Development of intelligent systems (RInS)

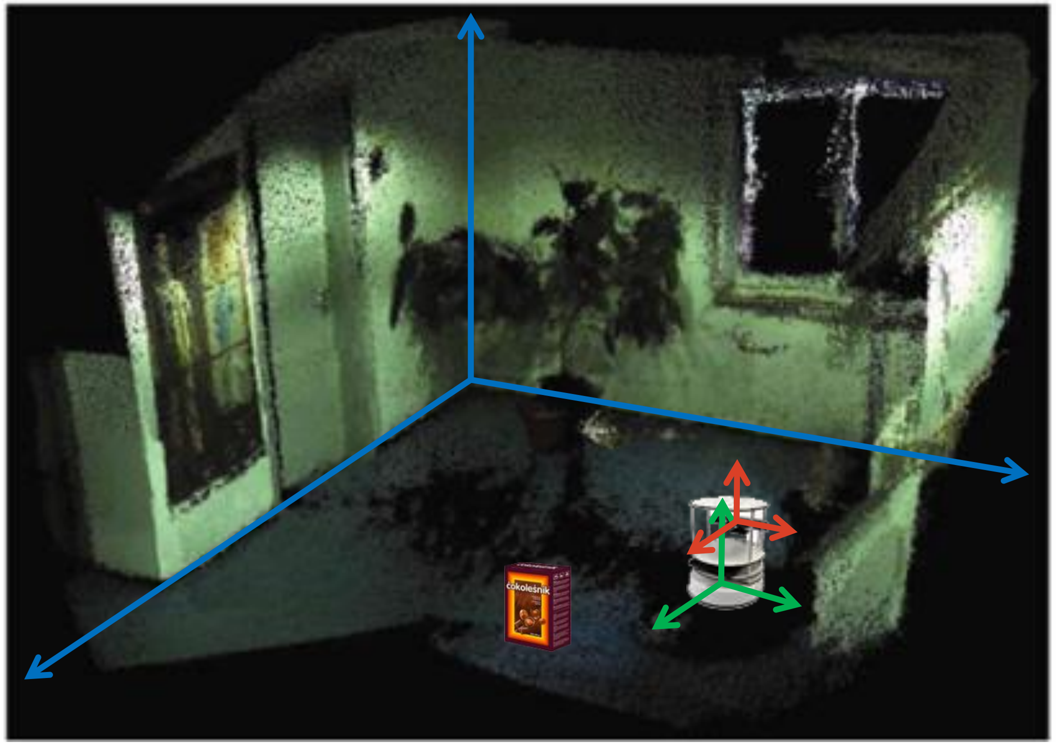
Object detection in 3D

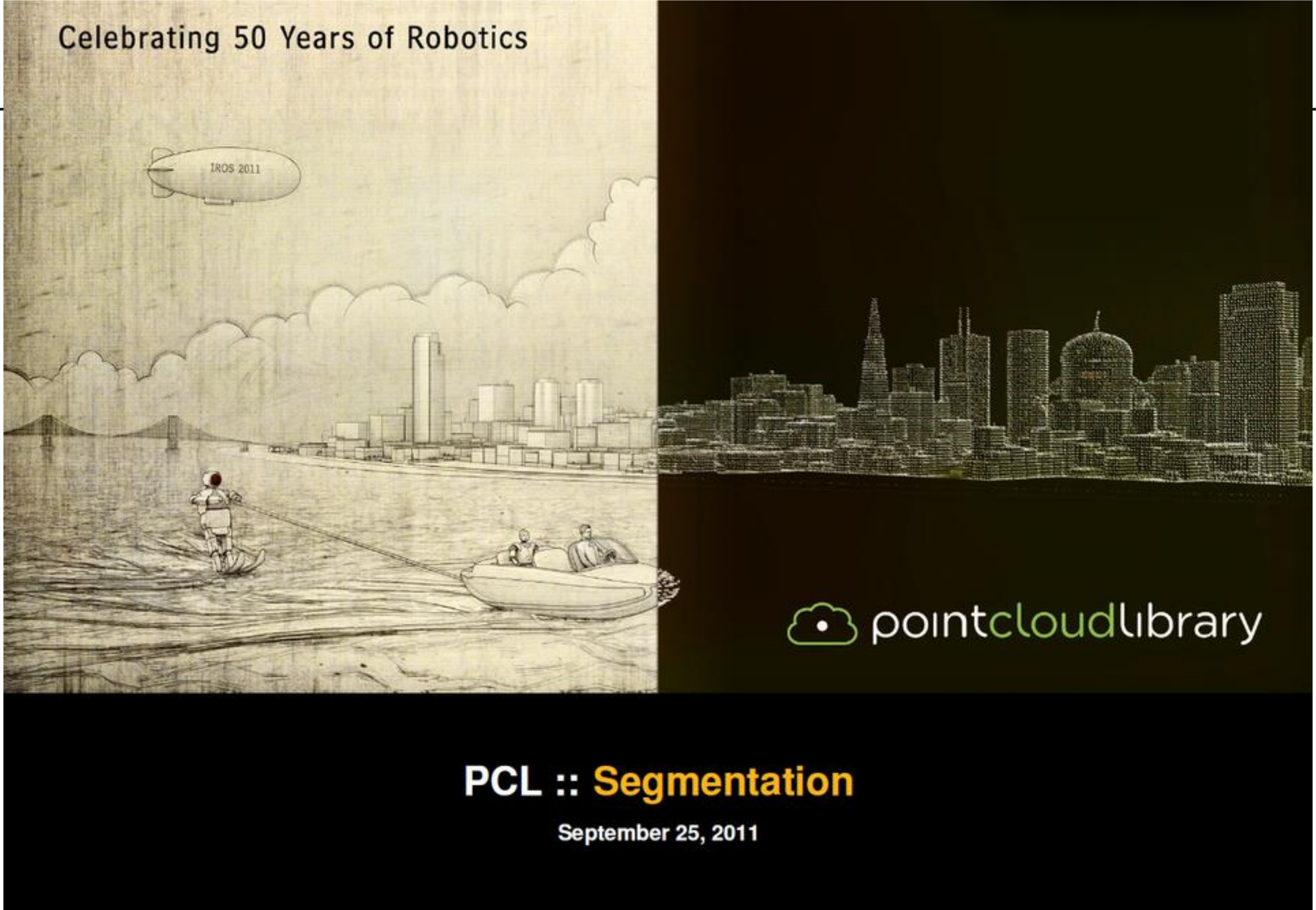
Danijel Skočaj
University of Ljubljana
Faculty of Computer and Information Science

Academic year: 2022/23

Detection of obstacles and objects

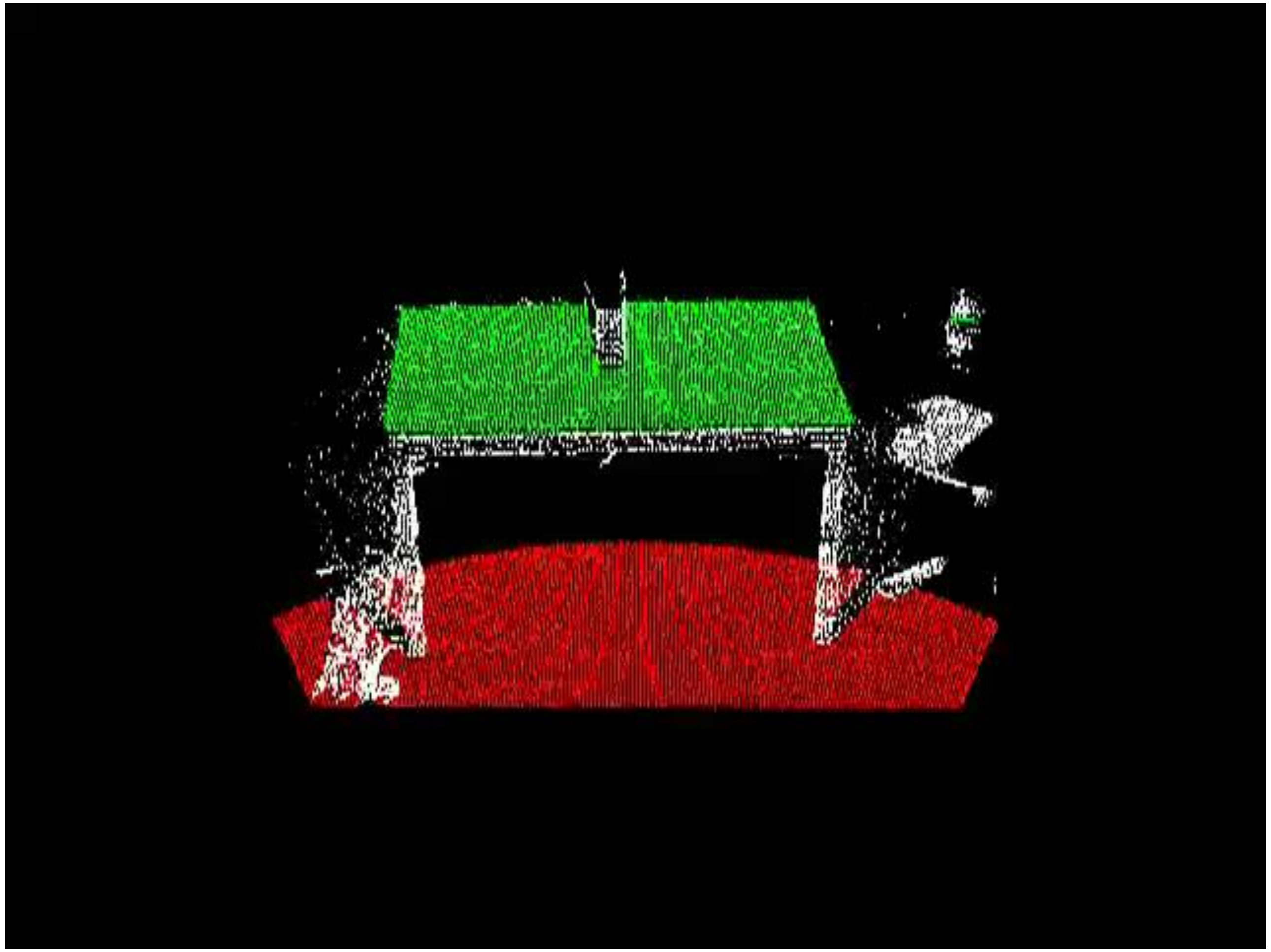
3D perception





Radu Bogdan Rusu, Nico Blodow, Willow Garage, PCL

Detection of planes



RANSAC

- Random Sampling Consensus [Fischler, Bolles '81]

In: $U = \{x_i\}$ set of data points, $|U| = N$

$f(S) : S \rightarrow p$ function f computes model parameters p given a sample S from U

$\rho(p, x)$ the cost function for a single data point x

Out: p^* p^*, parameters of the model maximizing the cost function

k := 0

Repeat until P{better solution exists} < η (a function of C^* and no. of steps k)

k := k + 1

I. Hypothesis

(1) select randomly set $S_k \subset U$, sample size $|S_k| = m$

(2) compute paramet $p_k = f(S_k)$

II. Verification

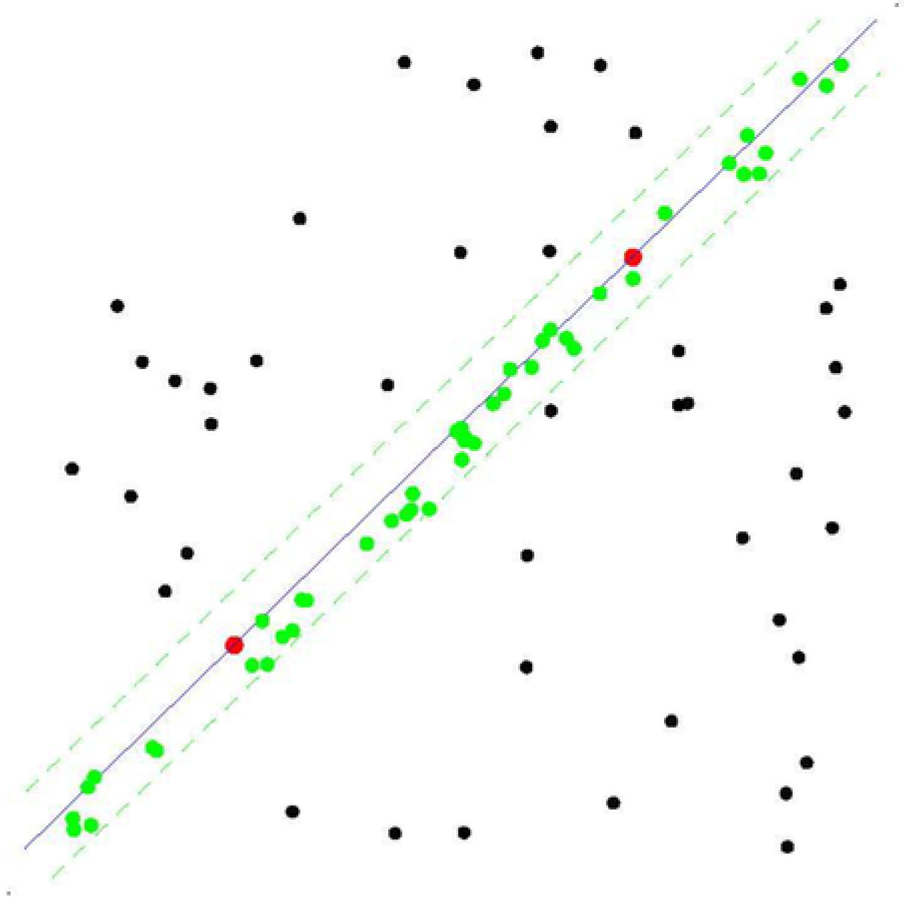
(3) compute cost $C_k = \sum_{x \in U} \rho(p_k, x)$

(4) if $C^* < C_k$ then $C^* := C_k$, $p^* := p_k$

end

J. Matas @ CVPR 11 Registration Tutorial

RANSAC



J. Matas @ CVPR 11 Registration Tutorial

ALL-INLIER SAMPLE

RANSAC time complexity

$$t = k(t_M + \overline{m}_s N)$$

k … number of samples drawn

N … number of data points

t_M … time to compute a single model

m_S … average number of models per sample

RANSAC

the "gold standard" algorithm:

In: $U = \{x_i\}$ set of data points, $|U| = N$

$f(S) : S \rightarrow p$ function f computes model parameters p given a sample S from U

$\rho(p, x)$ the cost function for a single data point x

Out: p^* p^*, parameters of the model maximizing the cost function

k := 0

Repeat until P{better solution exists} < η (a function of C^* and no. of steps k)

k := k + 1

I. Hypothesis

(1) select randomly set $S_k \subset U$, sample size $|S_k| = m$

(2) compute parameters $p_k = f(S_k)$

II. Verification

(3) compute cost

(4) if $C^* < C_k$ then $C^* := C_k$, $p^* := p_k$ $C_k = \sum_{x \in U} \rho(p_k, x)$

end Repeat

p^{out} = least square fit on the set of inliers to p^*

J. Matas @ CVPR 11 Registration Tutorial

RANSAC in PCL

Point Cloud Library (PCL) 1.10.1-dev

Main Page | Related Pages | Modules | Namespaces ▾ | Classes ▾ | Search

Classes | Functions

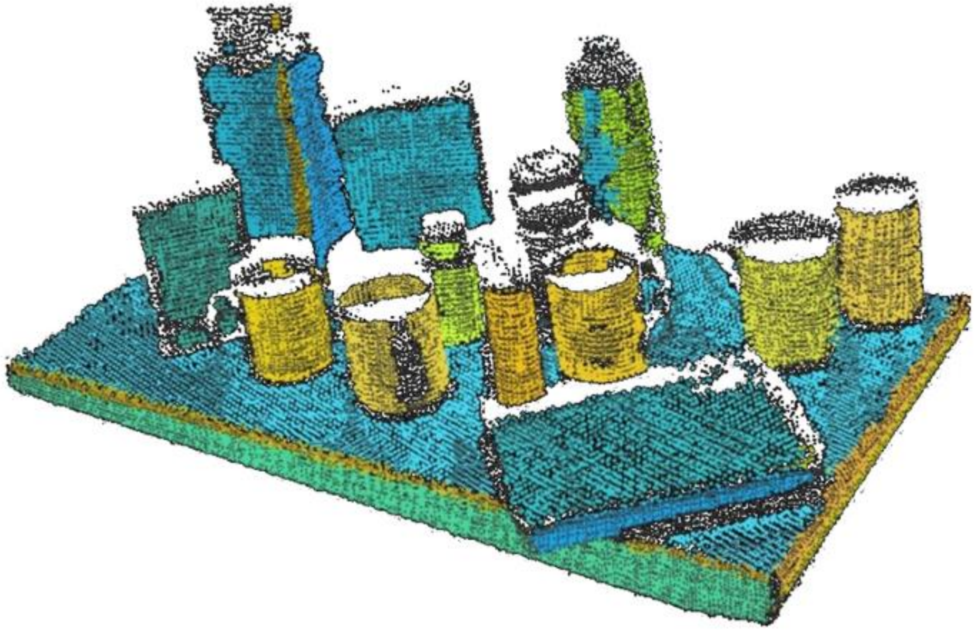
Module sample_consensus

Detailed Description

Overview

The **pcl_sample_consensus** library holds SAmple Consensus (SAC) methods like RANSAC and models like planes and cylinders. These can combined freely in order to detect specific models and their parameters in point clouds.

Some of the models implemented in this library include: lines, planes, cylinders, and spheres. Plane fitting is often applied to the task of detecting common indoor surfaces, such as walls, floors, and table tops. Other models can be used to detect and segment objects with common geometric structures (e.g., fitting a cylinder model to a mug).

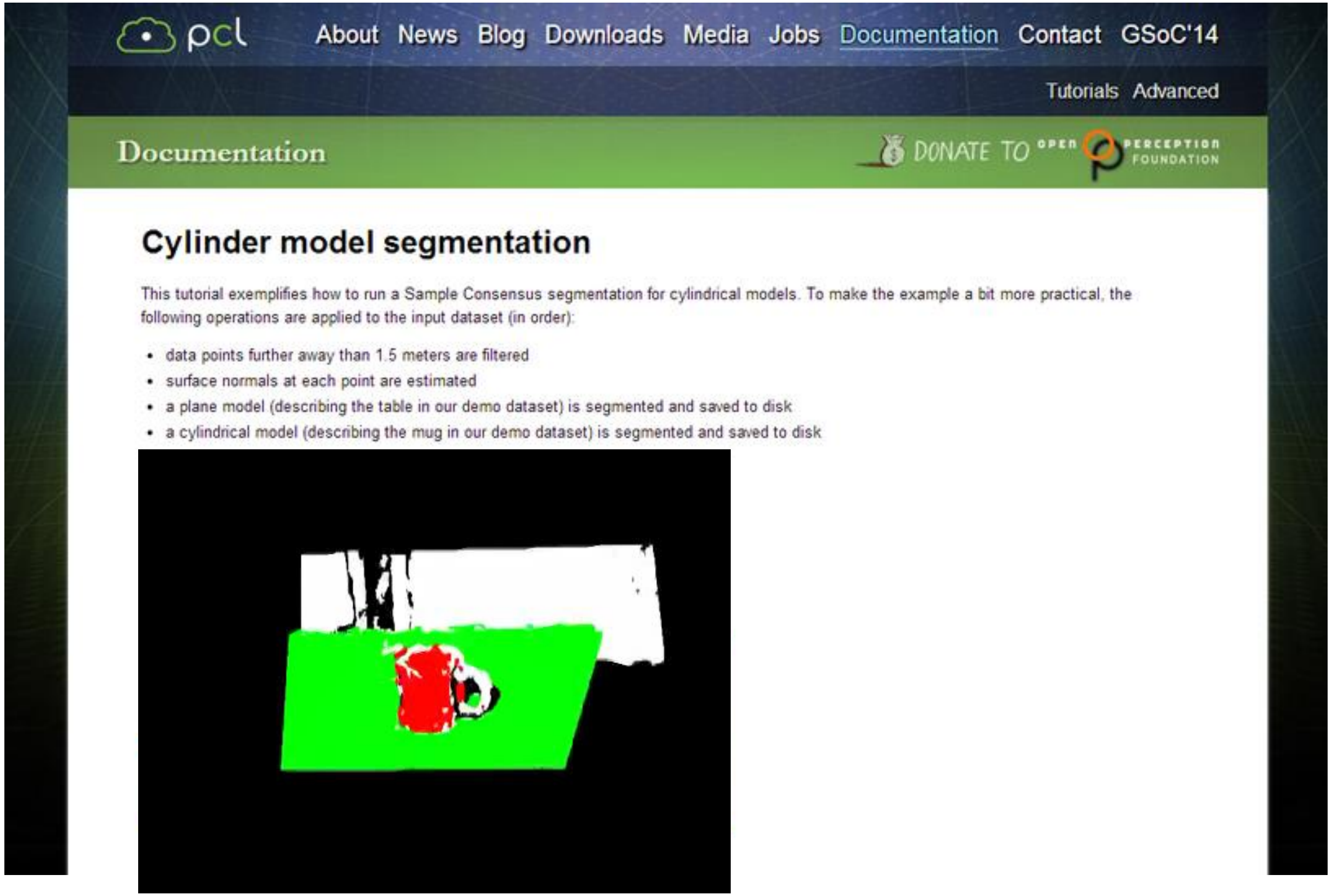


RANSAC in PCL

The following models are supported:

- SACMODEL_PLANE - used to determine plane models. The **four** coefficients of the plane are its Hessian Normal form: [**normal_x normal_y normal_z d**]
- SACMODEL_LINE - used to determine line models. The **six** coefficients of the line are given by a point on the line and the direction of the line as: [**point_on_line.x point_on_line.y point_on_line.z line_direction.x line_direction.y line_direction.z**]
- SACMODEL_CIRCLE2D - used to determine 2D circles in a plane. The circle's **three** coefficients are given by its center and radius as: [**center.x center.y radius**]
- SACMODEL_CIRCLE3D - used to determine 3D circles in a plane. The circle's **seven** coefficients are given by its center, radius and normal as: [**center.x, center.y, center.z, radius, normal.x, normal.y, normal.z**]
- SACMODEL_SPHERE - used to determine sphere models. The **four** coefficients of the sphere are given by its 3D center and radius as: [**center.x center.y center.z radius**]
- SACMODEL_CYLINDER - used to determine cylinder models. The **seven** coefficients of the cylinder are given by a point on its axis, the axis direction, and a radius, as: [**point_on_axis.x point_on_axis.y point_on_axis.z axis_direction.x axis_direction.y axis_direction.z radius**]
- SACMODEL_CONE - used to determine cone models. The **seven** coefficients of the cone are given by a point of its apex, the axis direction and the opening angle, as: [**apex.x, apex.y, apex.z, axis_direction.x, axis_direction.y, axis_direction.z, opening_angle**]
- SACMODEL_TORUS - not implemented yet
- SACMODEL_PARALLEL_LINE - a model for determining a line **parallel** with a given axis, within a maximum specified angular deviation. The line coefficients are similar to SACMODEL_LINE .
- SACMODEL_PERPENDICULAR_PLANE - a model for determining a plane **perpendicular** to a user-specified axis, within a maximum specified angular deviation. The plane coefficients are similar to SACMODEL_PLANE .
- SACMODEL_PARALLEL_LINES - not implemented yet
- SACMODEL_NORMAL_PLANE - a model for determining plane models using an additional constraint: the surface normals at each inlier point has to be parallel to the surface normal of the output plane, within a maximum specified angular deviation. The plane coefficients are similar to SACMODEL_PLANE .
- SACMODEL_NORMAL_SPHERE - similar to SACMODEL_SPHERE , but with additional surface normal constraints.
- SACMODEL_PARALLEL_PLANE - a model for determining a plane **parallel** to a user-specified axis, within a maximum specified angular deviation. The plane coefficients are similar to SACMODEL_PLANE .
- SACMODEL_NORMAL_PARALLEL_PLANE defines a model for 3D plane segmentation using additional surface normal constraints. The plane normal must lie **parallel** to a user-specified axis. SACMODEL_NORMAL_PARALLEL_PLANE therefore is equivalent to SACMODEL_NORMAL_PLANE + SACMODEL_PERPENDICULAR_PLANE. The plane coefficients are similar to SACMODEL_PLANE .
- SACMODEL_STICK - a model for 3D stick segmentation. A stick is a line with a user given minimum/maximum width.

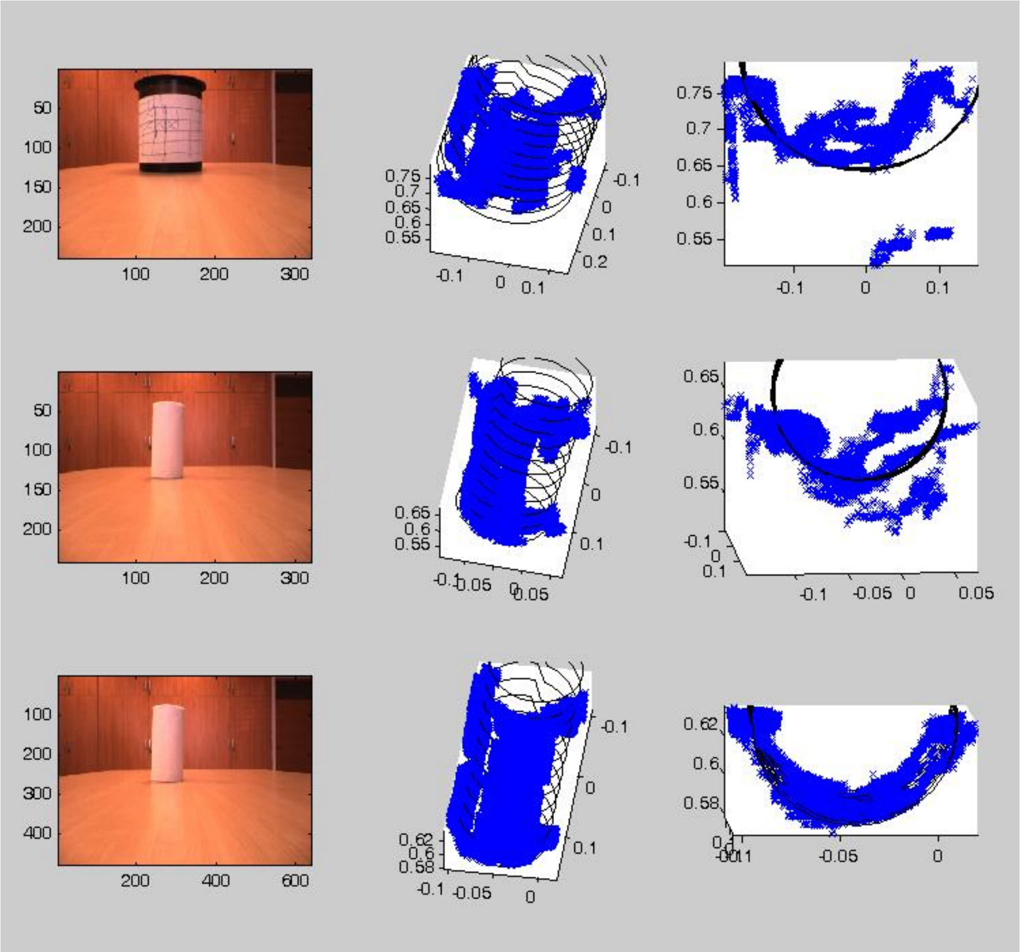
Detection of cylinders



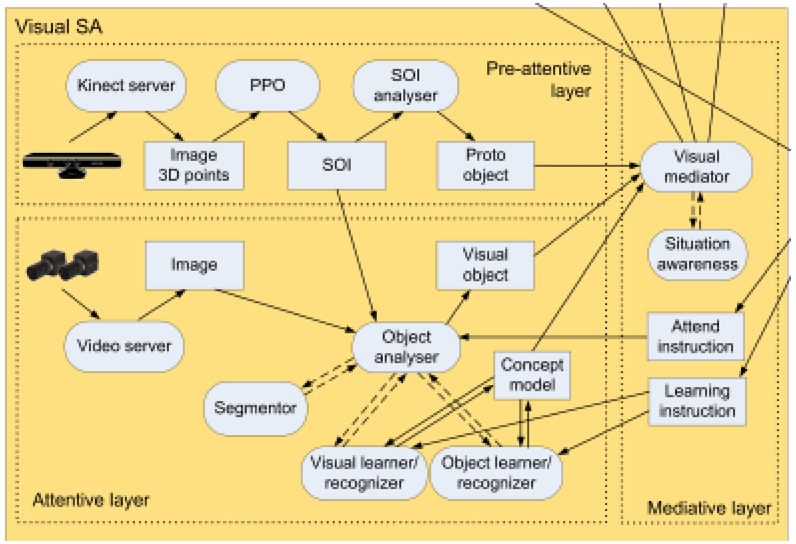
http://pointclouds.org/documentation/tutorials/cylinder_segmentation.php

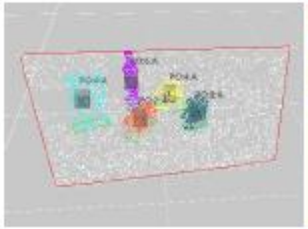
Detection of cylinders

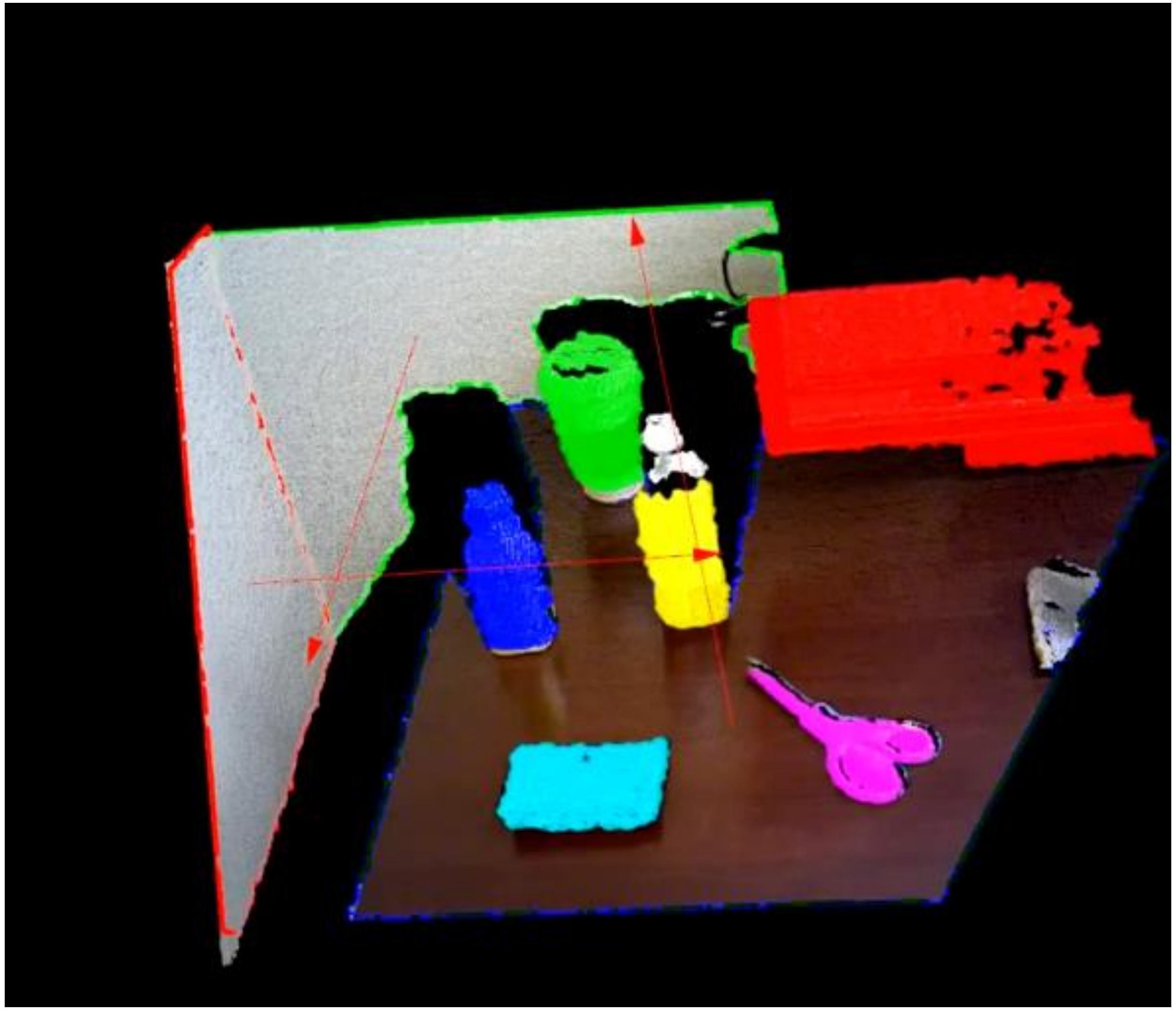
- Noisy data



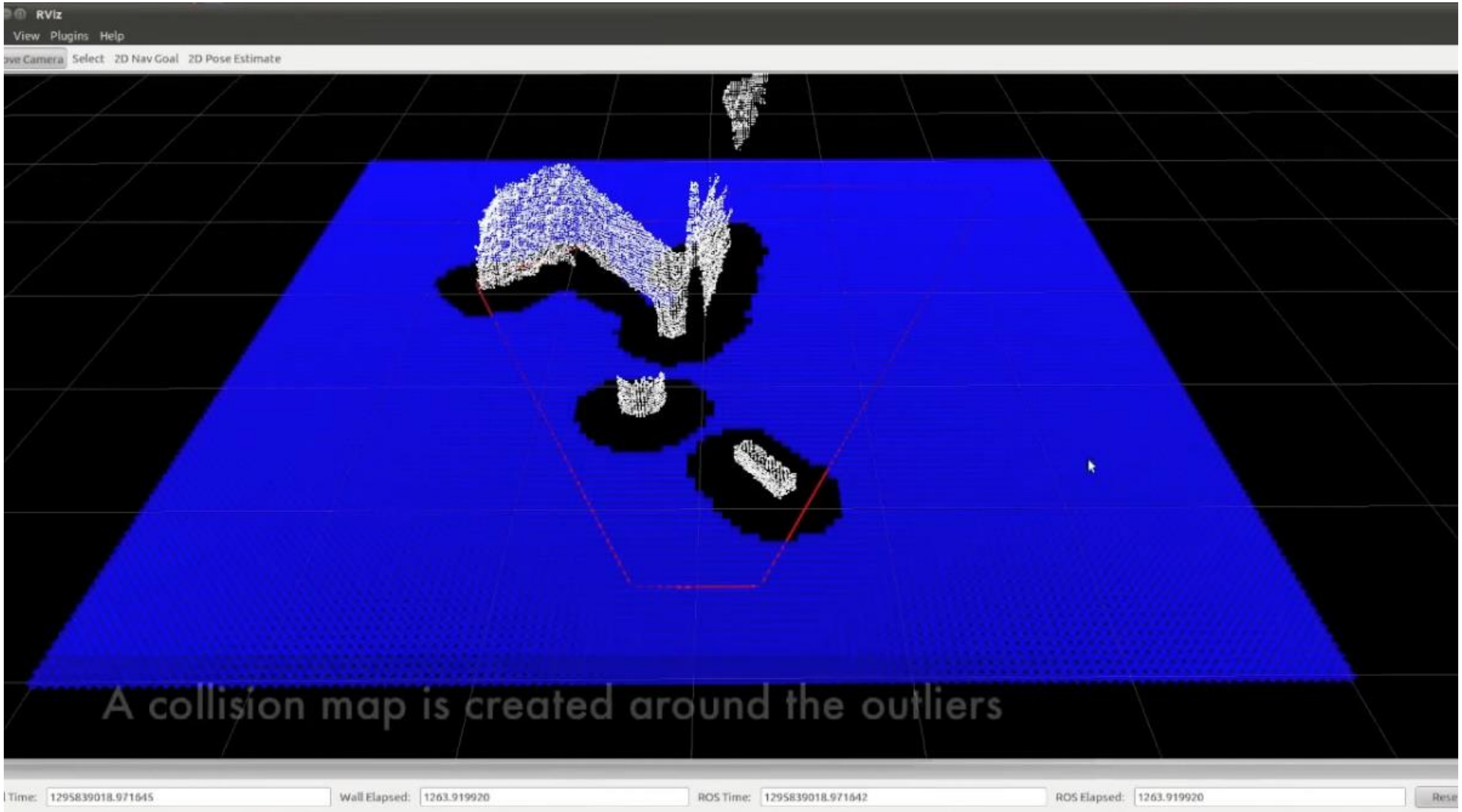
Detection of objects



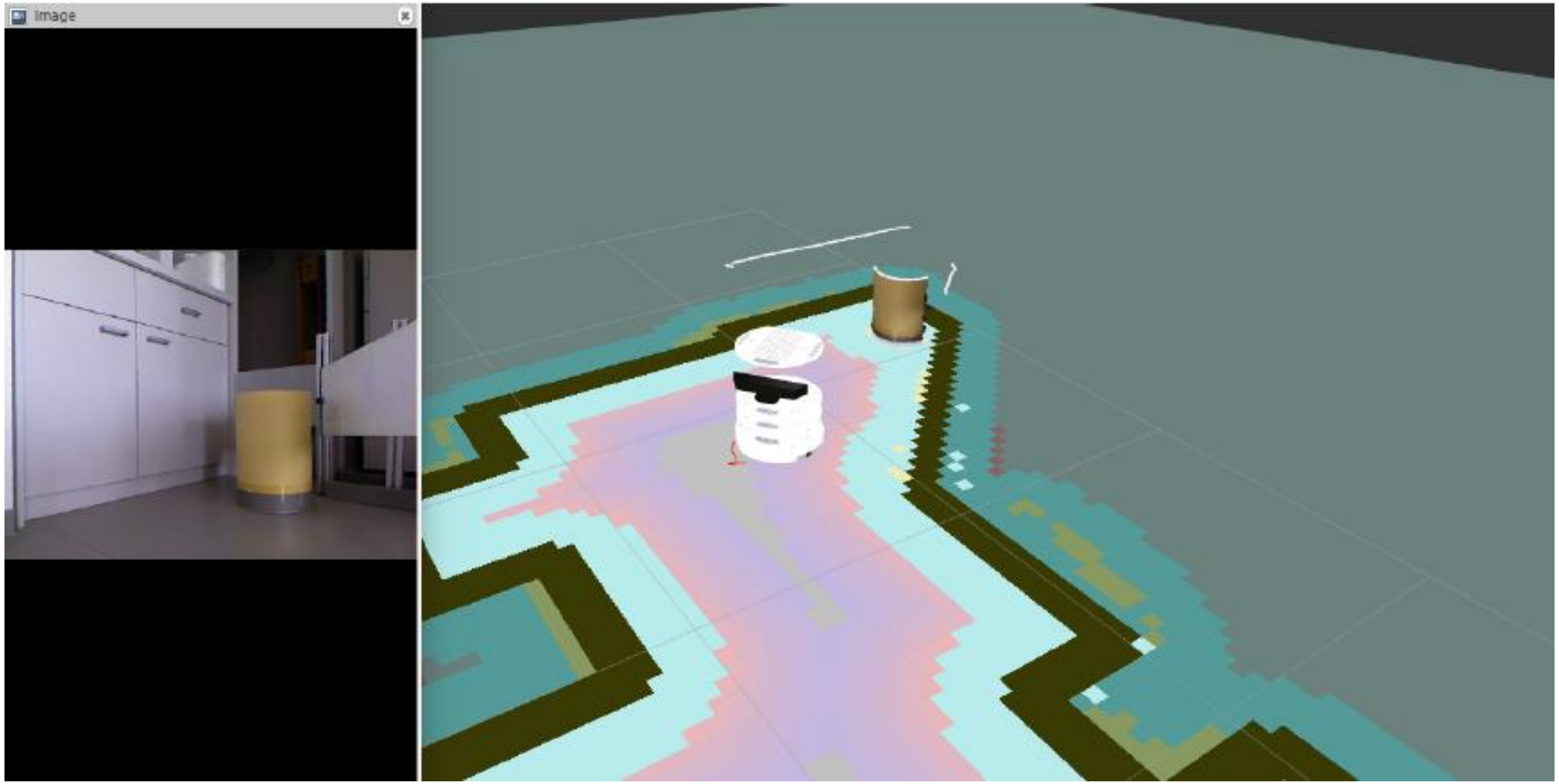




Collision map



Cylinder detection



Ring detection