# RoboCup@Home Practical Course

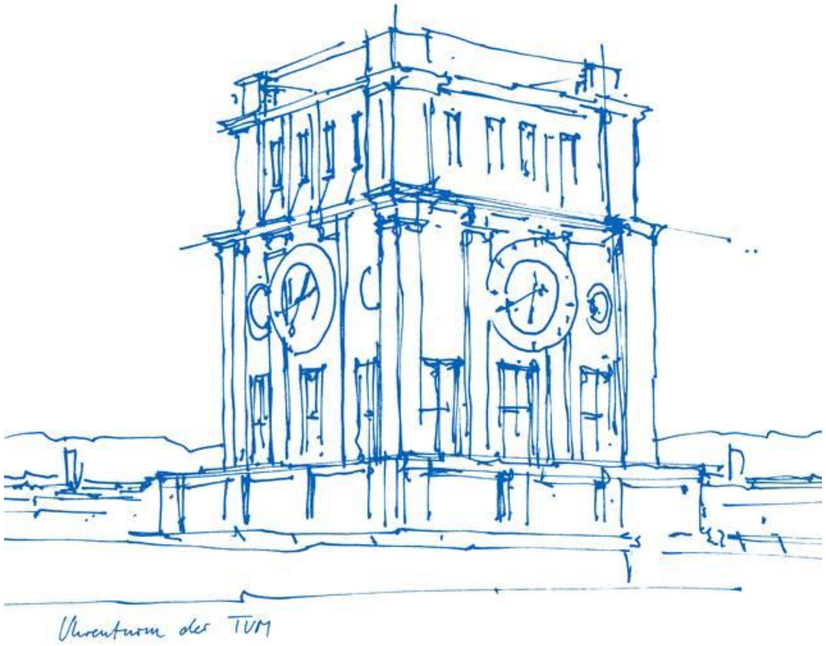
Dr. Pablo Lanillos

Technical University of Munich

Department of Electrical and Computer Engineering

Chair for Cognitve Systems

Munich, 29. Nov 2017

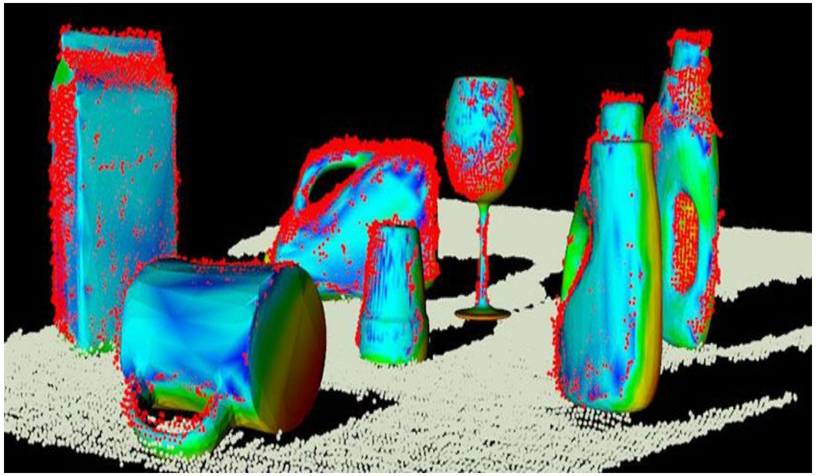


Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

# RoboCup@Home Practical Course

## Tutorial: object recognition and 3D

Dr. Karinne Ramirez-Amaro
Dr. Emmanuel Dean
**Dr. Pablo Lanillos**
M.Sc. Roger Guadarrama
Dr. Gordon Cheng



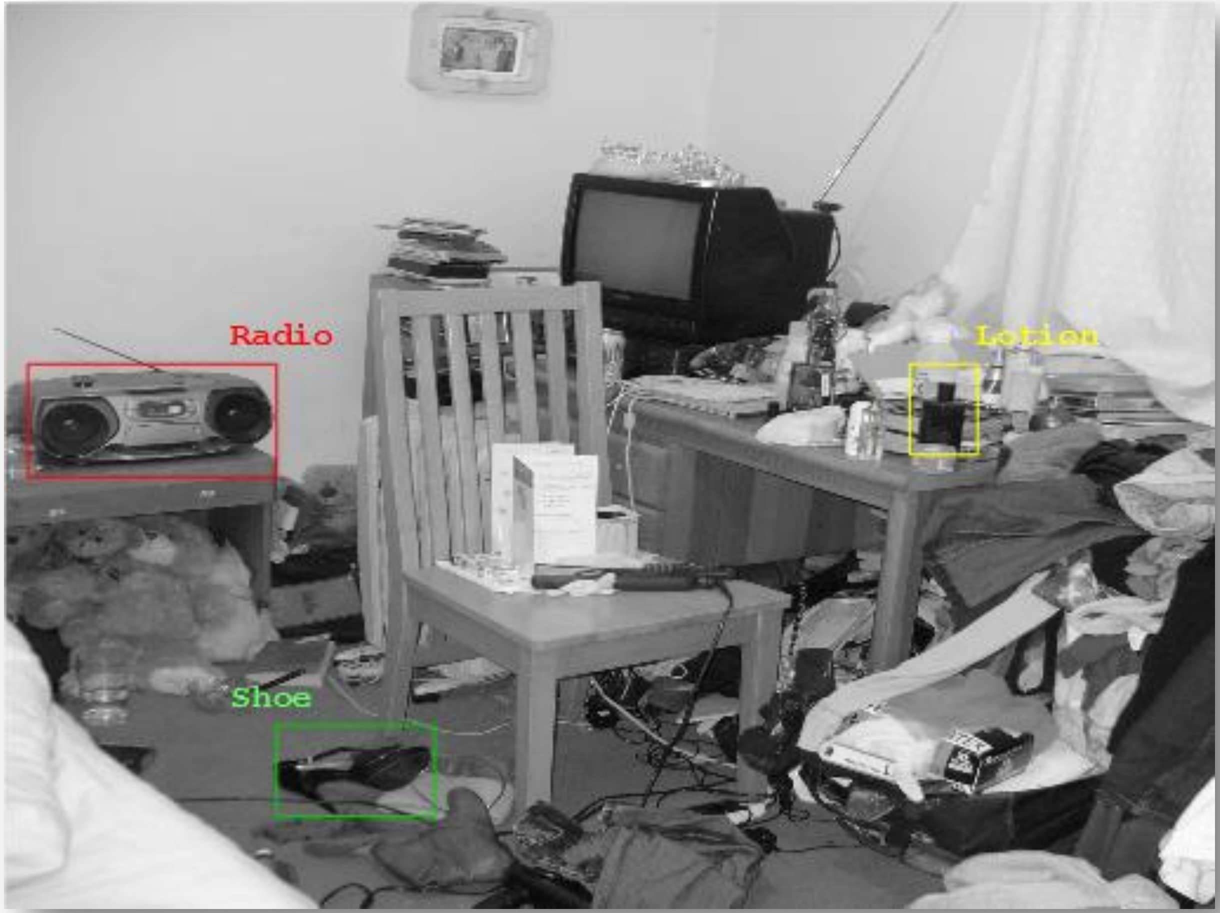
p.lanillos@tum.de
www.therobotdecision.com

# Exercises

**Email: robocup.atHome.ics@gmail.com**

- Compress all the folder containing the C++ nodes folders into one zip/rar/tar/gz file and name it as: Name_LastName_RCH_tutorial5.

# Motivation



# Goal: 3D object classification

3D point cloud algorithms and learn object recognition pipeline

Learn PCL in C++ under ROS

# RoboCup@Home qualification

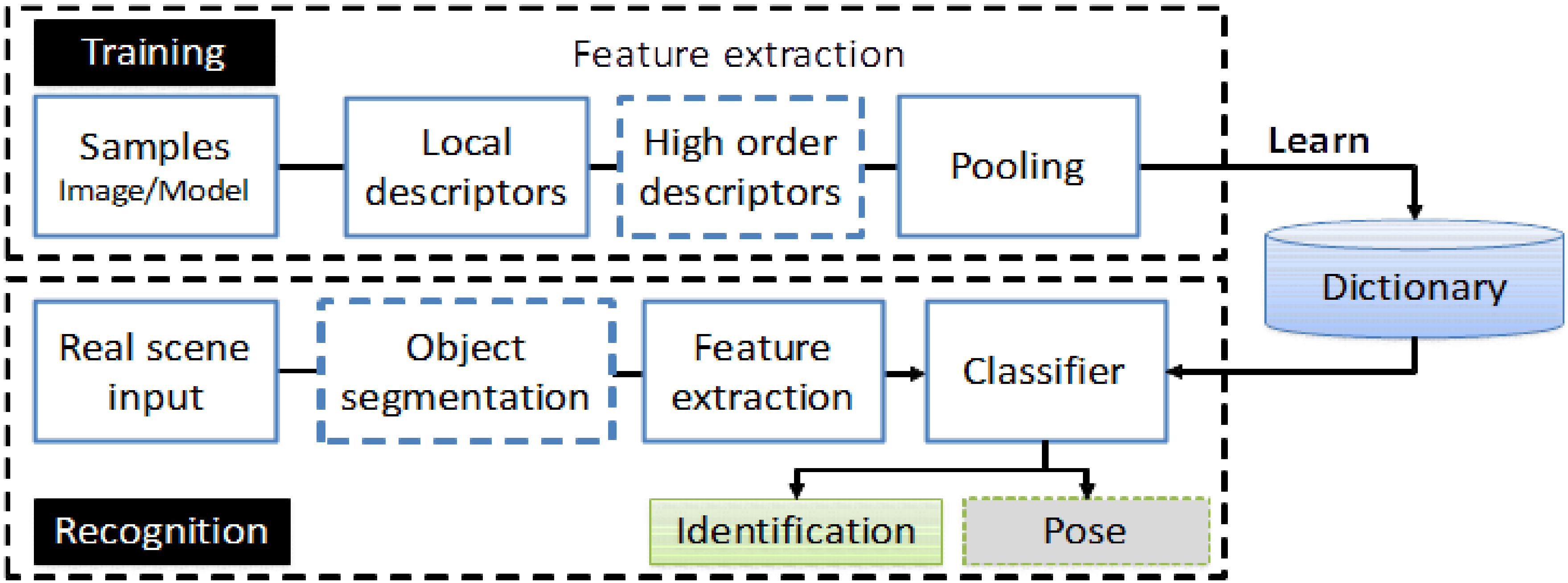
# Excercises

1. Implement a C++ ROS node that provides the 3D location of an object (optional: with respect to torso coordinate frame).

2. Implement a C++ ROS node that segments the table and the objects PointCloud in the scene.

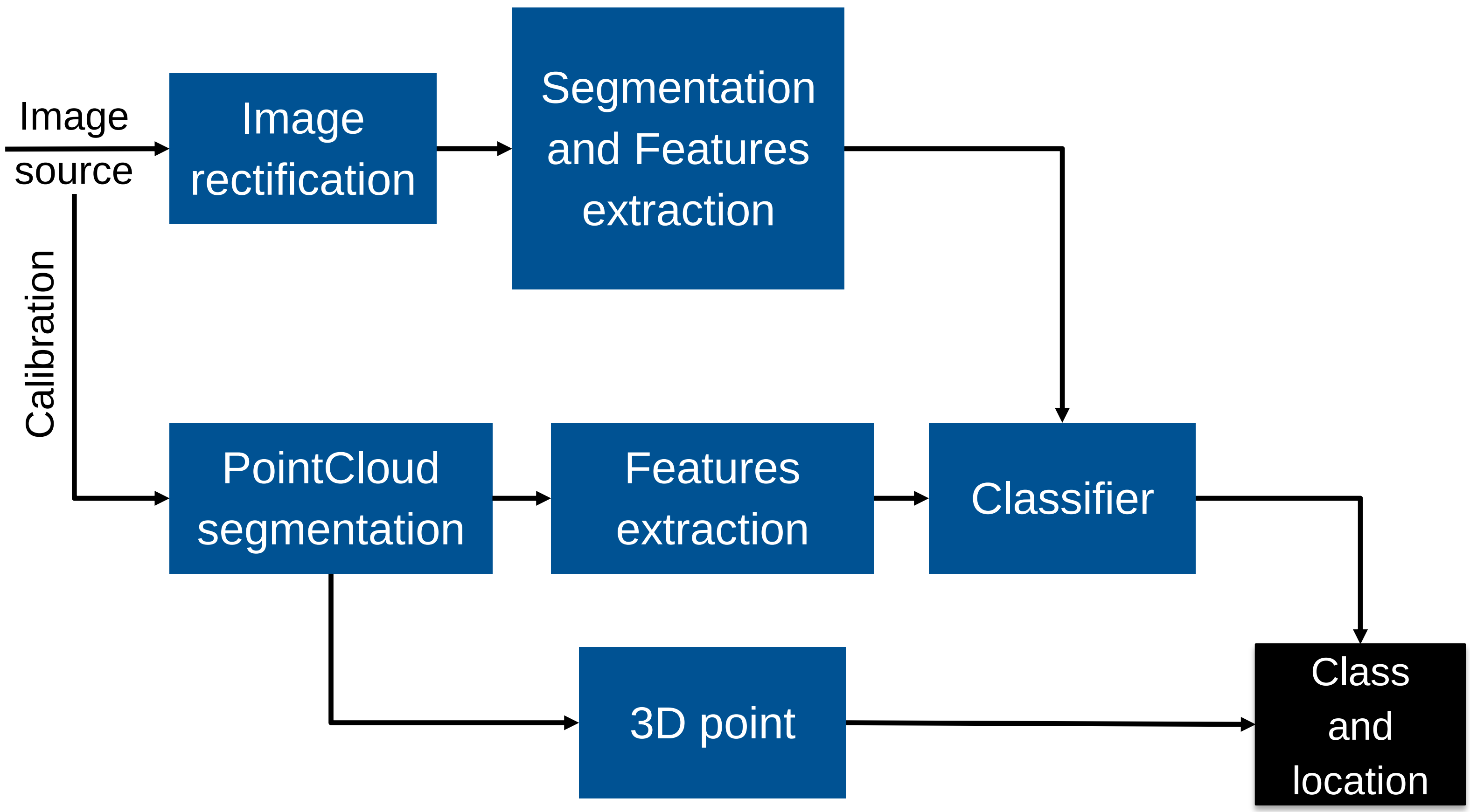
3. Implement a C++ ROS node that recognizes an object and provides its 3D location (template not provided).

# Classic pipeline



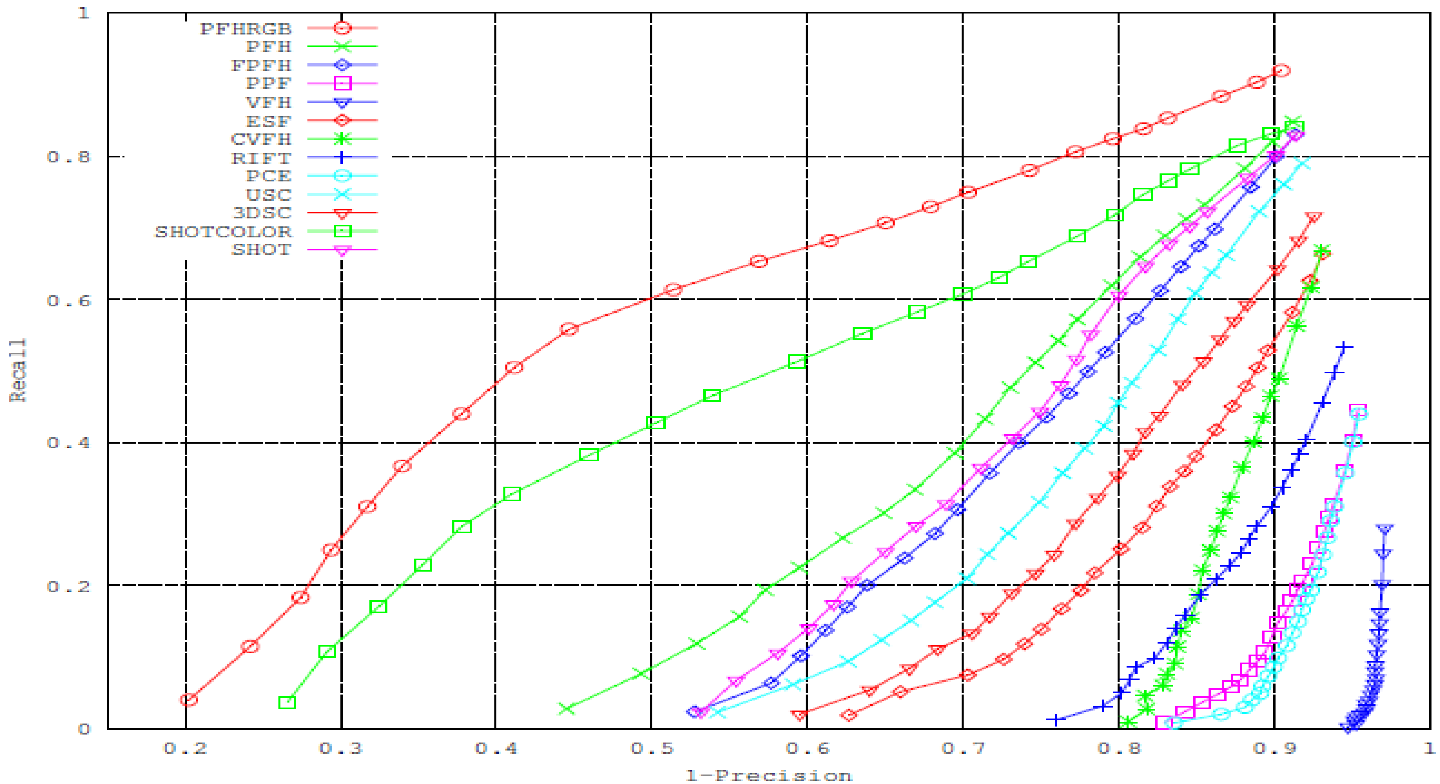

Adapted from: Wang, W., Chen, L., Liu, Z., Kühnlenz, K., & Burschka, D. (2015). Textured/textureless object recognition and pose estimation using RGB-D image. Journal of Real-Time Image Processing, 10(4), 667-682.

# ROS pipeline for the tutorial



# Descriptors

| | | | 2D | 3D |
|---|---|---|---|---|
| Gray scale | Geometric | Local | SIFT | RIFT |
| Color | Photometric | Global | HOG | SHOT |
| | | | SURF | FPFH |




Alexandre, L. A. (2012, October). 3D descriptors for object and category recognition: a comparative evaluation. In Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

# Exercise 1: Compute 3D point

1. Go to src/from2Dto3D

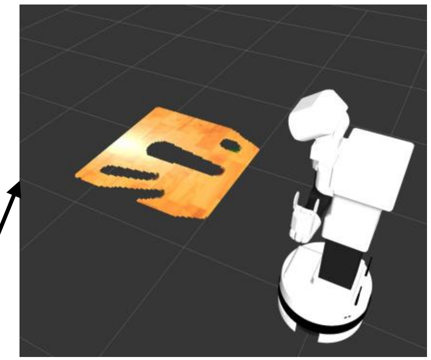
2. Implement a node that uses the kinect point cloud to transform the 2D point into a 3D coordinate

3. Publish to segmentation/point3D

**Input**: 2D point
**Output**: 3D point in torso coordinate frame

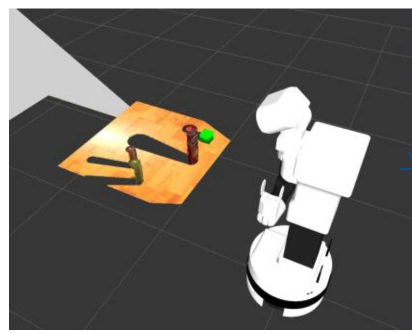
# Exercise 2: Point cloud segmentation - plane

1. Create a new package inside the catkin directory:
   /src/PlaneSegmentation/

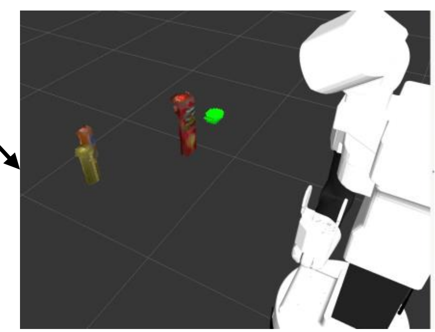


2. Copy the template from the wiki

3. Implement PlaneSegmentation.cpp using pcl library.



**Input**: Camera Point Cloud stream
**Output**:
1. Point cloud without the main plane
2. Point cloud of the main plane



http://wiki.ros.org/Robots/TIAGo/Tutorials/PointCloud

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

# Exercise 3: Object recognition

Groups of 2 people

1. Create a new package inside the catkin directory:
/src/object_recognition/
2. Implement ObjectRecognition.cpp
   - Object to recognize:
     (a) two different classes of the same object semantic class (orange, coke)
     (b) two different semantic classes (ball – brick, person – noperson)
   - You can use 3D features or 2D features or combined
   - You can use opencv and pcl libraries

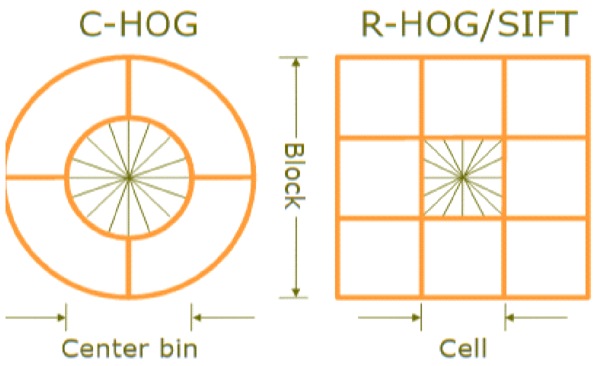
**Example 2: people detector**


**Example 1: ball and a brick**


# 2D Descriptors

## Histogram of Oriented Gradients (HOG)

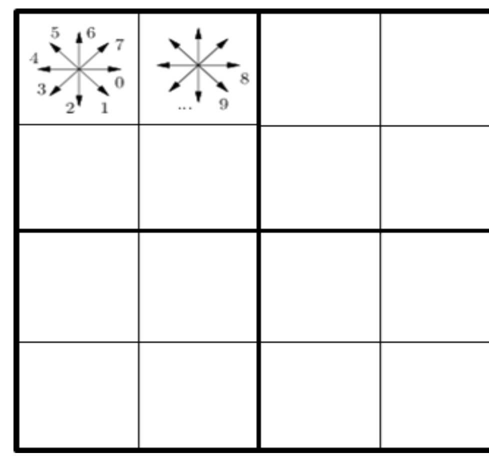
```
hx = [-1,0,1];
hy = hx';

% Compute the derivative in the x and y direction for
every pixel
dx = filter2(hx, double(img));
dy = filter2(hy, double(img));

% Convert the gradient vectors to polar coordinates
angles = atan2(dy, dx);
magnitude = ((dy.^2) + (dx.^2)).^.5;
```



# 2D Descriptors

**Scale-invariant feature transform (SIFT)**

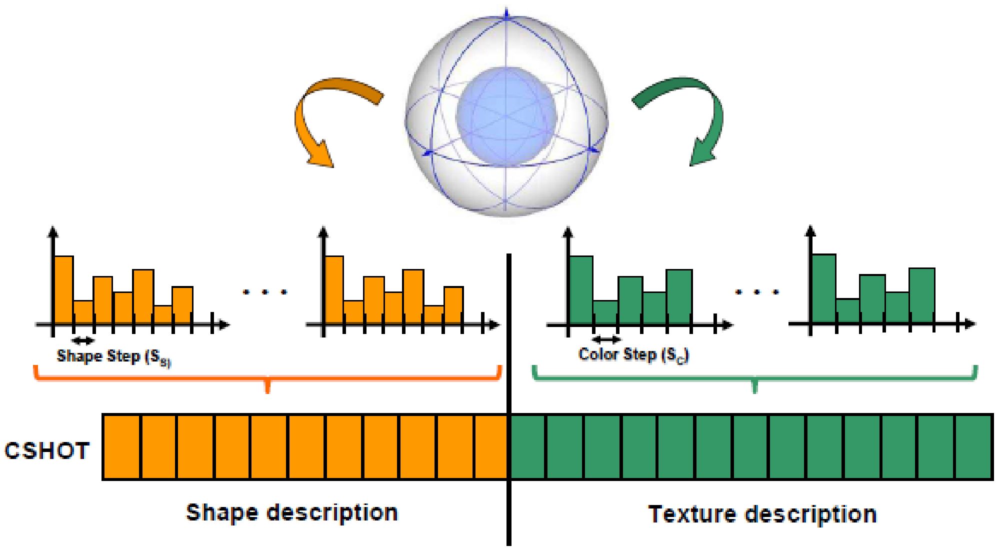
Descriptor= 4x4x8xN_Points = 128xN



# 3D Descriptors

**Signature of Histograms of OrienTations (SHOT)**

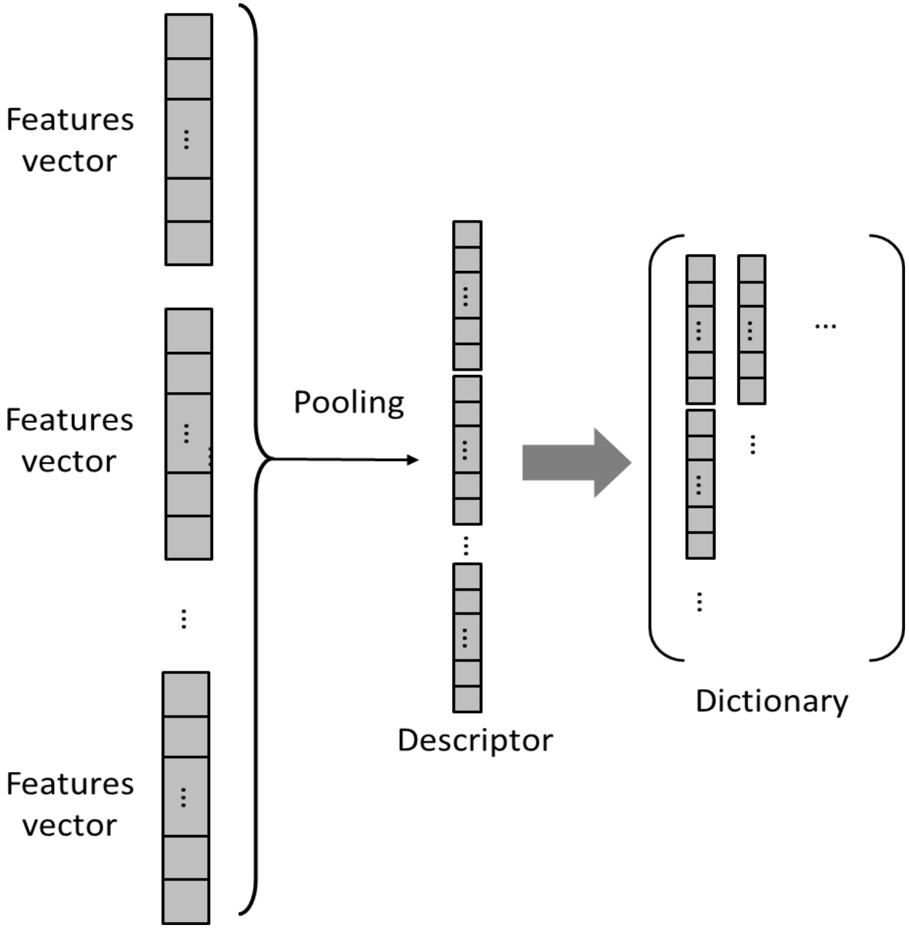
- Keypoints
- Normals




Salti, S., Tombari, F., & Di Stefano, L. (2014). SHOT: unique signatures of histograms for surface and texture description. Computer Vision and Image Understanding.

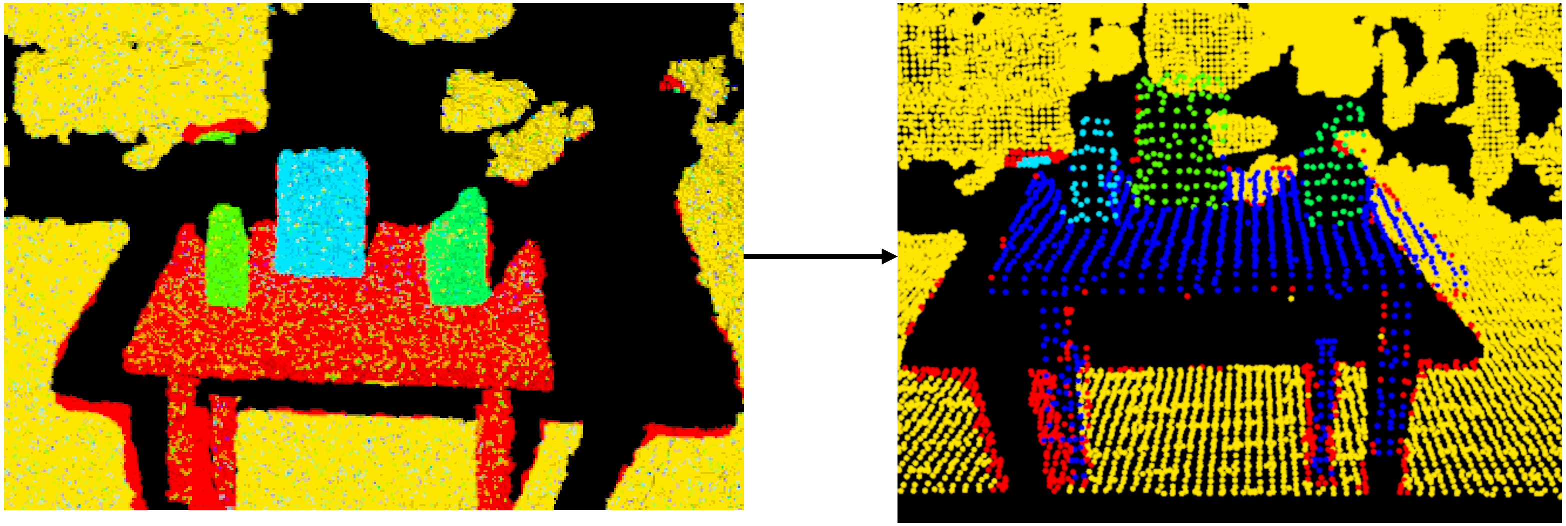
Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

# Descriptors packing and storing



# PCL segmentation

Voxel grid down sampling: pcl::VoxelGrid<pcl::PointXYZ>



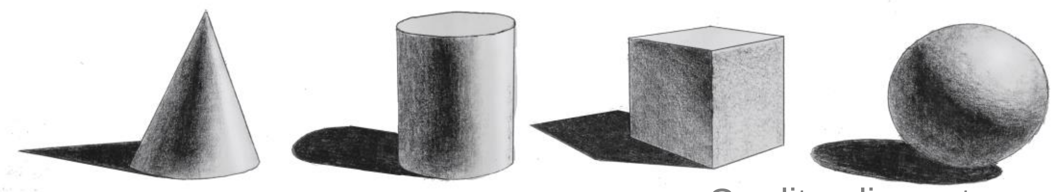
SetInputCloud( InputCloud )
SetLeafSize( x,y,z )
filter( Output PointCloud )

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

# PCL segmentation

## Consensus model segmentation: pcl::SACSegmentation

pcl::SACSegmentation<pcl::PointT>( pcl::PointIndices inliers, pcl::ModelCoefficients coefficients )



Credits: diaz-arts.com

SACMODEL_PLANE
SACMODEL_LINE
SACMODEL_CIRCLE2D
SACMODEL_CIRCLE3D
SACMODEL_SPHERE
SACMODEL_CYLINDER
SACMODEL_CONE
SACMODEL_TORUS
SACMODEL_PARALLEL_LINE
SACMODEL_PERPENDICULAR_PLANE
SACMODEL_PARALLEL_LINES
SACMODEL_NORMAL_PLANE
SACMODEL_NORMAL_SPHERE
SACMODEL_REGISTRATION
SACMODEL_REGISTRATION_2D
SACMODEL_PARALLEL_PLANE
SACMODEL_NORMAL_PARALLEL_PLANE
SACMODEL_STICK

RANSAC algorithm
pcl::SAC_RANSAC

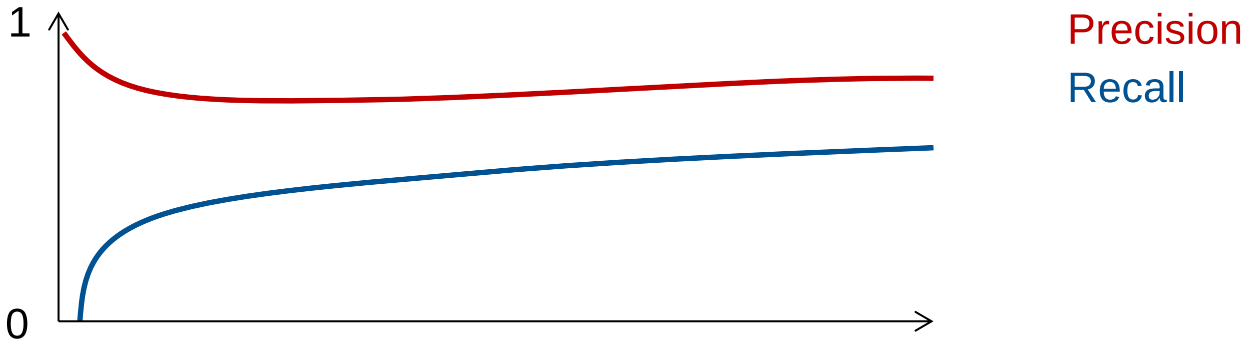
Extracting the output from segmentation
pcl::ExtractIndices<pcl::PointT>

setInputCloud( InputCloud )
setIndices( pcl::PointIndices )
setNegative( bool)
filter( Output PointCloud )

http://pointclouds.org/documentation/tutorials/walkthrough.php

# Tip: Generating labelled data

https://github.com/puzzledqs/BBox-Label-Tool



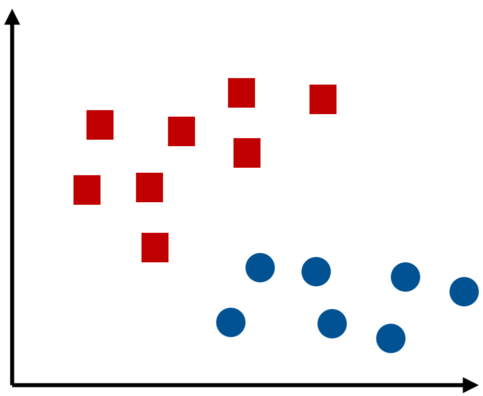

We know that the algorithm have this precision recall curve. Now we only have 30 images labelled.

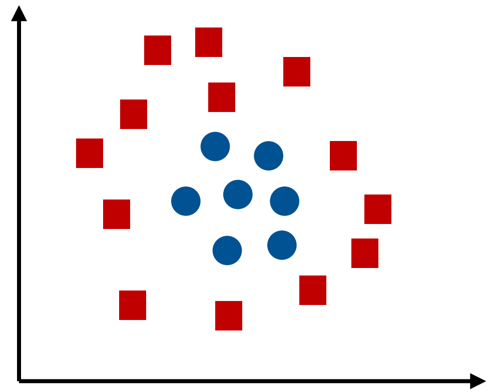
What we will do:

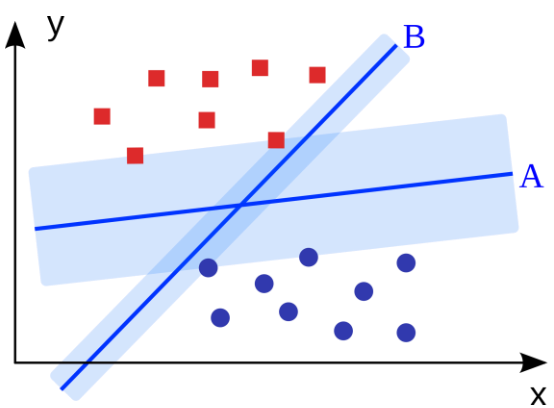
1) Improve the algorithm
2) Get more samples

3000 labeled samples → 1 min/sample → 3000 min → 50 h → 6,25 days

# Support Vector Machines (SVMs)








**Kernelized SVMs**

Boser, B. E., Guyon, I. M., & **Vapnik, V. N.** (1992, July). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory* (pp. 144-152). ACM.

# Tip: Recording data in a rosbag

> roslaunch kinect2_bridge kinect2_bridge.launch (for kinect)
> roslaunch openni2_launch openni2_launch.launch (for asus or orbbec)

> rosrun rviz rviz → add image → image_rectified

> rosbag record -a [-O session_name.bag]

To play

> rosbag play -l name_of_the_file
(-l makes the bag to play in a loop)

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

# MIGHT THE PIXELS BE WITH YOU!