Pre-Course Meeting – Building a Modular Robot

Advisors: Paul Maroldt, Constantin Dresel, Stefanie Manzinger, Stefan Liu, Martin Riedel, Matthias Mayer, Roman Hölzl

Supervisor: Prof. Althoff
Modular Robots

- Adaptable for any specific tasks
- Versatile for varying application needs
- Easy to repair
- Low cost
Video Demo: Our Modular Robot
Aim of the Hybrid Practical Course

- Understand and apply the theory behind collaborative, modular robots
- Learn about:
  - Robot programming
  - Robot control
  - Mechanics
  - Kinematics and dynamics
  - ROS
  - Project planning and team work

Hybrid character:
  I. Theory and ROS (1–2 sessions)
  II. Research and development (~10 sessions)
I. Theory and ROS

- History of robotics (briefly)
- Dynamics
- Kinematics
- Robot control
- Collaborative robotics
- Modular robotics
- Robotics work at the chair
- ROS Introduction
II. Research and Development Topics

A. Robot Visualization and Simulation
B. Robot Control Device and User Interface
C. Trajectory and Path Planning
D. Self-Collision Checking and Avoidance
E. Model-Based Controller with Accuracy Measurements
F. Computer Vision
G. 3D Use Case Scanning
H. Fail-Safe Planning
   I. Human Motion Synthesis for Reinforcement Learning

Topics with ROS

Topics without ROS
A) Robot Visualization and Simulation

User Story:
A realistic visualization and simulation of the modular robot and its environment should be provided to allow the user to perform fast and goal-oriented development and testing. The modelled robot should closely resemble the behavior of the real robot and should be able to manipulate its simulated environment.

https://www.youtube.com/watch?v=qCn8lkacJz0

Requirements:
• C/C++
• ROS or MATLAB/Simulink
• Basic robotics knowledge beneficial
• Previous experience with Gazebo beneficial

Expected working steps:
• Auto-generate modular robot models for Gazebo
• Create Gazebo-plugin for controlling the robot
• Control the robot from Simulink or ROS
• Include environment models in simulation
• Make objects in the environment manipulable

Supervisors:
Constantin Dresel, constantin.dresel@tum.de
Stefanie Manzinger, stefanie.manzinger@tum.de
B) Robot Control Device and User Interface

User Story:
A user wants to control a modular robot via various interface devices, such as touchscreens, mouse, keyboard, joystick or a spacemouse. These control inputs may move the robot in joint- or workspace (e.g. [https://youtu.be/8yGaMFDuFbE](https://youtu.be/8yGaMFDuFbE)).

Further, the user wants a graphic interface (ROS node) to enable a quick start and setup. It shall show and control the robot’s state, plots signals for reference and diagnosis, show a robot’s pose, etc.

Requirements:
• C/C++
• ROS
• Qt (optional)

Expected working steps:
• Identify and implement suitable control devices
• Conceptualize and implement a suitable user interface
• Perform user studies on control devices and interfaces

Supervisor:
Martin Riedel, martin.riedel@tum.de
Stefanie Manzinger, stefanie.manzinger@tum.de
C) Trajectory and Path Planning

User Story:
The user shall be able to pre-define points in joint and task space, which are then connected by a path planner without hitting any obstacles. The path is carried out with to the robot harmless motion profiles. The path planning can also be done online to utilize the results of computer vision.

Requirements:
• C/C++ (ROS)
• Basic robotics knowledge beneficial

Expected working steps:
• Selecting suitable path planning algorithms
• Implement path planning
• Implement motion profiles

Supervisor:
Paul Maroldt, paul.maroldt@tum.de
Roman Hölzl, roman.hoelzl@tum.de
D) Self-Collision Checking and Avoidance

User Story:
The robot prevents the user from driving the robot to close to a self collision. For this the robot detects when a self collision is about to happen and defines appropriate measures to prevent it. In scenarios where automatic path planning is performed, the algorithm prevents the path from being carried out. In scenarios where a human is controlling the robot directly, the robot prevents self collisions and gives feedback to the user.

Requirements:
• C/C++ (ROS)

Expected working steps:
• Research online collision checking algorithms
• Familiarize with geometrical and kinematic description of modular robots
• Implement collision detection algorithm
• Optimize collision detection algorithm for runtime
• Implement collision avoidance strategy

Supervisor:
Paul Maroldt, paul.maroldt@tum.de
Martin Riedel, martin.riedel@tum.de
E) Model-Based Controller with Accuracy Measurements

User Story:
In industrial robotics, accuracy and repeatability are key characteristics which highly depend on a good controller. In this topic a controller has to be implemented and measurements for repeatability and accuracy shall be carried out. For this the measurement systems have to be set up and a measurement routine must be developed.

Requirements:
• C/C++ (ROS)
• Basics in control theory
• MATLAB

Expected working steps:
• Choose controllers to implement
• Implement controllers in ROS
• Simulate controller performance
• Research accuracy and repeatability measures
• Implement accuracy and repeatability measurement using e.g. Vicon camera system, laser sensors and dial gauges
• Compare implemented controllers

Supervisor:
Paul Maroldt, paul.maroldt@tum.de
Roman Hölzl, roman.hoelzl@tum.de

F) Computer Vision

User Story:
One of the first steps towards scene understanding is the detection and localization of 3D objects in the workspace. Your goal is to detect and track a 3D object given a 2D camera image and a 3D model of its shape.

https://www.youtube.com/watch?v=nFP_9sl09t8

Requirements:
• C++, Python
• ROS
• Computer vision experience beneficial
• CAD experience beneficial

Expected working steps:
• Selection of adequate frameworks
• Selection of an adequate camera, lighting, and camera mount
• Implementation of camera calibration procedure
• Generation of a database of 3D shape models
• Implementation of a real-time capable method for 3D pose estimation using 3D models

Supervisor:
Constantin Dresel, constantin.dresel@tum.de
Martin Riedel, martin.riedel@tum.de
G) 3D Use Case Scanning

User Story:
To quickly and efficiently collect a large amount of use cases, a 3D scanning system is necessary. The goal of this project is to develop a system which is capable of 3D scanning a production environment, machines and periphery in order to use it for robot use cases.

https://www.youtube.com/watch?time_continue=120&v=71eRxTo1DaU&feature=emb_logo

Requirements:
- C/C++
- 3D scanning
- SLAM experiences beneficial
- ROS (optional)

Expected working steps:
- Select suitable capture device
- Implement 3D scanning
- Ensure real-life sizing
- Think of a way of including start and end point of robot trajectory
- Export as 3D file (and/or stream to ROS)

Supervisor:
Roman Hözl, roman.hoelzl@tum.de
Constantin Dresel, constantin.dresel@tum.de
H) Fail-Safe Planning

User Story:
Your goal is to make robots safe for humans: the robot should stop when it is approached by a human. When the robot is pouring a beer, it should bring the bottle into a vertical position, before the human reaches it.
https://www.youtube.com/watch?v=BYuhJqy5Wso

Requirements:
• MATLAB
• Motion planning
• Robot kinematics

Expected working steps:
• Program the trajectory for pouring beer
• Specification of safe states (bottle is vertical)
• Generation of trajectories, that lead to safe states
• Integration into a fail-safe planning framework

Supervisor:
Matthias Mayer, matthias.mayer@tum.de
Stefan Liu, stefan.liu@tum.de
User Story:
We need to generate training data for reinforcement learning of humans. To avoid real experiments, we would like to generate human motion as e.g. in computer games or movies.
https://www.youtube.com/watch?time_continue=250&v=vppFvq2quQ0&feature=emb_logo

Requirements:
• Reinforcement learning
• Unity
• Previous games engineering experience beneficial

Expected working steps:
• Extracting movement from motion capture data
• Generate virtual production environments
• Construct reward function
• Learning of natural human motion

Supervisor:
Matthias Mayer, matthias.mayer@tum.de
Stefan Liu, stefan.liu@tum.de
Administrative Matters

- 10 ECTS
- Intro and theory sessions, 2 hours each:
  - 23 April 2020
  - 30 April 2020
  - 7 May 2020
- Weekly 15 min scrum meetings with your supervisors
- Three milestone presentations (15 min) during the semester, 2 hours each:
  - 4 June 2020
  - 2 July 2020
  - 6 August 2020
- Grading:
  - Final report
  - Final presentation
  - Team performance in R&D projects
Contact

Please apply with:
- CV
- Transcript of records
- Letter of motivation (150 words)
- Your two favorite topics
- Please use email subject: „Application_PCMR“

Please send your application via email to roman.hoelzl@tum.de until 11 February 2020, 11 pm