pointcloudlibrary

KARLSRUHE ICRA 2013

Ensemble Learning for Object Recognition and Pose Estimation

Zoltan-Csaba Marton
German Aerospace Center (DLR)

May 10, 2013

pointcloudlibrary

Outline

pointcloudlibrary

Outline

pointcloudlibrary

Introduction

Why use "ensembles of experts" for object recognition?

- No silver bullet for all cases, different methods have complementing strengths
- Combining results from various sensors, segmentations, classifications, pose estimations is becoming a popular approach, thanks to the added robustness
- Methods of varying complexity already have been developed in the machine learning field
- This tutorial will present some application scenarios, evaluations and tutorial on methods that are relevant to 3D object recognition

pointcloudlibrary

Outline

pointcloudlibrary

Overview of Methods

Most gain for combinations of uncorrelated results

Many things can be used to produce complementing information:

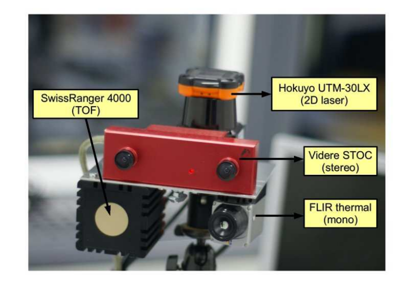
- Decision Trees and Ferns in PCL::ML, see IROS'12 tutorial
- Nearest neighbor classification class pcl::NNClassification using FLANN, see: `apps/src/nn_classification_example.cpp`
- local and global features for pose estimation and categorization, evaluated in [Aldoma et al. RAM'12]
- various segmentation methods, etc.

pointcloudlibrary

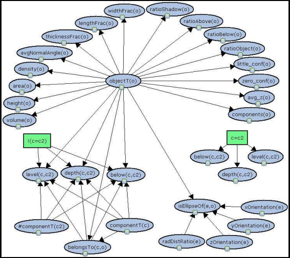
Outline

pointcloudlibrary

Complementing Modalities



[Marton et al. IROS'09]

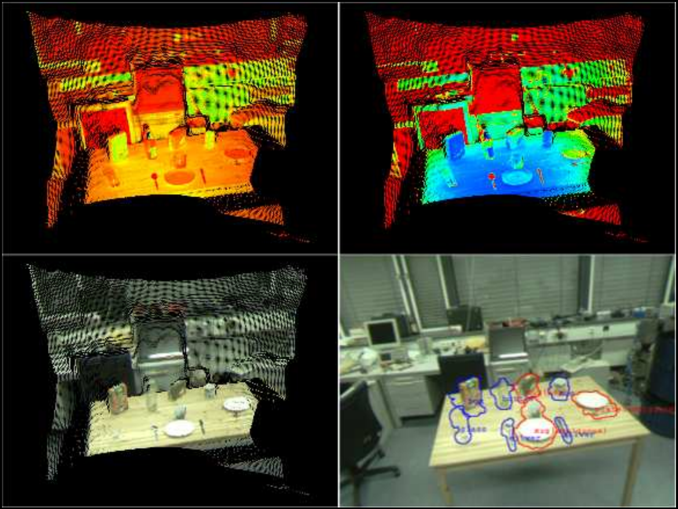


- SR4K: intensity, depth, confidence
- STOC: RGB, visual odometry
- probabilistic integration using a Bayesian Logic Network [Jain '09]

pointcloudlibrary

Complementing Modalities

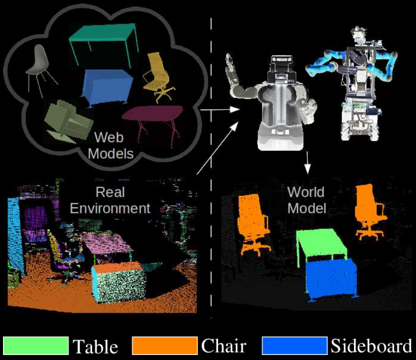
Multiple cues and logic ⇒ incorrect detections are not far off



1. intensity
2. confidence
3. RGB-D
4. blue: correct, red: incorrect (ground truth in parenthesis)

pointcloudlibrary

Unsupervised Part Learning



[Mozos et al. RAM'11]

PCL-based implementation:

`http://www.ros.org/wiki/furniture_classification`

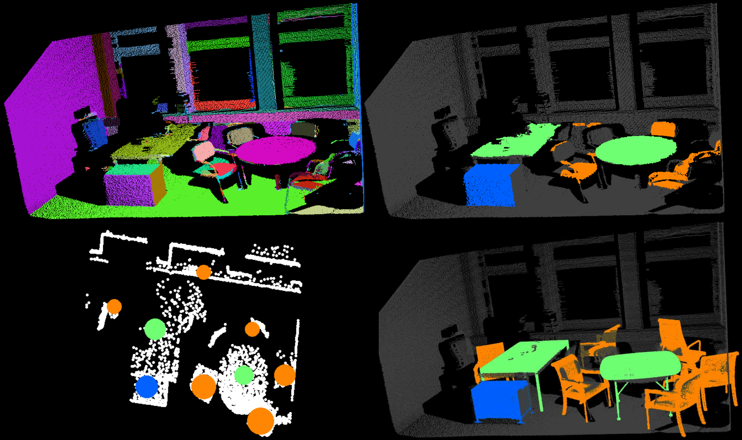
[by Vlad Usenko]

- Identification of furniture pieces for which similar CAD models are available from online stores.
- Common parts are grouped into a codebook based on simple statistics and they cast votes for object hypotheses
- Pose estimation by model matching and geometric verification.
- Can increase robustness by using multiple segmentations or views.

pointcloudlibrary

Unsupervised Part Learning

Results for Office

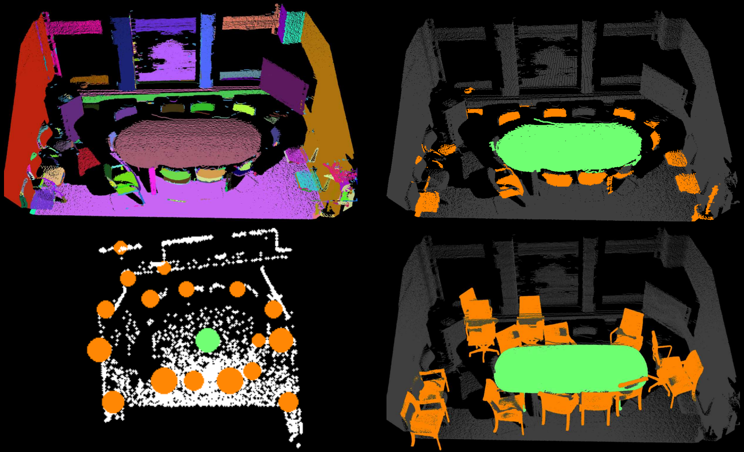


- remaining false matches due to high occlusions

pointcloudlibrary

Unsupervised Part Learning

Results for Seminar Room

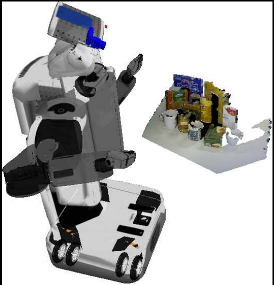


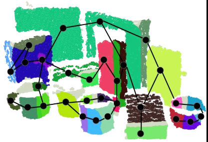
- remaining false matches due to high occlusions

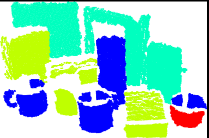
pointcloudlibrary

Scene Subgraphs

Considering all possible part groupings







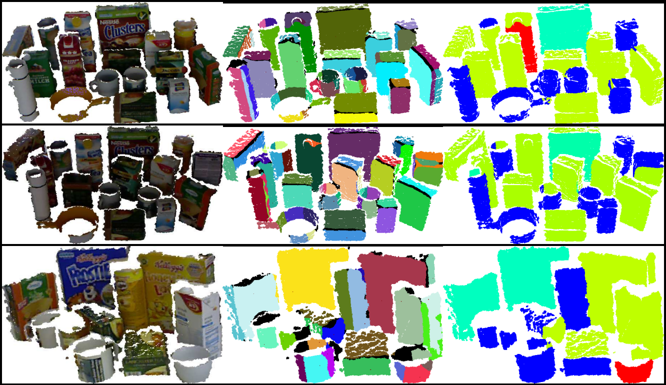
- Object categorization in cluttered scenes where accurate segmentation can be difficult.
- Over-segmentation and multiple hypotheses better than relying on a single, possibly bad segmentation.
- Approach based on scene- or part-graphs, using additive RGBD feature descriptors.

[Marton et al. SC'12]

pointcloudlibrary

Scene Subgraphs

Segmentation and classification on cluttered table scenes

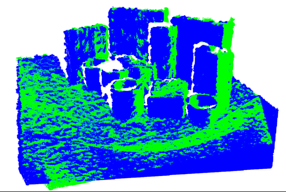


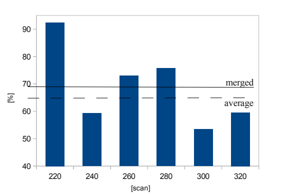
Object parts are segmented and categorized as spherical, box, flat and cylindrical (training and large-scale testing done using the RGB-D Object Dataset [Lai et al. ICRA'11]).

pointcloudlibrary

Exploiting Embodiment

Scenes From Multiple Views

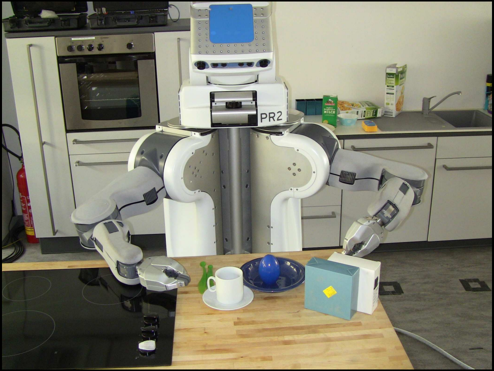


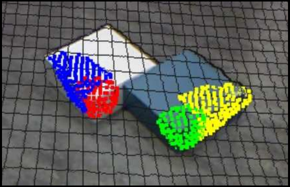


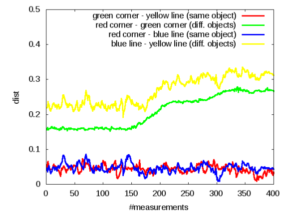
As the camera is moved (left), multiple frames can be captured that cover different parts of the objects in the scene (right).

pointcloudlibrary

Exploiting Embodiment







Detecting when and where to push and tracking 3D features in order to segment objects (using openni_tracking from U-Tokyo):

`http://www.ros.org/wiki/interactive_segmentation_textureless`

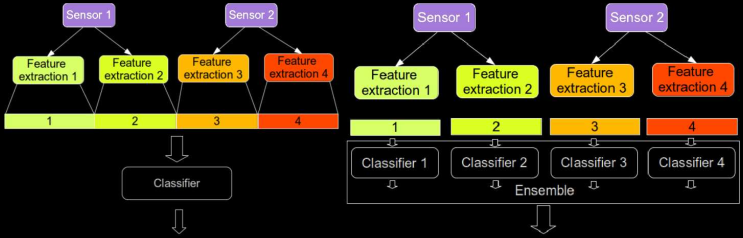
[Bersch et al. RSS'12/WS, Hausmann et al. ICRA'13]

pointcloudlibrary

Outline

pointcloudlibrary

Ensemble Learning



(a) Concatenation (b) Ensemble

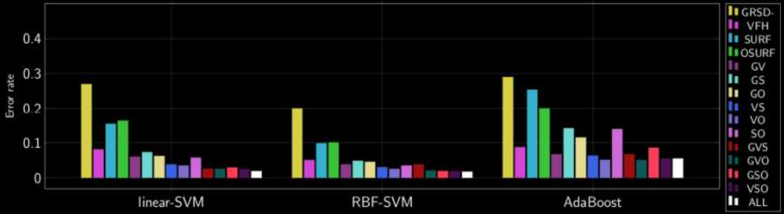
- Instead of monolithic classifiers on the concatenation of features use "ensembles of experts"
- Instead of weak learners use specialized strong learners
- Evaluated on the RGB-D Object Dataset [Lai et al. ICRA'11]

[Marton et al. PRL'12]

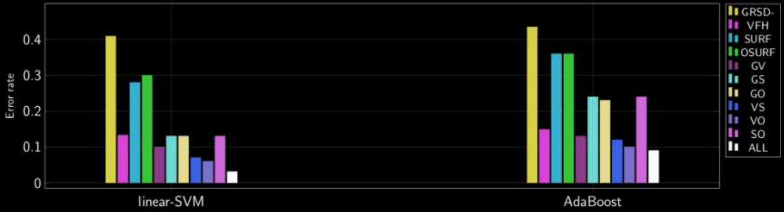
pointcloudlibrary

Ensemble Learning

Evaluation of 3D and RGB features – and of their combinations



(a) Error rate for the 20 classes problem



(b) Error rate for the 51 classes problem

pointcloudlibrary

Ensemble Learning

Final decision made based on the result of separate classifiers

Max accuracy: classifier with highest class-conditional accuracy

$$f(x) = d_{\arg\max_i p(a=d_i(x)|e_i=d_i(x))}(x) \tag{1}$$

Max confidence: classifier with highest confidence in decision

$$f(x) = d_{\arg\max_i p(e_i=d_i(x)|x)}(x) \tag{2}$$

Combined max confidence and accuracy: multiply the two

$$f(x) = d_{\arg\max_i p(a=d_i(x)|e=d_i(x))\cdot p(e_i=d_i(x)|x)}(x) \tag{3}$$

Product: Bayes rule, assuming independence of the decisions

$$f(x) = \arg\max_y \; p\,(a = y | c_1 = d_1(x), \ldots, c_M = d_M(x)) \tag{4}$$

$$\propto \prod_{i=1}^{|E|} p\,(c_i = y_i | a = y)\, p\,(a = y) \tag{5}$$

pointcloudlibrary

Ensemble Learning

Voting: using equal weights for each classifier

$$f(x) = \arg\max_y \sum_{i=1}^{|E|} I\left(d_i(x) = y\right) \tag{6}$$

Accuracy weighted voting: weighting by classifier accuracy

$$f(x) = \arg\max_y \sum_{i=1}^{|E|} I\left(d_i(x) = y\right) p\left(a = d_i(x) | e_i = d_i(x)\right) \tag{7}$$

Confidence weighted voting: weighting by classifier confidence

$$f(x) = \arg\max_y \sum_{i=1}^{|E|} I\left(d_i(x) = y\right) p\left(e_i = d_i(x) | x\right) \tag{8}$$

Confidence and accuracy weighted voting: like above

Error Correlation: closed form solution for the weights

pointcloudlibrary

Ensemble Learning

Simple rule ensembles don't require joint training

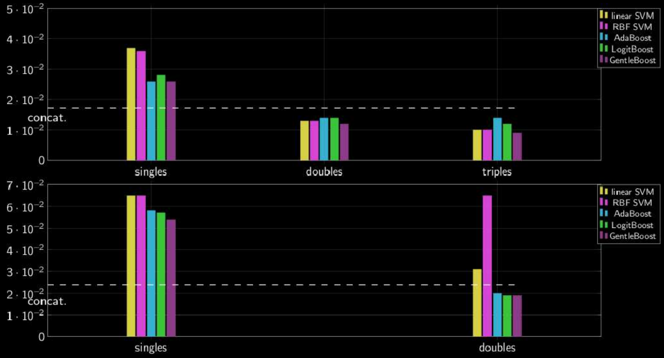
- Error rate for 20 classes with (from top to bottom) AdaBoost, linear SVM and SVM with RBF kernel.
- Accuracy and confidence based voting are the best simple rules.
- Hybrid model of simple rule ensemble trained on two element feature sets balances training time, accuracy, and modularity.

pointcloudlibrary

Ensemble Learning

Output of feature classifiers used as input for a new classifier

Improving on the best concatenation for 51 classes by stacking:



Stacking with AdaBoost as level-0 classifier and various level-1 classifiers, for 20 classes (top) and 51 classes (bottom)

pointcloudlibrary

Outline

pointcloudlibrary Representations and Mappings

Rotations in SO(3), available as Eigen classes:

- 3x3 rotation matrix (Matrix3f, Matrix3d)
  - describing the 3D axes of the new coordinate system
  - $R^{-1} = R^t$ and $det(R) = 1$

$$\mathbf{R} = \begin{bmatrix} \hat{\mathbf{u}}_x & \hat{\mathbf{v}}_x & \hat{\mathbf{w}}_x \\ \hat{\mathbf{u}}_y & \hat{\mathbf{v}}_y & \hat{\mathbf{w}}_y \\ \hat{\mathbf{u}}_z & \hat{\mathbf{v}}_z & \hat{\mathbf{w}}_z \end{bmatrix}$$

pointcloudlibrary Representations and Mappings

Rotations in SO(3), available as Eigen classes:

- 3x3 rotation matrix (Matrix3f, Matrix3d)
  - describing the 3D axes of the new coordinate system
  - $R^{-1} = R^t$ and $det(R) = 1$
- Angle-axis (AngleAxisf, AngleAxisd)
  - Eigen uses negative angles too!
- Quaternions (Quaternionf, Quaterniond)
  - are a double mapping of SO(3)
  - each rotation has two quaternion representations (q==-q)
  - (w, x, y, z) coordinates are points on 4D sphere (norm is 1)
  - (x, y, z) is the axis and $\alpha = 2acos(w)$ the rotation around that axis

pointcloudlibrary Representations and Mappings

You can convert between them, and perform operations easily using Eigen (multiplications, inversions, norms, etc.):

```
Quaterniond q (Matrix3d::Identity());
Matrix3d mat = q.toRotationMatrix();
double angle = AngleAxisd(q).angle();
```

In some cases it is more efficient to compute it yourself:

```
angle = 2*std::atan2(q.vec().norm(), q.w());
angle = acos((transformation_matrix<3,3>(0,0).trace() - 1)/2)
```

Make sure angle is between 0 and π (invert axis too if needed).

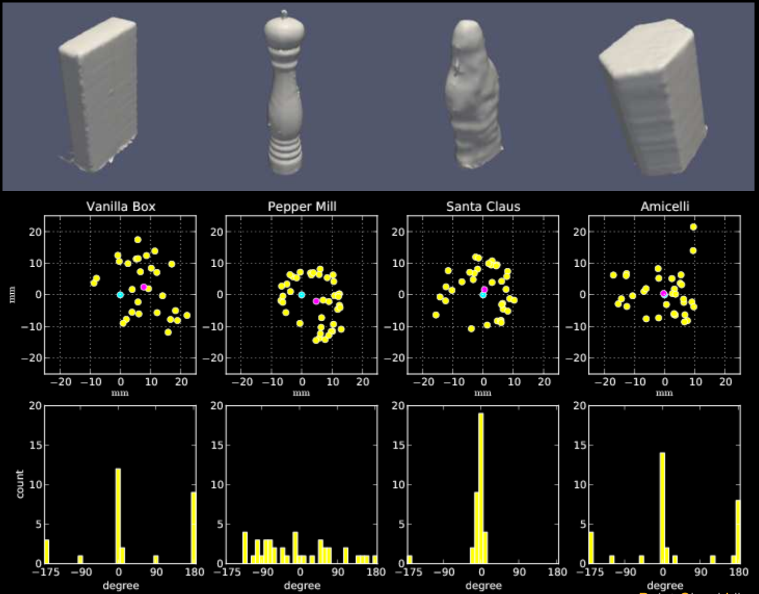
Further reading:

http://eigen.tuxfamily.org/dox/TutorialGeometry.html

http://en.wikipedia.org/wiki/Rotation_formalisms_in_three_dimensions

pointcloudlibrary

Pose Distances and Means



pointcloudlibrary

Pose Distances and Means

Discussion on the alternatives

Distance between translations is clearly the Euclidean distance: $||t_1 - t_2||_2$

Distances between rotations:

- The extrinsic (Euclidean, arithmetic) distance is $||R_1 - R_2||_F = (R_1 - R_2).norm()$
  - corresponding mean has a closed form solution
  - would be intuitive, but don't use it
  - it is a bad approximation of the real distance in SO(3)
- The intrinsic (Riemann, geometric) distance
  - it is the arc length of the shortest geodesic curve between the two rotations
  - just like in 2D, this is the rotation angle of $R_1^{-1} \cdot R_2$ (computed as detailed earlier)

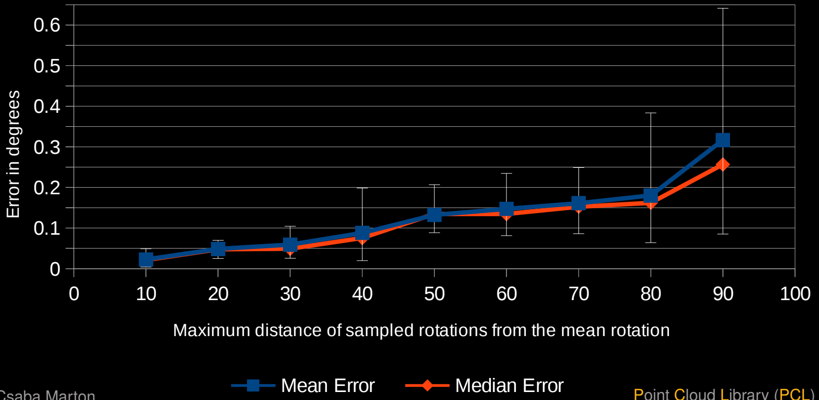
pointcloudlibrary

Pose Distances and Means

Arithmetic average of quaternions (normalized)

$$q_{avg} \approx QAA(q_{1,\dots,n}) = \left(\sum_{i=1}^{n} q_i\right) / norm \left(\sum_{i=1}^{n} q_i\right)$$

QAA approximation error (upper boound)



pointcloudlibrary

Pose Distances and Means

Combining rotational and translational differences (not recommended)

For distances between transformation matrices, you can use the rotation+translation part of the SRT Distance, analogous to an Euclidean distance [Pham et al. ICCV'11]:

$$d_{rt}(T_1, T_2) = \sqrt{d_r(R_1, R_2)^2 + (d_t(t_1, t_2)/\sigma)^2}$$

The paper proposes to use the Euclidean distance for $d_r(R_1, R_2)$, but the Riemann distance would be better (combined with the QAA mean).

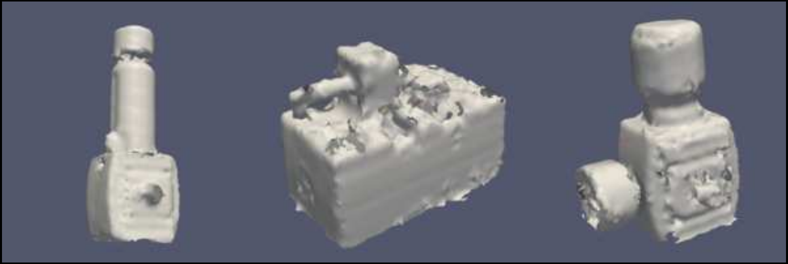
Further reading:

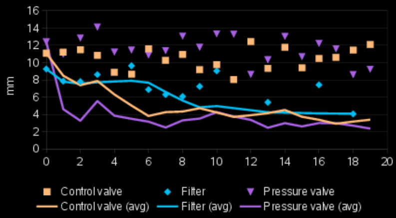
http://brml.technion.ac.il/publications_files/1307386738.pdf

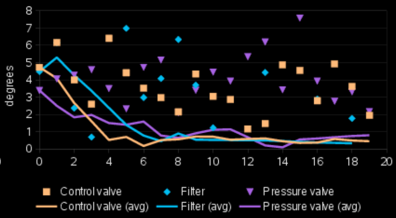
http://lcvmwww.epfl.ch/new/publications/data/articles/63/simaxpaper.pdf

Pose Distances and Clustering

Multi-view pose estimation of single objects

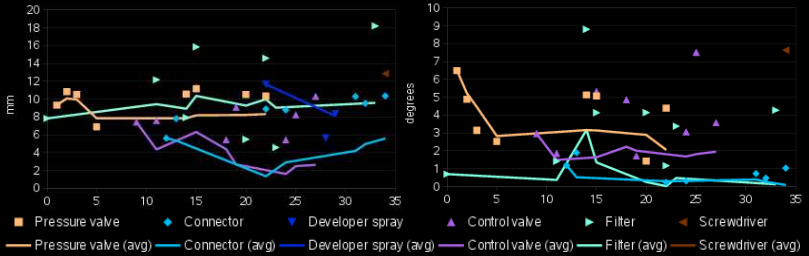






[Work done together with Simon Kriegel and Manuel Brucker]

pointcloudlibrary Pose Distances and Clustering



pointcloudlibrary

Pose Sampling

Random poses are useful for evaluating pose estimation methods, global and local registration methods, etc.

Evenly sampled rotations can be generated by:

- Making sure rotation matrix elements are evenly distributed:
  - brute force method not practical
  - solution to be released in PCL soon
- Simple solution:
  - evenly sample quaternion dimensions $(w, x, y, z) \in [0, 1] \times [-1, 1] \times [-1, 1] \times [-1, 1]$
  - discarding those that have a norm larger than 1
  - normalization of the rest

pointcloudlibrary

Pose Sampling

Note: evenly sampled axis and angle of rotation is not correct! Think of polar coordinates in 2D (or longitude-latitude on a sphere): selecting a uniformly sampled direction and distance will not fill a disk evenly.

You can specify a maximum angle too using rejection sampling (more efficient method to come in PCL).

Normal distribution (or something similar on the 4D sphere):

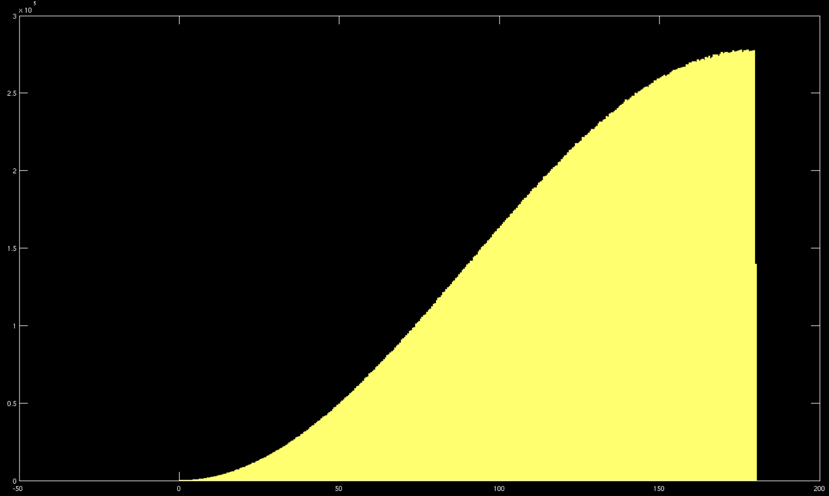
- Bingham distribution [Glover et al. RSS'11]
- uniform sampling and rejection with the probability assigned by the Gaussian of the angular difference to the desired mean

How to verify if some rotations are evenly distributed?

pointcloudlibrary

Pose Sampling

Distribution of distances (angles) is proportional to $sin^2(\alpha)$

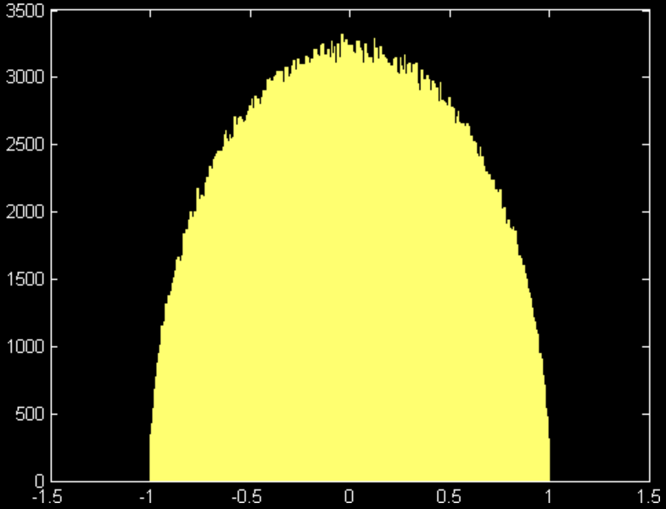


(Either between all rotation pairs, or distance to a fixed rotation)

pointcloudlibrary

Pose Sampling

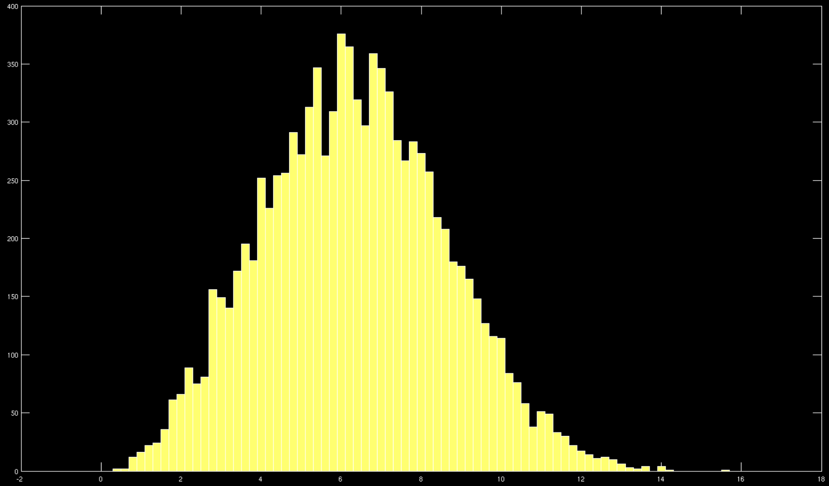
Distribution of quaternion dimensions (for *w* keep only positive side)



pointcloudlibrary

Pose Sampling

Nearest neighbor distances between 10000 samples



[Thanks to Christian Rink and Serkan Türker]

pointcloudlibrary

Outline

pointcloudlibrary

Summary

- Multiple methods are available for the same task
- It is simple to implement solutions that combine these
- Increases robustness for cases not covered during training

Summary

- Multiple methods are available for the same task
- It is simple to implement solutions that combine these
- Increases robustness for cases not covered during training

Questions?

Robotics and Mechatronics Center

Contact: zoltan.marton@dlr.de