Hardware & software interface for robot programming by manual guidance

Integrator Manual

Miha Deniša
Jožef Stefan Institute

April 2021

Abstract

The module for robot programming by manual guidance enables fast and intuitive programming of new robot tasks without using traditional robot programming systems. With manual guidance, called kinesthetic teaching, the operator physically guides the robot through the desired robot configurations and movements in order to program a sequence of robot movements that lead to a successful task execution. Manual guidance can be used for the programming of point-to-point movements or whole movements (trajectories). The first section of this manual presents the general overview of the approach and the required prerequisites. Hardware and software components of the module are presented in the second section, while instructions on using the graphical user interface are given in the last section.

Contents

1 Overview 1
- 1.1 Programming of robot tasks . . . 1
- 1.2 Kinesthetic teaching . . . 2
- 1.3 Robots supporting kinesthetic teaching . . . 2

2 Module components 3
- 2.1 Robot Operating System (ROS) . . . 3
  - 2.1.1 ROS installation . . . 4
- 2.2 Button interface . . . 5
- 2.3 Graphical user interface . . . 6
  - 2.3.1 Installation procedure . . . 6
- 2.4 Database . . . 6

3 Data acquisition by kinesthetic teaching 7
- 3.1 GUI configuration . . . 8
- 3.2 Using the GUI . . . 9
- 3.3 Point-to-point movements . . . 10
- 3.4 Adaptation, execution, and Cartesian DMPs . . . 10

1 Overview

To create a robotic workcell that is applicable to small batch size production and frequent changes in product demands, the cell must enable fast and easy programming of new robot tasks. With standard robot programming approaches, the programming is usually performed by experts at production sites. To support the programming process, many robots are equipped with teach pendants, which enable the programmer to control the robot through a relatively complex interface. Programming with a teach pendant is still relatively cumbersome and time consuming. An alternative is provided by a programming by demonstration (PbD) approach, where new robot operations are programmed by analyzing human demonstrations of the desired task [1]. Programming by demonstration enable non-expert users to generate new robot operations in a relatively short time. Kinesthetic teaching introduced in Section 1.2 is a form of programming by demonstration.

1.1 Programming of robot tasks

To program a new robot task, we typically need to program a sequence of robot movements and the associated operations that lead to a successful execution of the task. The associated operations typically involve control of robot end-effectors, e.g. grasp/release an object with a gripper, but can also relate to other peripheral equipment available in the workcell and not mounted on the robot's end-effector. As kinesthetic teaching is useful for the programming of robot movements, we focus in this manual on the programming of movements.

Some definitions are needed before we can explain the functionality of the module:

Robot con guration: a set of values specifying all robot joint positions.

Robot pose: position and orientation of the tip of the robot.

State of the robot, point: in this text, state of the robot / point refers to either robot configuration or pose.

Joint space trajectory: a dense or even continuous sequence of robot configurations and times at which the robot should reach the specified configurations.

Cartesian space trajectory: a dense or continuous sequence of robot poses and times at which the robot should reach the specified poses.

Point-to-point movement (motion): only the initial and the final robot configuration or pose and the duration of movement are specified. The in-between points are not specified. To generate the robot movement between the two specified points, the in-between robot configurations or poses are generated by computing a straight line or a higher order polynomial between the initial configuration and the final configuration, parametrized by time.

Whole movement (trajectory): the motion of the robot from the initial to the final configuration / pose is fully specified and does not necessarily follow a linear or higher-order polynomial.

Kinesthetic teaching / guidance is used to gather single points (robot configurations and poses) or dense sequences of points from which continuous trajectories can be generated.

1.2 Kinesthetic teaching

In kinesthetic teaching [2], the robot arm is held by a human demonstrator and guided in order to perform the desired task. The states of the robot during the task execution are recorded by using the robot's sensors. Kinesthetic teaching requires only limited user training and can be performed by users without in-depth knowledge about robotics. However, this does not mean that kinesthetic teaching can be performed effortlessly. The quality of the recorded data and consequently the quality of robot programs namely rely on the quality of the performed demonstrations.

1.3 Robots supporting kinesthetic teaching

The main prerequisite for kinesthetic teaching of robot configurations and trajectories is a robot arm that can be freely guided around by a demonstrator. This can be achieved by putting the robot in a gravity compensation mode, where the weight of the robot is compensated by its motors and the robot arm can be guided with a minimal effort. By applying an external force to the robot by his/her own hands, the user can guide the robot kinesthetically without any additional sensing equipment. Gravity compensation requires the availability of torque values at each joint. They can be gathered in two different ways:

1. by measuring torque values at each joint using a torque sensor,
2. by measuring the electric current applied to the motor mounted at a joint and transforming the measured currents to torque values using a dynamic model.

The first approach results in more accurate torque values as we can avoid the transformation of currents to torque values with a dynamic model. Consequently, the gravity compensation works better and the robot arm can be guided through the desired motion with less effort. The quality of the gathered data is therefore better. However, the price of robot arms that include joint torque sensors is generally significantly higher than the price of robot arms without such sensors. Robot arm equipped with joint torque sensors include Kuka LBR iiwa and Franka Emika Panda.

To reduce the price of the robot, the second approach is often used. For example, Universal Robot UR series arms do not have joint torque sensors but support gravity compensation out of the box by transforming electric currents

to torque values. This module is presented using this setup Universal Robot UR10 robot arm. An example demonstration using gravity compensation mode on a UR10 robot arm is shown in Fig. 1.

Figure 1: Kinesthetic guidance UR10 robot arm (from top-left to bottom-right).

2 Module components

This section presents all the components of this module, which consists of software and hardware components. Installation procedure for the software components is given. The module has been implemented for Universal robot UR10, but its adaptation to other robots that support gravity compensation is possible. Support for adaptation of the module to other robots can be provided by Jožef Stefan Institute on a contract basis[1].

A general overview of all main components of this module is shown in Fig. 2.

2.1 Robot Operating System (ROS)

The module requires the installation of ROS (Robot Operating System) [3, 4], which provides the database MongoDB, high level communication, and other

[1]Contact Miha Deniša: miha.denisa@ijs.si

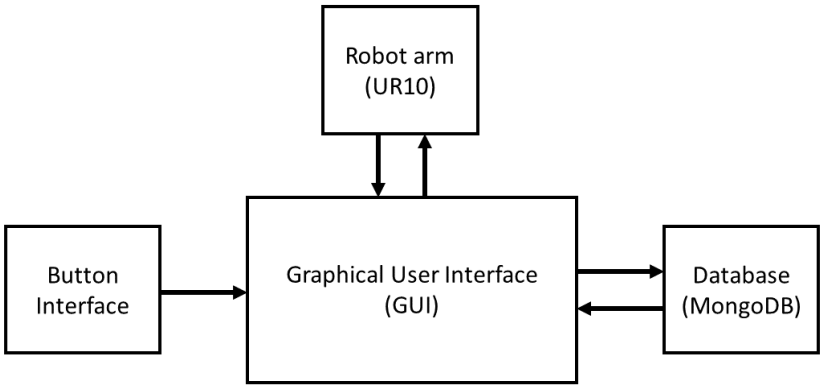

Figure 2: Overview of main module components. The GUI is at the core of the module. It receives triggers from the button interface, which are used to change the control mode of the robot, e.g. put the robot into gravity compensation mode, and initiate saving and reading the data from the database (MongoDB-based). The button interface is connected to the GUI via the robot's I/Os.

functionalities. While many features and tools are included in ROS, a subset of them is used to realize kinesthetic teaching within the cell: ROS nodes, MongoDB database, topics, services, action and parameters servers, messages, etc.

2.1.1 ROS installation

The ROS periphery interface is based on the Kinetic version of ROS. It can be installed on a machine running a Linux Ubuntu 16.04 LTS operating system. The minimal requirements for Ubuntu 16.04 are as follows:

- 2 GHz dual core processor or better
- 2 GB system memory
- 25 GB of free hard drive space
- Either a DVD drive or a USB port for the installer media
- Internet access is helpful

The tutorial covering the installation of Ubuntu 16.04 can be found at this link.

Once Ubuntu 16.04 is installed and running, ROS Kinetic can be installed by following instructions found at this link.