A Case-Study of ROS Software Re-usability: From Object Manipulation to In-Hand Manipulation

Guillaume Walck, Ugo Cupcic, Toni Oliver Duran and Veronique Perdereau
Content:
ROS: a versatile framework
(multi-robot / distributed system / stacks)
Adaptation of functionality
(extension / plugins / deriving / completing / combination)
New use of packages
Wrapping nodes
Architecture choices
(actionlibs / flow manager / integration)
A Case Study of ROS Software Reusability: From Object Manipulation to In-Hand Manipulation

Guillaume Walck, Ugo Cupcic, Toni Oliver Duran, and Veronique Perdereau

HANDLE robotic platform

- 4 DoF muscle arm
- 24 DoF motor hand
- RGB-D sensor
- Five 6-axis force and contact sensors
ROS handles multiple robots through wrapping of controller managers & merging of joint_states
Each computer of a distributed system MUST be synchronized with NTP.
Large amount of packages/libraries available in the community

**Environment Sensing**
- Openni / PCL filters / Robot self filters

**CollisionMap Processing**
- self occlusion / tabletop processing

**Environment Server**
- Real / Virtual env status

**Environment Analysis (PCL)**
- Cluster bounding box
- Tabletop object segmentation & recognition

**Household objects database**

**Object manipulation**
- Pick/Place
- Grasp planner / generator
- Gripper Action
- Reactive Grasp

**Reach & Grasp Planning and Object Manipulation (pick / place)**

**Control Interface**
- controller manager / controllers
- gripper or hand contact processors

**Arm Navigation**
- Motion planning (OMPL)
- IK (KDL)
- Trajectory filters
- move arm / trajectory controllers

**User Input : Select object**

**Real Robot**

**Simulated Robot**
Arm_navigation

Motion planning (OMPL)

IK (KDL)

Trajectory filters

move arm / trajectory controllers
Library extension

Orcos KDL

- WDLS solver
- 6D solvers
- ...

KDL coupling

- coupling manager
- 3D solvers

extension with a new package in the same namespace
**Environment Analysis (PCL)**

Cluster bounding box
Tabletop object segmentation & recognition

Household objects database
adapting for 5 fingered hand and new objects

Database completing

-edited-

hand_description.yaml

household object database

insertion

grasps

-5 fingers

objects

-mesh
ROS handles multiple robots through wrapping of controller managers & merging of joint states.
arm controller manager

controller manager wrapper
Plugin / Deriving

**pr2_controller_manager**

- position ctrl
- velocity ctrl
+ mixed pos vel ctrl: added

  robotstate (torque / pos / vel)

  ![Down Arrow]

**arm_controller_manager**

- N L position controller

  robotstate (pressure / pos / vel)
Each computer of a distributed system MUST be synchronized with NTP.

In-hand object tracking system

Combination

- AR marker
- ar_pose
- multi-marker tracker
- PTZ driver
- palm tracker
- tactile sensors
- contact location
- vision and tactile fusion

New functions through

functions combination

data combination

In-hand object tracking system
**Arm_navigation**

Motion planning (OMPL)
IK (KDL)
Trajectory filters
move arm / trajectory controllers
arm_navigation handles any serial manipulator

one finger is a serial manipulator

arm_navigation handles single finger planning (in-hand object is added as an obstacle)
Wrapping Database completing high-level actionlibs use wrappers to trigger functions. New high-level actions can re-use existing functionality.

**Object_manipulation**
- Pick/Place
- Grasp planner / generator
- Gripper Action
- Reactive Grasp

**Object Manipulation (pick / place)**
high-level actionlibs use wrappers to trigger functions
new high-level actions can re-use existing functionality

Wrapping

Pick
* detect object
* recognize object
* generate grasps
* test grasps
* plan movements
* reach
* approach
* grasp
* lift

Manipulate
* grasp transitions planning
* grasp transition execution
* arm motion planning
* arm motion execution
* in-hand actions planning
* in-hand actions execution

Place
* check place pose
* plan movements
* move
* approach
* release
* retreat

tabletop_object_detector
arm_navigation
household_objects_database
move_hand
grasp
move_arm
Step-based flow manager

Step1
Step2
StepN

Button

Gui

feedback: progress

param: checkbox state

service: button clicked?

Confirm PopUp

List display

Select  Cancel

Message for the request

Yes  No

service: display list
return selected item

service: display query
return response

Master Controller

ActionLib
Goal: steps to execute
Result: Execution result
Feedback: current step

* Execute step 1
* process result
if not autorun ask continue
* prepare next request with previous result
* Execute step N
* process result
...

MC Cancel
MC Goal (X-to-Y)
Next req
Next res
Param server

Cancel
Step1 Goal
Step1 Result
Step1 Feedback
StepN Goal
StepN Result
StepN Feedback

ActionLib Step1
Goal: data for step1
Feedback: current status
Result: step1 result

ActionLib StepN
Goal: data for stepN
Feedback: current status
Result: stepN result

ActionLibs
Services

Simulation
Real robot
Machine parameter to distribute computations
Master controller handles steps
Actionlib steps trigger actionlibs or services.
GUI + UI's and pop-ups handling service requests or feedbacks

Gui

Step1 (===== )
Step2 ( )
StepN ( )

Button

feedback: progress

checkbox

param: checkbox state

service: button clicked?

step specific UI 1

List display

Select

Cancel

Confirm PopUp

Message for the request

Yes

No

service: display list

return selected item

service: display query

return response
Grouping launch files by components

- top-level steps
  - steps actionlibs
  - master controller
- high-level nodes
  - arm_navigation
  - object_manipulation
- mid-level nodes
  - perception
  - planning
  - learning
  - database
  - utilities
- low-level drivers
  - hand
  - arm
  - kinect
  - ptz
  - tactile

Environment variables

- Simulation
- Real robot

Machine parameter to distribute computations