Accurate trajectory tracking on the KUKA youBot manipulator: Computed-torque control and Friction compensation

Jeyaprakash Rajagopal

Hochschule Bonn-Rhein-Sieg
University of Applied Sciences

b-it Bonn-Aachen International Center for Information Technology

Supervisors

Prof. Dr. Paul G. Plöger
Prof. Dr. Roustiam Chakirov
M.Sc. Sven Schneider

10th September, 2018

Motivation

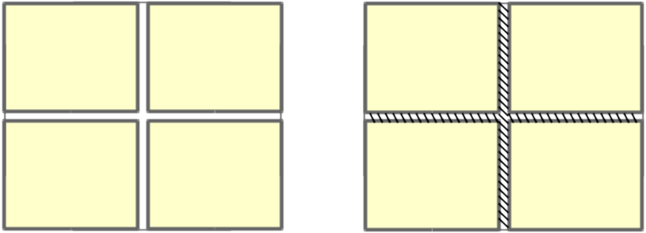


Figure 1 : Filling gaps between tiles

- Filling the gaps with waterproof liquid in between tiles
- Welding and surgical tasks also require high-precision tracking
- Don't damage environment and the robot

Related work

- Newton-Euler formulation has been used mostly in rigid-body algorithms [3] [4] [5] [8]
- In previous work [7], geometrical relation semantics were constructed in the numerical level and the parameter identification is incomplete
- Parameters identification methods based on the modified Newton-Euler formulation were proposed by [3] [6]
- The friction modeling and compensation brings us close to the complete dynamics [10] [11]
- Computed-torque control scheme has been proposed by many authors [2] [9] which discuss progress towards complete dynamics

Problem formulation

- Achieving accurate trajectory tracking in a non-linear system is difficult due to the following
  - Complexity of the dynamics
  - Uncertainties
- So, manipulator dynamics has to be considered in control scheme
- Construction of rigid-body algorithms has to consider the following
  - Geometric relations between rigid-bodies has to involve semantics checking in addition to numerical computations
  - Rigid-body algorithm's with this standard check avoids logical error, and perform meaningful operations
- Requirement of excitation trajectories for parameter identification

Approach

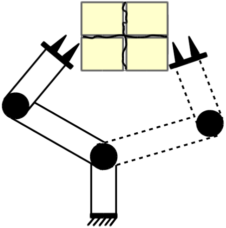


Figure 2 : Manipulation task

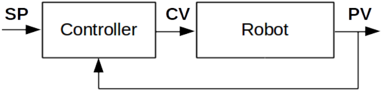


Figure 3 : Closed-loop system

- Problem: It could damage the environment and robot due to
  - system is non-linear
  - compromise on performance due to uncertainties

Approach

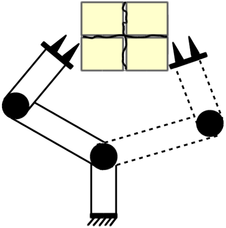


Figure 2 : Manipulation task

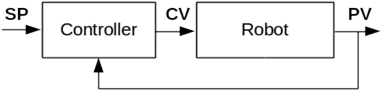


Figure 3 : Closed-loop system

- Problem: It could damage the environment and robot due to
  - system is non-linear
  - compromise on performance due to uncertainties
- Solution: Accounting manipulator dynamics and manipulator configuration

Basic and alternate approach

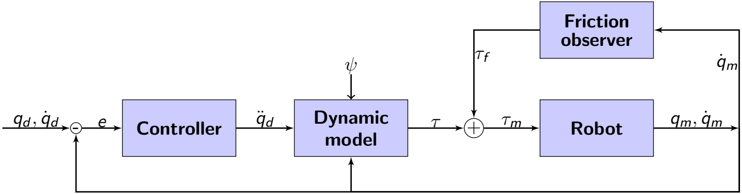


Figure 4 : Basic approach: inaccurate model parameters

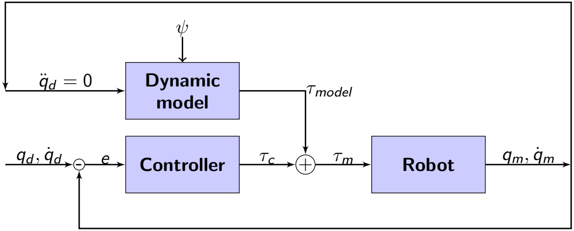


Figure 5 : Alternate approach with simple computed-torque control

Approach

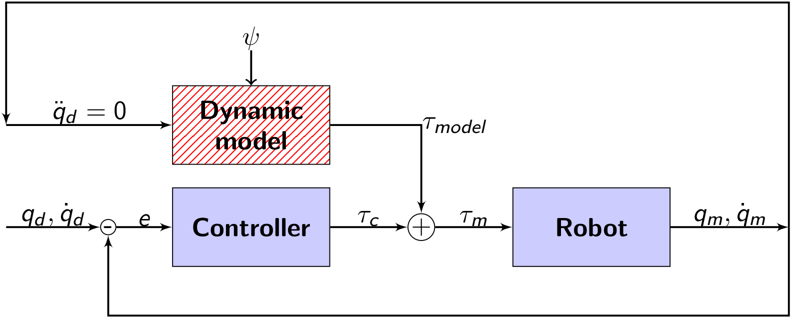


Figure 6 : Alternate approach

- Let's have a look into dynamic model of the robot manipulator

Dynamic model I

- The general equation of motion [2] can be given as

$$\tau_{model} = M(q,\psi) \overbrace{\ddot{q}}^{Joiningpoint} + \underbrace{C(q,\dot{q},\psi)\dot{q} + N(q,\dot{q},\psi)}_{non-linear} \quad (1)$$

- where M, C and N are non-linear functions of ψ (model parameters)
- $\ddot{q}$ can be replaced with linear controller i.e. PID [12]
- The non-linear terms C, N are handled by dynamic model which includes friction
- This work uses **Inverse Dynamics** solver from Orocos KDL [4]

Dynamic model II - findings

- Kinematic chain of youBot manipulator is constructed in KDL
- Dynamic model parameters ψ are taken from manufacturers [13] defined w.r.t. reference frame
- Center of mass and inertial frames are depicted w.r.t. reference frame

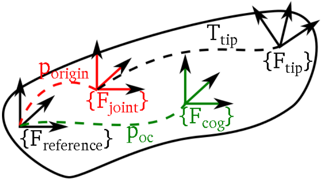


Figure 7 : Segment image [4]

Dynamic model II - findings

- Kinematic chain of youBot manipulator is constructed in KDL
- Dynamic model parameters ψ are taken from manufacturers [13]

  defined w.r.t. reference frame
- Center of mass and inertial frames are depicted w.r.t. reference frame

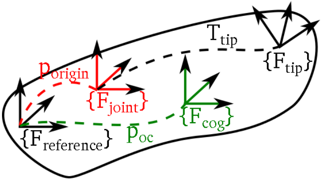


Figure 7 : Segment image [4]

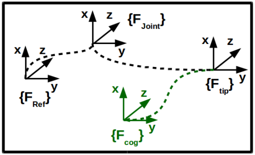


Figure 8 : Modified

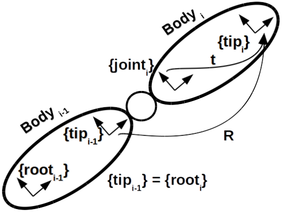
- But implementation considers tip frame as reference
- Relations between rigid-bodies are defined implicitly

Dynamic model III

- In previous work [7], implementation was based on coordinate computations without semantics checkings
- Geometric relation semantics of rigid-body algorithms are analyzed
- It includes pose, motion and force related semantics
- Why is it important to include semantics checking?
    - Rigid body conventions differ from one author to another
    - To avoid logical errors in rigid body algorithm construction
    - i.e. Transforming twists between rigid bodies requires a common point and same coordinate frame

Dynamic model IV

Figure 9 : KDL segment



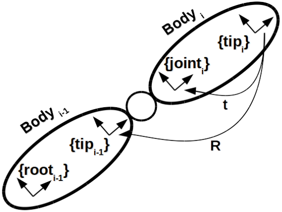
- t: Position($tip_i, joint_i$)

  R: Rotation($[tip_i], [root_i]$)

- $V_J = \text{Twist}(joint_i | Body_i,$

  $Body_{i-1}, [root_i])$

Figure 10 : Featherstone

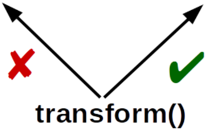


- t: Position($tip_i, joint_i$)

  R: Rotation($[root_i], [tip_i]$)

- $V_J = \text{Twist}(root_i | Body_i,$

  $Body_{i-1}, [root_i])$



Parameter identification - Excitation trajectories

- Trajectory parameterization and optimization is needed to get excitation trajectories [6]
- Finite Fourier series is used to get parameterized trajectories
- Optimizing an objective function $K^T K$ over the coefficients a, b, q(0)
    - where q(0) represents initial position of the joint
    - a, b are the coefficients that covers the workspace of the joint
    - Constraints are $q, \dot{q}, \ddot{q}$ within the joint limits and no collision constraint

Approach

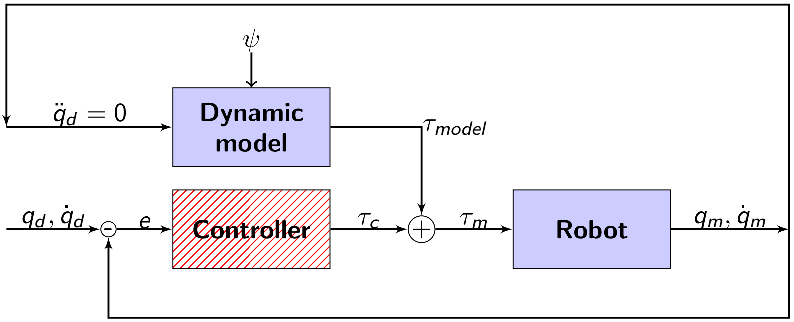


Figure 11 : Alternative approach

- Let's have a look into controller
- Computed-torque control runs at $\approx$ 1 KHz due to EtherCAT sampling constraint

Safety controller

- Safety of the hardware is the primary focus to save equipments
- Before commanding **torque** to manipulator joints, three different checks are performed
  - **Position** limit check based on measured joint positions
  - **Velocity** limit check based on measured joint velocities
  - **Torque** limit check based on measured joint torques
- If any of these check fails controller sets joint velocities to zero and quits
- Artificial joint limits are imposed and clamping

Cascade PI control I

- The outer-loop controller is position PI and inner-loop is velocity PI

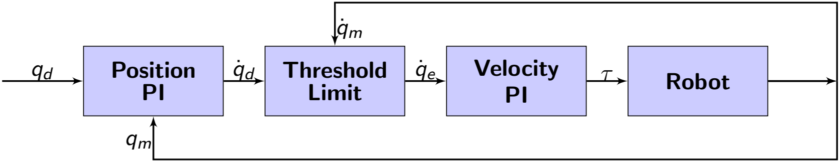


Figure 12 : Cascade PI controller

- Position controller output is fed as an input to velocity PI controller

$$\dot{q}_d = q_e \cdot K_p + \sum q_e \cdot K_i \tag{2}$$

- Control output of position PI is fed to inner-loop

Cascade PI control II

- Produces higher velocity if disturbances are not handled
- Limiting velocity of the joint

$$\dot{q}_e = limit(\dot{q}_d - \dot{q}_m) \tag{3}$$

- Cascaded PI control along with gravity compensation looks like

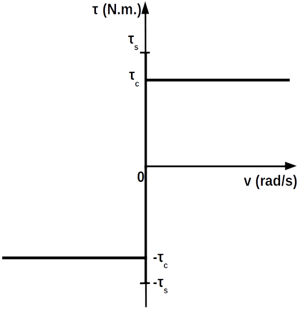
$$\tau = \overbrace{(\dot{q}_e \cdot K_p + \sum \dot{q}_e \cdot K_i)}^{ControlVariable} + \overbrace{\tau_g(q)}^{Torqueduetogravity} + bias \tag{4}$$

- The above equation specifies computed-torque control scheme

Friction observer

- Friction is a non-linear phenomenon that needs to be modeled and it can be classified into
  - Static friction is applied when joint is in equilibrium state
  - Coulomb friction is applied when joint is at motion

Figure 13 : Simple model of friction



Flow diagram and libraries used

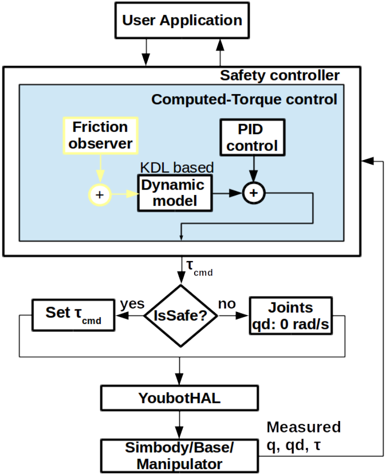
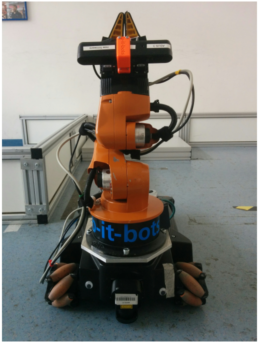


Figure 14 : Software flow

Experimentation



Experimentation steps

- Safety controller on base and arm
- Tuning controller gains
- Cascade PI control
- Gravity compensation
- Validation of model-based controller

Empirical tuning of controller gains

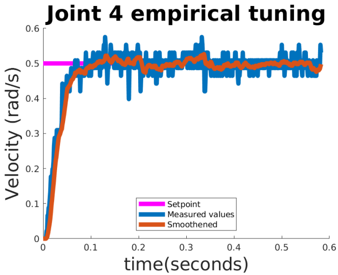


Figure 15 : Velocity PI tuning

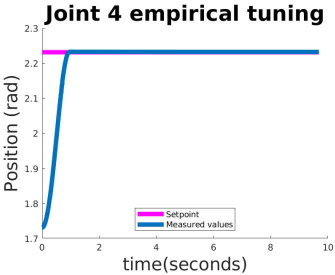


Figure 16 : Position PI tuning

- Inner-loop controller is tuned at first then outer-loop
- Increase P-gain till it reaches the set-point
- Gradually increase I-gain till steady-state error is close to zero

Validation of computed-torque controller

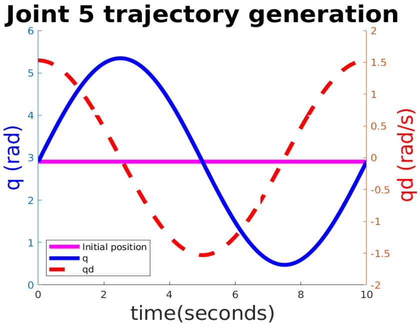
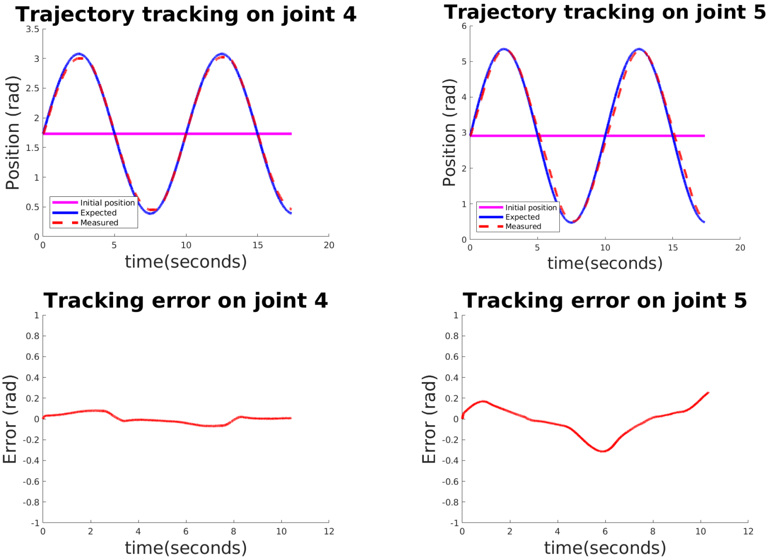


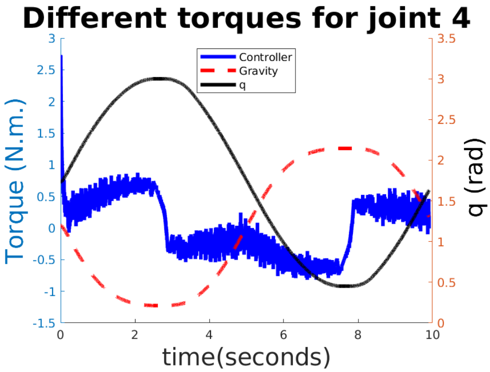
Figure 17 : Analytical trajectory

- Simple sine wave is selected
- It is simpler to derive the velocity from q
- Compute maximum and minimum velocity of joints
- To validate the model-based controller

Validation of computed-torque controller



Model and controller torques



- Model has no high-frequency noises in gravity torques
- Controller handles disturbances

Conclusions I

- Attempt to achieve complete dynamics which improves the performance of computed-torque control
- Kinematic chain of youBot manipulator is constructed manually
- The basic friction model that comprises static, Coulomb friction is implemented and compensation terms are added in dynamic model
- Safety controller ensures the safety of the robotic joint with an acceptable delay in response
- Empirical tuning of the controller gains

Conclusions II

- Controller is implemented on youBot base due to the semantic mismatches between Simbody and KDL
- Cascaded PI controller in youBot manipulator runs in approximately 1 KHz
- Gravity compensation is achieved with model-based controller
- Model-based control scheme is validated by using analytical trajectories
- In spite of inaccurate model parameters, the proposed control scheme produces accurate tracking with some deviations

Future work

- Geometric relation semantics correction and identification
- The basic approach can be used once accurate model parameters are identified
- Safety controller has to be adapted to check joint constraints actively than defining a static value
- Tuning controller gains based on auto-tuning procedures
- Inclusion of D-term increases stability
- Controller's performance can be improved in a hard real-time operating systems

References I

[1] KUKA youBot developers, Dynamic robot model . *online at* *http://www.youbot-store.com/developers/kuka-youbot-kinematics-dynamics-and-3d-model-81, 2018.*

[2] Benjamin Keiser, Elias Müggler, Matthias Fässler, Davide Scaramuzza, Stephan Huck, and John Lygeros. Torque control of a kuka youbot arm. (2013) *source online at https://github.com/uzh-rpg/rpg_youbot_torque_control/tree/master/youbot_arm_model.*

[3] Chae H An, Christopher G Atkeson, and John M Hollerbach. Estimation of inertial parameters of rigid body links of manipulators. *In Decision and Control, 1985 24th IEEE Conference on, volume 24, pages 990 to 995. IEEE, 1985*

[4] Ruben Smits, H Bruyninckx, and E Aertbelien Kdl: Kinematics and dynamics library(2011). *online at http://www.orocos.org/kdl.*

References II

[5] Roy Featherstone. Rigid body dynamics algorithms. *Springer, 2014*.

[6] Jan Swevers, Walter Verdonck, and Joris De Schutter. Dynamic model identification for industrial robots. *IEEE Control Systems, 27(5):5871, 2007*

[7] Jeyaprakash Rajagopal. Dynamic robot model parameter identification via domain specific optimization. WS17 HBRS - Plöger, Schneider Supervising, *December, 2017*

[8] Sherman, Michael A and Seth, Ajay and Delp, Scott L. Simbody: multibody dynamics for biomedical research 2011. Procedia Iutam, *Online at https://simtk.org/projects/simbody*

[9] C. H. An, C. G. Atkeson, J. D. Griffiths, and J. M. Hollerbach. Experimental evaluation of feedforward and computed torque control. IEEE Transactions on Robotics and Automation, *5(3):368373, Jun 1989*

References III

[10] Olsson, Henrik and Åström, Karl Johan and De Wit, Carlos Canudas and Gäfvert, Magnus and Lischinsky, Pablo. Friction models and friction compensation, *Eur. J. Control, 4(3):176195, 1998*

[11] Ruiz, Miguel Corberán, Haptic Teleoperation of the youBot with friction compensation for the base, *PhD thesis, MA thesis. Departamento de Ingenier a de Sistema y Autom atica, Universidad Carlos III de Madrid, 2012. URL: http://hdl. handle. net/10016/16267 (cit. on p. 168), 2012*

[12] W. K. Chung, L.-C. Fu, and T. Kr oger. Motion Control, pages 163194. *Springer International Publishing, Cham, 2016.-*

[13] youBot developers. Dynamic robot model. *online at - https://www.youbot-store.com/developers/kuka-youbot-kinematics-dynamics-and-3d-model-81, Nov 2016.*

Questions???

The End