Researcher of the Week: superquadric-model module

Goal

Object detection and modeling

iCub looks at the object

the object is detected

a 3D model is computed

the model is visualized

Requests:

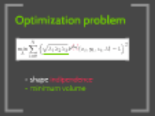
① Detection ✓
② Modeling ?
③ Visualization ?
④ ... and everything robust and in real-time!

Theory

Superquadric fuctions

inside-outside function:
- F > 1: point outside
- F < 1: point inside
- F = 1: point on surface

5 parameters for shape + 6 parameters for pose

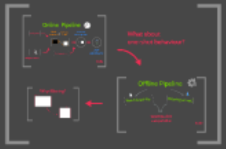


Optimization problem

- shape independence
- minimum volume

Code

Pipeline



Dependencies

Dependencies:
- YARP
- iCub
- OpenCV
- IPOPT [illegible]

External application
- IOL (Interactive Object Learning)
  - BlobExtractor [illegible]
  - SFM [illegible]
  - OPC [illegible]

Code

Connections

Giulia Vezzani - 13/05/2016

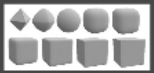
Goal

Object detection and modeling

iCub looks at the object

the object is detected

a 3D model is computed



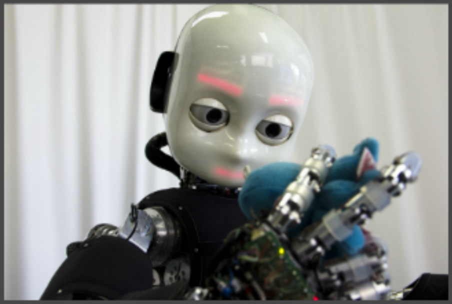
the model is visualized

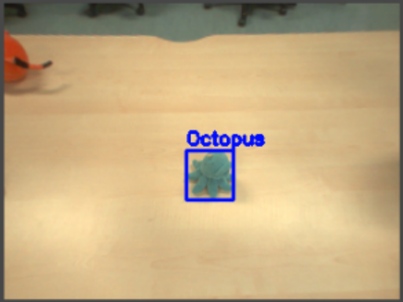
Requests:

1. Detection ✓
2. Modeling ?
3. Visualization ?
4. ... and everything robust and in real-time !

Object detection and modeling

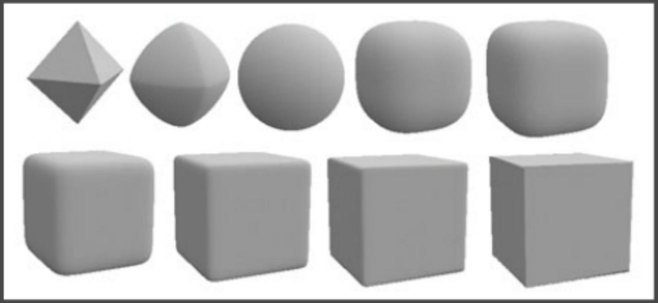
iCub **looks** at the object

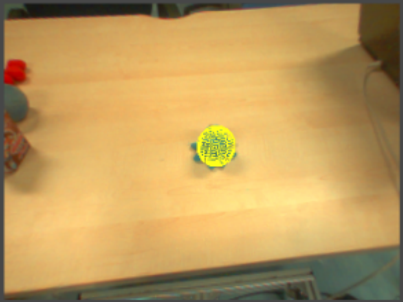




the object is **detected**

a **3D model** is computed





the model is **visualized**

Requests:

1. Detection ✓
2. Modeling ?
3. Visualization ?
4. ... and everything robust and in real-time!

Theory

Superquadric fuctions

inside-outside function:

- F > 1: point outside
- F < 1: point inside
- F = 1: point on surface

$$F(x, y, z, \lambda) = \left(\left(\frac{x}{\lambda_1} \right)^{\frac{2}{\lambda_5}} + \left(\frac{y}{\lambda_2} \right)^{\frac{2}{\lambda_5}} \right)^{\frac{\lambda_5}{\lambda_4}} + \left(\frac{z}{\lambda_3} \right)^{\frac{2}{\lambda_4}}$$

[5 parameters for shape] + [6 parameters for pose]

Optimization problem

$$\min_{\lambda} \sum_{i=0}^{N} \left(\underline{\sqrt{\lambda_1 \lambda_2 \lambda_3}} F^{\lambda_4}(x_i, y_i, z_i, \lambda) - 1 \right)^2$$

- shape indipendence
- minimum volume

Superquadric fuctions

inside-outside function:

- F > 1: point outside
- F < 1: point inside
- F = 1: point on surface

$$F(x, y, z, \lambda) =$$

$$\left(\left(\frac{x}{\lambda_1}\right)^{\frac{2}{\lambda_5}} + \left(\frac{y}{\lambda_2}\right)^{\frac{2}{\lambda_5}}\right)^{\frac{\lambda_5}{\lambda_4}} + \left(\frac{z}{\lambda_3}\right)^{\frac{2}{\lambda_4}}$$

5 parameters for shape + 6 parameters for pose

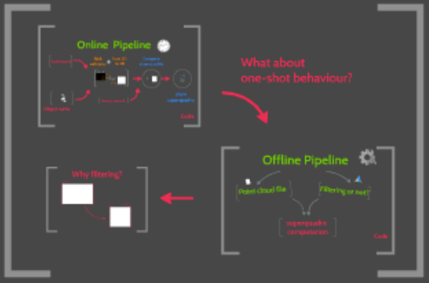
Optimization problem

$$\min_{\lambda} \sum_{i=0}^{N} \left(\underline{\sqrt{\lambda_1 \lambda_2 \lambda_3}} F^{\enclose{circle}{\lambda_4}}(x_i, y_i, z_i, \lambda) - 1 \right)^2$$

- shape indipendence
- minimum volume

Code

Pipeline



Dependencies

Dependencies:

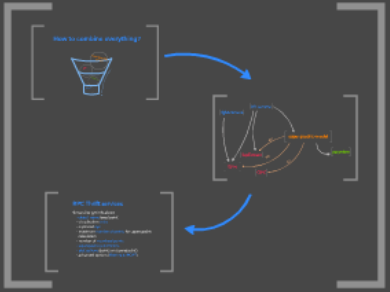
- YARP
- iCub
- OpenCV
- IPOPT Interior Point OPTimizer

External application

- IOL: (Interactive Object Learning)
  - lbpExtract (Local Binary Pattern Extraction)
  - SFM (Structure from Motion)
  - OPC (Object Property Collector)

Code

Connections



Dependencies:

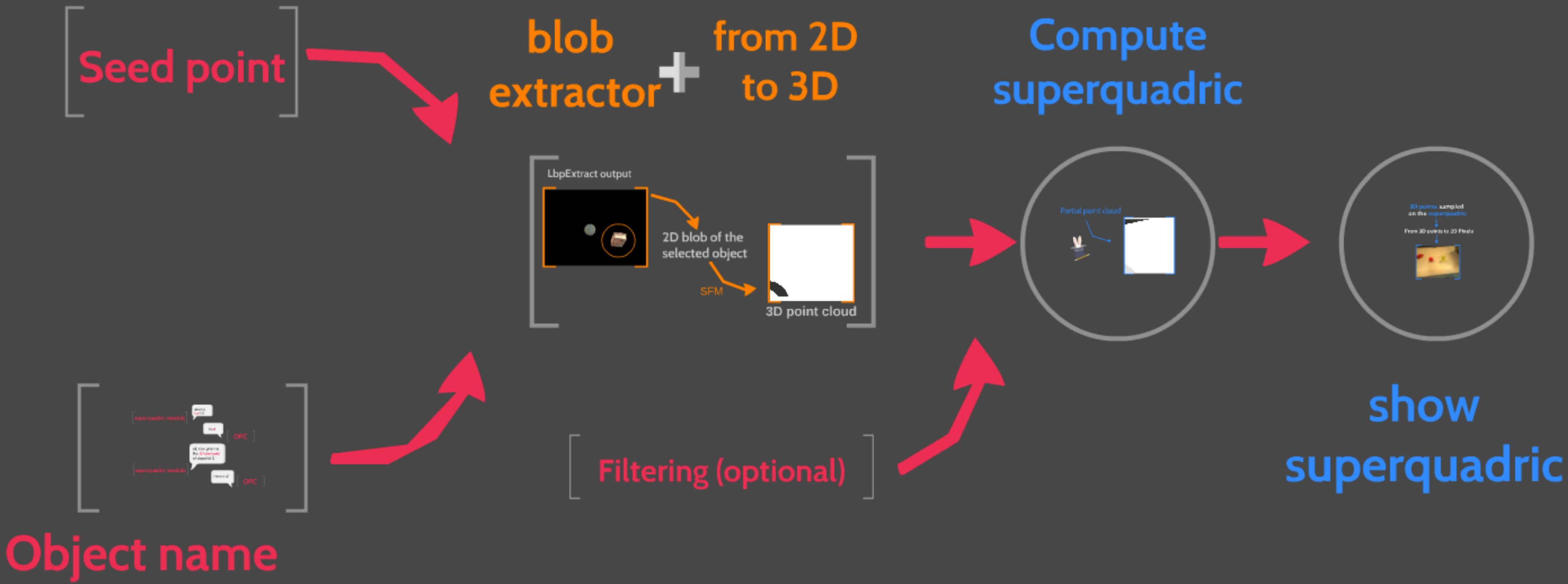
- YARP
- iCub
- OpenCV
- IPOPT Interior Point OPTimizer

External application

- IOL: (Interactive Object Learning)
  - lbpExtract (Local Binary Pattern Extraction)
  - SFM (Structure from Motion)
  - OPC (Object Property Collector)

Code

Online Pipeline



Seed point

Object name

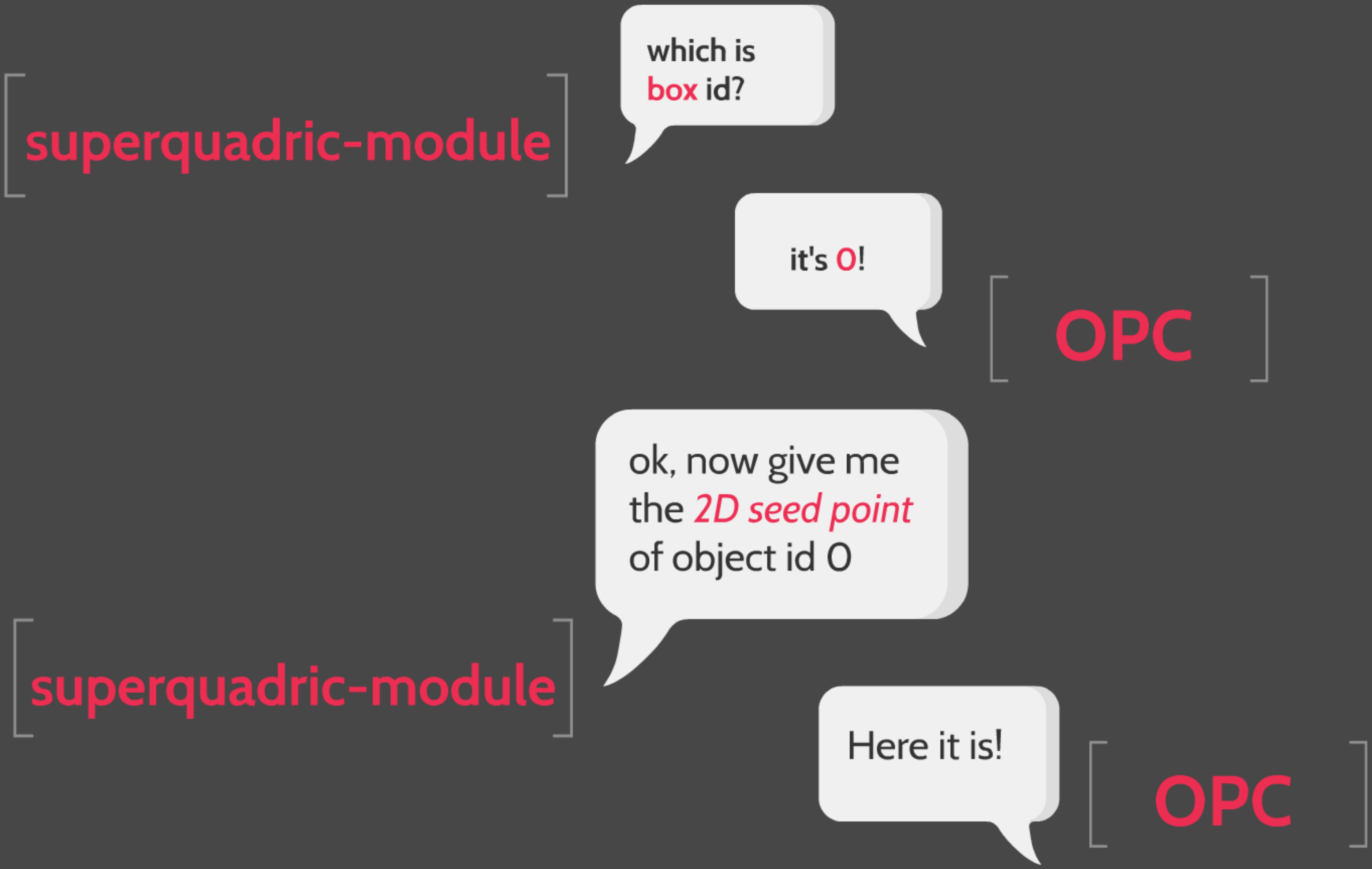
blob extractor + from 2D to 3D

Filtering (optional)

Compute superquadric

show superquadric

Code



superquadric-module

which is box id?

OPC

it's 0!

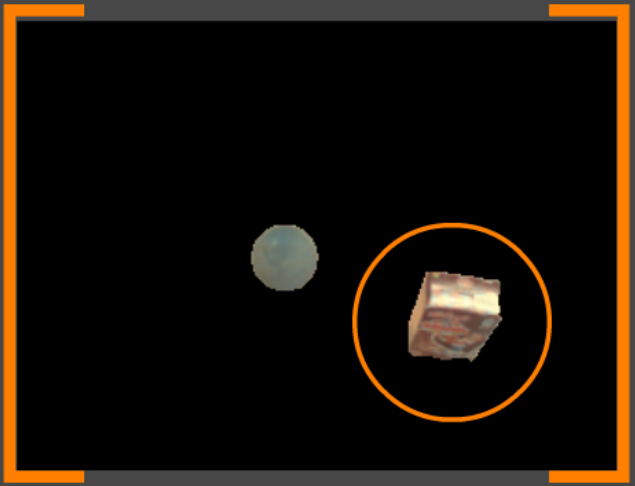
superquadric-module

ok, now give me the *2D seed point* of object id 0

OPC

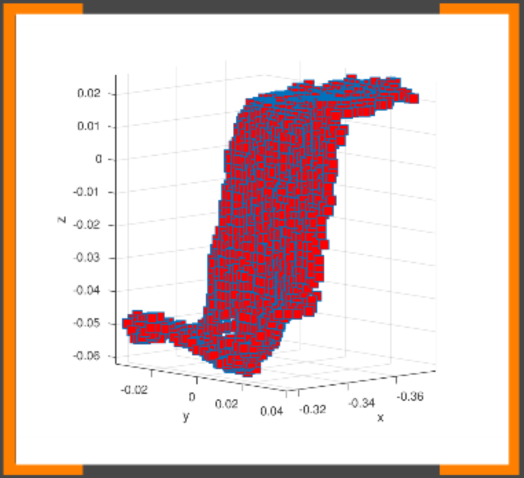
Here it is!

LbpExtract output



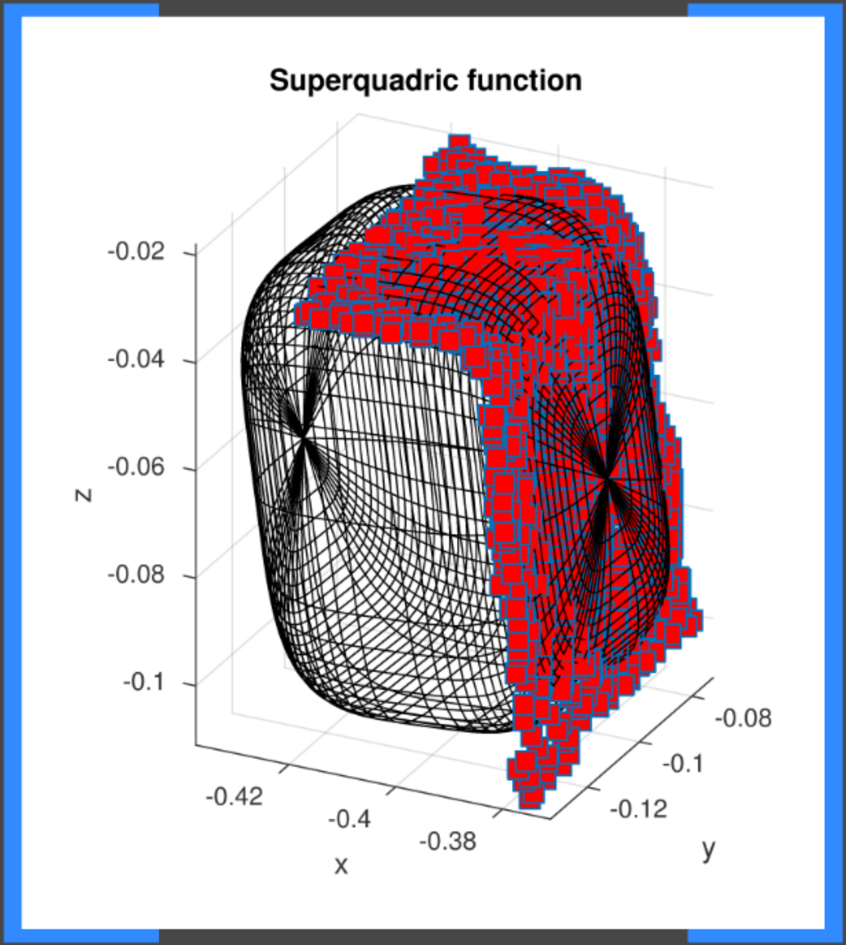
2D blob of the selected object

SFM



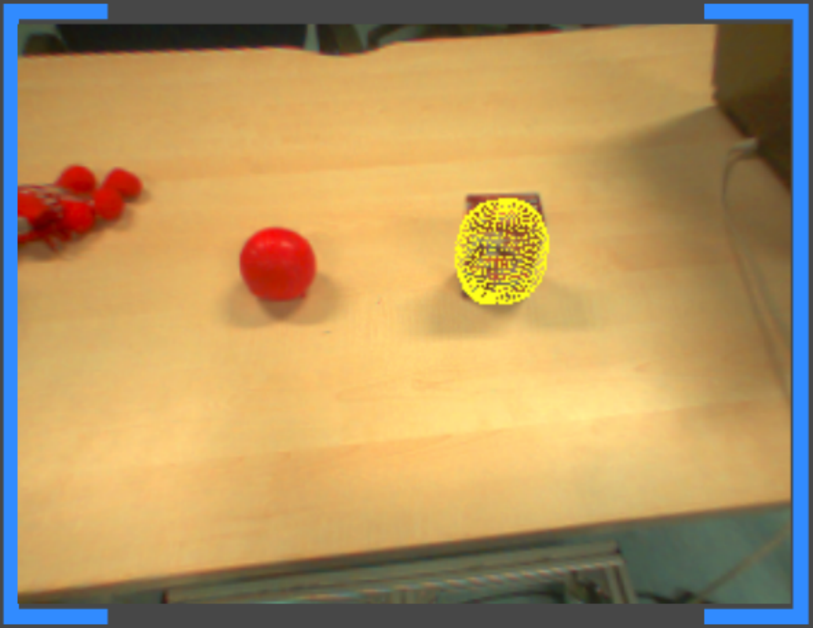
3D point cloud

Partial point cloud



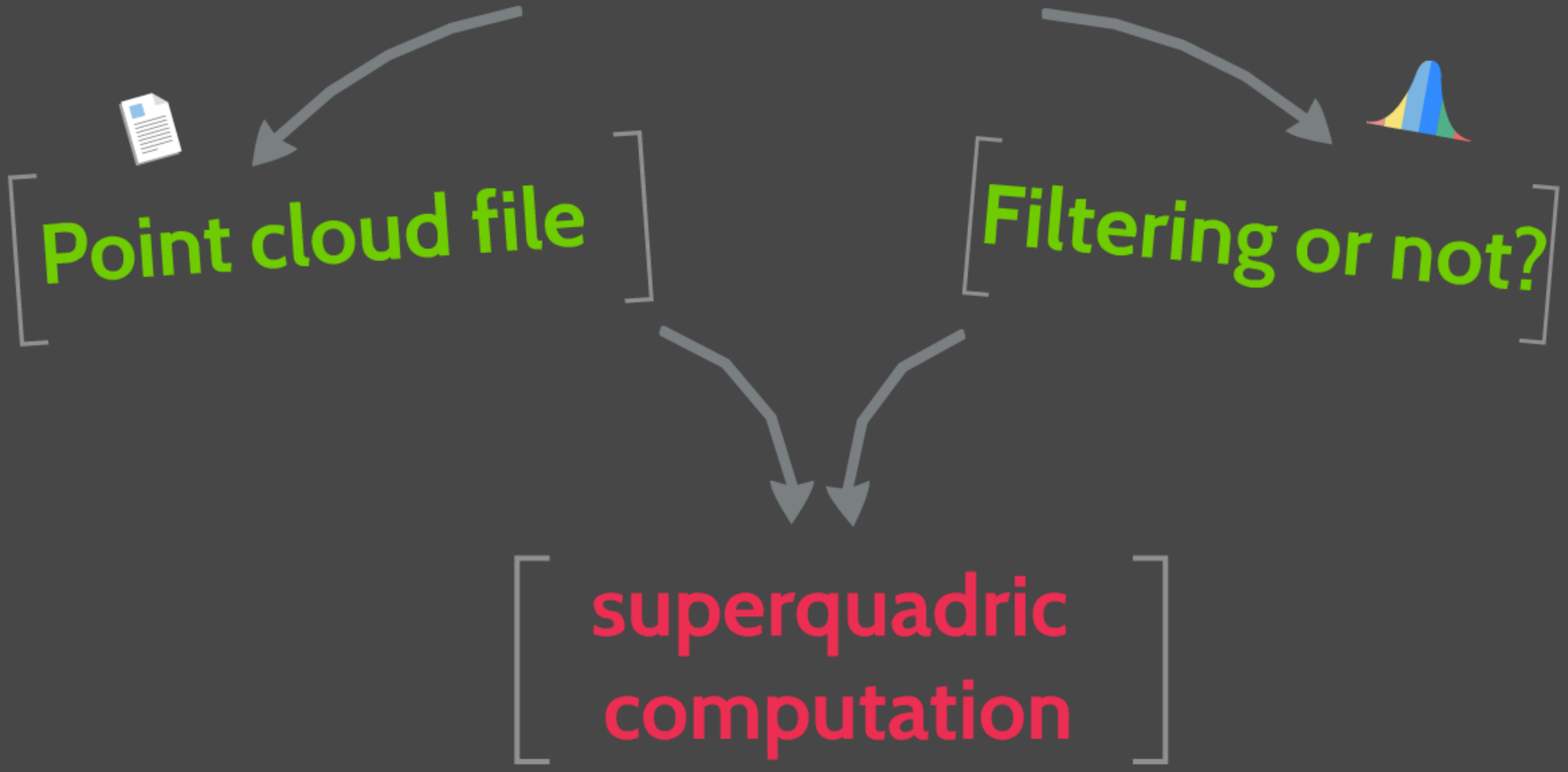
3D points sampled on the superquadric

From 3D points to 2D Pixels



What about one-shot behaviour?

Offline Pipeline



Point cloud file

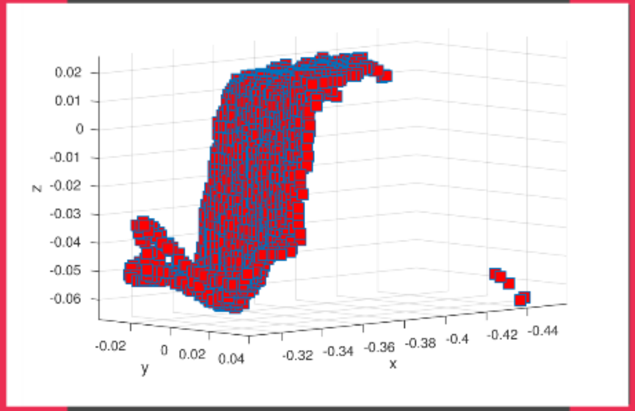
Filtering or not?

superquadric computation

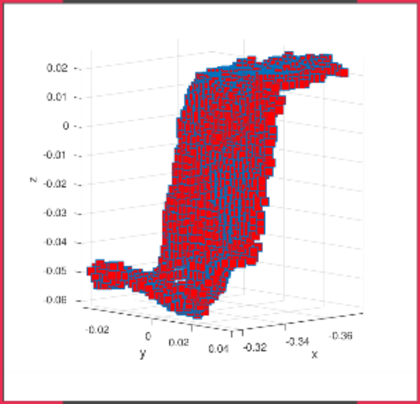
Code

Why filtering?

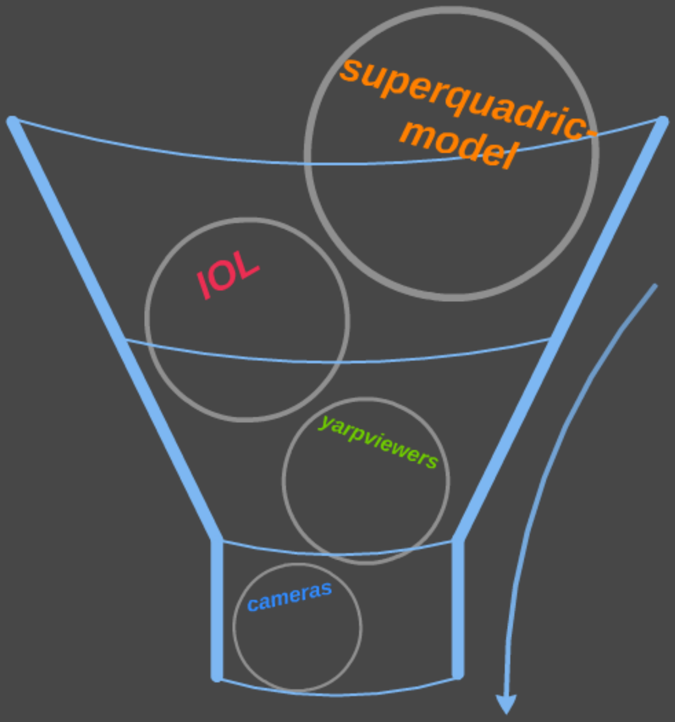
Before …

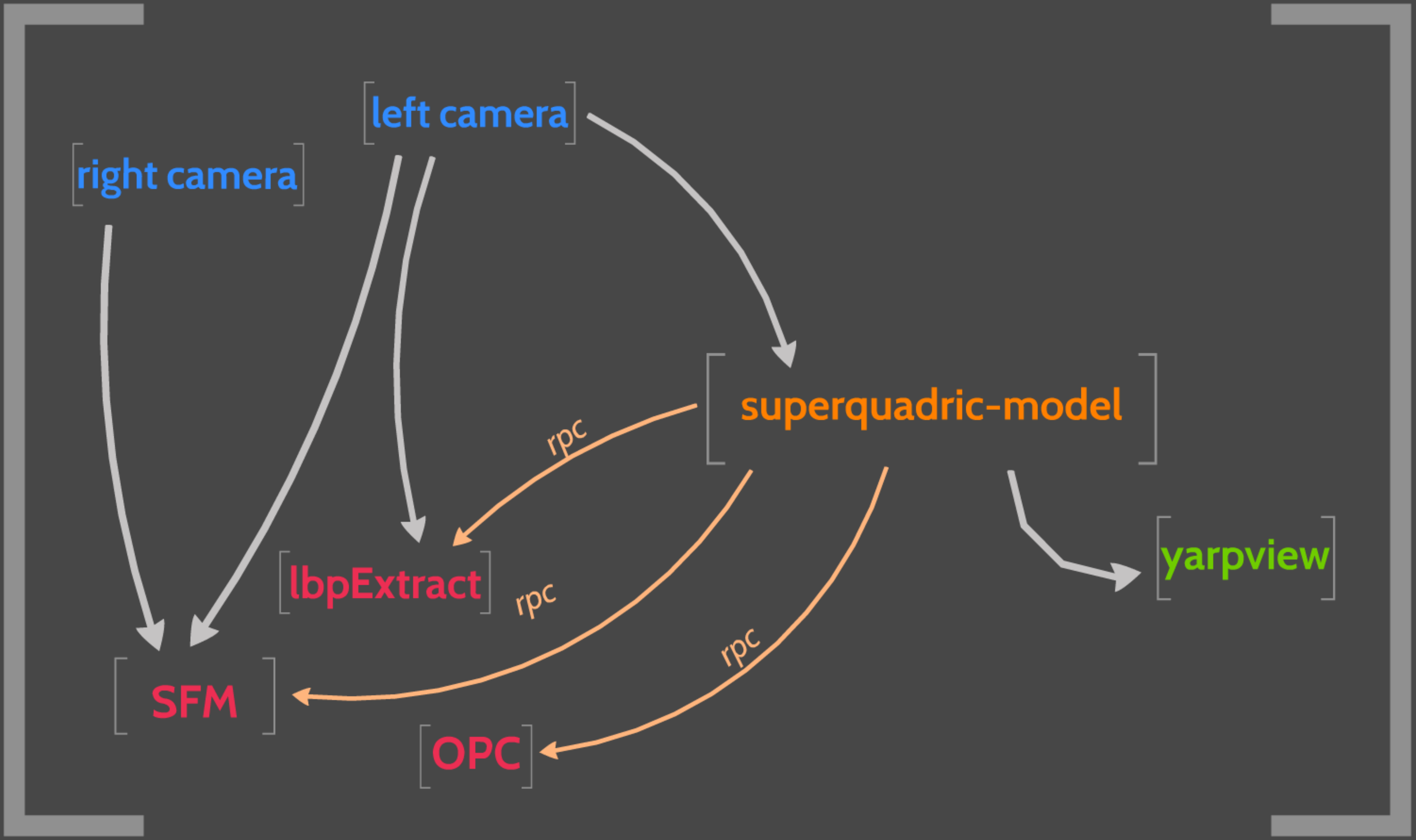


… after!



How to combine everything?





left camera

right camera

superquadric-model

rpc

rpc

rpc

lbpExtract

yarpview

SFM

OPC

RPC Thrift services

Set and/or get info about:

- object name/seed point
- visualization color
- exploited eye
- maximum number of points for superquadric calculation
- number of visualized points
- superquadric parameters
- plot options (points or superquadric)
- advanced options (filtering & IPOPT)

Researcher of the Week: superquadric-model module

Goal

Object detection and modeling

iCub looks at the object

the object is detected

a 3D model is computed

the model is visualized

Requests:

① Detection ✓
② Modeling ?
③ Visualization ?
④ ... and everything robust and in real-time!

Theory

Superquadric fuctions

inside-outside function:
- F > 1: point outside
- F < 1: point inside
- F = 1: point on surface

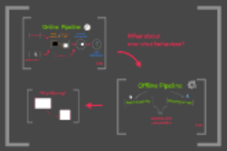
5 parameters for shape + 6 parameters for pose

Optimization problem

- shape independence
- minimum volume

Code

Pipeline



Dependencies

Dependencies:
- YARP
- iCub
- OpenCV
- IPOPT [illegible]

External application
- IOL (Interactive Object Learning)
  - lbpExtract [illegible]
  - SFM [illegible]
  - OPC [illegible]

Code

Connections

Giulia Vezzani - 13/05/2016

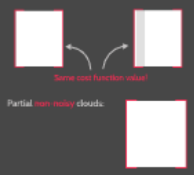
You can find
all the information about superquadric-model module on the github repo:

https://github.com/·robotology/superquadric-model

… What happens next?

Noteworthy ...

Wrong solution (at first glance)



Same cost function value!

Partial non-noisy clouds:

Possible solutions

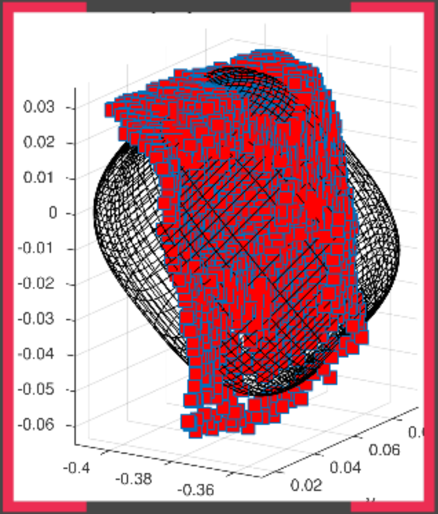
Ideas

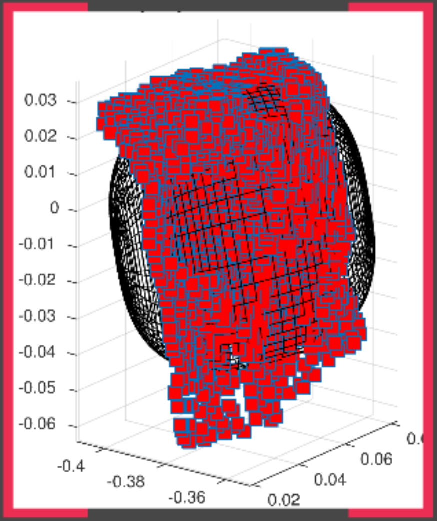
1. Merged point clouds (ICP)
2. Point cloud time refinement and filtering (Voxelization)
3. 2D surface overlapping
4. Different optimization problem formulation

Noisy point clouds

What the robot sees...

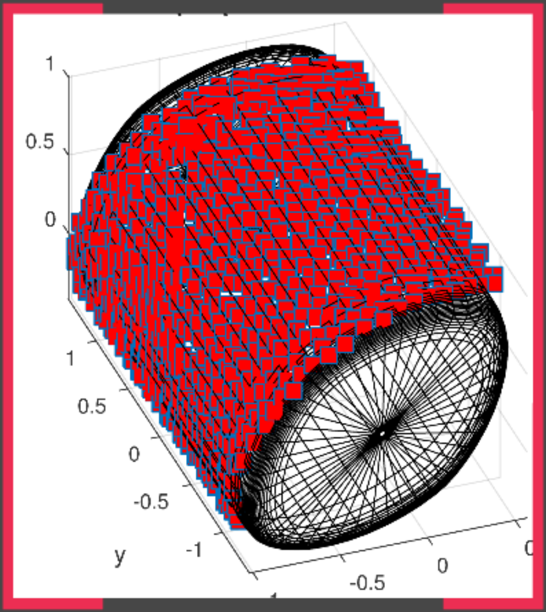
...what actually the robot sees



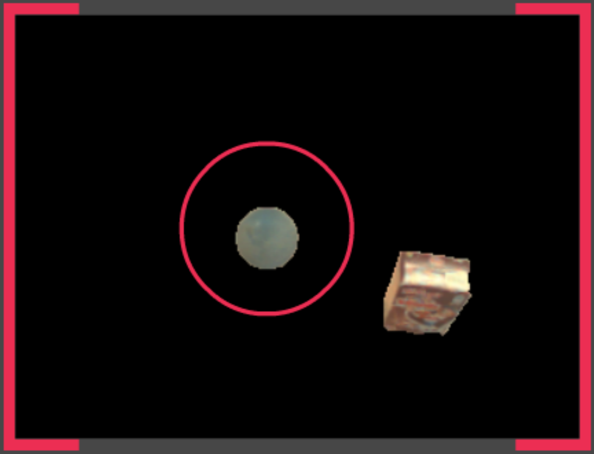


Same cost function value!

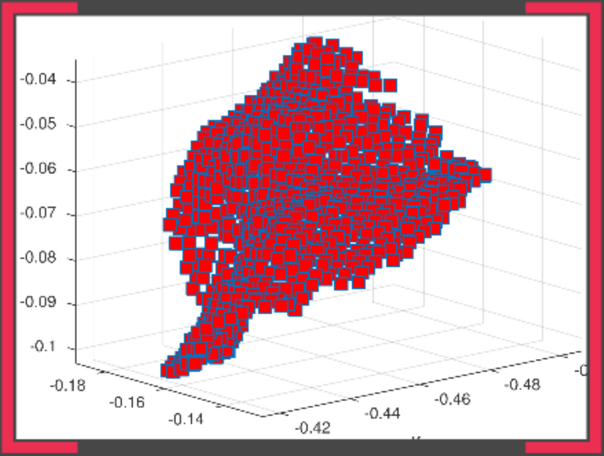
Partial non-noisy clouds:



What the robot sees..



..what actually the robot sees



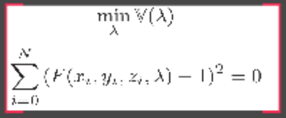
Ideas

1. Merged point clouds (ICP)

2. Point cloud time refinement and filtering (Voxelization)

3. 2D surface overlapping

4. Different optimization problem formulation



$$\min_{\lambda} \mathbb{V}(\lambda)$$

$$\sum_{i=0}^{N} (F(x_i, y_i, z_i, \lambda) - 1)^2 = 0$$

$$\min_{\lambda} \mathbb{V}(\lambda)$$

$$\sum_{i=0}^{N} (F(x_i, y_i, z_i, \lambda) - 1)^2 = 0$$

Future works

[coding & applications]

- Superquadric visualization ?

VTK/OpenGL

- Grasping application

Good grasping pose

Trajectory planning

Obstacle avoidance

Thank you for your attention!

..any questions or comments?

Then, let's try our code!