# VIBRATION CONTROL OF ELASTIC JOINT ROBOTS BY INVERSE DYNAMICS MODELS

Michael Thümmel, Martin Otter, Johann Bals
*DLR, Institute of Robotics and Mechatronics, Germany*

**Abstract**: In this article tracking control of nonlinear plants using a two degree of freedom controller structure is considered. The work described herein focuses on the design of feedforward controllers based on automatic generation of inverse plant models. The method is applied to the problem of vibration control of elastic joint robots and is demonstrated with simulations and experimental results.

**Key words:** inverse dynamics model, elastic joint robot, two degree of freedom controller, Modelica, Dymola

## 1. INTRODUCTION

This article addresses model based control of mechanical systems, in particular robots. When the tracking problem of such a plant is considered, the controller often includes some kind of non-linear inverse dynamics of the system. Depending on the controller, the inverse dynamics may be used in the feedback part, e.g. in feedback linearization, or in the feedforward part, e.g. in output regulation. For complex systems with fast dynamics like robots, computation of the inverse dynamics involves a large computational effort. In particular use of the inverse dynamics in the fast feedback loop often fails due to real-time requirements. Therefore, considering practical relevant robots, the inverse dynamics is used in the feedforward controller. Here sampling times can be larger and there are no hard real-time constraints.

In the following section a general framework is discussed to automatically derive a non-linear inverse model from a given plant model. This inverse model is used in the feedforward path of an appropriate two degree of freedom controller which is introduced in section 3. Section 4

highlights practical aspects of inverse model generation. In section 5, the explained technique is applied to a robot with elastic joints. Section 6 presents experimental results from our laboratory robot illustrating the reduction of vibrations at the tool. Section 7 concludes the paper.

## 2. DERIVATION OF INVERSE MODELS

The problems related to computation of non-linear inverse models shall be illustrated by calculation of the inverse dynamics of a simple robot having one elastic joint as shown below.

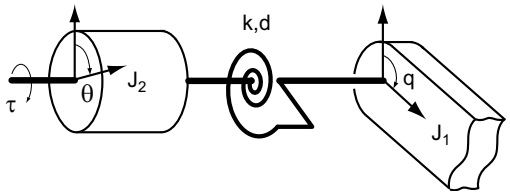


*Figure 1.* Robot with one elastic joint

The equations of motion are given by

$$\begin{aligned} 0 &= J_1\ddot{q} + k(q-\theta) + d\cdot(\dot{q}-\dot{\theta}) + g(q) \\ \tau &= J_2\ddot{\theta} - k(q-\theta) - d\cdot(\dot{q}-\dot{\theta}) \end{aligned} \tag{1}$$

with joint position q(t), motor position θ(τ), motor torque τ(τ), link inertia $J_1$, motor inertia $J_2$, a strictly increasing stiffness characteristics $k(.)\in C^1$ which is symmetric with respect to the origin, damping constant d and torque due to gravity g(q). To obtain the inverse dynamics, τ must be computed from a given q(t) and higher derivatives of q(t). In a first step, θ is calculated from the upper part of Eq. (1). It can be shown that this differential equation with state θ, which is known as the "internal dynamics" of the system, is stable and therefore the solution θ(t) can be further utilized. Note, since k(.) is a non-linear function, the solution θ(t) is usually calculated by numerical integration. In order to compute τ from the lower part of Eq. (1), the second derivative of θ must be calculated by differentiation of the first equation. This results in a new algebraic equation.

$$\ddot{\theta} = \frac{1}{d}\left( J_1 q^{(3)} + \frac{\partial k}{\partial(q-\theta)}\cdot(\dot{q}-\dot{\theta}) + \frac{\partial g}{\partial q}\dot{q} \right) + \ddot{q} \tag{2}$$

As illustrated with this simple example, the derivation of the inverse dynamics model may require to differentiate a subset of the equations, select appropriate states and solve the resulting system of differential and algebraic equations numerically. Further, derivatives of the desired trajectory must be provided up to a certain order. Below it will be shown how these tasks can be handled automatically using algorithms for solution of differential algebraic equations.

These algorithms start with a general model representation described by a set of differential and algebraic equations (= DAE)

$$\mathbf{0} = \mathbf{f}(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{y}, \mathbf{u}) \tag{3}$$

where $\mathbf{x}(t)$ are variables that appear differentiated in the DAE, $\mathbf{y}(t)$ are algebraic and $\mathbf{u}(t)$ are known input functions of time t. In order to solve Eq. (3) it is, at least numerically, possible to perform an index reduction step and transform the system to state space form

$$\begin{bmatrix} \dot{\mathbf{x}}_1 & \mathbf{x}_2 & \mathbf{y} & \mathbf{w} \end{bmatrix}^T = \mathbf{f}_S(\mathbf{x}_1, \mathbf{u}) . \tag{4}$$

Here $\mathbf{x}_1$ and $\mathbf{x}_2$ form the vector $\mathbf{x}$ such that the subset vector $\mathbf{x}_1$ is the *state vector* and contains the independent variables of $\mathbf{x}$. For example in multi-body systems, $\mathbf{x}_1$ are the minimal coordinates and $\mathbf{x}_2$ are the coordinates that are constrained. The vector $\mathbf{w}$ contains higher order derivatives of $\dot{\mathbf{x}}_1$, $\mathbf{x}_2$ and $\mathbf{y}$ that emerge when differentiating equations of $\mathbf{f}(.)$. They are treated as algebraic variables. The equations to be differentiated can be determined with the algorithm of Pantelides (1988). The selection of the state variables $\mathbf{x}_1$ can be performed with the "dummy derivative method" of Mattsson and Söderlind (1993). Both algorithms are, for example, available in the Modelica (Modelica Association 2005) simulation environment Dymola (Dynasim 2005). Computation of $\mathbf{f}_S(.)$ might include solution of linear and/or non-linear algebraic systems of equations.

In a plant model for controller design and simulation, the control variables are chosen as inputs and the variables that are required to track as outputs. In robotics this kind of model is known as forward dynamics.

Equivalently an *inverse* model of the plant can be constructed by exchanging the meaning of variables: A subset of the control variables, $\mathbf{u}_{inv}$, with dimension $n_{inv}$, is no longer treated as known but as *unknown*, and $n_{inv}$ previously unknown tracking variables are treated as *known* inputs. This kind of model is named inverse dynamics in robotics. It can still be written in form of Eq. (3) and handled with the same methods as any other DAE. Some more practical aspects of derivation of inverse models from plant models can be found in section 4.

Use of the inverse model in the controller as described in the next section is only possible if its DAE has a *unique solution* and if it is *stable*. Therefore the internal dynamics of the plant associated with the output to be tracked has to be stable. For linear systems this means, that the plant may not have unstable transmission zeros. For a general DAE no stability proof exists, but for certain classes of DAEs, including models of robots with elastic joints, it is possible to prove the stability of the inverse model. Since the internal dynamics is usually non autonomous, only boundedness of the internal states can be guaranteed. This can e.g. be analyzed using the notion of input-to-state stability from Sontag (1989). For the one link robot of Eq. (1), after a coordinate transformation $(\gamma,\delta)=(q-q_0,q-q_0-\theta)$ with $q_0$ such that $g(\gamma=0)=0$, one can show that the integral of $k(\delta)$ is an ISS-Lyapunov function for the inverse model. Therefore $\theta$ is bounded, provided $q(t) \in C^2$.

In case it is not possible to prove stability, simulations can be performed to check whether the inverse model is stable in the desired operation region. An alternative is to linearize the plant model around several stationary operating points and check whether the transmission zeros are stable. Of course, none of these checks can guarantee that the inverse DAE is stable.

If the inverse model is unstable, *approximate* inversion might be used for the controller. Therefore, since the internal dynamics is associated with the output, often an output redefinition approach is used. For linear single-input/single-output systems this can e.g. be achieved by removing unstable zeros before inverting the plant. For a non-linear DAE, one might choose other outputs that are similar to the original outputs, but don't lead to instability. Alternatively, the DAE might be modified before inversion, e.g., by introducing additional damping terms.

Since the transformation from Eq. (3) to Eq. (4) might differentiate a subset of the model equations, the known inputs of the model may be differentiated too. Therefore, the derivatives of these inputs must exist and must be provided up to a certain order. These derivatives can be computed, e.g., if the inputs are available as functions that can be differentiated sufficiently often. Alternatively, a desired reference model might be used that, in combination with the inverse DAE, results in a DAE that does not require derivatives of inputs. The reference model could, e.g., be selected as a filter such that a combination of the filter states yields the needed derivatives.

## 3. CONTROLLER USING AN INVERSE MODEL

The question arises how to use an inverse DAE model in a controller. Kreisselmeier (1999) has proposed and analyzed a controller structure with

two structural degrees of freedom for linear, single-input/single-output systems. A generalization of this controller structure to *nonlinear* multi-input/multi-output systems is shown in Fig. 2.

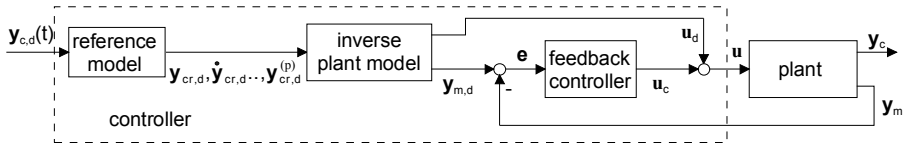


*Figure 2.* Controller with two structural degrees of freedom

In Fig. 2 the multi-input/multi-output plant has inputs $\mathbf{u}$, measured signals $\mathbf{y}_m$ and outputs $\mathbf{y}_c$ that are primarily controlled. The number of inputs must be identical to the number of controlled variables: $\dim(\mathbf{y}_c) = \dim(\mathbf{u})$. In this case the inverse plant model with (known) inputs $\mathbf{y}_c$ and unknown outputs $\mathbf{u}$ is used in the feedforward path of the controller to compute the desired actuator inputs $\mathbf{u}_d$ to the plant.

A "reference model" is used to provide a smooth trajectory $\mathbf{y}_{cr,d}$ and its derivatives, which are the inputs of the inverse plant model. The inverse plant model computes the desired measurement signals $\mathbf{y}_{m,d}$ and the desired plant inputs $\mathbf{u}_d$. For stabilization and robustness purposes a correction is added that is computed by a feedback controller that has the error between the measured signals and the desired measured signals as input. The feedback controller might be a simple state controller.

The *feedback controller* must be able to stabilize the system around the desired trajectory. Assuming the inverse plant model computes the same value for $\mathbf{y}_{m,d}$ as the measured quantity $\mathbf{y}_m$, then control error $\mathbf{e}$ is zero. Further, the feedback controller starts from a stationary state which is selected such that $\mathbf{u}_c=0$. Then $\mathbf{u}$ equals $\mathbf{u}_d$ which is computed in the inverse plant model. If the inverse model of the plant is ideal and starts at the same initial values as the plant, the plant will produce trajectories $\mathbf{y}_m(t)=\mathbf{y}_{m,d}(t)$ (because the inverse plant model was constructed in this way). Then again the preliminary assumption of zero control error $\mathbf{e}$ is fulfilled and the controller has no effect.

In case a control error occurs, the controller has to stabilize the system and cope with the imprecise inverse plant model. The proposed controller structure is advantageous because design of the feedback controller is decoupled from the feedforward controller that is responsible for tracking the given reference input.

Note, that due to the particular controller structure, the *reference model* determines essentially the input/output behaviour $\mathbf{y}_{c,d} \rightarrow \mathbf{y}_c$ of the overall system: According to the analysis above, the "feedback controller" in Fig. 2 has no effect for an ideal inverse plant model. In this case, the overall system

consists of a series connection of reference model, inverse plant model and plant model. Using the same assumptions as above, the output of the plant model is identical to the input of the inverse plant model. As a result, the reference model can be interpreted as the ”desired transfer function” of the overall system. Since the inverse plant model is in the feedforward path, the calculation of $\mathbf{u}_d$ and of $\mathbf{y}_{m,d}$ might be performed *offline*, so that hard real-time requirements for the solution of the inverse plant model are not present.

## 4. CONSTRUCTING INVERSE MODELS USING MODELICA AND DYMOLA

Deriving a non-linear inverse model manually can be a challenging task. For complex models this is not practical. Our approach is to use the Modelica modeling language (Modelica Association 2005) to define the model and the Modelica simulation environment Dymola (Dynasim 2005) to derive the inverse model automatically and perform simulations. The object-oriented modeling language Modelica is designed to allow convenient, component-oriented modeling of complex physical systems, e.g., systems containing mechanical, electrical, electronic, hydraulic, thermal, control, electric power or process-oriented subcomponents. The free Modelica language and libraries are available, ready-to-use and have already been utilized in demanding industrial applications. In Fig. 3, some example models of available Modelica libraries useful for robot simulation are shown.

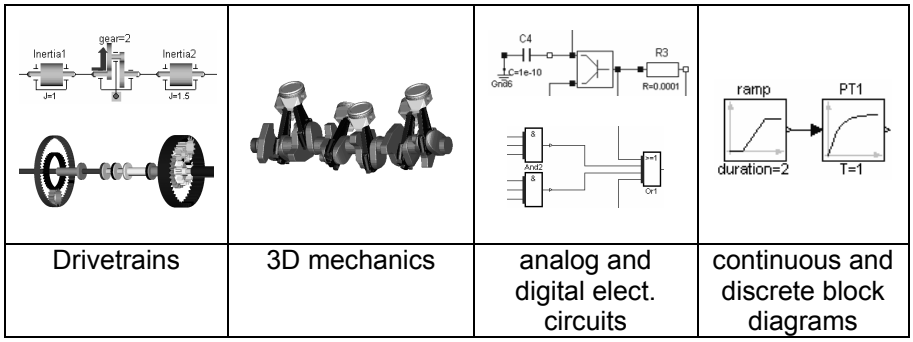


*Figure 3.* Modelica component libraries

With the Modelica simulation environment Dymola, the practical derivation of inverse models is straightforward, even for complex systems:

1. Define the plant model and include input and output signals of the plant over which the inversion shall take place.
2. If necessary, provide a reference model or input filter of appropriate relative degree. The relative degree may be known from physical

knowledge of the plant dynamics, or can be automatically derived as described below.

3. Connect the components using a twoInputs block as shown in Fig. 4. The twoInputs block is necessary because in block diagrams it is not allowed to connect two output signals with each other.

On the left side of Fig. 4 the plant model with one input and one output is present.

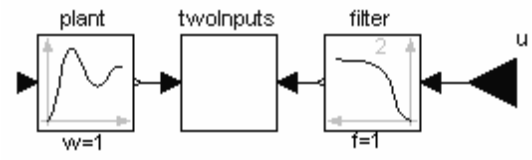


*Figure 4.* Definition of inverse model with Modelica

On the right side, a filter is used as reference model. If the filter order is too low the DAE is not causal and Dymola prints an error message of the following form:

Error: The model requires derivatives of some inputs as listed below:
Order of derivative Input
4 u1
2 u2
3 u3
Error: Failed to reduce the DAE index

In the second column the Modelica names of the input signals are listed that need to be differentiated according to the differentiation order of the first column. The numbers in the first column are therefore the minimum order of the corresponding filters. The higher the required filter order, the more problems usually occur when applying the inverse model in a control system. In such cases, one might simplify the plant model.

## 5. ELASTIC JOINT ROBOTS

The computation of the inverse dynamics for rigid robots is a standard task. Using the well-known equations of motion for tree-structured rigid multi-body systems

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} \tag{5}$$

and given the desired joint positions $\mathbf{q}(t)$ and their derivatives with respect to time up to a order of two, the generalized forces exerted by the drives $\boldsymbol{\tau}$ can be calculated algebraically. De Luca (1996) showed that, when more realistic models including drive dynamics and elasticity in the joints are taken into account the computation becomes more complex. Derivatives of the desired trajectory $\mathbf{q}(t)$ must be provided up to a certain order, because the equations of motion must be differentiated several times in order to calculate $\boldsymbol{\tau}$. The necessary number of derivatives of $\mathbf{q}(t)$ depends on the robot kinematics and cannot be given easily in closed form. An upper bound for an robot having N joints is 2(N+1) without damping in the joints, and 2+N, if damping is present in all joints. A first application of the proposed controller using an inverse dynamics model for elastic joint robots can be found in Thümmel (2001).

The framework presented in the previous sections is now applied to the DLR lightweight robot LBR2 shown in Fig. 5. This manipulator arm of approximately 1 m length is able to handle a payload of 8 kg with a total weight as low as 17 kg. It is a chain-structured mechanism with seven revolute joints. The motors are mounted in the joints, their rotors' axes of rotation coincide with the joint axes, as depicted in the right part of Fig. 5. The drive trains are known to be elastic due to harmonic drive reduction gears and torque sensors. They are modeled like in the example of section 2, with non-linear joint stiffness that are roughly $10^4$ Nm/rad and damping coefficients that are roughly 15 Nms/rad. A more detailed description of the LBR2 is given in Albu-Schäffer (2002). In Höpler (2004) the inverse dynamics problem for this robot was solved using spatial operator algebra and the reasons for the necessity of differentiation of the equations of motion were given as well.

*Figure 5.* DLR light weight robot LBR 2.
Left part: complete arm, right part: joint with rotor and drive

Starting from a detailed Modelica model of the LBR 2, the inverse dynamics model of the robot was automatically derived, as described before. It was found, that this model requires derivatives of the equations of motion up to an order of 4, what is undesirable for use in a controller. This higher

order results from gyroscopic effects introduced by the motors which are mounted on the robot arms. It is well known, that these effects decrease with increasing gear ratio. Taking into account the high gear ratios of the LBR2, it seems justified to neglect gyroscopic effects *of the motors*. Therefore, in a simplified model the motors are assumed to be not moving with the bodies to which they are attached but mounted on the ground. This significantly simplifies the model and reduces computation time since now the equations of motion only need to be differentiated once. A comparison of the models can be found in Tab. 1.

*Table 1.* Comparision of required differentiability

| Type of model | Detailed model | Simplified model |
|---|---|---|
| Required order of derivatives of the equations of motion | 4 | 1 |
| Required order of derivatives of q(t) | 6 | 3 |

In order to verify the assumption that gyroscopic effects of the motors can be neglected, simulations have been performed to check the errors introduced by the simplification of the model. In the example shown in Fig. 6 the desired trajectory for all joints was a step input from one to two radian filtered with a 8th order filter with cut-off frequency of 5Hz.

As can be seen, the errors with the simple model are roughly one percent of the overall torque and much smaller than the errors of a rigid model where the elasticity in the joints is neglected. Therefore the simple model is a good approximation for control purposes.

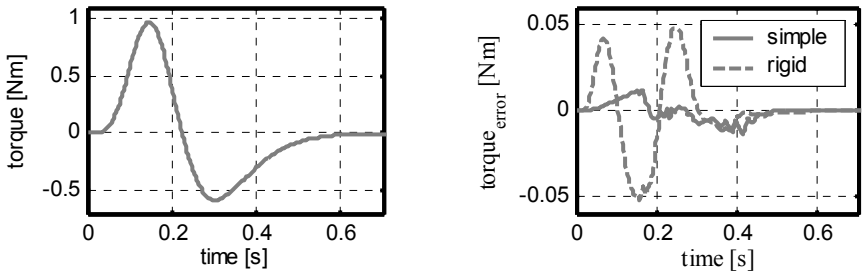


*Figure 6.* Torques computed with inverse dynamics models
Left part: torque for detailed model. Right Part: torque errors with respect to detailed model

## 6. EXPERIMENTAL RESULTS

In order to check the performance of the proposed controller, tests with our laboratory robot have been carried out using the controller structure

shown in Fig. 2. The feedback controller was chosen to be a PD controller for every joint and a low-pass filter was used as a reference model.

On the left hand side of Fig. 7 the tool vibrations after a robot motion are shown. The motion was executed two times: one time without feedforward controller (PD) and one time with the inverse dynamics model in the feedforward part (PD+inv.). As can be seen the inverse dynamics model reduces vibrations for positioning tasks.

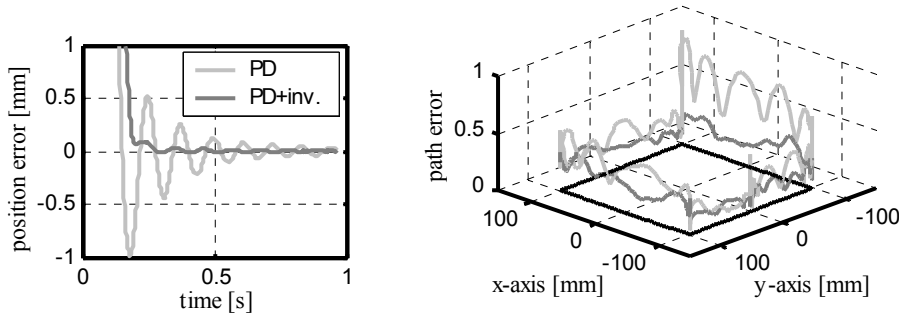


*Figure 7.* Comparison of control errors. Bright line: controller without feedforward. Dark line: controller using inverse dynamics model
Left part: positioning experiment. Right part: tracking experiment

On the right hand side of Fig. 7 the results of a trajectory tracking experiment are shown. The desired trajectory was chosen to be a square with start point [-100, 0, 0] mm traversed in clockwise direction as indicated by the black line. Again this movement was executed with and without the inverse dynamics model in the feedforward part and measured with an optical coordinate measurement system. The figure shows normalized absolute deviations from the desired path displayed on the z axis. As can be seen also the tracking error can be reduced significantly when using the inverse dynamics model.

## 7. CONCLUSION

Vibration control of elastic joint robots with automatically generated inverse dynamics models in the feedforward controller was considered. It has been shown, that inverse models of general plants can be derived automatically using existing algorithms for DAE systems. The method was illustrated by simulations with a model of the LBR2, and a simplified model for use in a controller was derived. As shown in experiments, the proposed

controller can reduce robot vibrations in positioning tasks as well as tracking errors significantly.

## 8. ACKNOWLEDGEMENT

This work was in parts supported by „BMBF Verbundprojekt PAPAS“ (Plug And Play Antriebs- und Steuerungskonzepte für die Produktion von Morgen) under contract number 02PH2060.

## 9. REFERENCES

Albu-Schäffer, A., 2002, Regelung von Robotern mit elastischen Gelenken am Beispiel der DLR-Leichtbauarme,. *Ph.D. dissertation*, Technische Universität München.

De Luca, A. and Tomei, P., 1996, Elastic joints, in *Theory of robot control*, de Wit, C. C., Siciliano, B. and Bastin, G., ed., Springer Verlag, Berlin, Heidelberg, New York, Tokyo, pp. 179-217.

Dynasim, 2005, Dymola – Users Manual; http://www.dynasim.com.

Höpler, R. and Thümmel, M., 2004, Symbolic Computation of the Inverse Dynamics of Elastic Joint Robots, *Proc. of the 2004 IEEE Intern. Conference on Robotics and Automation* pp 4314-4319.

Kreisselmeier, G., 1999, Struktur mit zwei Freiheitsgraden. *Automatisierungstechnik* **6**:266-269.

Mattsson, S.E. and Söderlind, G., 1993, Index reduction in differential-algebraic equations using dummy derivatives, *SIAM Journal of Scientific and Statistical Computing* **14**:677-692.

Modelica Association, 2005, Modelica™ - A Unified Object-Oriented Language for Physical Systems Modeling. Tutorial,Version 1.4; http://www.modelica.org.

Pantelides, C.C., 1988, The consistent Initialization of differential-algebraic Systems, *SIAM Journal of Scientific and Statistical Computing* **9**:213-231.

Sontag, E. D, 1989, Smooth stabilization implies coprime factorization, *IEEE Trans. Aut. Control **34**:435-443.*

Thümmel, M., Otter, M., Bals, J., 2001, Control of Robots with Elastic Joints based on Automatic Generation of Inverse Dynamics Models, *Proc. of 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems* pp. 925-930.