6th International Conference
ðComputational Mechanics and Virtual Engineering ð
COMEC 2015
15-16 October 2015, Braşov, Romania

IMITATION-BASED ROBOT PROGRAMMING METHOD FOR PREDICTING THE MOTION

Aurel Fratu[1]
[1] "Transilvania" University of Brasov, ROMANIA, e-mail fratu@unitbv.ro

Abstract: *One of the main barriers to automating a particular task with a robot is the amount of time needed to program the robot. In this paper the decrease in programming time is accomplished by combining off-line and on-line programming techniques .The method consists in using a programming platform that allows us to expediently compose robot programs. On the programming platform there is carried out the virtual prototype of the physical robotic arm to be programmed and the real working space wherein it is intended to work. In this paper, we propose a new approach to robot programming based on virtual robot prototype behavior in experimental scenery. The actions for the each robotic task are computed for virtual robot prototype and are transferred online, with a central coordination, to corresponding physical robot, which must imitate her virtual "homonymous".*
Keywords: *virtual robots, path learning by imitation, motion programming, behavioral control*

1. INTRODUCTION

A characteristic feature of robot programming is that usually it is dealing with two different worlds; the real physical world to be manipulated and the abstract models - in particular virtual prototypes models - representing this world in a functional or descriptive manner by programs and data. In the simplest case, these models are pure imagination of the programmers; in high level programming languages, e.g. it may consist of CAD data.
Creating accurate robot path points for a robot application is an important programming task. It requires a robot programmer to have the knowledge of the robot's reference frames, positions, operations, and the work space.
In the conventional "lead-through" method, the robot programmer uses the robot teach pendant accessory to position and to orientation the robot joints and end-effector and record the satisfied robot pose [3].
The basic idea behind these approaches is to relieve the programmer from knowing all specific robot details and liberate him from coding every small motion.
Today's robot simulation software provides the robot programmer with the functions of creating virtual robot and virtual path points in an interactive virtual 3D design environment.
By the time a robot simulation design is completed; the simulation robot program is able to move the virtual robot end-effector to all desired virtual path points for performing the specified operation. By different scenarios one creates particular motion paths in the virtual space that are selected for particular motion paths in physical work space.
However, because of the inevitable dimensional differences of the components between the physical robot work space and the simulated robot work space, the virtual robot path points must be adjusted relative to the actual position of the components in the physical robot work space. This task involves the techniques of calibrating the position coordinates of the simulation device models with respect to the physical robot path points [1].
Learning by imitation based on our method represents a new research topic in robotics being a promising approach, towards effective robot programming.
This work follows a recent trend in Programming by Imitation. We present a new method to programming physical robot through the imitation based on the virtual robot prototype.
Movement imitation requires a demonstrator. In our approach the demonstrator is the virtual robot prototype and the imitator is the physical robotic manipulator. Conformity our method the dynamics of the motion of the virtual robots is reproduced by the physical robot.
In this paper one use the virtual robot prototypes and the motion capture systems to obtain the reference motion data, which typically consist of a set of trajectories in the Cartesian space.

We restrict our study to imitation of manipulative tasks. In particular, the ability to imitate virtual robots gestures and pursue task-relevant paths is essential skills for physical robotic manipulators, in the spirit of our method.
Inspired by the personal patent [8] and based on the motion imitation concept this paper develops a general policy for robot motion programming based on virtual robot prototypes.
The strong point of the proposed method is that it provides fast on-line re-planning of the motion in the face of spatial-temporal perturbations and it allows generalizing a motion to unseen context based on particular scenarios. The objective of the present paper is giving an overview of a new robot programming method based on the virtual prototypes.

2. PROGRAMMING FROM A SIMULATED VIRTUAL MODEL

Programming by imitation has appeared as one way to respond to growing need for intuitive control methods and is one of the most promising programming techniques for robotic manipulators. This technique allows even inexperienced users to easily program the robots based on the teaching by imitation paradigm.
We present a description of the theoretical aspects of the programming from a simulated virtual model and of the strategy which is at foundation of this approach. The advantages of such programming approach as an alternative to the classical methods (e.g. vision guided trajectory imitation) are on-line adaptation to the motion of the virtual prototype.
A solution to the above problem is to construct a virtual prototype model and transfer the virtual trajectory by interacting with the physical model.
Designing a model would be an option; however, the behavior of the robots is very difficult to model. Moreover, the use of system knowledge is contrary to our research aim. Therefore we focus on creating a virtual prototype model from experimental data obtained from the physical robot. Thus, we will prove that our method guarantees the motion optimization for each robotics tasks using a planning module and a visualization module.
The planning package communicates primarily with simulation environment. A planning module can send messages to the simulation system such as computed plans for the robots The planning module can further send trajectory and planning structure information to visualization; so users can see the results of this algorithm. The planning module also receives control signals from the simulation module, such as when to start planning joint trajectories [2].
The visualization module is in charge for visualizing any aspect needed by the programmer for the optimization process. Users interact with the simulation environment through the visualization. This includes, but not limited to, computer screen. The visualization provides an interface to develop interactive implementations based on imitation strategy.
In our work, we assume that learning of the deterministic part for description motion dynamics should be sufficient to design the corresponding robot control.
We particularly refer to the ability of the system to react to changes in the environment that are reflected by motion parameters, such as a desired target position and motion duration. Therefore, the system is able to manage with uncertainties in the position of a manipulated object, duration of motion, and structure limitation (e.g., joint velocity and torque limits).

3. ONLINE IMITATION METHOD OF THE VIRTUAL DYNAMICAL SYSTEMS

In robotics, one of the most frequent methods to represent movement strategy is by means of the learning from imitation. Imitation learning is simply an application of supervised learning. One goal of imitation of the dynamical systems is to use the ability of coupling phenomena to description for complex behavior [6].
In this paper, we propose a generic modeling approach to generate virtual robot prototype behavior in experimental scenery. The actions for the each task are computed for virtual robot prototype and are transferred online, with a central coordination, to corresponding physical robot, which must imitate her virtual "homonymous". Notice the similarity between moves of the virtual robot prototype in the virtual work space and the "homonymous" moves in the real work space of the physical robot. We assume to use the virtual robot prototypes and the motion capture systems to obtain the reference motion data, which typically consist of a set of trajectories in the operational space.
Our method consists in using a programming platform on which there is carried out the virtual prototype of the physical robotic arm to be programmed and the real working space wherein it is intended to work. The method combines off-line and on-line programming techniques
In the robot program there is written a source code intended to generate the motion paths of the virtual robotic arm prototype. The numerical values of the prototype articulation variables are sent to the data register of a port

of the information system which, via a numerical interface, is on-line transferred into the data registers of the controllers of the actuator of the physical robotic arm.
Finally, there are obtained tracking structures due to which the moving paths of the virtual robotic arm joints are reproduced by the physical robotic arm joints, thereby generating motion within the real working space.
Imitation learning from our strategy demonstrates how to obtain dynamical virtual models with CAD systems. Those online adjusted virtual models are among the most important properties offered by a dynamical systems approach, and these properties cannot easily be replicated without the feed-back from physical robot of our proposed structure.
The objective of a movement is to generate a reaching movement from any start state to a goal state [8]**.** The discrete dynamical system is initialized with a minimum movement, which is frequently used as an approximate model of smooth movement.
The proposed structure uses a virtual demonstrator for planning the movements of the physical robot. We investigate the potential of imitation in programming robotic manipulators with multiples degrees of freedom when the associated joints must be coordinated concurrently.
The imitation strategy consists in a proportional real-time mapping between each virtual joint and the corresponding physical joint.
The system requires the programmer to perform an initial calibration routine to identify the range of motion for each movement. Each range is divided into as many intervals as the number of feasible discrete configurations of the corresponding joint.

3.1 Predicting Robot Control Latency

The imitation method supposes a delay between the action of the virtual prototype and physical robot. The time elapsed between making an action decision and perceiving the consequences of that action in the environment is called the control delay.
A predictor approximates a negative delay. It has access to the delayed robot state as well as to the un-delayed controller actions and is trained to output the un-delayed robot state. The predictor contains a forward model of the robot and provides instantaneous feedback about the consequences of action commands to the controller. If the behavior of the robot is predictable, this strategy can simplify controller design and improve control performance. All physical feedback control loops have a certain delay, depending on the system itself, on the input and output speed and, of course, on the speed at which the system processes and transfer the information from virtual environment to physical environment.
In our system we use the concept of motor prediction and the paper describes a method to reduce the effects of the system immanent control delay. It explains how we solved the task by predicting the movement of the robots, using a virtual prototype model. Just virtual robot positions and orientations as well as the most recent motion commands sent to the physical robot are used as input for the prediction.
In our system the global sensorial module is used to determine the positions of all robot joints. In order to control the behavior of the robot in a way that is appropriate for the situation of her virtual homonym on the virtual scene we need the exact positions of them at every moment. Because of the delay inherent in the control loop, however, the behavior control system actually reacts to the environment with finite delay (about few ms size).When moving faster the delay becomes more significant as the error between the real position and the virtual position used for control grows.
In order to correct this immanent error we have developed an informatics network which processes the positions and the orientations of each joint. We use recorded preprocessed data of moving virtual robot to train the network. It predicts the actual positions of the robots. These predictions are used as a basis for control. The action commands will be sent to the physical robots during the real time.
Using the concept of motor prediction one predicts the joints' position of the robot arm, rather than sensing it by sensorial system. In this model the predictions are based on a copy of the motor commands acting on the virtual joints. In effect, the virtual joints' position of the virtual robot arm is made available before physical sensory signals become available.
These predictions are used in an inner control loop to generate sequences of actions that guide the robot towards a target state. Since this cannot account for disturbances, the robot predictions are delayed by the estimated dead time and compared to the sensed robot state. The deviations reflect disturbances that are feedback into the controller via an outer loop. The fast internal loop is functionally equivalent to an inverse-dynamic model that controls a robot without feedback.
A simple approach to implement the predictor would be to use a mathematical method. This method is very effective to handle linear effects, for instance the motion of a free joint [5]. It is however inappropriate for robot arm that contain significant non-linear effects, e.g. caused by the slippage or by the behavior of its motion

controller. For this reason, we must use a virtual prototype to predict non-linear systems. But this approach requires predicting first a good virtual model of the robot.
We then measure the time between sending the command to change the direction of motion and perceiving a direction change of the robot's movement. There are many possibilities to counter the adverse effects of the control delay. The easiest way would be to reduce precision requirements, but this would lead to unnecessary collisions and uncontrolled behavior. Control researchers have made many attempts to overcome the effects of delays.

3.2 System Architecture

Action commands are sent via a wireless link from virtual robot to the physical robot that contains only minimal local intelligence. Our imitation experimental system is illustrated in Figure 1.

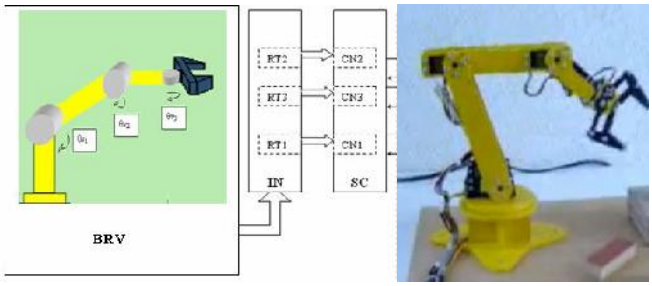

Figure 1 Experienced robotic manipulator

For behavior control of the physical robot is use one Video camera. It looks at the field from above and produces an output video stream, which is forwarded to the central PC. Images are captured by a frame memory and given to the vision module.
The global computer vision module analyzes the images, finds the paths of the physical robots and produces as output the positions and orientations of the robot's joints, as well as the position of the arm. It is described in detail in [3].
Based on the gathered information from virtual robot and the physical robot, the behavior control module then produces the commands for the physical robot: desired rotational velocity, driving speed and direction, as well as the activation of the logical device. The central PC then sends these commands via a wireless communication link to the physical robots. The hierarchical reactive behavior control system of the team virtual robot – physical robot is described in [5]. For each robot joint is needed a microcontroller for omni directional motion control. It receives the commands and controls the movement of the robot using PID controllers (see [4]). Feedback about the speed of the joints is provided by the pulse generators which are integrated in each servo-motor

4. EXPERIMENTAL CONFIGURATION FOR ROBOT PROGRAMMING BY IMITATION

The approach was implemented using a simulation software platform called Robot Motion Imitation Platform (RMIP) developed at University of Brasov [8], who its feasibility has been demonstrated in some simulation settings for manipulative works, using Virtual Reality.
We extended the simulation software platform to programming a physical robot. We see them as intelligent structure that can be used to transfer - in real time - the gesture of the virtual robot to the physical robots for generating complex movements in real work space. One transfers via intelligent interface, the virtual joint angles data from a motion capture system to a kinematic model for a physical robotic manipulator.
The RMIP is an architecture that provides libraries and tools to help software developers in programming.
Platform RMIP is focused on 3D simulation of the dynamics systems and on a control and planning interface that provides primitives for motion planning by imitation.
Also is a object-oriented infrastructure for the integration of controllers, as well as the integration of planners with controllers to achieve feedback based planning, In particular, RMIP is used to provide concrete implementations of motion planners.
The platform RMIP utilizes a framework which allows both high level and low level controllers to be composed in a way that provides complex interactions between virtual and physical robots.

In our strategy we just included the possibility of functional coupling virtual and physical robot model. The functional coupling consists to map of the motion of the demonstrator robot arm with the imitator robot arm. The former is displayed on the screen, giving the visual information to programmer and is connected to the latter through intelligent interface. In our experiments we have adapted as imitator a small robotic manipulator, named EXMAN and we start with a 3 degree-of-freedom (DOF) discrete movement system that models point-to-point attained in a 3D Cartesian space.
The arm moved in a free space without obstacles, had three active joints (shoulder, elbow and wrist) and was driven by electrical actuators.
Figure 2 shows our experiment involving the imitation learning for a physical robotic arm with 3 degrees-of-freedom (DOFs) for performing the manipulate tasks. We demonstrated the imitation of elbow, shoulder and wrist movements. Importantly, these tasks required the coordination of 3 DOFs, which was easily accomplished in our approach.

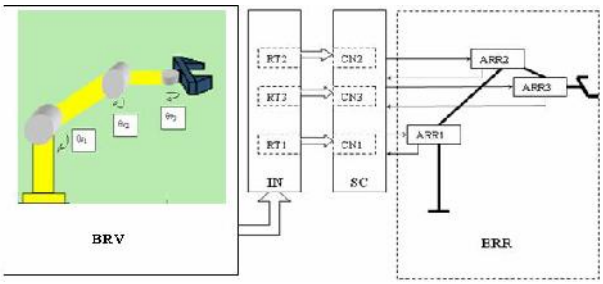

Figure 2. Imitation system architecture

The imitated movement was represented in joint angles of the robot. Indeed, only kinematic variables are observable in imitation learning. The robot was equipped with a controller (a PD controller) that could accurately follow the kinematic strategy (i.e., generate the torques necessary to follow a particular joint angle trajectory, given in terms of desired positions, velocities, and accelerations) [7].
Figure 2 also displays (left image) the user interface of a virtual robotic manipulator arm, which has been created which a dynamical simulator.
To generate the motion sequence for the real robot, one captures the motions from a virtual robot model and maps these to the joint settings of the physical robot.
Initially, a set of virtual desired postures is created to the virtual robotic arm BRV and the pictures' positions are recorded for each posture, during motion.
These recorded pictures' positions provide a set of Cartesian points in the 3D capture volume for each posture. To obtain the physical robot postures, the virtual pictures' positions are assigned as positional constraints on the physical robot. To obtain the physical joint angles one use standard inverse kinematics (IK) routines. The IK routine then directly generates the physical joint angles on the physical robot for each posture.
Referring the Figure 2 we comment the following: on programming platform, a robot program is carried out off-line, and one sends into the data registers of a port of the hardware structure, the numerical values of the joint variables of the virtual prototype of the robotic arm (BRV) and displays on a graphical user interface, the evolution of the virtual prototype during the carrying out of the robotic task. Via numerical interface (IN) the virtual joint dataset, from the data registers of the port of the hardware structure of the programming platform are transferred into the data registers of the numerical comparators of the controllers. These datasets are reference inputs of the pursue loops, resulting a control system (SC).
The symbolic spatial relations specifying the virtual work space can be used for the automatic pursuits of possible virtual path as well as for planning of appropriate behavior of the physical robot arm BRR, which may guide the motion process during task execution.
The easiest way to generate the spatial relations explicitly is the interactively programming of the behavior of the virtual prototype in his virtual environment, in order to specify suitable positions θ_{v1}, θ_{v2}, θ_{v3}. This kind of specification provides an easy to use interactive graphical tool to define any kind of robot path; the user has to deal only with a limited and manageable amount of spatial information in a very comfortable manner.
The robot programming system has to recognize the correct robot task type and should map it to a sequence of robot operations [7]. The desired pathways are automatically transferred and parameterized in the numerical interface IN, using the path planner.
The applicable robot tasks are designed and the desired pathways are programmed off-line and stored in the buffer modules RT1, RT2, RT3.

The comparative modules CN1, CN2, CN3 furnish, to the pursuit controllers, the datasets involving the expected state of the virtual robot prototype and the measured state of the physical robot.
While motion execution is in progress, the real robot joints ARR1, ARR2, ARR3 are activates into the real work space. Each actuator was connected by a sensor in the closed-loop. Each time, a skill primitive is executed by the robot control system SC; it changing the robot joints position. As no time limit for the motion is specified, the physical robot imitates the behavior of the virtual robot.
In our laboratory currently we are developing Cartesian control architecture able to interpret the physical robot commands in the above given form. The basis of our implementation is a flexible and modular system for robot programming by imitation.
In our experimental configuration in order to prove the correctness of the robot programming by imitation we have chosen an robotic manipulator equipped with electrical actuators, mounted on the physical robot's joints.
The robot's control unit is connected via TCP/IP to a PC equipped with the interface card; the PC is running the simulation and control process. The robot control system receives and executes each 16 ms, an elementary move operation.
Due to the kinematics limitations of physical robot, the resolution of the joints is quite limited. After the calibration phase, physical robot starts imitating the gestures of the virtual robot demonstrator. The user can decide to activate all the three degrees of freedom concurrently or with a restricted subset by deactivating some of the sensors attached to each joint of the physical robot.
Our system requires an essential step in that one converts the position errors into motor commands by means of the PD controller [4].

5. CONCLUSION

We propose to use a virtual robot as a predictor for the robot motion, because this approach doesn't require an explicit model and can easily use robot commands as additional input for the prediction. This allows predicting future movement changes before any of them could be detected from the visual feedback. As with all control systems, there is some delay between making an action decision and perceiving the consequences of that action in the environment. We have successfully developed, implemented, and tested a imitation system for predicting the motion of the physical robots. The prediction compensates for the system delay and thus allows more precise motion control. The virtual robot is trained with data recorded from real robots. We have successfully field-tested the system at several scenarios. The predictions using the virtual prototypes improve speed and accuracy of the physical robot motion.
The virtual tracking paths of the virtual joints are reproduced in the real work space by the actuators of the physical robotic arm. The reference datasets are obtained using a motion capture channel taking into account the joints motion range.. In future work we will investigate these idea and we will examine the possibility of how to implement them.

REFERENCES

[1] R. Diankov and J. Kuffner, "Openrave: A planning architecture for autonomous robotics". Technical report, CMU-RI-TR-08-34, The Robotics Institute, Carnegie Mellon University, 2008.
[2] M. Turpin, N. Michael, V. Kumar," Trajectory planning and assignment in multi robot systems". Proc. Workshop on Algorithmic Foundations of Robotics, 2012
[3] J. Snape, J. van den Berg, S. J. Guy, and D. Manocha, "Independent navigation of multiple mobile robots with hybrid reciprocal velocity obstacles", IEEE/RSJ Int. Conf. Intelligent Robots and Systems, St. Louis, Mo., pp. 5917–5922, 2009.
[4] M. Nicolescu, M. Mataric, *Natural methods for robot task learning- Instructive demonstrations, generalization and practice,* In Proc. Intl Joint Conf. on Autonomous Agents and Multi-agent Systems (AAMAS), Melbourne, Australia, pp. 241-248, 2003.
[5] S. Calinon, F. Guenter and A. Billard, *On learning, representing and generalizing a task in a humanoid robot,* In IEEE Trans. on Systems, Man and Cybernetics, Part B, vol. 37, no. 2, pp., 286-298, 2007
[6] M. Pardowitz, R. Zoellner, S. Knoop, and R. Dillmann, *Incremental learning of tasks from user demonstrations, past experiences and vocal comments*, In IEEE Trans. on Systems, Man and Cybernetics, Part B, vol. 37, no. 2, pp. 322-332, 2007.
[7] J. Kober, J. Peters, *Learning motor primitives for robotics,* In Proceedings of the IEEE International Conference on Robotics and Automation, pp. 2112–2118, 2009, Piscataway, NJ: IEEE
[8] A. Fratu, B., Riera and V., Vrabie : *Predictive strategy for robot behavioral control,* Proceedings in Manufacturing Systems, ISSN 2067-9238, Volume 9, Issue 3, 2014, pp. 125-130.