RAFCON: A Graphical Tool for Engineering Complex, Robotic Tasks

Sebastian G. Brunner, Franz Steinmetz, Rico Belder, Andreas Dömel

DLR German Aerospace Center
Institute of Robotics and Mechatronics
Department of Cognitive Robotics

November 4, 2016
Outline

1. Introduction
2. Video
3. RAFCON Core
4. RAFCON GUI
5. Plugins
6. Use Cases
Introduction

Problem:

- Robotic tasks become increasingly complex
- No scalable solution to program robotic systems efficiently for specific scenarios

Our Solution:

RAFCON

- Tool for (visual) creation and execution of robotic tasks [1]
- State machine based
- Includes feature-rich GUI
- Result of implementation of long requirement list elaborated by experienced roboticists of DLR Robotic and Mechatronic Center (RMC)
Video

https://www.youtube.com/watch?v=35dUykJandU
RAFCON Core

- High level system control
- State machine based
- Coarse State Machine design:
  - States and Transitions
  - Hierarchy and concurrency states
  - Re-usage with library and template concept
  - Explicit data flow modeling between states
  - Not event based
- Completely separated from GUI
- Written in Python:
  - No tedious development cycles (modify, compile, restart)
  - State machine modifications possible during runtime
  - Glue language: easy integration of heterogeneous components
Formal State Machine Notation

- Formal notation uses predicate logic and set theory
- Left out for brevity (see RAFCON paper for details)
- Definitions enable comparison with other state machine approaches
- Future work: Reachability analysis and deadlock detection
- Inspired by:
  - Other state machine definitions: DFA (deterministic finite state machine) and FST (finite state transducer) [2]
  - Statecharts [3]
  - SyncCharts [4]
  - Flowcharts [5]
Core Features

- Human readable file format supports easy versioning (with e.g. git)
  - JSON

- Programmatic state machine generation
- Flexible execution: Pause, Resume, Step, Backward step

- Arbitrary starting points

- Offers easy error and preemption handling integration
RAFCON GUI

Inspired by:

- Bubbles (invented at RMC) [6]
- Lego Mindstorms’ NXT-G [7]
- Scratch (for education purposes) [8]
- ROS Commander [9]
- RobotFlow [10]
- Gostai Studio [12]
- Smach (graphical viewer only) [13]
RAFCON GUI - Details

1. Graphical State Machine Editor
2. State Editor
3. Library Manager
4. State Machine Tree
5. Global Variable Manager
6. Modification History
7. Execution History
8. Logging View
GUI Features

- Intuitive task development and fast prototyping
- Visualize huge state machines with deep hierarchies
- Panning and zooming mechanism like in a digital map (mental model of robotic task)
- Easy collaboration
- Different viewing modes
- Cooperation with professional designers (https://interaktionswerk.de)
- Improve effectiveness of information retrieval by using the four gestalt principles (Chang et al.[15]): proximity, similarity, continuity and closure

Image from TNW [14]
Plugins

- SME assembly planner visualization
- Model checking plugin using Divine
- ROS info plugin
- DDS based monitoring and remote execution plugin
RAFCON for reliable autonomous robotic systems: The LRU mobile robot at the SpaceBotCamp 2015 [16]
More Use Cases

RACELab:
HRI in industrial scenarios

SAPHARI:
HRI in medical scenarios

ROBEX:
Space Robotics
Questions?

http://rmc.dlr.de/rafcon

Open Source Release 2017


