Proceedings of the RIC Project Day

Workgroups ‘Framework & Standardization’ and ‘Manipulation & Control’

Frank Kirchner (Editor)
Thomas M. Roehr, Bertold Bongardt (Associate Editors)

06/2014
Founded in 1988, DFKI today is one of the largest nonprofit contract research institutes in the field of innovative software technology based on Artificial Intelligence (AI) methods. DFKI is focusing on the complete cycle of innovation – from world-class basic research and technology development through leading-edge demonstrators and prototypes to product functions and commercialization.

Based in Kaiserslautern, Saarbrücken and Bremen, the German Research Center for Artificial Intelligence ranks among the important ‘Centers of Excellence’ worldwide. An important element of DFKI's mission is to move innovations as quickly as possible from the lab into the marketplace. Only by maintaining research projects at the forefront of science DFKI has the strength to meet its technology transfer goals.

The key directors of DFKI are Prof. Wolfgang Wahlster (CEO) and Dr. Walter Olthoff (CFO). DFKI's research departments are directed by internationally recognized research scientists:

- Knowledge Management (Prof. A. Dengel)
- Cyber-Physical Systems (Prof. R. Drechsler)
- Robotics Innovation Center (Prof. F. Kirchner)
- Innovative Retail Laboratory (Prof. A. Krüger)
- Institute for Information Systems (Prof. P. Loos)
- Embedded Intelligence (Prof. P. Lukowicz)
- Agents and Simulated Reality (Prof. P. Slusallek)
- Augmented Vision (Prof. D. Stricker)
- Language Technology (Prof. H. Uszkoreit)
- Intelligent User Interfaces (Prof. W. Wahlster)
- Innovative Factory Systems (Prof. D. Zühlke)

In this series, DFKI publishes research reports, technical memos, documents (eg. workshop proceedings), and final project reports. The aim is to make new results, ideas, and software available as quickly as possible.

Prof. Wolfgang Wahlster
Director
Proceedings of the RIC Project Day

Workgroups ‘Framework & Standardization’ and
‘Manipulation & Control’

Frank Kirchner\textsuperscript{(1,2)} (Editor)
Thomas M. Roehr\textsuperscript{(1)}, Bertold Bongardt\textsuperscript{(1)} (Associate Editors)

\textsuperscript{(1)} DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany
\textsuperscript{(2)} Universität Bremen, Arbeitsgruppe Robotik, Robert-Hooke-Straße 1, 28359 Bremen, Germany

06/2014

Document D-14-03 des
Deutschen Forschungszentrums für Künstliche Intelligenz (DFKI)
Abstract

This document is the current edition of a publication series which records the topics, discussions and efforts of the workgroups at the DFKI Robotics Innovation Center (RIC). Each edition contains presentation slides and posters of a project day which is organized by two workgroups.

Workgroups provide a platform for cross-project communication and knowledge transfer. They are formed by peers dedicated to a specific topic. Each workgroup has one administrator. In 2008, the workgroups started to present their results and efforts in an open presentation format called brown-bag talk. From 2009 onwards, these presentations were held at so-called project days. Since 2014, a project day consists of two main parts: an oral session and a poster session. Both sessions are documented in a proceedings using the DFKI Document format.

Zusammenfassung

Dieses Dokument enthält die aktuelle Ausgabe einer Tagungsbandserie, welche die Themen, Diskussionen und Bemühungen der Arbeitsgruppen am DFKI Robotics Innovation Center (RIC) protokolliert. Jede Ausgabe enthält Vortragsfolien und Poster eines Projekttaages, der von je zwei Arbeitsgruppen gestaltet wird.

## Contents

Abstract vii

1 Editorial 2

2 ‘Framework & Standardization’ 3

2.1 FS-T-01: ‘AG Framework and Standardization’
   Alexander Duda, Thomas M. Roehr 3

2.2 FS-T-02: ‘Cross Compiling’
   Martin Zenzes 8

2.3 FS-T-03: ‘Artemis Rover – Model Based Engineering’
   Stefan Haase, Thomas M. Roehr et. al 22

2.4 FS-T-04: ‘Distributed compilation using IceCC’
   Steffen Planthaber 31

2.5 FS-T-05: ‘Data Distribution Service (DDS)’
   Ronny Hartanto 39

2.6 FS-P-01: ‘Planning with reconfigurable multi-robot systems’
   Thomas M. Roehr, Ronny Hartanto 48

2.7 FS-P-02: ‘FIPA-based multi-robot infrastructure for Rock’
   Satia Herfert, Thomas M. Roehr 50

3 ‘Manipulation & Control’ 52

3.1 MC-T-01: ‘Work Group Activities’
   Bertold Bongardt, Benjamin Girault, et al 52

3.2 MC-T-02: ‘Control of Flexible Link Manipulator’
   Ajish Babu 67

3.3 MC-T-03: ‘Motion Planning for Manipulator – MoveIt!’
   Sankaranarayanan Natarajan 76

3.4 MC-T-04: ‘Motion Planning Library for Robotic Application’
   Behnam Asadi 84

3.5 MC-T-05: ‘Trajectory generation using the Reflexxes library’
   Benjamin Girault, Malte Wirkus 89

3.6 MC-T-06: ‘Cascaded Robot Joint Control’
   Vinzenz Bargsten 103

3.7 MC-P-01: ‘Autonomous Steering Controller for Path Following’
   Mohammed Ahmed, Ajish Babu 118

3.8 MC-P-02: ‘Motion Planning for Manipulators’
   Behnam Asadi, Sankaranarayanan Natarajan 120

3.9 MC-P-03: ‘Distributed Dynamics Computation’
   Vinzenz Bargsten 122

3.10 MC-P-04: ‘Complex Numbers and Quaternions – A unified view on rotations’
   Bertold Bongardt 124

3.11 MC-P-05: ‘iTaSC: Application and Rock Integration’
   Dennis Mronga 126
1 Editorial

This is the first edition of a new format to document the efforts of the DFKI-RIC thematic workgroups. Workgroups are formed by peers and provide a means for cross-project communication on a deep content level and facilitate knowledge transfer amongst the peers. In 2008 we first started forming workgroups on specific topics around robotics and AI research. Among them were topics as ‘system design & engineering’, ‘machine learning’, ‘planning & representation’ as well as ‘frameworks & architectures’ and ‘man-machine interaction’. These workgroups were established with the intention to provide a platform for interested DFKI-RIC personnel for discussing the start of the art, recent achievements, and future developments in the respective fields.

Over time the workgroups gathered a collection of material in form of presentations, short papers, and posters which were worthwhile to be presented also to the rest of the institute. Due to this development, in 2009, we started to have a project day once every quarter. Each project day provided a platform for two of the workgroups to present their material and to discuss it with the further colleagues of the institute. Nowadays, the project day is organized as a one-day workshop with oral presentations, poster sessions, and a free pizza lunch for everybody who attends. Until now, the talks and posters have only been collected on our servers but were not assembled in a citable document.

This format at present is the next evolutionary step and it aims at eliminating this deficit by compiling the material of the workgroups presented during a project day into a single, citable document of unified format. We will see which steps can be taken in the future to enhance the presentation quality of this material.

Frank Kirchner

This year’s second project day presented the material of the workgroups ‘Framework & Standardization’ and ‘Manipulation & Control’.

The workgroup ‘Framework & Standardization’ focuses its efforts on continuously improving the software development workflow and aims at supporting a software framework which fulfills the special needs in the domain of robotics. The workgroup’s main motivation is to facilitate and accelerate routine tasks and to increase the robustness of the developed software. The workgroup has successfully established the Robot Construction Kit (Rock) as the main in-house development framework which can coexist with the well-known Robot Operating System (ROS). Furthermore, it has introduced a flexible git-based hosting for software projects in parallel to the existing subversion-based infrastructure. The presentations of this year are dealing with workflow optimization and the requirements of distributed systems with respect to modeling and infrastructure.

The purpose of the workgroup ‘Manipulation & Control’ is to discuss problems and solution approaches in the fields of kinematics, dynamics, and joint control. Questions in these areas arise commonly during the development of the various, unique robotic systems that are manufactured at the DFKI-RIC. The findings of the workgroup’s members enable to apply the control and planning approaches to the mechatronic systems of the institute. These include walking machines, wheeled systems, robotic arms, exoskeletons, and the humanoid AILA. The contributions of the workgroup to the project day 2014 distribute across the areas ‘motion planning along waypoints’, ‘generation of dynamic trajectories’, ‘real-time computation of manipulator dynamics’, and ‘representation of spatial rotations’.

We would like to thank the authors of the second project day 2014 for their contributions and for the effort to provide their material in a standardized format.

Thomas M. Roehr, Bertold Bongardt
2 ‘Framework & Standardization’

2.1 ‘AG Framework and Standardization’ (FS-T-01)

Alexander Duda(1), Thomas M. Roehr(1)

(1) DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany
(2) Universität Bremen, Arbeitsgruppe Robotik, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: alexander.duda@dfki.de, thomas.roehr@dfki.de

Abstract

This introduction provides a compact outline of the activities of the work group over the past year. The activities cover the past organization of Rock workshops and improving the build and development infrastructure and also points to upcoming changes.
Ongoing: Rock workshops

- Outreach
  - Require installation: 15
  - Newbie: 17
  - Beginner: 12
  - Intermediate: 9
  - Advanced: 1 (does not include maintainers ;)

- Workshops
  - Installation, Guided / Mentored Tutorial sessions
    - average 4-5 participants p/workshop, i.e. 8-10 p/Topic
  - Mentors: AG members
Ongoing: Distributed Compilation

icemon screenshot – when using distributed compilation

Ongoing: Moving from gitorious to github
Ongoing: Building Debian packages

Using OpenSUSE for building Debian packages for Rock

TBD: Upgrading spacegit

Upgrading Spacegit from „experimental“ to production, i.e. to latest version of gitorious
... and still going ...

Spacegit

Buildserver

Robot Construction Kit (Rock)

... suspended ...

Reference architecture, though there is
2.2 ‘Cross Compiling’ (FS-T-02)

Martin Zenzes$^{(1)}$

$^{(1)}$DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: martin.zenzes@dfki.de

Abstract

This presentation provides a general introduction into the challenges of cross compiling for different types of target systems. It illustrates the need for cross compilation and presents the existing cross compiling approach. Furthermore, it motivates the need for an integrated cross-compilation workflow to fully support cross platform software development.
hello_world.cpp

#include <iostream>

int main(int argc, char* argv[])
{
    std::cout << "hello world!\n";
    return 0;
}
ELF 64-bit LSB executable, x86-64, version 1 (SYSV)

ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV)
Contents

Introduction

Cross-Compiler

Software Development Kit
Introduction

- Heterogeneous environments
- Use case → architecture → machine code
  
  \textbf{x86} lot of power and volume, fast  
  \textbf{ARM} efficient, flexible PCB designs, slow  
  \textbf{MIPS} even more efficient, even fewer resources  
  \textbf{$other}$ interesting quirks, different instruction set  

- Some architectures are cumbersome to use  
- Native development sometimes impossible  
- Not always need for build-environment \textit{inside} target system  
- Pre-compiling and binary distribution speeds up co-workers
Virtualization
They use it in the cloud, don’t they?

- Simulate target system, with fewer constraints
- Use simulated native tools for development
- VirtualBox, LXC, Qemu, VMware...
- Unsupported syscalls, sloooow

Full system:
- Full OS (kernel+userland)
  - Can boot production image
- Filesystem access a hassle
- Reference system to be shared among developers

User mode:
- Detect foreign binaries, emulate on the fly
- Combine with chroot, LXC
- Use hosts kernel (+userland)

U-Boot + Linux on Qemu

Contents

Introduction

Cross-Compiler

Software Development Kit
Cross-Compiler
Dammit, just gimme what I need...

Tasks:
▶ Compile and link on developers system (host)
▶ Create binaries to be executed on target system
▶ Headers and libraries present on the host, suitable for target
▶ Keep target hardware out of the loop

Get one:
▶ Ubuntu: shipped by default!
sudo apt-get install gcc-arm-linux-gnueab
▶ Debian: from Emdebian repository
▶ Configure & build your own (buildroot, crosstool-NG, bitbake)
Cross-Compiler
Split development and execution

▶ Cross-Compiler:

host  System on which the compiler is building software

target This is the platform it will generate machine code for

▶ Identify compiler with triplet:

- arch[-vendor][-os]-abi

▶ 30 years of erosion: only loose policy survived...

```
guy@host:~$ gcc -dumpmachine
x86_64-linux-gnu
# others (non exhaustive):
  arm-linux-gnueabihf
  arm-none-eabi
  x86_64-pc-cygwin
  i386-unknown-openbsd
  i686-apple-darwin10-gcc-4.2.1
```

Cross-Compiler
Split build, development and execution

▶ Canadian-Cross-Compiler:

build  System on which the compiler is building a compiler

host  System executing the resulting compiler to compile

target This is the platform it will generate machine code for

▶ Identify compiler with triplet:

- arch[-vendor][-os]-abi

▶ 30 years of erosion: only loose policy survived...

```
guy@host:~$ gcc -dumpmachine
x86_64-linux-gnu
# others (non exhaustive):
  arm-linux-gnueabihf
  arm-none-eabi
  x86_64-pc-cygwin
  i386-unknown-openbsd
  i686-apple-darwin10-gcc-4.2.1
```
Cross-Compiler
Split build, development and execution

- Canadian-Cross-Compiler:
  - build System on which the compiler is building a compiler
  - host System executing the resulting compiler to compile
  - target This is the platform it will generate machine code for

- Identify compiler with triplet:
  - arch[-vendor][-os]-abi

- 30 years of erosion: only loose policy survived...

```
guy@host:~$ gcc -dumpmachine
x86_64-linux-gnu
# others (non exhaustive):
arm-linux-gnueabihf
arminone-eabi
x86_64-pc-cygwin
i386-unknown-openbsd
i686-apple-darwin10-gcc-4.2.1
```

Vocabulary

- **binutils** Tools for creating and managing binaries, object files, libraries, profile data, and assembly code
- **prefix** Where to install the software in the target system
- **sysroot** Location for toolchain on the build machine
- **gnueabihf** *Embedded ABI* for GNU on ARM with hardfloat
- **bare metal** No OS, binary executed directly on target CPU
- **native** Binaries which can be executed on *this* system

- Proper compiler location on the host:
  - $SYSROOT/$PREFIX/bin/$TARGET-gcc
- A cross-toolchain cannot be used inside target!
2.2 ‘Cross Compiling’ – Martin Zenzes

guy@host:~$ ls
hello_world.cpp

guy@host:~$ g++ -print-sysroot
/

guy@host:~$ g++ -dumpmachine
x86_64-linux-gnu

guy@host:~$ g++ hello_world.cpp -o hello_world-x86_64

guy@host:~$ file hello_world-x86_64
hello_world-x86_64: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.32, BuildID[sha1]=8c2504a1f446f0751b51319fc6fa269513e764c6, not stripped

guy@host:~$ ./hello_world-x86_64
hello world!

guy@host:~$ arm-poky-linux-gnueabi-g++ -print-sysroot
/opt/poky/1.4.3/sysroots/armv7a-vfp-neon-poky-linux-gnueabi

guy@host:~$ arm-poky-linux-gnueabi-g++ -dumpmachine
arm-poky-linux-gnueabi

guy@host:~$ arm-poky-linux-gnueabi-g++ hello_world.cpp -o hello_world-arm

guy@host:~$ file hello_world-arm
hello_world-arm: ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV), dynamically linked (uses sharedlibs), for GNU/Linux 2.6.16, BuildID[sha1]=e5d3665c193b9d864378f9b5410ba9bfb564ec66, not stripped

guy@host:~$ qemu-arm-static -L $(arm-poky-linux-gnueabi-g++ -print-sysroot) ./hello_world-arm
hello world!
hello_world.cpp

x86_64-linux-gnu

hello_world-x86_64: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.32, BuildID[sha1]=8c2504a1f446f0751b51319fc6fa269513e764c6, not stripped

hello world!

hello_world-arm: ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.16, BuildID[sha1]=e5d3665c193b9d864378f9b5410ba9fb564ec66, not stripped

hello world!
guy@host:~/.dev$ ls
hello_world.cpp

guy@host:~/.dev$ g++ -print-sysroot
/

guy@host:~/.dev$ g++ -dumpmachine
x86_64-linux-gnu

guy@host:~/.dev$ g++ hello_world.cpp -o hello_world-x86_64

guy@host:~/.dev$ file hello_world-x86_64
hello_world-x86_64: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.32, BuildID[sha1]=8c2504a1f446f0751b51319fc6fa269515e764c6, not stripped

guy@host:~/.dev$ ./hello_world-x86_64
hello world!

Arne@poky:~/.poky$ arm-poky-linux-gnueabi-g++ -dumpmachine
arm-poky-linux-gnueabi

Arne@poky:~/.poky$ arm-poky-linux-gnueabi-g++ hello_world.cpp -o hello_world-arm

Arne@poky:~/.poky$ file hello_world-arm
hello_world-arm: ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.16, BuildID[sha1]=e5d3665c193b9d864378f9b5410ba9bf5b64e66, not stripped

Arne@poky:~/.poky$ qemu-arm-static -L $(arm-poky-linux-gnueabi-g++ -print-sysroot) ./hello_world-arm
hello world!

Contents

Introduction

Cross-Compiler

Software Development Kit
You need more than a Cross-Compiler
Configure, compile, link, test, deploy, execute...

Piece it all together:

- Initialize environment variables: $CC, $CFLAGS etc
- provide build-time dependencies
- compile and link into executables
- create binary packages of generated software
- check whole process for common pitfalls
- prevent native tools from creeping in
- provide simulated environment for testing

This quickly becomes tedious...

You need more than a Cross-Compiler
Configure, compile, link, test, deploy, execute...

Piece it all together:

- Initialize environment variables: $CC, $CFLAGS etc
- provide build-time dependencies
- compile and link into executables
- create binary packages of generated software
- check whole process for common pitfalls
- prevent native tools from creeping in
- provide simulated environment for testing

This quickly becomes tedious...
Questions? Remarks?
Abstract

Future application of robotic missions in the space context will require the systems to have both mobility and manipulation capabilities. The limited direct communication with the systems due to visibility, and severe time delays also make it a requirement for the system to perform its actions mainly autonomously. The increasing complexity of the task, as well as the strict requirements for reliability and fault tolerance pose a significant challenge to both engineering and research activities. The SpaceBot Cup was held in November 2013 to probe those capabilities in the context of a competition. In this paper we present the Artemis rover and its software architecture as well as the competition results and lessons learned. Special attention is given to the modular design based on the Robot Construction Kit (Rock); a component based software framework, which uses a component model based on the Orocos Real-Time-Toolkit (RTT).
The Artemis Rover as an Example for Model Based Engineering in Space Robotics

Jakob Schwender, Thomas M. Roehr, Stefan Haase, Malte Wirkus, Marc Manz, Sascha Arnold and Janosch Machowinski

presented by Stefan Haase

DFKI Bremen & Universität Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
robotics@dfki.de

The Spacebot Cup

• Development of an autonomous mobile manipulation system within 8 months

• Task:
  ▪ unknown exploration area (21x21.5m²) containing three target objects
  ▪ find and collect two objects
  ▪ find main object and assemble all

• Constraints:
  ▪ remote operation allowed up to three times up to 5 minutes each
  ▪ communication delay of 2s (one-way)
Development approach

- Top down
  - high-level mission decomposition and identification of required capabilities
  - distribution of tasks to specialized (sub-)teams
  - maximization of component and library reuse

- Main development lines / (sub) teams
  - Hardware
    - Arm
    - Manipulator
    - Rover
    - Wheels
  - Software
    - Navigation
    - Manipulation
    - Exploration
    - Object detection
    - Integration

Artemis - Hardware

- Velodyne HDL-32E
- AVT Prosilica
- XSens IMU MTi 10
- Cup Storage
- W-Lan Modem
- Intel Core i7 Computer
- Battery Storage
- Wheel Module
- Elastic Wheel
- Six DoF Arm
- Two Finger Gripper
Artemis - Software

• Application of Robot Construction Kit (Rock) as basis
  ▪ model-based
  ▪ component-based
  ▪ established workflows and infrastructure for efficiently
    ▶ embedding external library
    ▶ performing library and component updates
    ▶ managing network of components

• Rock allows to interface with ROS components (nodes)
  ▪ managing component networks can deal with Rock and ROS components

Artemis – Model-based

Workflow

Levels of functionality

Actions

Compositions

Components

Drones / Libraries (C/C++)

+ ruby glue code

+ ruby glue code

move_to

Navigation

Universität Bremer
Model-based Components

- Specification uses a domain specific language (DSL)
  - Orocos RTT as component model

```
using_library "message_drivers/";
import_types_from "message_driver/message.hpp";

task_context "Task" do
  output_port "messages", "message_driver/message", periodic(0.2)
end
```

- Specification is applicable to other component models, e.g. ROS Nodes

<table>
<thead>
<tr>
<th>Orocos</th>
<th>ROS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task Context</td>
<td>Node</td>
</tr>
<tr>
<td>Port</td>
<td>Topic</td>
</tr>
<tr>
<td>Deployment</td>
<td>Launcher</td>
</tr>
</tbody>
</table>

Artemis - Overview
Navigation

Manipulation
Systems Management

Workflow

Manage compositions (aka component networks)

An Overview

Component development

Management of complex systems

Data analysis

autoproj  Build system

orocos.rb  Ruby-interface for components

data_logger  High-performance data logging

RTT  C++ component implementation (Orocos RTT)

Syskit  Model-based deployment and system supervision

pocolog  Log file handling

oroGen  Model-based component development

vizkit  Data visualization and log replay

typegen  Standalone typekit generation

Universität Bremer
Summary

• Artemis served to validate the current state of our model-based development approach
  – It showed to us that we made a good step towards a ‘less painful’ integration process for robotics

• Our robotic systems will become more complex

• Managing complexity will be our main challenge

• Rock is open-source:
  http://rock-robotics.org

Outlook

• DLR announced the next SpaceBot Cup in 2015!!
• Possibility to test the systems in a Mars like environment in October 2014 in Noordwijk (ESA’s ESTEC technical centre)
Thank you!

DFKI Bremen & Universität Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
robotics@dfki.de
2.4 ‘Distributed compilation using IceCC’ (FS-T-04)

*Steffen Planthaber*\(^{(1)}\)

\(^{(1)}\) DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: steffen.planthaber@dfki.de

**Abstract**

This presentation details the usage of IceCC. IceCC serve as distributed compilation platform and can speed up the compilation process using a cloud-like service. Details are provided on how to monitor IceCC which runs as a system service and how to tune it to maximize compilation speed.
Distributed compilation using IceCC
by Steffen Planthaber

DFKI Bremen & Universität Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
robotics@dfki.de

Icecream (IceCC)

- Originally developed by SUSE
- Distributes compile jobs in the network
- High parallelism possible >100 jobs in parallel
  - Depends on local memory
- No dependencies on other machines
  - Local preprocessing and linking
- Works for linux and gcc
  - GCC version is checked
- In use for about a year
How it works:

• Scheduler
  ▪ Organizes compiler nodes and jobs
  ▪ Is aware of the load and speed of the clients

• Client
  ▪ Local machine (iceccd daemon)
  ▪ If no sheduler reachable, compiles locally
  ▪ Sheduler can assign compile tasks to the client (low priority)

Setup

• “Normal“
  ▪ Install apt-get install icecc
  ▪ Icecc has own versions of c++, cc, g++, gcc stored under:
    /usr/lib/icecc/bin
  ▪ All you need to to is to set the PATH to find the icecc versions first
  ▪ Regenerate CMakeFiles!

• Autoproj
  ▪ Add the - spacegit: compilerspeedup/package_set to your manifest
  ▪ autoproj update and autoproj force-build
  ▪ Icecc will be enabled by sourcing the env.sh
icemon

- Size defines the current load
- Color the origin

icemon

- Color: Origin of files compiled by node
- White: Linking
Setup

- /etc/icecc/icecc.conf
  - Nice level of running compilers (default is 5: low priority)
  - Number of jobs to run in parallel
  - Turn compilations by other nodes on this machine off
    - Nice for robots, compiling cannot disturb demos or experiments ;-)
  - If the daemon can't find the scheduler by broadcast (e.g. because of a firewall you can specify it.
    - Might be needed in DFKI-WPA wlan
      » Scheduler is "eurex1"

Numbers (from icecc doc)

- Single file
  - g++ on my machine: 1.6s
  - g++ on fast machine: 1.1s
  - icecream using my machine as remote machine: 1.9s
  - icecream using fast machine: 1.8s

- Two files
  - g++ -j1 on my machine: 3.2s
  - g++ -j1 on the fast machine: 2.2s
  - using icecream -j2 on my machine: max(1.7,1.8)=1.8s
  - using icecream -j2 on the other machine: max(1.1,1.8)=1.8s
Numbers

- Rebuild: base/orogen/types
  - Easily parallelizable
    - Normal (4 threads)
      - real 5m25.422s
    - IceCC (15 threads allowed (default, recommended))
      - real 1m47.574s
    - IceCC (50 threads allowed):
      - real 1m40.066s
    - IceCC (150 threads allowed):
      - real 1m45.972s
  
- Not linear, linking and preprocessing local + overhead due to network communication
- 150 Threads allowed does not mean they are all used
- When more parallel builds are used, slower nodes are selected

Attention

- Having more parallelism results in more memory usage
  - Extremely slow when HD swapping starts
  - Configuration option for autoprog update -- reconfigure

- No distributed builds, if --march= parameter is passed to gcc/g++
  - PCL has --march=native
Disabling

- IceCC works from environment variables:
  - Autoproj update –reconfigure
  - When asked about icecc say „no“
    - env.sh is now updated, but not re-loaded
  - Open new console
  - autoproj force-build
    - To re-generate CMakeFiles

Buildserver: „jenkins“

- [http://buildsrv01:8080](http://buildsrv01:8080)
- Jenkins can build and test your code on different OS setups
  - Ubuntu LTS, Ubuntu Current and Debian Testing
    - Currently 12.04, 14.04
  - Also builds autoproj –based projects
    - Demo
- Notification via email or RSS
- Full log files available (build != node)
  - Logs are in the node folder, not in the build folder
- Also can use icecc for faster compiling
- Different build types, incremental, bootstrap, auto
Thank you!

DFKI Bremen & Universität Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
robotics@dfki.de
2.5 ‘Data Distribution Service (DDS)’ (FS-T-05)

Ronny Hartanto$^{(1)}$

$^{(1)}$ DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: ronny.hartanto@dfki.de

Abstract

This presentation describes the application of the Data Distribution Service (DDS) to establish a multi-robot communication infrastructure. Experience with DDS has been gained in the project IMPERA where OpenSpliceDDS has been used as actual implementation for DDS.
Data Distribution Service – (DDS)

Dr. Ronny Hartanto
Project Day, 19.06.2014

DFKI - Labor Bremen & Universität Bremen
Forschungsgruppe Robotik
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
robotics@dfki.de

Outline

• Multi-robot Communication Challenges
• Some Communication Middleware
  ▪ DDS Applicability
  ▪ Communication Paradigm (Client Server vs Publish/Subscribe)
• Topics
• Domains and Partitions
• Consuming Data and Message History
• Anatomy of a DDS Application
• OpenSpliceDDS Versions
Multi-robots Communication Challenges

- Sharing worldmodel
  - Perception
  - Plan
- Send command to another agents
- Task status update between agents
- Roaming network
- Persistence
  - Messages history
- Scalability
  - 0...x robots
  - Loosely coupled
  - Plug and Play

Some Communication Middleware

<table>
<thead>
<tr>
<th>Middleware</th>
<th>Class</th>
<th>Specification</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenSplice</td>
<td>MOM</td>
<td>DDS</td>
<td>opensplice.com</td>
</tr>
<tr>
<td>CoreDX</td>
<td>MOM</td>
<td>DDS</td>
<td>twinoakscomputing.com/coredx.php</td>
</tr>
<tr>
<td>Apache Qpid</td>
<td>MOM</td>
<td>AMQP</td>
<td>qpid.apache.org</td>
</tr>
<tr>
<td>ZeroMQ</td>
<td>MOM</td>
<td>AMQP</td>
<td>zeromq.org</td>
</tr>
<tr>
<td>webMethods</td>
<td>MOM</td>
<td></td>
<td>softwareag.com/corporate/default.asp</td>
</tr>
<tr>
<td>mom4j</td>
<td>MOM</td>
<td>JMS</td>
<td>mom4j.sourceforge.net</td>
</tr>
<tr>
<td>Boost.MPI</td>
<td>POM</td>
<td>MPI</td>
<td>boost.org/doc/libs/1_39_0/doc/html/mpi.html</td>
</tr>
<tr>
<td>TIPC</td>
<td>POM</td>
<td></td>
<td>tipc.sourceforge.net</td>
</tr>
<tr>
<td>D-Bus</td>
<td>OOM</td>
<td></td>
<td>freedesktop.org/wiki/Software/dbus</td>
</tr>
<tr>
<td>LCM</td>
<td>MOM</td>
<td></td>
<td>code.google.com/p/lcm/</td>
</tr>
<tr>
<td>Spread</td>
<td>POM</td>
<td></td>
<td>spread.org</td>
</tr>
<tr>
<td>omniORB</td>
<td>OOM</td>
<td>CORBA</td>
<td>omniorb.sourceforge.net</td>
</tr>
<tr>
<td>JacORB</td>
<td>OOM</td>
<td>CORBA</td>
<td>jacob.org</td>
</tr>
<tr>
<td>TAO</td>
<td>OOM</td>
<td>CORBA</td>
<td>theaceorb.com</td>
</tr>
<tr>
<td>ORBACUS</td>
<td>OOM</td>
<td>CORBA</td>
<td>progress.com/orbacus/index.html</td>
</tr>
<tr>
<td>ICE</td>
<td>OOM</td>
<td></td>
<td>zeroic.com</td>
</tr>
<tr>
<td>XMLRPC++</td>
<td>POM</td>
<td>XMLRPC</td>
<td>xmlrpcpp.sourceforge.net</td>
</tr>
</tbody>
</table>

* More detail in BRICS Deliverables 2.1
DDS Applicability

Data Distribution Service (DDS)

- A High Performance Real-Time Data-Centric Publish/Subscribe Middleware
  - The right data, at the right place, at the right time – all the time
  - Fully distributed, high performance, highly scalable, and high availability

- Perfect Blend of Data-Centric and Real-Time Publish/Subscribe Technologies
  - Content based subscriptions, (continuous) queries, filters, and windows

- Loosely coupled
  - Plug and play architecture with dynamic discovery
  - Time and space decoupling
2.5 ‘Data Distribution Service (DDS)’ – Ronny Hartanto

Communication/Coordination Paradigms

Examples: CORBA, COM+, Java RMI, .Net Remoting

The ‘4Ws’ of Client/Server
- Who/Where: SPACE Coupling
- What: Structural Coupling
- When: Time Coupling

Examples: OpenSplice DDS, TIBCO Rendezvous, JMS

Client Server vs Publish Subscribe

Client/Server
- Complex Deployment
- Tight Coupling
- Fragile to Fault
- Inherently One-to-One

Publish/Subscribe
- Plug & Play
- Loosely Coupled
- Fault Resilient
- Inherently Many-to-Many
Topics

• **Unit of information atomically exchanged** between Publisher and Subscribers

• **An association between a unique name, a type and a QoS setting**

Examples:

```c
struct LeaderType {
    short id;
    string robotname;
};
#pragma keylist LeaderType
```

Keyless topic

```c
struct PoseType {
    short id;
    float x;
    float y;
    float z;
};
#pragma keylist PoseType id
```

Keyed topic

---

**Topic Types and Keys**

Publisher 1

```
1
10.5 3.95 0.45
1.25 3.25 5.45
1 12.5 3.95 0.45
```

Publisher 2

```
2
12.5 3.95 0.45
1.25 3.25 5.45
1 10.5 3.95 0.45
```

Subscriber

```
1
Pioneer 1

2
Quadro
```

DDS

```
1
12.5 3.95 0.45
```

```
2
12.5 3.95 0.45
1.25 3.25 5.45
```

```
1
Pioneer 1

2
Quadro
```

```
2
1.25 3.25 5.45
```
Domains and Partitions

- **Domain**
  - A Domain is one instance of the DDS Global Data Space
  - DDS entities always belong to a specific domain
- **Partition**
  - A partition is a scoping mechanism provided by DDS
  - E.g.
    - sensors.pioneer.pose
    - sensors.quadro.pose
  - Data for all sensors is available via
    - sensors.*

Consuming Data and Message History

- **Read**
  - History Depth = 5

- **Take**
Anatomy of a DDS Application

OpenSpliceDDS Versions

- Community Edition (LGPL3)
- Compact Edition
- Professional Edition
- Enterprise Edition
Resources

- The DDS Tutorial Part I & II (Angelo Corsaro, Ph.D., PrismTech)
- DDS: A Next-Generation Approach to Building Distributed Real-Time Systems (Gerardo Pardo-Castellote, Ph.D., Real-Time Innovations)
- Addressing the Data-Distribution Challenges of Next-Generation Business- and Mission-Critical Systems (Angelo Corsaro, Ph.D., PrismTech)
- BRICS: Best Practice in Robotics Deliverable D2.1 (BRSU)

Thank you!

DFKI Bremen & Universität Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
robotics@dfki.de
2.6 ‘Planning with reconfigurable multi-robot systems’ (FS-P-01)

Thomas M. Roehr(1), Ronny Hartanto(1)

(1) DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: thomas.roehr@dfki.de, ronny.hartanto@dfki.de

Abstract

A number of space missions have proven the effectiveness of applying robots for planetary exploration. Those missions are usually handicapped by long distances and limited resources on the communication network which making such operations less efficient. One possible approach to overcome this problem is by increasing the level of autonomy of the deployed robotic systems. However, this is only cautiously being accepted as a tool for existing space missions and thus applied in a very limited fashion. Some reasons for these are limited experience with this technology in space missions, development costs, and a low to no risk tolerance in space missions. Meanwhile, the “pressure to reduce manpower during routine mission operation” is real, though thus such robotic missions are far from becoming routine missions and demanding further research on novel mission design concepts. Therefore, this poster presents an approach for applying reconfigurable multi-robot systems to allow for more and safer autonomy in upcoming missions.
Planning with reconfigurable multi-robot systems
Thomas M. Roehr and Ronny Hartanto

A multi-robot architecture based on ESA's Functional Reference Model

ESA's Functional Reference Model (FRM) serves as main architectural template for the design of our multi-robot system. The FRM suggest the application of mission (C), task (B) and action (A) layer. To achieve full distribution of the (semi-) autonomous multi-robot system we:
- maintain a task layer and an action layer on each robot
- allow for horizontal extension of task and action layer by using a (FIPA-based) communication supporting distributed systems
- allow robot to robot communication

Organization modeling

The capability of reconfiguration of a multi-robot system introduces new challenges for planning:
- considering permutations, i.e. new robotic actors resulting from combining two or more other actors
- handling effects of reconfiguration, e.g. operative and dormant actors

It also opens up new opportunities:
- increasing efficiency and robustness of the multi-robot system
- perform actor and resource management to optimize safety constraints

Modeling of the set of atomic actors and their respective capabilities allows to inference of resulting capabilities after reconfiguration. Furthermore, it provides the basis for performing analysis of organization properties, e.g., looking at resource redundancy and simulating the effects of the loss of resources.

Planning architecture based on organization modeling

To exploit the capabilities of a reconfigurable multi-robot system we suggest a planning architecture that consists of:
- an organization model that holds domain knowledge and allows inference of capabilities of recombined systems
- a component for planning and optimization which generates plan candidate and which is operates upon domain knowledge from the organization model and dynamic information about the system
- an arbiter component that evaluates the plan candidates based on predefined policies set by an operator
- an execution engine which manages the activities of a single robot and synchronizes with other robots using direct interaction based on predefined protocols

In order to maximize reuse of existing technologies the organization modeling uses Ontology Web Language (OWL) as its representation. Furthermore, planning domain and problem are generated in PDDL to interface with existing PDDL planner such as LAMA.

Supported by:

Universität Bremen

Grant Number 50RA1201

Supported by:

DLR
2.7 ‘FIPA-based multi-robot infrastructure for Rock’ (FS-P-02)

Satia Herfert\textsuperscript{(1)}, Thomas M. Roehr\textsuperscript{(1)}

\textsuperscript{(1)} DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: satia.herfert@dfki.de, thomas.roehr@dfki.de

Abstract

The application of teams of autonomous, cooperating robots in areas which are hazardous and inaccessible for humans is highly desirable. Furthermore, a decentralized approach allows for more fail-prune algorithms, but has its own challenges. To improve the capability of such multi-robot systems, we built an infrastructure that allows to share resources in a distributed system and to easily find and contact agents, even from other ecosystems. This poster provides a motivation for this research and illustrates parts of the architecture.
FIPA-based multi-agent infrastructure for Rock
Distributed locking algorithms and platform-interoperability in TransTerrA

Satia Herfert and Thomas M. Roehr

Multi-agent infrastructure

The application of teams of autonomous, cooperating robots in areas which are hazardous and inaccessible for humans is highly desirable. Furthermore, a decentralized approach allows for more fail-prone algorithms, but has its own challenges. To improve the capability of such multi-robot systems, we built an infrastructure that allows:
• to share resources in a distributed system and
• to easily find and contact agents, even from other ecosystems.

Distributed locking for shared resources

Going for a decentralized solution, robots need to perform their resource sharing with distributed algorithms, i.e. Sherpa and Coyote will have to communicate about camera and battery usage. The challenge is that each software resource corresponds to a physical device belonging to exactly one agent – existing algorithms do not model this.

Good candidates for handling this problem are Ricart-Agrawala and Suzuki-Kasami. The first one uses a broadcast mechanism whereas the second one relies on a software-token granting exclusive access. Both have been implemented and extended. To guarantee that agents can continue their work, even if one agent fails for some reason, they have been equipped with agent-failure-detection and handling. This detection is realized sending heartbeat messages to resource holders in intervals. If no instant response is received, the agent is assumed to have failed/crashed.

As an example Coyote could request exclusive access to a payload item camera, that Sherpa carries. If the communication is cut, which is detected, Coyote will consequently have to modify its behavior.

FIPA and JADE-interoperability

The before mentioned algorithms exchange messages that are compliant to the FIPA standards for inter-operation of heterogeneous agents. FIPA’s reference implementation, JADE, is written in Java. Rock was extended to be able to connect to JADE systems. Being able to cross-platform communicate with JADE enables developers to take advantage of the broad JADE infrastructure. JADE has a built-in GUI that facilitates testing and external developers wishing to use Rock’s functionality are able to remain in JADE’s ecosystem.

The agents are able to easily find each other with a distributed service directory based on zeroconf/avahi network entries. This is platform independent. They continue to communicate via a set of reusable TCP/UDP sockets, over which they send XML encoded data.

The given infrastructure provides a basis for implementing advanced cooperation strategies such as auction-based algorithms that solve task assignment problems. In the long run, this enables us to use reconfigurable multi-robot teams to a maximum extent.

Typical message flow for the FIPA-brokering protocol. The initiator asks the broker to find an agent for him that provides a certain service. The broker finds one, and informs the initiator about the progress.

Support:

Grant Number 50RA1201

Structure of the FIPA services. The message transport service (MTS) forwards messages to other MTS. They look up the addresses of agents in both ecosystems in the distributed service directory.
3 ‘Manipulation & Control’

3.1 ‘Work Group Activities’ (MC-T-01)

Bertold Bongardt\textsuperscript{(1)}, Benjamin Girault\textsuperscript{(1)}, et al.

\textsuperscript{(1)} DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: bertold.bongardt@dfki.de, benjamin.girault@dfki.de

Abstract

In this introduction, a compact outline of the activities of the work group during the last year is given. Further, the posters of the afternoon session are introduced briefly.
### Project Day

**‘Manipulation & Control’**

and

**‘Framework & Standardization’**

June 19, 2014

#### Schedule 2014-06-19

<table>
<thead>
<tr>
<th>Time</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>09:00 - 09:01</td>
<td>Hello!</td>
</tr>
<tr>
<td>09:01</td>
<td>“Overview of the new RIC project day structure” (Sirko Straube)</td>
</tr>
<tr>
<td>09:10 - 09:30</td>
<td>Block I – Talks – AG Framework &amp; Standardization</td>
</tr>
<tr>
<td>09:10</td>
<td>General Introduction (Alexander Duda)</td>
</tr>
<tr>
<td>09:20</td>
<td>Cross platform development (Martin Zenses)</td>
</tr>
<tr>
<td>09:40</td>
<td>The Artemis Rover as an Example for Model Based Engineering in Space Robotics (Stefan Haase)</td>
</tr>
<tr>
<td>10:00 - 10:20</td>
<td>Distributed compilation and Buildserver (Steffen Planthuber)</td>
</tr>
<tr>
<td>10:40 - 11:00</td>
<td>Coffee Break</td>
</tr>
<tr>
<td>11:00 - 12:45</td>
<td>Block II – Talks – AG Manipulation &amp; Control</td>
</tr>
<tr>
<td>11:00</td>
<td>Work Group Activities – Introduction (Bertold Bongardt, *)</td>
</tr>
<tr>
<td>11:15</td>
<td>Control of Flexible Link Manipulator (Ajish Babu)</td>
</tr>
<tr>
<td>11:30</td>
<td>Motion Planning for Manipulator using Moveit (Sankar Natarajan)</td>
</tr>
<tr>
<td>11:45</td>
<td>A Library for Motion Planning (Behnam Asadi)</td>
</tr>
<tr>
<td>12:00</td>
<td>Trajectory generation using the library Reflexxes (Benjamin Girault, Mehn Wirkus)</td>
</tr>
<tr>
<td>12:15</td>
<td>Cascaded Robot Joint Control (Vinzenz Bargsten)</td>
</tr>
<tr>
<td>12:30 - 12:45</td>
<td>Lunch Break (Move to Foyer)</td>
</tr>
<tr>
<td>13:00 - 15:30</td>
<td>Block III – Posters &amp; Discussion – AG Framework &amp; Standardization</td>
</tr>
<tr>
<td>13:00</td>
<td>FIPA-based multi-agent infrastructure for Rock (Satia Hofert, Thomas Rühr)</td>
</tr>
<tr>
<td>13:15</td>
<td>Planning with Reconfigurable Multi-Robot Systems (Thomas Rühr, Romy Hartanto)</td>
</tr>
<tr>
<td>13:30</td>
<td>Distributed Dynamics Computation (Vinzenz Bargsten)</td>
</tr>
<tr>
<td>13:45</td>
<td>Complex Numbers and Quaternions - A unified view on representations and metrics for rotations (Bertold Bongardt)</td>
</tr>
<tr>
<td>14:00 - 14:15</td>
<td>An Application of Constraint-Based Motion Control (ITiSC) (Dennis Munra)</td>
</tr>
<tr>
<td>16:20 - 16:30</td>
<td>Clean-Up!</td>
</tr>
</tbody>
</table>

---

3.1 ‘Work Group Activities’ – Bertold Bongardt, Benjamin Girault, et al.
"Current activities in 'AG Manipulation & Control'"
3.1 ‘Work Group Activities’ – Bertold Bongardt, Benjamin Girault, et al.

‘AG Manipulation and Control’ – People

Figure: Visitors 2013.

‘AG Manipulation and Control’ – People

Figure: Visitors 2014.

Two Sides of the Coin

Manipulation     Kontrolle

Euklidischer Raum Konfigurationsraum

Kinematik  Dynamik  Regelung

Figure: Euclidean and configuration space.
Outline since last project day 2013

- click https://svn.bb.dfki.de/trac/Workgroups/wiki/Manipulation
- (only) nine meetings
- approx. seven people
- workgroups ⊥ projects
- posters!

Poster #01 – Mohammed & Ajish
Autonomous Steering Controller for Path Following

Mohammed Ahmed and Ajish Babu
Project-day AG Manipulation / AG Framework & Standardization, 19th June 2014

DFKI Bremen & Universität Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
robotics@dfki.de

Path Following Controller

• Controller to enable a robot to follow a predefined trajectory
• Uses the Kinematic Model of the robot
• Transforms the kinematic model to chained form
• Controller for unicycle and car-like mobile robots
• Compensates for distance and orientation error

Implementation and results

• Integrated into ROCK as library and orogen module
• Results for differential steering from IMoby project
• Results for car-like steering from Simulation
• Being used in many project at DFKI
Poster #02 – Behnam & Sankaranarayanan

Motion Planning for Manipulator
Behnam Asadi and Sankaranarayanan Natarajan

DFKI Bremen & Universität Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
robotics@dfki.de
• **MoveIt!** – A ROS based motion planning library
  - Software framework for developing mobile manipulation application
  - State of the art motion planners, Kinematics solver
  - Collision Checking, Control and Navigation

• Features MoveIt can offer?
  - Environment representation from sensor data
  - Trajectory execution and monitoring
  - Manipulation task - Pick and place task
  - Constraint representation
  - Easy setup assistant (live demo)

• A Stand alone motion planning library
  - Sampling based planners
  - C++
  - Analytical and numerical kinematic Solver
  - Hierarchical Brute-force search for the robots with more than 6 DOF for minimum movement
  - Self collision and environment collision checker.
  - Replanning for dynamic environment.
Poster #03 – Vinzenz

Introduction to the Poster:

**Distributed Dynamics Computation**

Vinzenz Bargsten

DFKI Bremen & Universität Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
robotics@dfki.de
Distributed Dynamics Computation

Motivation
- Taking the robot dynamics into account in the control system
- Simulation of motion dynamics
  ⇒ Require computation of the dynamic robot model

Distributed Dynamics Computation
- Structure of the recursive Newton-Euler-Algorithm is exploited
  - Each FPGA controlling an actuator computes a recursion step of the algorithm

Poster #04 – Bertold
Complex Numbers and Quaternions

A unified view on representations and metrics for rotations

Bertold Bongardt
Robotics Innovation Center DFKI, Bremen, Germany.

Motivation

"In mathematics, the quaternions are a number system that extends the complex numbers." [Wikipedia]

How? If a complex number defined as

\[ z = x + iy, \]

The traditional answer

a quaternion is

\[ q = w + ix+jy+zk, \]

with

\[ i^2 = j^2 = k^2 = i \cdot j \cdot k = -1. \]

Yet another answer

a quaternion is

\[ q = w + jx + ky + lz, \]

with

\[ i \neq j \neq k \neq l. \]

... a simple concept with useful implications!
3.1 ‘Work Group Activities’ — Bertold Bongardt, Benjamin Girault, et al.

More Information

Complex Numbers and Quaternions
A unified view on representations and metrics for rotations.

Unified View on Complex Numbers and Quaternions with Applications to Measuring and Averaging Rotations
Bertold Bongardt
June 17, 2014

Abstract

1 Introduction

Introduction. The aim of this paper is to show the advantages and the potential benefits of using quaternions in kinematics and robot manipulators' reaching area. A thorough analysis of how to apply quaternions for calculating the rotation of a rigid body with respect to another reference body is presented. The presented methods are easy to implement and use in kinematics and robot manipulators' reaching area.

Application: Relations of Rotation Metrics.

Metrics for rotations and displacements are used in several applications. In robotics, the choice of a specific distance function for a specific rotation representation influences the solutions of these tasks, multiple methods were defined in the past. For specifying a rotation in space, several rules for composition. However, these methods are often preferred to communicate about orientations in robotics. In this paper, we provide a unified view on complex numbers and quaternions with applications to measuring and averaging rotations. The notation with four distinct imaginary units allows to unify the view of these metrics. The squared metrics are developed in dependence on the different rotation representations mentioned above.

Imaging and Vision

Rigid Bodies.

Imaging and Vision Rigid Bodies.

1. Introduction

Introduction. The aim of this paper is to show the advantages and the potential benefits of using quaternions in kinematics and robot manipulators' reaching area. A thorough analysis of how to apply quaternions for calculating the rotation of a rigid body with respect to another reference body is presented. The presented methods are easy to implement and use in kinematics and robot manipulators' reaching area.

Application: Relations of Rotation Metrics.

Metrics for rotations and displacements are used in several applications. In robotics, the choice of a specific distance function for a specific rotation representation influences the solutions of these tasks, multiple methods were defined in the past. For specifying a rotation in space, several rules for composition. However, these methods are often preferred to communicate about orientations in robotics. In this paper, we provide a unified view on complex numbers and quaternions with applications to measuring and averaging rotations. The notation with four distinct imaginary units allows to unify the view of these metrics. The squared metrics are developed in dependence on the different rotation representations mentioned above.

Imaging and Vision

Rigid Bodies.

Imaging and Vision Rigid Bodies.

1. Introduction

Introduction. The aim of this paper is to show the advantages and the potential benefits of using quaternions in kinematics and robot manipulators' reaching area. A thorough analysis of how to apply quaternions for calculating the rotation of a rigid body with respect to another reference body is presented. The presented methods are easy to implement and use in kinematics and robot manipulators' reaching area.

Application: Relations of Rotation Metrics.

Metrics for rotations and displacements are used in several applications. In robotics, the choice of a specific distance function for a specific rotation representation influences the solutions of these tasks, multiple methods were defined in the past. For specifying a rotation in space, several rules for composition. However, these methods are often preferred to communicate about orientations in robotics. In this paper, we provide a unified view on complex numbers and quaternions with applications to measuring and averaging rotations. The notation with four distinct imaginary units allows to unify the view of these metrics. The squared metrics are developed in dependence on the different rotation representations mentioned above.
iTaSC: Application and Rock Integration
Specifi cation and Control of Complex Sensor-Based Tasks

Dennis Mronga
Robotics Innovation Center DFKI, Bremen, Germany.

• What is iTaSC and what are its capabilities?
• How is it integrated in Rock?
iTaSC: Instantaneous Task Specification using Constraints

Schedule
Current activities in ‘AG Manipulation & Control’
### Schedule 2014-06-19

<table>
<thead>
<tr>
<th>Time</th>
<th>Activity</th>
<th>Speaker(s)</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>09:00</td>
<td>Hello!</td>
<td></td>
<td></td>
</tr>
<tr>
<td>09:01</td>
<td>&quot;Overview of the new RIC project day structure&quot; (Sirko Straube)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>09:10</td>
<td>General Introduction (Alexander Duda)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>09:20</td>
<td>Cross platform development (Martin Zenses)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>09:40</td>
<td>The Artemis Rover as an Example for Model Based Engineering in Space Robotics (Stefan Hause)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10:00</td>
<td>Distributed compilation and Buildserver (Steffen Planthaber)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10:20</td>
<td>Data Distribution Service (DDS) (Ronny Hartanto)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10:40</td>
<td>Coffee Break</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11:00</td>
<td>Work Group Activities – Introduction (Bertold Bongardt, *)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11:15</td>
<td>Control of Flexible Link Manipulator (Ajish Babu)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11:30</td>
<td>Motion Planning for Manipulator using Moveit (Sankar Natarajan)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11:45</td>
<td>A Library for Motion Planning (Behnam Asadi)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12:00</td>
<td>Trajectory generation using the library Reflexxes (Benjamin Graiel, Malte Wirkus)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12:15</td>
<td>Cascaded Robot Joint Control (Vinzenz Bargsten)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12:30</td>
<td>Lunch Break (Move to Foyer)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14:00</td>
<td>Work Group Activities – Introduction (Bertold Bongardt, *)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14:15</td>
<td>Control of Flexible Link Manipulator (Ajish Babu)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14:30</td>
<td>Motion Planning for Manipulator using Moveit (Sankar Natarajan)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14:45</td>
<td>A Library for Motion Planning (Behnam Asadi)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15:00</td>
<td>Trajectory generation using the library Reflexxes (Benjamin Graiel, Malte Wirkus)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15:15</td>
<td>Cascaded Robot Joint Control (Vinzenz Bargsten)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16:00</td>
<td>Clean-Up!</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Block I – Talks – AG Framework & Standardization**

- General Introduction (Alexander Duda)
- Cross platform development (Martin Zenses)
- The Artemis Rover as an Example for Model Based Engineering in Space Robotics (Stefan Hause)
- Distributed compilation and Buildserver (Steffen Planthaber)
- Data Distribution Service (DDS) (Ronny Hartanto)

**Block II – Talks – AG Manipulation & Control**

- Control of Flexible Link Manipulator (Ajish Babu)
- Motion Planning for Manipulator using Moveit (Sankar Natarajan)
- A Library for Motion Planning (Behnam Asadi)
- Trajectory generation using the library Reflexxes (Benjamin Graiel, Malte Wirkus)
- Cascaded Robot Joint Control (Vinzenz Bargsten)

**Block III – Posters & Discussion – AG Framework & Standardization**

- FIPA-based multi-agent infrastructure for Rock (Satia Herfert, Thomas Röhr)
- Planning with Reconfigurable Multi-Robot Systems (Thomas Röhr, Ronny Hartanto)

**Block IV – Posters & Discussion – AG Manipulation & Control**

- Autonomous Steering Controller for Path Following (Mohammed Ahmed, Ajish Babu)
- Motion Planning for Manipulators (Behnam Asadi, Sankar Natarajan)
- Distributed Dynamics Computation (Vinzenz Bargsten)
- Complex Numbers and Quaternions - A unified view on representations and metrics for rotations (Bertold Bongardt)
- An Application of Constraint-Based Motion Control (iTaSC) (Dennis Mronga)

**16:20 - 16:30 Clean-Up!**

---

**Project Day ‘Manipulation & Control’ and ‘Framework & Standardization’ 2014**

---

66
Abstract

Light-weight robot design have a certain degree of flexibility, which in turn introduces unwanted vibrations. This talk introduces different methods of vibration attenuation, combining feed-forward and feedback techniques. Feed-forward compensation based on Input Shaping and feedback control based on Strain gauges and Inertial Measurement Unit are used. The effectiveness of these controllers are studied using a two-link manipulator test setup.
Control of Flexible Link Manipulator
by Ajish Babu

Project-day AG Manipulation / AG Framework & Standardization
19th June 2014

DFKI Bremen & Universität Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
robotics@dfki.de

Motivation

- AILA has some degree of flexibility.
- This induces unwanted vibration due to motion or environmental interaction.
- This problem is general for most of the light weight robots

Objective

- Generate vibration free motion and cancel the existing vibration.
- Stop any vibration on AILA body propagating to AILA arms.
- Try and cancel the vibration on the body by movement of the arms.
Test Bench

- Starting point for testing some controllers
- 2 Links planar manipulator
- Using AILA motors
- Flexible Aluminium link
- Weight at the end
- Strain gauge with DFKI μDMS board
- Flexspline strain gauge torque measurement

Strain Measurement

- Strain gauges
- Full Wheatstone bridge
- Sensitive to strain
- Temperature compensation
- Resistance compensation
Vibration Controller

- Feed-Forward: Input shaper
- Feedback: Strain
- Feedback: Proportional Controller
- Position control at Joint level

Input Shaper

- Feed-Forward technique
- Cancels its own vibration
- Convolution in Frequency domain
- No stability concerns
- Sensitive to parameters
Parameter Identification

- Parameters: Natural Frequency and Damping Ratio
- Impulse response
- Second order system
- Omega = 4.5, Zeta = 0.01

External Disturbance
Sinusoidal Input

![Graph showing reference position vs. time with different input scenarios: no attenuation, input shaper only, and input shaper + strain feedback.](image)

Sinusoidal Input Response

![Graph showing strain vs. time with different input scenarios: no attenuation, input shaper only, and input shaper + strain feedback.](image)
3.2 ‘Control of Flexible Link Manipulator’ – Ajish Babu

**Sinusoidal Input Response**

![Image of experimental setup](image)

**IMU Instead of Strain Gauge**

- XSENS IMU
- Rudimentary motion estimation

![Graph of acceleration over time](image)
Conclusion

- Simple Input Shaper based controller reduces vibration.
- Input Shaper with proportional strain feedback controller gives good results.
- IMU based controller is not as effective.

Challenges

- Source of vibration is not always accessible for measurement.
- Motion Estimation from IMU
- Extending this to multiple links

Future Work

- Try Input Shaper with AILA base motion
- Try with a better IMU (STIM300 from Sensonor)
- Vibration Observer based on IMU measurement
- Vibration free trajectories
- Vibration compensation trajectories
Thank you!

DFKI Bremen & Universität Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfk.de/robotics
robotics@dfki.de
3.3 ‘Motion Planning for Manipulator – MoveIt!’ (MC-T-03)

Sankaranarayanan Natarajan

(1) DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: sankaranarayanan.natarajan@dfki.de

Abstract

This talk gives a brief introduction on a motion planner framework MoveIt! . The MoveIt! framework provides the state of the art motion planners, kinematic solver, collision checker, control and navigation software modules. The key features of the MoveIt! like environment representation and set up assistant are discussed.
Motion Planning for Manipulator

- MoveIt! -

DFKI Bremen & Universität Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
robotics@dfki.de

Content

• Introduction

• MoveIt! – A Motionplanner Library

• Robots using MoveIt! in DFKI

• (LiveDemo)
Introduction

- **MotionPlanning**
  - Ability of a robot to plan its own motion considering the given constraints
    - Constraints: obstacles in the environment, kinematic constraints.
  - Motion planner is a collection of software modules
    - Object modeling
    - Collision detection
    - Search algorithm

Developing a motion planner is not an easy task

- **Model-based motion planning**
  - Known environment

- **Known Libraries**
  - MTK – motion planning kernel (Stanford)
  - OOPSMP (Kavrakilab)
  - MSL – Motion Strategy Library (University of Illinois)
  - OMPL (Kavrakilab)
  - openRAVE

- Many of the available planner are not easy to implement and are not maintained any more
• Model-based motion planning
  ▪ Known environment

• Known Libraries
  ▪ MTK – motion planning kernel (Stanford)
  ▪ OOPSMP (Kavakilab)
  ▪ MSL – Motion Strategy Library (University of Illinois)
    ▪ OMPL (Kavakilab)
    ▪ openRAVE

• Many of the available planner are not easy to implement and are not maintained any more

**MoveIt! – A Motionplanner Library**

• What is *MoveIt!*?
  ▪ Motionplanner library developed in Willowgarage
  ▪ Software framework containing
    ▪ State of the art motion planners
    ▪ Kinematics solver
    ▪ Collision Checking
    ▪ Control and Navigation

• Why we need *MoveIt!*?
  ▪ Developing new higher-level capabilities from scratch is time consuming
  ▪ To build mobile manipulation applications
• Features *MoveIt!* can offer?
  ▪ State of the art Sampling-based planner (OMPL)
    ▸ RRT, RRTConnet, LazyPPRT, PRM, KPIECE, LBKPIECE, BKPIECE, etc.
    ▸ Benchmarking custom planner
  ▪ Search-based planner (SBPL)
  ▪ Kinematic analysis

• Features *MoveIt!* can offer?
  ▪ Environment representation from sensor data
  ▪ Trajectory execution and monitoring
  ▪ Manipulation task
    ▸ Pick and place task
  ▪ Constraint representation
    ▸ Joint constraints
    ▸ Position constraints
    ▸ Orientation constraints
    ▸ Visibility constraints
  ▪ Easy setup assistant (live demo)
3.3 ‘Motion Planning for Manipulator – MoveIt!’ – Sankaranarayanan Natamjan

- **MoveIt! Architecture**

- **MoveIt! used on over 65 robots**

![BDI Atlas](image1)
![PR2](image2)
![Care-O-Bot](image3)
![PAL Robotics](image4)
![TUM Rosie](image5)
![Baxter Research Robot](image6)
![HRP4](image7)
![Caspera Robot](image8)
![MEKA – M3](image9)
![Hubo](image10)
![Robonaut Robonaut2](image11)
![HRP 2](image12)
![REEEM-C](image13)
Robots using *Movelt!* in DFKI

- **Amparo**
  - Arm-navigation
    - predecessor of Movelt.
  - Only Self-collision.
  - Arm placement bad in Amparo.

- **Artemis**
  - *Movelt!*
  - Self-collision and collision with environment.
  - ROCK – ROS bridging.

- **Environment representation**
  - Pointcloud (octomap)
  - env_video
  - Kinematic Constrained planning
    - constraint_video
LiveDemo

- Easy to setup

Thank you!

DFKI Bremen & Universität Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfk.de/robotics
robotics@dfki.de
3.4 ‘Motion Planning Library for Robotic Application’ (MC-T-04)

Behnam Asadi(1)

(1) Universität Bremen, Arbeitsgruppe Robotik, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: behnam@informatik.uni-bremen.de

Abstract

A stand alone library for motion planning task, based on sample based motion planners, implemented in C++. Functionalities of the library:

- Computing analytical and numerical solutions for forward and inverse kinematics queries.
- Hierarchical brute-force search for the robots with more than 6 DOF for minimum movement.
- Self collision and environment collision checker.
- Replanning for dynamic environment.
- Dual Arm manipulation.
Overview of Different Approaches for Motion Planning Task

- Combinatorial planning.
- Search Based.
- Optimization Based.
- Sample Based.
Sample Based Motion Planning Pipe Line Algorithm

Artemis executing trajectory From Motion Planner Library
3.4 ‘Motion Planning Library for Robotic Application’ – Behnam Asadi

Schunk Arm executing trajectory From Motion Planner Library

AILA executing trajectory From Motion Planner Library
Replanning in Dynamic Environments

- In many applications the planning scene changes during the execution of trajectory.
- The model of environment and the position of obstacles might be different from what we observed during planning.

Our approach:

- After execution of each step of trajectory we check the future states for collision.
- If there is a collision, we find the broken states and by calling the planner we try to find a valid path to connect the disconcerted part of the path.
3.5 ‘Online trajectory generation using the Reflexxes library’ (MC-T-05)

Benjamin Girault(1), Malte Wirkus(1)

(1) DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: benjamin.girault@dfki.de, malte.wirkus@dfki.de

Abstract

This talk presents the case of online trajectory generation and how the library Reflexxes (www.reflexxes.com) solved this problem. The first part explains why we need trajectory generation algorithms that can deal with arbitrary initial states, can be computed online and allows the joints of the robot to be time synchronized and reach a target position and speed. The second parts describes the library Reflexxes, its features and how it works. Finally, examples of implementation at the DFKI are presented.
Online Trajectory generation using the Reflexxes Library

Project Day
AG Manipulation & Control
19th June 2014
Benjamin Girault, Malte Wirkus, DFKI RIC

Content

1. Online Trajectory Generation
   - What is it?
   - What do we expect from it?
2. Reflexxes Library
   - Where does it come from?
   - What does it do?
   - How does it work?
3. Using the Library
1. Online Trajectory Generation

Desirable properties
- Limited speed / acceleration / jerk
- Time synchronization of several DOFs
- Phase synchronization of the DOFs
- Can deal with any new target state (also non-static) at any time (state of motion)
- Online: Can be computed within control loop

Application
The robot needs to react to unforeseen events:
- Switching between controllers
- Trajectory replanning (e.g. obstacle avoidance) without stopping

Source (Kröger 2010)
2. Reflexxes Library

Reflexxes = name of the spin-off created by Torsten Kröger (www.reflexxes.com) from the TU Braunschweig

Before Reflexxes Library, no solution for (says the Author):
- More than 1 DOF that need to be synchronized
- Limited acceleration and jerk
- Initial state with velocity and acceleration not equal to zero
- Target state with velocity not equal to zero
- Time optimal trajectory
- Online trajectory computation

Based on his research papers such as:

Algorithm Outline

1. Compute synchronization time \( t_{\text{sync}} \)
2. Selection of acceleration / speed profiles
3. Generate new motion state for each DOF
2. Reflexxes Library

**1. Compute synchronization time $t_{sync}$**

**1.1. Minimum execution time for each DOF**

- Target position = 300mm
- Initial position = 50mm
- Initial speed = 80mm/s
- Target speed = 70mm/s
- Initial acceleration = 20mm/s²

$T_i$ $T_{min}$

Minimum time to reach the target with the desired speed

---

Use of
- Acceleration Profiles
- Decision Trees

---

3.5 ‘Trajectory generation using the Reflexxes library’ – Benjamin Girault, Malte Wirkus
2. Reflexxes Library

**Acceleration Profiles:**
- Trapezoid profiles always reach max. acceleration
- Profiles without zero phase do not reach the max. velocity (time optimality)

**Decision Trees for:**
- Determination which Acceleration Profile to apply (e.g. for Time optimal 1D trajectory)
2. Reflexxes Library

Decision Trees for:
- Determination which Acceleration Profile to apply (e.g. for Time optimal 1D trajectory)

Is velocity after we stopped accelerating slower than v_{trgt}?

Acceleration positive?

Trap Profile required to reach v_{trgt}?
2. Reflexxes Library

Decision Trees for:
- Determination which Acceleration Profile to apply (e.g. for Time optimal 1D trajectory)

1. Compute synchronization time $t_{\text{sync}}$
2. Find inoperative time intervals for each DOF

Target position = 300mm
Initial position = 50mm
Initial speed = 80mm/s
Target speed = 70mm/s
Initial acceleration = 20mm/s²

In this time interval, the target can be reached for any time $T$. 

2. Reflexxes Library

1. Compute synchronization time $t_{sync}$

2. Find inoperative time intervals for each DOF

Target position = 300mm

Initial position = 50mm

Initial speed = 80mm/s

Target speed = 70mm/s

Initial acceleration = 20mm/s²

In the interval, the target cannot be reached with the desired speed.

The target can be reached for any time $T > T_{end}$. 
2. Reflexxes Library

1. Compute synchronization time $t_{\text{sync}}$

1.3. Determine synchronization time $t_{\text{sync}}$

- DOF1
- DOF2
- DOF3
- DOF4

$t_{\text{min}}$ $t_{\text{begin}}$ $t_{\text{end}}$ $t_{\text{sync}}$

2. Selection of acceleration / speed profiles

- Similar to 1.1. But $t_{\text{sync}}$ is given this time
- Synchronization

Subset of acceleration profiles

Part of the decision tree to find the acceleration profile

Source (Kröger 2010)
2. Reflexxes Library

Algorithm Outline

1. Compute synchronization time \( t_{\text{sync}} \)
   1.1. Minimum execution time for each DOF
   1.2. Find inoperative time intervals for each DOF
   1.3. Determine synchronization time \( t_{\text{sync}} \)

2. Selection of acceleration / speed profiles

3. Generate new motion state for each DOF
   • Trivial: apply acceleration profile to current motion state

3. Using the Library

Available Libraries:
• Reflexxes Motion Libraries Type II
• Reflexxes Motion Libraries Type IV
→ Handle different sets of constraints

Source (Kröger 2010)
3. Using the Library

**ROCK:**
- Type II Library → on gitorious, „rock-control/reflexxes“
- Type IV Library → on spacegit, „besman-rock/reflexxes_type_iv“
- oroGen component → on gitorious, „rock-control/orogen/trajectory_generation“

**Matlab / Simulink:**
Library Type IV embedded in a S-Function (project EO2)

**Projects that use Reflexxes:**
- Besman (control of Aila, collision avoidance)
- Cmanipulator demo (control of the Orion arm)
- Artemis arm
- Sherpa legs
References


Thanks you for your attention!
Jerk limitation for 1 DOF

Case of a car that goes only forward:
- max speed = 10 m/s
- max acceleration = 5 m/s² (~0.5g)
- max jerk = 10 m/s³

Initial state:
- Position = 0 m
- Velocity = 0 m/s
- Acceleration = 0 m/s²

Target state:
- Position = 50 m
- Velocity = 0 m/s

→ Very simple, could be computed manually

2. Reflexxes Algorithm

There are two C++ libraries:
- Reflexxes Motion Libraries Type II
- Reflexxes Motion Libraries Type IV

→ Corresponds to different constraints on the Online Trajectory Generator (OTG)
3.6 ‘Cascaded Robot Joint Control’ (MC-T-06)

Vinzenz Bargsten\(^{(1)}\)

\(^{(1)}\) Universität Bremen, Arbeitsgruppe Robotik, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: bargsten@uni-bremen.de

Abstract

High-Level robot control algorithms rely on the low-level control of robotic joints. This talk presents a FPGA-based, cascaded control approach for robotic joints. The approach allows to deal with different control objectives and joint limits.
Cascaded Robot Joint Control
Vinzenz Bargsten

DFKI Bremen & Universität Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
robotics@dfki.de

Motivation
FPGA controlled robot joints with BLDC Motor
**Motivation**

Robot Joint Control with:
- Various control objectives
  - not all are safe on their own
  - hard limits are not an answer
- Different joint limits
Motivation

Robot Joint Control with:

- Various control objectives
  - not all are safe on their own
  - hard limits are not an answer
- Different joint limits
  - position, velocity, current
- Need for consistent and independent controller settings

FPGA controlled robot joints with BLDC Motor

Cascaded Robot Joint Control
June 19, 2014
Motivation

Robot Joint Control with:
- Various control objectives
  ⇒ not all are safe on their own
  ⇒ hard limits are not an answer
- Different joint limits
  ⇒ position, velocity, current
- Need for consistent and independent controller settings
  ⇒ from experimental tuning
  ⇒ from theoretical design

⇒ FPGA-based cascaded control component

Implementation

General Properties of Cascaded Controllers:
- One could say inner loops linearize
Implementation

General Properties of Cascaded Controllers:
- One could say inner loops linearize
- Disturbances can be cancelled out on the level they occur
- Inner loops have to have a higher control frequency
- Keeps all control variables within their limits
**Implementation**

**General Properties of Cascaded Controllers:**

- One could say inner loops **linearize**
- Disturbances can be cancelled out on the level they occur
  ⇒ better disturbance rejection
- Inner loops have to have a higher control frequency
- Keeps all control variables within their limits
Implementation

General Properties of Cascaded Controllers:
- One could say inner loops linearize
- Disturbances can be cancelled out on the level they occur
  ⇒ better disturbance rejection
- Inner loops have to have a higher control frequency
- Keeps all control variables within their limits

Motivation Implementation Experiments Summary

Implementation

General Properties of Cascaded Controllers:
- One could say inner loops linearize
- Disturbances can be cancelled out on the level they occur
  ⇒ better disturbance rejection
- Inner loops have to have a higher control frequency
- Keeps all control variables within their limits
Motivation Implementation Experiments Summary

**Implementation**

**General Properties of Cascaded Controllers:**
- One could say inner loops *linearize*
- Disturbances can be cancelled out on the level they occur
  - better disturbance rejection
- Inner loops have to have a higher control frequency
- Keeps all control variables within their limits

![Cascaded Robot Joint Control Diagram](image)

**Implementation Properties:**
- VHDL component implements array of PID-Controllers
- Interconnected as cascade
Implementation Properties:
- VHDL component implements array of PID-Controllers
- Interconnected as cascade
- Different control modes
  - Control loops can be partly deactivated
  - Still take action on limit violation
  - Can be switched during run-time
### Implementation

**Implementation Properties:**
- VHDL component implements array of PID-Controllers
- Interconnected as cascade
- Different control modes
  - Control loops can be partly deactivated
  - Still take action on limit violation
  - Can be switched during run-time

![Cascaded Robot Joint Control Diagram](image)

### Position Step Input
- Current and velocity limitation
- Full cascade is active
Position Step Input

- Current and velocity limitation
- Full cascade is active

First: Current controller limits at high acceleration
Then: Velocity controller limits velocity

2: fast, 3: low current

Cascaded Robot Joint Control
June 19, 2014
Position Step Input

- Current and velocity limitation
- Full cascade is active
- First: Current controller limits at high acceleration
- Then: Velocity controller limits velocity

Friction Compensation with Current Control

- Friction model computes compensation current
- Compensation current is fed into current controller
- Position and velocity controllers control against limit violation
Summary

Current Implementation

- covers many use scenarios for low level joint control
- allows us to test and use a broader range of high level control schemes
  - Control schemes with compliance, force control
  - Still compatible with basic position or velocity control
- should be used with feed-forward values for tracking
<table>
<thead>
<tr>
<th>Motivation</th>
<th>Implementation</th>
<th>Experiments</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Summary</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Current Implementation**

- covers many use scenarios for low level joint control
- allows us to test and use a broader range of high level control schemes
  - Control schemes with compliance, force control
  - Still compatible with basic position or velocity control
- should be used with feed-forward values for tracking
- supports arbitrary interconnection as well
- can be used for different purposes (I know from hand control)

**Future Aspects:**

- Auto Tuning
3.7 ‘Autonomous Steering Controller for Path Following’ (MC-P-01)

Mohammed Ahmed(1), Ajish Babu(1)

(1) DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: mohammed.ahmed@dfki.de, ajish.babu@dfki.de

Abstract

This poster presents an autonomous path following controller for mobile robots. Controller designs using the kinematic model of the robot is discussed for both differential and car-like steering. The kinematic model of the robot is transformed to chained form, from which the controller is developed. It is intergrated in ROCK frame work and are being used by many projects at DFKI. Results of the controller from the IMoby and EO2 projects are also presented.
Autonomous Steering Controller for Path Following
Mohammed Ahmed and Ajish Babu

Introduction
• enables a robot to track a reference path via control of its actuators.
• assumes workspace free of obstacles.
• controller is embedded in hierarchical architecture
• higher-level planner solves obstacle avoidance problem
• provides desired path (series of motion goals) to lower control layer
• measurements of variables involved in control loop are available (typically):
  • position
  • orientation of robot w.r.t fixed frame or a path the robot should follow

Differential Steering Robots

Unicycle-like model
Equations of motion
\[
x = u_i \cos \theta \\
y = u_i \sin \theta \\
\dot{\theta} = \frac{d}{L} - \omega \sin \theta
\]
point at mid-distance of actuated wheels \(P_i\), chassis orientation angle \(\theta\), distance between rear and front wheel axes \(d\), driving and steering velocity inputs \(u_1\) and \(u_2\).

Robots with Steerable Wheels

Car-like model
Equations of motion
\[
x = u_i \cos \theta \\
y = u_i \sin \theta \\
\phi = \frac{u_2}{u_1} \\
\dot{\theta} = \frac{u_2}{L} - \omega \sin \theta
\]
steering wheel angle \(\phi\), curve parameter \(s\), curvature \(\kappa(s) = \frac{d}{L}\).

Kinematic Model into Chained Form
• canonical form for nonholonomic robot kinematic models
• utilized for systematic development of open/closed-loop control
• through change of coordinates and control variables
Model: \((x, \dot{x}, \theta, \dot{\theta}, u_1, u_2) \mapsto \text{chained form: } (x_1, x_2, \ldots, x_n, f_1, f_2)\)

Transformation from Kinematic to Chained Form.

Proportional Input-Scaling Controller
• control law for the vehicle to follow a path in stable manner.
• asymptotically stabilize \((d = 0, \theta = 0)\)
\[
v = -\frac{1}{n} \sum \frac{f_i}{\| \mathbf{f} \|_2} \left( \frac{d}{n} \right)
\]
Controller for differential (left) and car-like (right) steering.

Implementation
• implemented as a stand-alone C++ library
• encapsulated in a MATLAB/Simulink S-Function block
• integrated into ROCK (library and oxygen component)

Results for Differential Steering from iMoby Project

Asphalt following DFN outdoor track in iMoby project.

Results for Steerable Wheels from Simulation

MATLAB/Simulink/Adams/View Cosimulation. ISO3888-2 (Double Lane Change Test) standard test track is used as desired path.
3.8 ‘Motion Planning for Manipulators’ (MC-P-02)

Behnam Asadi(1), Sankaranarayanan Natarajan(2)

(1) Universität Bremen, Arbeitsgruppe Robotik, Robert-Hooke-Straße 1, 28359 Bremen, Germany
(2) DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: behnam@informatik.uni-bremen.de, sankaranarayanan.natarajan@dfki.de

Abstract

On this poster, two libraries for motion planning for manipulator is introduced (MoveIt!, MotionPlanner Library). MoveIt! is state of the art motion planner framework for mobile manipulation applications. A framework independent motionplanning library is developed using sample-based motionplanning algorithm. This library also supports dual arm manipulation and replanning for dynamic environment.
Motion Planning for Manipulators

Behnam Asadi, Sankaranarayanan Natarajan

Sample Based Motion Planning

Motion planning in robotics realm refers to the motions of a robot in a 2D or 3D world occupied with obstacles. Sample based planner are common and widely applied in many robotics tasks and they have the following advantages:

- Probabilistically completeness.
- Capability of Dealing with high-dimensional C-space.
- No need for creating C-space.

Schunk arm performing a collision free trajectory.

MoveIt! - A ROS based Motion Planner

MoveIt! is state of the art sampling based motion planner using OMPL [SMK12]. It provides an easy-to-use platform for developing manipulation applications:

- Easy to setup with new robots - using URDF
- Integrated Collision Checker (FCL) [PCM12] and Kinematics Solver (KDL, InFast)
- Environment awareness through 3D perception (PointCloud)
- Kinematics constrained motion planning

MoveIt Architecture

MoveIt in Rock: The Rock framework can interoperate with ROS at dataflow level. This feature helps to use MoveIt in Rock.

Motion Planner Library

Due to the dependency of the MoveIt! motion planner with ROS framework, a motionplanner library which is independent of any framework has been developed.

- The library supports sampling based planner such as RRT, SBL, EST, LBKPIECE, BKPIECE, KPIECE, RRT, RRTConnect, RRTstar, PRM and PRMstar.
- Planning in a dynamic environment.
- The library also supports dual arm manipulation.
- Planning under Kinematics constraints.

Incorporating Foot, Knees, Hip and Waist into the motion planning.

Next steps

- Integrate the library to Rock framework.
- Including Optimization based planner to the library.
- Representing the environment from the sensor data like PointCloud.

References:


3.9 ‘Distributed Dynamics Computation’ (MC-P-03)

Vinzenz Bargsten\textsuperscript{(1)}

\textsuperscript{(1)} Universität Bremen, Arbeitsgruppe Robotik, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: bargsten@uni-bremen.de

Abstract

This poster presents a method to compute robot motion dynamics distributedly at joint level. In this method, FPGAs controlling the joint actuators are utilized to compute the recursion steps of the Newton-Euler-Algorithm.
3.9 ‘Distributed Dynamics Computation’ – Vinzenz Bargsten

Distributed Dynamics Computation
FPGA-based method to compute robot dynamics at joint level

Vinzenz Bargsten

Introduction
- Robot Motion Dynamics
  - relates actuator torques/forces with the resulting motion
  - often highly coupled and non-linear system
- Linear controllers require high gains to compensate for unmodelled dynamics

Motivation
- Taking the robot dynamics into account in the control system
  - simplifies control problem
  - allows more compliant control schemes instead of stiff position control
- Simulation of motion dynamics
  - Requires computation of the dynamic robot model,
  ⇒ linear controllers require high gains to compensate for unmodelled dynamics

Novel Approach – Distributed Dynamics Computation
- Structure of the recursive Newton-Euler Algorithm is exploited
  ⇒ distributed computation of the dynamic robot model
- Each FPGA controlling an actuator computes a recursion step of the algorithm
  - implemented as component programmed in VHDL
- Linear dependence on the dynamic parameters is preserved

Application Example: Arm Computed-Torque Control
In Computed-Torque-Control, the use of a dynamic model:
- Decouples the motion dynamics with the inertia matrix \( B(q) \)
- Compensates for non-linear effects from gravity \( g(q) \), friction \( \tau(q) \), centrifugal and Coriolis forces \( C(q, \dot{q}) \)
- Trajectory tracking is possible with much lower feed-back gains
- Basis for compliant motions
- Contact forces at the end-effector can be estimated through the Jacobian

Future Aspects
- Online adaption of dynamic parameters
- Experiments with different model-based control schemes
- Hybrid control of contact forces
- Online model adaption to changes in the link structure

Distributed Computation
- Each robot joint communicates with its neighbours
- Forward Recursion starts with the first link, incorporating base velocity and acceleration including gravity
- According to joint motion and link geometry, the current link’s velocity and acceleration is propagated to the next one
- The last node initiates the Backward Recursion: the vector of forces and moments is propagated back to the previous node. Projection on the rotation axis yields the torque \( \tau \) at the joint

Structure of Computed-Torque-Control

References
3.10 ‘Complex Numbers and Quaternions – A unified view on representations and metrics for rotations’ (MC-P-04)

Bertold Bongardt(1)

(1) DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: bertold.bongardt@dfki.de

Abstract

On this poster, a novel view on complex numbers and quaternions is motivated by introducing a five-dimensional complex space which is defined as the union of the complex plane \( \mathbb{C} \) with the quaternion space \( \mathbb{H} \). For rotations with a fixed rotation axis, the complex 5-space can be visualized by \( \mathbb{R}^3 \) and by \( \mathbb{R}^2 \). In these visualizations, the representations of a rotation via a complex number, quaternions, and a rotation matrix appear in an elementary-geometric setup generalizing the unit circle. The definition of the complex 5-space is based on an explicit distinction of four different imaginary units. The poster illustrates one usage of these novel concepts with a comparison of distance measures for rotational displacements.
Complex Numbers and Quaternions
A unified view on representations and metrics for rotations

Introduction.
In mechanics, complex numbers and quaternions are used to encode rotations: Complex numbers for planar and quaternions for spatial rotations. How are both related – algebraically and geometrically?

A Precise Algebraic Notation.
A complex number is defined as \( z = x + iy \) with \( i^2 = -1 \).
A quaternion, with \( q = (q_0, q_1, q_2, q_3) \) and \( j = (0, 1, 0, 0) \), is:
\[
q = q_0 + q_1i + q_2j + q_3k
\]
For the four distinct units \( i, j, k, l \) with
\[
i \cdot j = k, j \cdot l = i, k \cdot i = -j, l \cdot j = -k
\]
the squares equal to \( i^2 = j^2 = k^2 = l^2 = -1 \).

Planar Rotations with Complex Numbers and Quaternions (2 \( \times \) 2)-Matrices.
For a given angle, \( \phi \in [-\pi, \pi] \), the corresponding unit complex number \( z \) is determined via the exponential map (Euler’s formula) as
\[
z = \cos \phi + i \sin \phi
\]
the planar rotation matrix \( R = R(z) \) is determined as
\[
R = \begin{bmatrix}
\cos \phi & -\sin \phi \\
\sin \phi & \cos \phi
\end{bmatrix}
\]
where \( x = x_2 \) and \( y = y_2 \).

Spatial Rotations with Quaternions and (3 \( \times \) 3)-Matrices.
For a rotation, given by an angle \( \phi \in [-\pi, \pi] \) and an axis \( \omega \in S^2 \), a corresponding unit quaternion \( q \) is determined via the equation [1]
\[
q = \exp(\frac{\phi}{2} \omega) = q_0 + q_1i + q_2j + q_3k
\]
the (3 \( \times \) 3) rotation matrix \( R = R(q) \) is determined as
\[
R = \exp(\frac{\phi}{2} q^*) \exp(\frac{\phi}{2} q)
\]
where \( \times \) denotes the vector product.

Complex 5-Space and Complex 3-Space.
Define a complex 5-space (see Equation 1):
\[
\mathbb{C} \times \mathbb{C} \cong \mathbb{C}^5 \cong \mathbb{C}^3 \oplus \mathbb{C}^3
\]
Define a complex 3-space (‘unit’) \( \mathbb{C}^3 \cong \mathbb{C}^3 \) and plane \( \mathbb{C}^3 \subset \mathbb{C}^5 \):
\[
\mathbb{C} \times \mathbb{C} \cong \mathbb{C}^5 \cong \mathbb{C}^3 \oplus \mathbb{C}^3
\]

Extended Unit Circle Geometry.
To planarize the view of Figure 1, ‘fold’ the two vertical coordinate planes in Figure 1 so that they coincide, see Figure 2.

Fig. 1: Combined planar geometry of unit complex number and unit quaternion.

Fig. 2: Appr. of an angle in units ii.

Fig. 3: Graphs of distance function in 3D.

Fig. 4: Graphs of distance function in 3D.

Application: Relations of Rotation Metrics.
Metrics for rotations and displacements are used in several applications in mechanics. Recent works are, for example, [2, 3, 4].

Overview of Rotation Metrics.
With previous visualizations (Figures 3 – 6), five types of rotation metrics are observed ‘geometrically’, see Table 1.

Table 1: An overview of the five considered classes of distance functions.

<table>
<thead>
<tr>
<th>Metric Type</th>
<th>Geodesic</th>
<th>Chordal</th>
<th>Versine</th>
<th>Sine</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td></td>
<td>)</td>
<td>( \log</td>
<td></td>
</tr>
</tbody>
</table>

Conclusion.
The rotation with four distinct imaginary units allows to unify the view of complex numbers and quaternions. Using this, rotation metrics are studied in analogy observations and two novel concepts are revealed.

References.
3.11 ‘iTaSC: Application and Rock Integration – Specification and Control of Complex Sensor-Based Tasks’ (MC-P-05)

Dennis Mronga(1)

(1) DFKI GmbH, Robotics Innovation Center, Robert-Hooke-Straße 1, 28359 Bremen, Germany

Contact: dennis.mronga@dfki.de

Abstract

On this poster an introduction to the iTaSC Software (Instantaneous Task Specification using Constraints) is given. iTaSC helps the user to specify complex sensor-based robot tasks by separating them into more easily defined sub-problems, which are the constraints to an hierarchical optimization problem.

The poster illustrates use-cases, mathematical background and experimental results, as well as the integration of iTaSC in the Rock framework.
iTaSC: Application and Rock Integration

Specification and Control of Complex Sensor-Based Tasks

**What is iTaSC?**
- iTaSC - Instantaneous Task Specification using Constraints [1, 2]
- Can be used to describe reactive robot control tasks by specifying constraints between robot frames and the environment
- Automatic computation of the control solution
- Prioritization and weighting between constraints

![Example: Moving an object with two arms along a pre-defined trajectory](image)

### Establishing Constraint Hierarchies

**Given:** A proportional controller $y_i - k_i (x_i - x)$ with gain $k_i$ that controls the pose between two robot frames. A solution in joint space$q_i$ that regulates the constraint velocity $y_i$ to zero is:

$$q_i - J_{i}^T y_i = (I - J_{i} J_{i}) \xi$$  

(1)

**Iterative extension to $N$ constraints**

$$q_{i+1} = q_{i+1} - J_{i}^T P_{i} (y_{i+1} - J_{i} q_{i+1})$$  

(3)

$$q_{i+1} = 0, i = 0$$  

(4)

$$P_{i+1} = P_{i} + (J_{i} P_{i} J_{i})^{-1} (J_{i} P_{i} J_{i}) P_{i} - I$$  

(5)

$I$ - Jacobian of the $i$th priority level

$y_i$ - Constraint velocity of the $i$th priority level

$P_i$ - Nullspace projector of priority level $i$

$q_i$ - Overall solution on priority level $i$

**Integration in Rock**
- Whole body control component (WBC):
  - Configuration via URDF (robot model) and a WBC configuration (constraints, priorities, task frames, ...)
  - Dynamically creates port interfaces for each constraint
  - Sorts constraints by priority, creates equation system
  - Solver implements the iTaSC algorithm
  - Control Library: Collection of generic controllers that provide the input for the WBC component

### Experimental Results

<table>
<thead>
<tr>
<th>Name</th>
<th>Constraint description</th>
<th>TF Root</th>
<th>TF Tip</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero Force R</td>
<td>Maintain zero force on right wrist</td>
<td>Hand R</td>
<td>Hand L</td>
<td>0</td>
</tr>
<tr>
<td>Zero Force L</td>
<td>Maintain zero force on left wrist</td>
<td>Hand L</td>
<td>Hand R</td>
<td>0</td>
</tr>
<tr>
<td>Hold Pose R</td>
<td>Hold the pose of the right arm</td>
<td>Hand R</td>
<td>Hand L</td>
<td>1</td>
</tr>
<tr>
<td>Keep Parallel</td>
<td>Keep left &amp; right wrists parallel</td>
<td>Hand R</td>
<td>Hand L</td>
<td>2</td>
</tr>
</tbody>
</table>

Constraints used in the experiment, ordered by priority (0 = highest)

### Integration of Sensor-Based Reactions to Global Motion Planning
- Connection of sensor-based reactions to global motion planning
- Extension to torque controlled motions

### Conclusion
- iTaSC allows specification of complex sensor-based tasks
- Easier configuration and monitoring of the constraints in Rock

### Next Steps
- Execution of task sequences with context-specific adaptation of task hierarchies, weights and controller configurations
- Extension to torque controlled motions
- Connection of sensor-based reactions to global motion planning
German Research Center for Artificial Intelligence (DFKI) GmbH

DFKI Bremen
Robert-Hooke-Straße 1
28359 Bremen
Germany
Phone: +49 421 178 45 0
Fax: +49 421 178 45 4150

DFKI Saarbrücken
Stuhlsatzenhausweg 3
Campus D3 2
66123 Saarbrücken
Germany
Phone: +49 681 875 75 0
Fax: +49 681 857 75 5341

DFKI Kaiserslautern
Trippstadtter Straße 122
67608 Kaiserslautern
Germany
Phone: +49 631 205 75 0
Fax: +49 631 205 75 5030

DFKI Projektbüro Berlin
Alt-Moabit 91c
10559 Berlin
Germany
Phone: +49 30 238 95 0

E-mail:
reports@dfki.de

Further information:
http://www.dfki.de