SkiROS2
Skill-based robot control system for ROS V.2

Bjarne Grossmann
Aalborg University Copenhagen
SkiROS

Software platform for the coordination industrial robots

Main features

• Skill-based robot control architecture
• Behavior trees execution system, for reactive behavior in dynamic environments
• Hardware-abstracted task description
• Semantic database server
• Integrated with PDDL task planner

ROS - enabled
SkiROS - A brief introduction

With SkiROS the user can teach and store variable features like:

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under the Grant Agreement No 723658
SkiROS in Scalable

**High-level**
Integration and synchronization with PM/APM

**SkiROS**
Task design and execution on robot

**Low-level**
Integration of hardware through HAL
SkiROS architecture

World model
The semantic database, manages task data and offers services to modify it and reason on it

Skill manager
The execution engine, manages information about available skills and offers services to execute and monitoring

Skills
Complex primitive skills can be exposed as ROS actions
SkiROS overview

World model
The semantic database, manages task data and offers services to modify it and reason on it

Skill manager
The execution engine, manages information about available skills and offers services to execute and monitoring

Skills
Complex primitive skills can be exposed as ROS actions
World model

Ontology
Semantic definition of concepts, data structures and relations of objects in the domain

Scene graph
Model of the current world state for reasoning, planning and execution
Scene graph

Skill manager

Scene

User / APM defined

Robot

Devices

Skills

Containers

Parts

TransformationPoses

Automatic synchronization with TF

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under the Grant Agreement No 723658
SkiROS overview

World model
The semantic database, manages task data and offers services to modify it and reason on it.

Skill manager
The execution engine, manages information about available skills and offers services to execute and monitoring.

Skills
Complex primitive skills can be exposed as ROS actions.
Skill manager

World model sync
Uploads robot description and available skills on world model

Execution services

<robot_name>/get_skills service: return the list of available skills
<robot_name>/command service: start/stop a skill execution
<robot_name>/monitor publisher: output feedback from skills
SkiROS overview

World model
The semantic database, manages task data and offers services to modify it and reason on it

Skill manager
The execution engine, manages information about available skills and offers services to execute and monitoring

Skills
Complex primitive skills can be exposed as ROS actions
## Skill concept revisited

<table>
<thead>
<tr>
<th>Skill</th>
<th>A process that can change the state of the robot and its environment.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primitive skill</td>
<td>A command that resides at the lowest level of the hierarchy and can be directly executed by the robot platform</td>
</tr>
<tr>
<td>Compound skill</td>
<td>A hierarchically organized collection of skills.</td>
</tr>
<tr>
<td>World state</td>
<td>Contains the current state of the world known by the robot</td>
</tr>
</tbody>
</table>

### World state

- Drive
- Locate
- Pick
- Compound skills
  - Place
- Primitive skills
  - Move
  - Grasp
  - Register
  - Drive
Skills

Definition:
A skill allows to transition from one world state to another, if its preconditions are met.
Skills

Problem:
A skill might be an action with a long duration. The system is **not reactive** during its execution. Complementary checks have to be encoded within the skill.
Skills

Idea:
Time slicing of a skill into n steps.

A skill becomes a Markov chain that is interruptible at each step.
**Solution:**

With the right design, a skill can be expressed in terms of an iterative function system

\[ s(x) = s_n(s_{n-1}(s_{n-2}(...(s_1(x))...))) \]
Compound skills

Hierarchical structure of set of skills

Often only used as sequences

- Compound Skill
  - do Skill A
  - do Skill B
  - if condition
    - true: do Skill C
    - false: do Skill D

Sequences are not flexible enough!
Intuitive way of representing and coordinating iterative skills are behavior trees

- Tree where leaves are actions to execute and nodes define the execution sequence
- Actions can return 3 states: Success, Failure or Running
- Execution is divided into discrete ticks, which propagate from the root node
- Allows change of execution at each tick

SkiROS extends this concept!
Motion Generators Combined with Behavior Trees: a Novel Approach to Skill Modeling

F. Rovida, D. Wuthier, B. Grossmann, M. Fumagalli and V. Krüger

Robotics, Vision and Machine Intelligence (KVI) Laboratory
Aalborg University Copenhagen
francesco.daw@bjarne.mf@vdk@mp.aau.dk
SkiROS overview

World model
The semantic database, manages task data and offers services to modify it and reason on it

Skill manager
The execution engine, manages information about available skills and offers services to execute and monitoring

Skills
Complex primitive skills can be exposed as ROS actions
## Task planning skill

<table>
<thead>
<tr>
<th>Task description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Put an alternator into the kit and drive it to station 1</td>
</tr>
</tbody>
</table>
Task planning skill

<table>
<thead>
<tr>
<th>Process flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task description</td>
</tr>
</tbody>
</table>

InKit(Alternator, Kit)
at(robot, Station-1)
Task planning skill

Process flow

- Task description
- Define PDDL goal
- Generated PDDL plan

InKit(Alternator, Kit)

at(robot, Station-1)

- drive(LargeBox-25)
- pick(Alternator)
- place(Kit-7)
- drive(Station-1)
Task planning skill

**Process flow**

- Task description
- Define PDDL goal
- Generated PDDL plan
- Generated Behavior tree

**Task description**

- InKit(Alternator, Kit)
- at(robot, Station-1)

**Generated PDDL plan**

- drive(LargeBox-25)
- pick(Alternator)
- place(Kit-7)
- drive(Station-1)
Task planning skill

Process flow

Task description → Define PDDL goal → Generated PDDL plan → Generated Behavior tree → Expansion

InKit(Alternator, Kit) → at(robot, Station-1)

- drive(LargeBox-25)
- pick(Alternator)
- place(Kit-7)
- drive(Station-1)

Task description

- Define PDDL goal
- Generated PDDL plan
- Generated Behavior tree
- Expansion

Task description skill

- Task description
- Define PDDL goal
- Generated PDDL plan
- Generated Behavior tree
- Expansion
Task planning skill

Process flow

- Task description
- Define PDDL goal
- Generated PDDL plan
- Generated Behavior tree
- Expansion
- Expansion

InKit(Alternator, Kit) -> drive(LargeBox-25) -> pick(Alternator) -> place(Kit-7) -> drive(Station-1)

Task description

- Define PDDL goal
- Generated PDDL plan
- Generated Behavior tree
- Expansion
- Expansion

Behavior tree

- drive
  - locate
  - build_observation_pose
  - gripper_oc
  - move_arm(Observable) -> plan_move
  - move_exe
  - registration
  - build_grasping_pose
  - move_arm(Grasp) -> plan_move
  - move_exe
  - hold
  - gripper_oc
  - move_arm(Home) -> plan_move
  - move_exe
  - place
  - locate_kit
  - build_placing_pose
  - move_arm(Place) -> plan_move
  - move_exe
  - release
  - gripper_oc
  - move_arm(Home) -> plan_move
  - move_exe
  - drive
SkiROS - Kit planning
SkiROS - Kinesthetic teaching

TEACHING PHASE

requirement #1: integrated teaching strategies

>> x4
Future of SkiROS

Source code publicly available (soon)

Founded RiACT as spin-off from Scalable
The RiACTivists

Magnus Philip Ritzau
Master / Business developer

Bjarne Grossmann
Post Doc / Software engineer

Francesco Rovida
Post Doc / Control engineer

Volker Krueger
Professor / Advisor
Hands-on!
Let’s program some turtles
git clone https://github.com/Bjarne-AAU/skiros-demo.git

cd skiros-demo

./scripts/install-skiros-repo.sh .

roslaunch demo_skills main.launch
The development process can be summarized in the following steps:

- Create a new OWL file with a new robot description, including hardware and other relevant properties.
- Develop necessary plug-ins:
  - Primitive skills
  - Compound skills
- Create a new ROS launch file running a skill manager with the new robot description and skills