Dynamic simulation of *n*-R planar manipulators

Leon Žlajpah[a]

[a]Jožef Stefan Institute, Jamova 39, 61000 Ljubljana, Slovenia

In the paper a simulation package for dynamic simulation of *n*-DOF planar manipulators with revolute joints is presented. Numerical effective equations to solve the direct kinematics of *n*-DOF manipulator are developed. The dynamic model based on Lagrangian equation is derived. The simulation package is implemented in MATLAB. It consists of several M-files for the calculation of the model, utility functions and the user interface. The graphical user interface provides easy selection of main simulation parameters and different simulation tasks. For illustration, simulation examples of trajectory generation and dynamic control in task space are presented.

1. INTRODUCTION

In practice, a variety of tasks require sophisticated motion and so redundant manipulators get more and more important due to their flexibility. Most of the study of redundant mechanisms has been done without consideration of any particular mechanism [1,2]. A theoretical approach to the control problem does usually incorporate a dynamic model of the mechanism and also pure kinematic redundancy resolution schemes often use dynamic models or its parts in performance criterion. Some authors use in their analysis mechanical systems with only one degree of redundancy. Actually, they use simple 3-DOF planar mechanism. The reason is the complexity of models of the mechanisms, which increase very fast with the number of DOF. For evaluation of theoretical results it would be of benefit to have dynamic models of mechanisms with more than one degree of redundancy.

In this paper we present a simulation package for dynamic simulation of *n*-DOF planar manipulators with revolute joints — *n*-R manipulator. First, numerical effective equations to solve the direct kinematics of *n*-DOF manipulator are developed. In algorithm recursive equations are used having the benefit of partial results being utilized for further calculations. Next, the dynamic model based on Lagrangian equation is derived. In the next part a simulation system for *n*-R manipulator based on the derived dynamic model is presented. The simulation is implemented in MATLAB. It consists of M-files for calculation of the model and utility functions and scripts for the user interface. The basic M-file functions are for the calculation of the kinematics and dynamics of *n*-R manipulators. Beside these functions, some additional M-files have been developed for different control algorithm, task definition, etc. The graphical user interface provides easy selection of main simulation parameters and selection of different simulation tasks. The simulation results are presented simultaneously as an animation of the motion of the manipulator or can be analyzed after the simulation. For illustration some simulation results of trajectory generation and dynamic control in task space are presented.

2. MATHEMATICAL MODEL

In the following subsections we give the kinematic and the dynamic model of the n-DOF planar manipulator with revolute joints. The manipulator is supposed to move in the vertical plane x-y and the task coordinates $\mathbf{x}$ are the positions in x-y plane, $\mathbf{x} = [x, y]^T$.

2.1. Kinematics of n-R planar manipulator

With respect to n joint coordinates $\mathbf{q}$ and m task coordinates $\mathbf{x}$ the kinematics of the manipulator can be described with the following equations [3]

$$\mathbf{x} = \mathbf{p}(\mathbf{q}) \qquad \dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \qquad \ddot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} \tag{1}$$

where $\mathbf{x}$ is m-dimensional vector of task positions, $\mathbf{p}$ is m-dimensional vector function representing direct kinematics, $\mathbf{J}$ is the Jacobian matrix and $\dot{\mathbf{J}}$ is its time derivative, $\dot{\mathbf{J}} = \mathrm{d}\mathbf{J}/\mathrm{d}t$. As we deal with redundant manipulators, $m = 2$ and $n > m$.

For the planar manipulator with revolute joints the end effector positions $\mathbf{x}$, $\mathbf{x} = [x_1, y_1]^T$, can be expressed by the following recursive equations [4]

$$\varphi_i = \varphi_{i-1} + q_i, \qquad i = 1, \ldots, n \tag{2}$$

$$x_i = x_{i+1} + l_i \cos(\varphi_i), \qquad y_i = y_{i+1} + l_i \sin(\varphi_i), \qquad i = n, \ldots, 1 \tag{3}$$

where l_i is the length of the i-th link and $\varphi_0 = 0$, $x_{n+1} = y_{n+1} = 0$. From (3) it follows that

$$\frac{\partial x_i}{\partial q_j} = -y_k, \qquad \frac{\partial y_i}{\partial q_j} = x_k, \qquad k = \max(i, j) \tag{4}$$

Hence, the components of $\mathbf{J}$ can be specified without any further calculations

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial x_1}{\partial q_1} & \cdots & \dfrac{\partial x_1}{\partial q_n} \\ \dfrac{\partial y_1}{\partial q_1} & \cdots & \dfrac{\partial y_1}{\partial q_n} \end{bmatrix} = \begin{bmatrix} -y_1 & \cdots & -y_n \\ x_1 & \cdots & x_n \end{bmatrix} \tag{5}$$

To obtain $\dot{\mathbf{J}}$ we have to differentiate $\mathbf{J}$ with respect to time, $\dot{\mathbf{J}} = \sum_{k=1}^{n} \left(\frac{\partial \mathbf{J}}{\partial q_k} \dot{q}_k \right)$. In the calculation, the following relations are used

$$\frac{\partial^2 x_i}{\partial q_j \partial q_k} = \begin{cases} -x_r, & j \le i \quad \text{and} \quad k \le i \\ 0, & j > i \quad \text{or} \quad k > i \end{cases} \qquad \frac{\partial^2 y_i}{\partial q_j \partial q_k} = \begin{cases} -y_r, & j \le i \quad \text{and} \quad k \le i \\ 0, & j > i \quad \text{or} \quad k > i \end{cases} \tag{6}$$

where $r = \max(j, k)$.

2.2. Dynamics of n-R planar manipulator

Using the Lagrangian formulation the dynamic model of the rigid manipulator can be given in a form

$$\tau = \mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{B}\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) \tag{7}$$

where $\mathbf{H}$ is the inertia matrix, $\mathbf{h}$ the vector of Coriolis, centrifugal forces, $\mathbf{B}$ the viscous friction matrix and $\mathbf{g}$ is the vector of gravity forces.

The inertia matrix $\mathbf{H}$ can be obtained using the following equation [3]

$$\mathbf{H} = \sum_{i=1}^{n} \left(m_i \mathbf{J}_L^{(i)T} \mathbf{J}_L^{(i)} + \mathbf{J}_A^{(i)T} \mathbf{I}_i \mathbf{J}_A^{(i)} \right) \tag{8}$$

where $\mathbf{J}_L^{(i)}$ and $\mathbf{J}_A^{(i)}$ are the Jacobian matrices for the center of mass of the i-th link associated with linear and angular task velocities, respectively. The position of the center of mass of the i-th link is expressed as

$$\mathbf{x}_{c,i} = [x_1 - x_i + l_{c,i}\cos(\varphi_i), y_1 - y_i + l_{c,i}\sin(\varphi_i)]^T \tag{9}$$

where $l_{c,i}$ is the length between the i-th joint and the center of mass of the i-th link. Differentiation of (9) with respect to $\mathbf{q}$ results in

$$\begin{array}{l} \dfrac{\partial x_{c,i}}{\partial q_j} = \begin{cases} -y_j + y_i - l_{c,i}\sin(\varphi_i), & j \le i \\ 0, & j > i \end{cases} \\ \dfrac{\partial y_{c,i}}{\partial q_j} = \begin{cases} x_j - x_i + l_{c,i}\cos(\varphi_i), & j \le i \\ 0, & j > i \end{cases} \end{array} \quad \text{and} \quad \mathbf{J}_L^{(i)} = \begin{bmatrix} \dfrac{\partial x_{c,i}}{\partial q_1} & \cdots & \dfrac{\partial x_{c,i}}{\partial q_n} \\ \dfrac{\partial y_{c,i}}{\partial q_1} & \cdots & \dfrac{\partial y_{c,i}}{\partial q_n} \end{bmatrix} \tag{10}$$

Note that in the case of planar manipulators with revolute joints the terms $\mathbf{J}_A^{(i)T} \mathbf{I}_i \mathbf{J}_A^{(i)}$ can be simplified and rewriten into the form

$$\mathbf{J}_A^{(i)T} \mathbf{I}_i \mathbf{J}_A^{(i)} = I_i \begin{bmatrix} \mathbf{1}_{i\times i} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}_{n\times n} \tag{11}$$

where I_i is the moment of inertia of the i-th link about the center of mass. Matrices $\mathbf{1}$ and $\mathbf{0}$ are matrices of ones and zeros of corresponding dimensions, respectively.

Next, the vector of Coriolis and centrifugal forces $\mathbf{h}$ with components

$$h_i = \sum_{j=1}^{n} \sum_{j=1}^{n} h_{ijk} \dot{q}_j \dot{q}_k, \qquad h_{ijk} = \frac{\partial H_{ij}}{\partial q_k} - \frac{1}{2} \frac{\partial H_{jk}}{\partial q_i} \tag{12}$$

is determined. In the calculation the derivatives of $\mathbf{J}_L^{(i)}$ are used (see [3])

$$\frac{\partial \mathbf{J}_L^{(i)}}{\partial q_k} = \begin{bmatrix} \dfrac{\partial^2 x_{c,i}}{\partial q_1 \partial q_k} & \cdots & \dfrac{\partial^2 x_{c,i}}{\partial q_n \partial q_k} \\ \dfrac{\partial^2 y_{c,i}}{\partial q_1 \partial q_k} & \cdots & \dfrac{\partial^2 y_{c,i}}{\partial q_n \partial q_k} \end{bmatrix} \tag{13}$$

From (10) it follows

$$\begin{array}{l} \dfrac{\partial^2 x_{c,i}}{\partial q_j \partial q_k} = \begin{cases} -x_r + x_i - l_{c,i}\cos(\varphi_i), & j \le i \;\text{ and }\; k \le i \\ 0, & j > i \;\text{ or }\; k > i \end{cases} \\ \dfrac{\partial^2 y_{c,i}}{\partial q_j \partial q_k} = \begin{cases} -y_r + y_i - l_{c,i}\sin(\varphi_i), & j \le i \;\text{ and }\; k \le i \\ 0, & j > i \;\text{ or }\; k > i \end{cases} \end{array} \qquad r = \max(j,k) \tag{14}$$

The gravity forces of the n-DOF planar robot $\mathbf{g}$ can also be expressed recursively. The components g_i are computed by

$$\begin{array}{l} g_n = 9.81\, m_n l_{c,n} \cos(\varphi_n) \\ g_i = g_{i+1} + 9.81 \left(m_i l_{c,i} \cos(\varphi_i) + \displaystyle\sum_{k=i}^{n} m_k\, l_i \cos(\varphi_i) \right) \qquad i = n-1, \ldots, 1 \end{array} \tag{15}$$

Table 1
Execution time and number of operations versus number of DOF (PC-Pentium/90MHz)

	`dynmodel`		`kinmodel`	
n	Exec. time (ms)	No. of FPO	Exec. time (ms)	No. of FPO
2	18	422	5	74
3	30	1426	9	152
4	46	3694	12	258
5	67	8054	17	392
10	271	105614	55	1482
20	1543	1560734	197	5762

3. IMPLEMENTATION IN MATLAB

In this section we describe a software package for simulation of n-DOF planar manipulators. The package is implemented in MATLAB as M-Files. MATLAB was selected mainly due to its capabilities of solving problems with matrix formulations, easy extensibility and because of variety of tools for the graphic representation and the user interface.

The kinematic and dynamic model represent the kernel of the simulation system. The corresponding M-Files are

- `[x,J,Jd]=kinmodel(q,L,qd)` representing direct kinematics
- `[x,J,Jd,H,h,g]=dynmodel(q,qd,nj,L,Lc,m,ml,I,B)`]representing the complete model (kinematic and dynamic).

As described in the previous section most of the equations of the model are in the form of loops. To obtain the most speed of MATLAB all equations representing the kinematic model and some equations of the dynamic model have been vectorized, e.g., equations (2) and (3) have been transformed into the form

```
fi=(robll*q)'; cL=L.*cos(fi); sL=L.*sin(fi);
x=cl*robll; y=sl*robll; X=[x(1),y(1)];
```

where `robll` is a lower-left triangular matrix of ones. Table 1 shows the execution time and number of floating point operations (FPO) versus number of DOF of the manipulator for both main functions. We can see that the execution time for calculation of the kinematic model is increasing with the number of DOF as $O(n^2)$ and the complete model as $O(n^3)$. The numerical complexity (No. of FPO) is little higher due to additional operations contributed by the vectorization.

The model functions are used in main simulation routines, which enable simulation of dynamic and kinematic close-loop control systems with variety of control schemes common in the analysis and control of redundant manipulators [2,1]. For that purpose, M-files have been developed for different control algorithms and task specification. To make the usage of the package as easy as possible a graphical user interface has been built based on different MATLAB *Uicontrol* objects. Fig. 1 shows the main simulation window which enables the specification of the manipulator, task and control parameters. It also controls the simulation run and simultaneously shows the animation of the resulting motion in task plane.

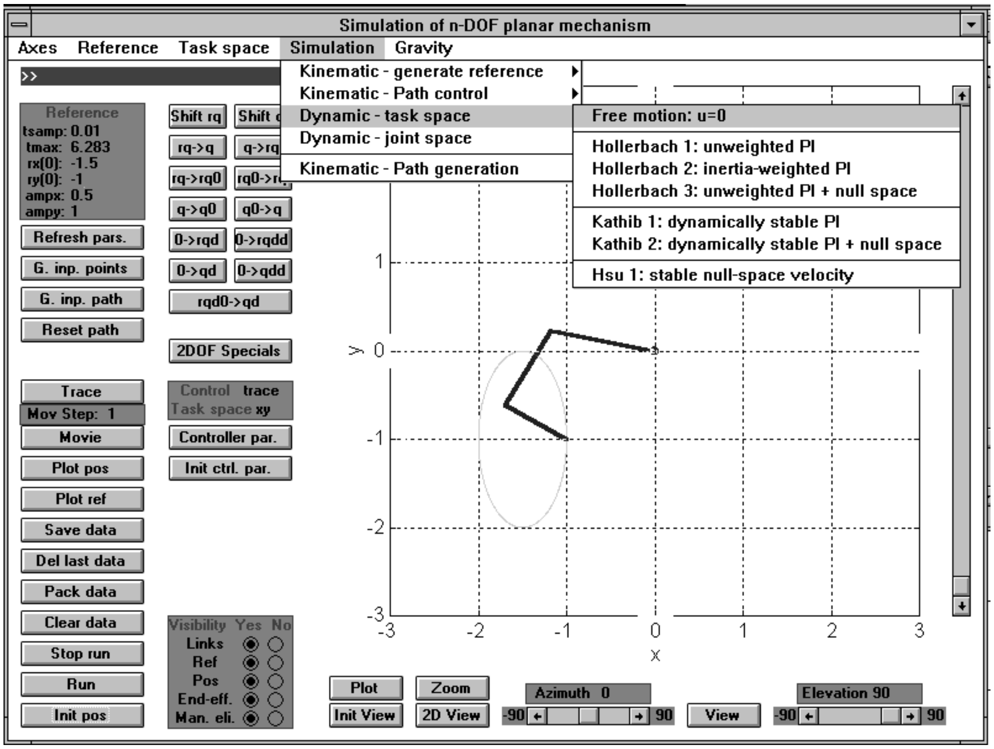

Figure 1. User interface

4. SIMULATION EXAMPLES

For illustration we have chosen two examples. The first example is a dynamic simulation of a 5-R manipulator moving along a straight line with the control

$$\tau = \mathbf{H}\mathbf{J}^{\dagger}(\ddot{\mathbf{x}}_r + \mathbf{K}_v(\dot{\mathbf{x}}_r - \dot{\mathbf{x}}) + \mathbf{K}_p(\mathbf{x}_r - \mathbf{x}) - \dot{\mathbf{J}}\dot{\mathbf{q}}) + \mathbf{h} + \mathbf{B}\dot{\mathbf{q}} + \mathbf{g} \tag{16}$$

where $\mathbf{J}^{\dagger}$ the pseudoinverse of $\mathbf{J}$ and $\mathbf{K}_p$, $\mathbf{K}_v$ are the close-loop gains. Fig. 2 shows the manipulator motion in x-y plane and control signals versus time. The second example deals with the trajectory generation for 4-R manipulator. The task is to generate joint trajectories for a given path defined by points and splines between points. Additionally, the path velocity should be constant. The results are shown in Fig. 3. Applications of the simulation system to other problems are presented in [5], where also some additional functions for calculation of different performance criteria for n-R planar manipulators are given.

5. CONCLUSION

In the paper, a simulation package for dynamic simulation of n-R planar manipulators with revolute joints is presented. Numerical effective equations to solve the direct kinematics and dynamics of n-R manipulator are developed. Their basic benefit is to utilize partial results in further calculations. The simulation is implemented in MATLAB. The package consist of several M-files for calculation of the model, utility functions and the user interface. The graphical user interface provides easy selection of main simulation parameters and selection of different simulation tasks. The package can be used for many purposes: kinematic simulation, dynamic

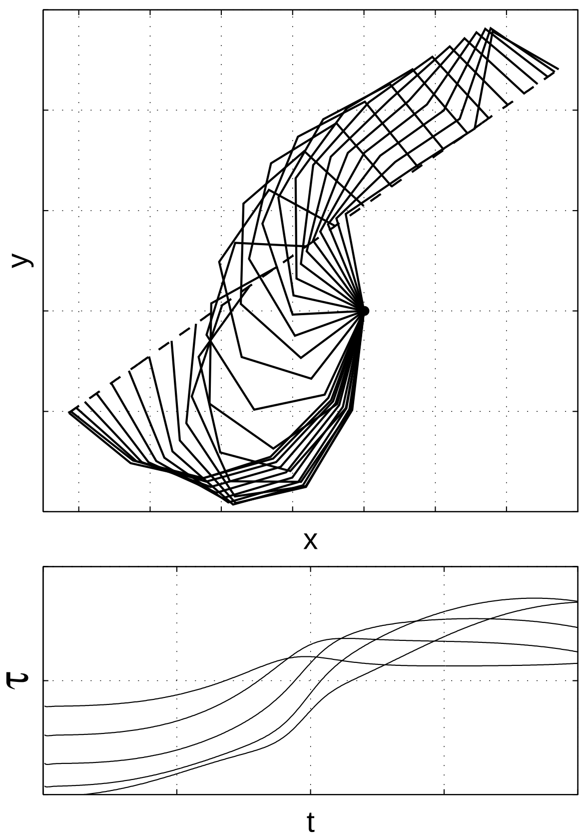

Figure 2. Pseudoinverse control of 5-R manipulator

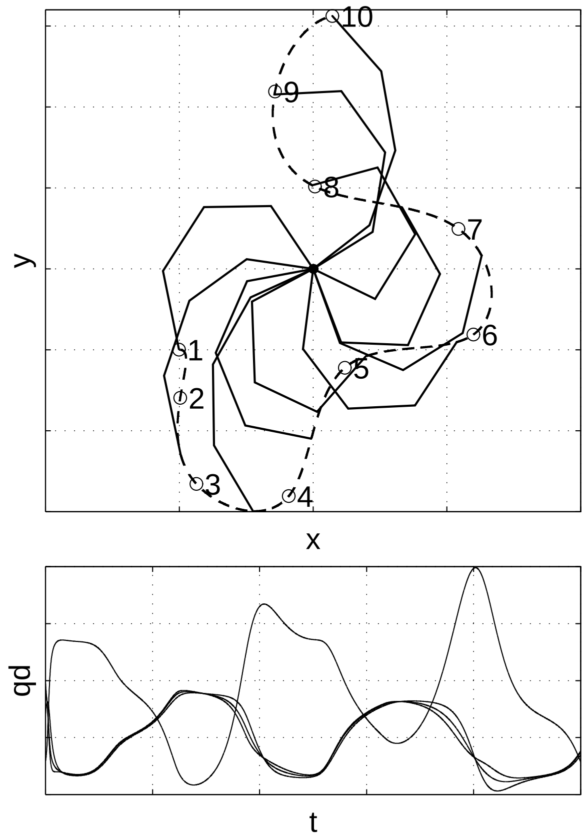

Figure 3. Trajectory generation for 4-R manipulator

simulation, analysis and synthesis of control systems, trajectory generation, etc.

The derived simulation system has proved to be a very useful and effective tool for testing different control algorithms and for motion planing of redundant manipulators. As the complexity of the model is increasing slower with the number of DOF as in the case of general manipulators, the derived system can be used for the simulation of manipulators with many DOF. The simulation package is very easy to extend and to adapt to different requirements.

REFERENCES

1. C. A. Klein and C. H. Huang. Review of pseudoinverse control for use with kinematically redundant manipulators. *IEEE Trans. on Systems, Man, Cyb.*, SMC-13(3):245 – 250, 1983.
2. D. N. Nenchev. Redundancy resolution trough local optimization: A review. *J. of Robotic Systems*, 6(6):769 – 798, 1989.
3. H. Asada and J.-J. E. Slotine. *Robot Analysis and Control*. John Wiley & Sons, 1986.
4. J. Lenarčič. Optimum configurations of planar *n*-r hyper-redundant mechanism. In *Int. Conf. on Advanced Robotics ICAR '93*, Tokyo, 1993.
5. L. Žlajpah and J. Lenarčič. On Line Minimum Joint Torque Motion Generation for Redundant Manipulators. In *25th Int. Symp. on Ind. Robots*, 757 – 762, Hannover, 1994.