTracker
    - KLDAdaptiveParticleFilterOMPTracker
  - PointCloudCoherence
    - NearestPairPointCloudCoherence
      - ApproxNearestPairPointCloudCoherence
  - PointCoherence
    - DistanceCoherence
    - HSVColorCoherence
    - NormalCoherence
Here comes a tracking library to PCL!

Our approach is based on Particle Filter

- 6-D pose tracking
- Optimized towards real-time tracking

Future works
- Implementing on GPU

Videos:
- http://youtube.com/garaemon1