Proceedings of the RAAD 2009
18th International Workshop on Robotics in Alpe-Adria-Danube Region
May 25-27, 2009, Brasov, Romania

An Approach to Solving Kinematics Models and Motion Planning for Manipulators with Mobile Base

Mihai Duguleana[a]

[a] *Department of Robotics, Transylvania University, Romania*
E-mail: mihai.duguleana@unitbv.ro
URL: www.unitbv.ro

Abstract. In this paper it is presented an up-to-date review of different approaches to solving inverse kinematics problem for 6 DOF manipulators. There are also briefly described the most common motion planning algorithms. Furthermore, we are proposing a simple particular solution using a combination between the Reach Hierarchy and the standard analytical solution for Powerbot Mobile Robot equipped with a 6 DOF PowerCube robotic arm. We conclude by listing research issues and further development directions.

Keywords. Forward Kinematics, Inverse Kinematics, Motion Planning, Redundant Manipulators, 6 DOF

1. Introduction

Kinematics problem is one of the most discussed issues in fields like Robotics and Computer Animation. There are 2 different types of Kinematics: Forward Kinematics and Inverse Kinematics. These are widely used for synthesizing the motion of linked bodies (also called kinematic chains). Such structures are any robotic arms, generally formed by linked bodies kept together by joints.

There are 6 possible types of joints: prismatic, revolute, screw, cylindrical, planar and spherical. Each prismatic, screw and revolute joint introduces 1 degree of freedom (DOF) to the structure while a cylindrical joint introduces 2 DOF and a spherical or planar joint introduces 3 DOF. If a robotic arm has 6 DOF, it can reach any point in 3D its working space. This perfectly constrained structure is also known as the holonomic arm.

If a link structure can reach the same position of the end-effector by several different intermediate positions of the bodies kinematic from the chain, it is called redundant. Redundancy can be used to meet secondary tasks such as obstacle avoidance, satisfying joint limits, singularities avoidance or variable optimization (i.e. torque optimization, dexterity optimization, energy saving optimization).

While kinematics handles joint angles and Cartesian coordinates positioning, the motion itself is handled by path planning algorithms (Zoppi, 2002). The most used types of motion planning algorithms are sample-based motion planning and combinatorial motion planning. The main idea for the first type of algorithm is to apply a sampling scheme to solution space, while combinatorial approaches to motion planning find paths through the continuous configuration space without resorting to approximations (La Valle, 2006). The main issue that is handled by sampling algorithms is the transformation of the infinite space of solutions (resulted from the infinite sampling loop) to a finite space of solutions (resulted from early closing).

There are also other types of algorithms that seem slightly adequate for PowerCube arm (Bertram et al., 2006). Looking at the problem from the path planning point of view, the probabilistic roadmap planner is one of the most used solutions. This approach searches for the shortest path between the initial given configuration and the goal configuration on a pre-computed graph. The graph is usually generated in a collision-free space. Rapid-exploring random trees are also used to generate a graph that always offers the optimal path; however because the trees explore the joint space uniformly, the algorithm exhibits rather slow convergence (Kopicki, 2007).

Other approaches presume cognition. Robot learning options are desired in tasks that are performed in fast evolving environments. Motion can be assisted by neural networks which are trained to offer the optimal trajectories depending on several parameters (such as load weight, shortest path, smallest energy consume and others) (Oiama et al., 2005). Some of these approaches also use sampling algorithms in order to make decisions.

The goal of this paper is to present a simple yet effective arm solution that will enable us to gain better control over a robotic manipulator mounted on a mobile robot.

2. Physical system structure

The work presented in this paper is conducted on Powerbot robot from Active Robots (Fig. 1). Powerbot comes equipped with a 6 DOF robotic arm called PowerCube, produced by AMTEC, a subdivision of Schunk. Active Robots provide an API for basic control over the arm joints via CANBUS interface, a wrapper of the libraries developed by AMTEC. Using PowerCube signals manual, these classes are further extended into a control driver.

The arm is driven by seven 24v DC motors and it can reach up to 90 cm from the center of its rotating base to the tip of its closed gripper. The arm can be reconfigured for many different work situations, but the initial configuration seems to have a good correspondent in real life. As seen in Fig. 2, the joints include a rotating base, a pivotating shoulder, two rotating links, a pivotating elbow, a pivotating wrist and a 1 DOF gripper that can grasp objects 6 cm wide. All the joints except gripper are rotary. The arm can lift weights up to 2 kg and each joint encoder offers current position and speed control capabilities on every motor.

The control of the arm is made by setting each joint angle at the desired value. This type of interaction is not useful for anything beyond a very basic demonstration. Still, a great function provided by Active Robots in their API is armPark(). Using this procedure at the end of each application written in C/C++, the PowerCube can be safely parked to a protected configuration.

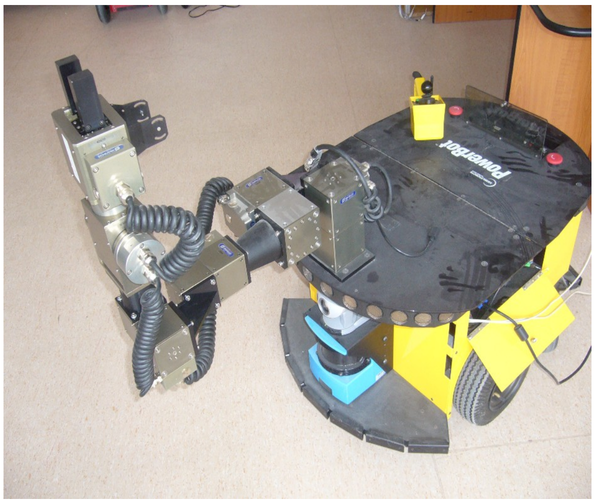
Fig. 1. Powerbot with PowerCube arm

In order to use the manipulator for higher-level projects, it was decided to implement an interface that will allow solving forward and inverse

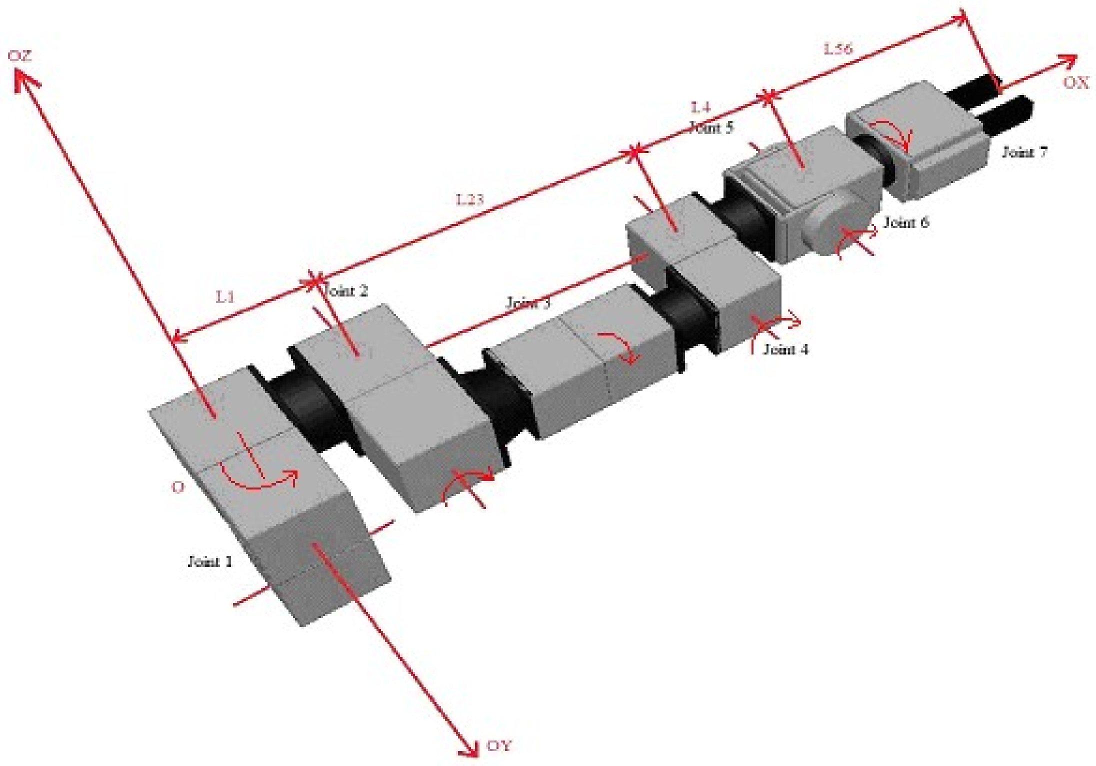

Fig. 2. Modelling and setting up the main coordinate frame

kinematics problems while avoiding collisions and singularities, thus providing control in Cartesian coordinates.

3. Forward kinematics

Forward kinematics (FK) explicitly resolves the position and orientation of each segment in Cartesian coordinates at a specific time using joint angles as input, as seen in Eq. (1).

$$F_k(\phi_{1-n}) = (X;Y;Z)_{1-n} \quad (1)$$

Given the angles Φ for each of the 1-n joints, Fk provides the Cartesian coordinates (X;Y;Z).

For most robotic structures the main goal of FK is to determine the position and orientation of the last link (the end-effector).

A commonly used convention for selecting frames of reference in robotic applications is the Denavit-Hartenberg (D-H). The basic idea behind D-H algorithm is to assign coordinate frames to each joint from the chain (Spong et al., 2005). Each new frame provides a new homogeneous transformation matrix. While applying D-H for Powercube arm, some conventions specific to our case have been made. One major approximation which deeply simplifies our problem is the suppressing of joint 3 movements. Joint 3 can be safely blocked if it can be proved that the solution space remains the same. As our manipulator is redundant and its base is mobile, it is trivial that each solution is available to the constrained link chain (as the mobile property of the manipulator's base introduces 2 more DOF – by suppressing 1 DOF, PowerCube still remains redundant).

Another factor for this particular link structure is the small length of L1, which is only 13 cm. Being the only rotary joint after OZ axis, this characteristic enables a better formed solution space and reduces singularity problems.

While fixed manipulators need redundancy for secondary tasks like obstacle avoidance or trajectory optimization, a mobile manipulator can always change its position, thus changing the origin point of the main chosen coordinate frame. Our approach considers providing the robot possibly movement instructions for cases where targets are out of reach, for cases where we have obstacles or for better grasping motions with fixed orientation and positioning of the last body of the kinematic chain. Later in this article it will be presented an optimal end-effector region that is used in cases where no solution is found.

Another convention that has been made is fixing the main coordinate frame as seen in Fig. 2. The origin O has been chosen in the middle of the first joint, just over the rotary base. Axis OX has been chosen along the straight line given by the zero angle position for each joint.

The links 2 and 3 are merged into L23 and the links 5 and 6 are merged into L56 (where L1 = 13cm, L23 = 36 cm, L4 = 18 cm, L56 = 26 cm).

The resulting coordinates for the end-effector (the point is placed in the center of the gripper) are calculated as the product between all the homogenous matrices resulted from D-H table (see Eq. (2), Eq (3) and Eq. (4)).

The D-H table for this link structure is presented in Tab. 1.

Tab. 1. D-H table for PowerCube

Link	a_i	α_i	d_i	Φ_i
1	L_1	90°	0	Φ_1*
23	L_{23}	0	0	Φ_2*
4	L_4	0	0	Φ_4*
56	L_{56}	0	0	Φ_5*

$$EE_x = \cos\phi_1 \cdot [L_1 + L_{23}\cdot\cos\phi_2 + L_4\cdot\cos(\phi_2+\phi_4) + L_{56}\cdot\cos(\phi_2+\phi_4+\phi_5)] \quad (2)$$

$$EE_y = \sin\phi_1 \cdot [L_1 + L_{23}\cdot\cos\phi_2 + L_4\cdot\cos(\phi_2+\phi_4) + L_{56}\cdot\cos(\phi_2+\phi_4+\phi_5)] \quad (3)$$

$$EE_z = L_{23}\cdot\sin\phi_2 + L_4\cdot\sin(\phi_2+\phi_4) + L_{56}\cdot\sin(\phi_2+\phi_4+\phi_5) \quad (4)$$

4. Inverse kinematics and motion planning

Opposite to FK that computes the position of the end-effector, inverse kinematics (IK) provides direct control over the position of the end-effector by solving each of the joint angles from the kinematic chain (Kang, 2000).

$$Ik(X;Y;Z) = Fk^{-1}(X;Y;Z) = (\phi_{1-n}) \quad (5)$$

The Eq. (5) doesn't have a unique solution in most cases, since Fk may not have an inverse.

There are 2 different approaches to IK problem: analytical and numerical. Analytical methods attempt to give an exact solution by directly inverting the FK equations. This is only possible on relatively simple chains. Numerical methods use approximation and iteration to converge to a solution. These tend to have a more general purpose but require most of the times computational resources.

The most used numerical method for solving IK is based on Jacobian matrices (Rotenberg, 2005). The main idea that stands behind this approach is that a Jacobian matrix contains all the information necessary to relate a change in any component of x to a change in any other component of the Fk function we wish to invert. The approximation which is usually made is presented in Eq. (6), where ΔEE is the desired incremental change of the end-effector position.

$$\Delta\phi = J^{-1} \cdot \Delta EE \quad (6)$$

All it needs to be done now is to pick a step and iterate the equation on a loop until EE reaches the goal or a position which is fairly satisfying near the goal.

Variations on this technique rely on constructing J^{-1}; most used are the pseudo-inverse $(J^{+})^{-1}$ and the transposed $(J^{T})^{-1}$.

Another IK numerical approach is the Lagrangian method. This one tries to extend an underconstrained redundant system to a perfectly constrained one using Lagrange multipliers (Kang, 2000).

Another interesting IK numerical method is called Reach Hierarchy (RH). The main idea behind this algorithm is that each subset of links has a working environment. RH relies on pre-computed workspaces for the kinematic chain, for each link subset of the kinematic chain and for each link body in particular (Kang, 2000).

For the specific case of the PowerCube arm, we propose a solution based on a combination between the analytical approach and RH. This approach ensures fast computation times, lowers self-collision opportunities and can be easily implemented because it is clearly separated into logic cases.

RH will be used for link bodies L1 and L23 and the exact analytical solution will be computed for the last 2 links. Using RH, the value of the first angle will be determined. Once the first link is aligned, the last 3 links will be coplanar with the goal, as it can be seen in Fig. 4. There are also represented in the figure the workspaces for L1 (W1) and L23-L4 (W45). We will later refer to the workspace for L23-L4-L56 as W456.

From the motion planning point of view, a sample-based hierarchic approach will be used. If a solution is computed, it will be tested on a loop using static grade sampling to ensure that no collisions occur and if the solution passes the test, joints will be commanded in hierarchic order, starting from the base and ending with the end-effector. If an error is received, there are many possible scenarios that can be followed (changing angle 2 – by ameliorating RH algorithm, moving the robot and others).

If a specific direction to the end-effector is given, the solution can be further constrained to the analytical approach that will solve the configuration of L23 and L4. In this case, if no solution is suitable after applying the collision test function, the only escape is to move the robot until a solution is validated.

Other constrains that must be imposed are the physical rotation capabilities of each joint, limiting the collision with ground (G_z > H_robot), with the robot itself. A collision test between the link bodies of the kinematic chain is also imposed.

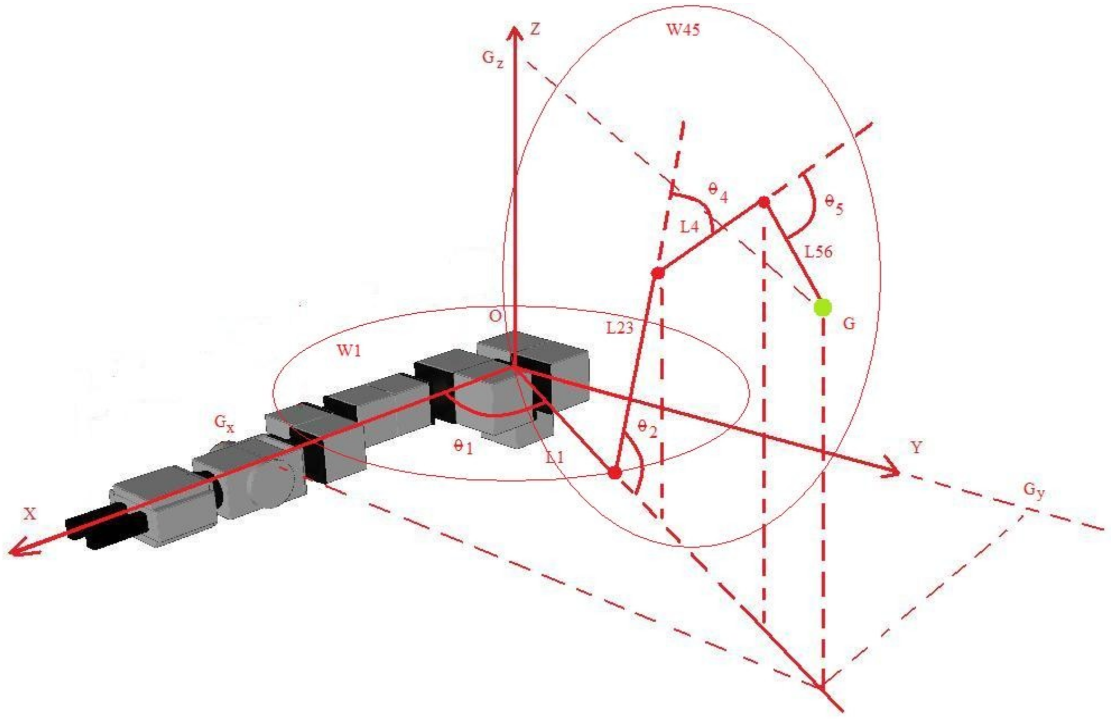

Fig. 3. Geometric model of PowerCube

5. Software architecture

The arm controlling program has been built around the libraries provided by Active Robots, in C/C++ environment. One of the most important features of the software is its flexibility. Sophisticated manipulation capabilities will be needed for real time interaction within dynamic test scenes.

As it can be seen in Fig. 4, each characteristic of our problem has been separated in an object oriented style. Subscribing to this approach ensures that other modules can be later added and the ones that are already functioning can be easily modified.

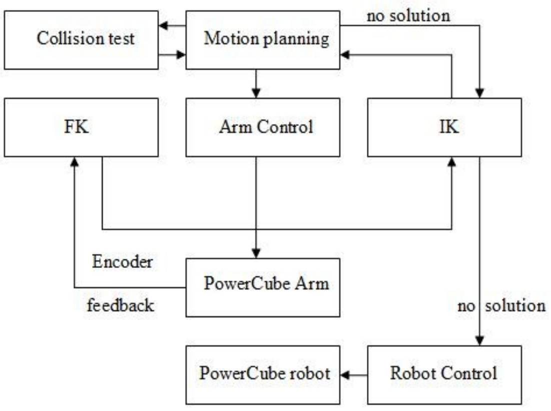

Fig. 4. Software architecture

The encoders from each joint provide angle information that is used by FK module to find out the actual position of link of the arm. The information is transmitted to IK module which provides the 4 angles presented in Fig. 3: Φ_1, Φ_2, Φ_4 and Φ_5. Φ_1 can have only 2 values for any goal set in the main coordinate frame. After setting up Φ_1, L23, L4 and L56 are coplanar with the goal. We will further be analyzing the inverse kinematics of 3 links in 2D space.

This also deeply simplifies the collision test function. After setting up angle restrictions for each joint, the only link collisions that have to be taken care of are the ones between L1-L56 and L23-L56, as all the other combinations are neighbors. Also L4 can not reach L1 because of the length of L23 (see Eq. (7), where Δd is added because of the link width).

$$L_1 + L_4 + \Delta d < L_{23} \tag{7}$$

If the IK module returns a set of solutions but none of them passes the collision test, the program enables the robot control module which commands the robot to move in a desired position where the goal is reachable. For this type of cases it has been defined an optimal end-effector region (based on the work in (Hadi and Sukhan, 2005) where the manipulator is having a well defined solution, considering link and environment constrains (see Fig. 5; at left - the optimal zone is presented facing the profile of the robot, at right – the optimal zone is viewed from the top of the robot, which is represented as the gray rectangle).

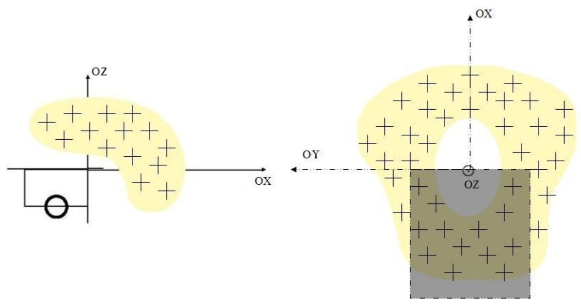

Fig. 5. Optimal end-effector space

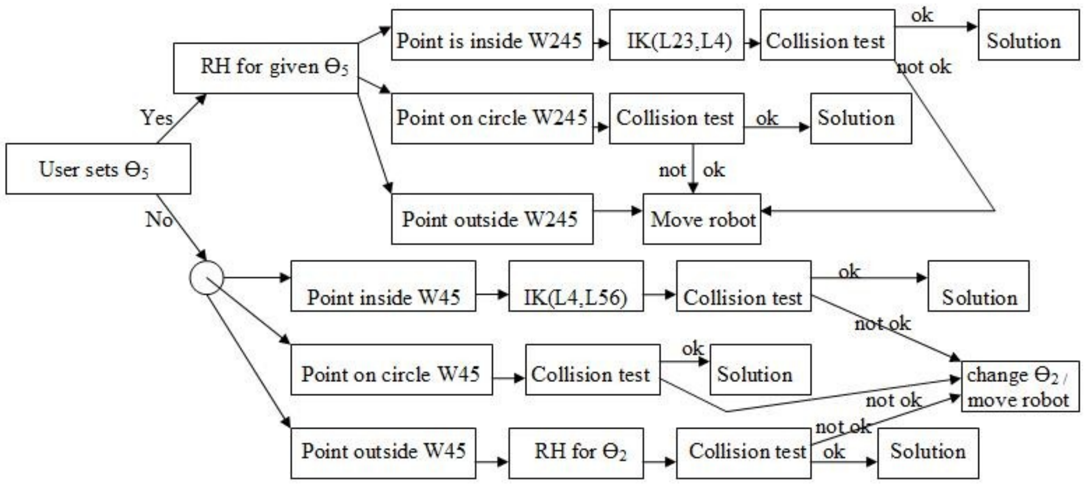

Fig. 6. Main program structure

Based on the cited research and on our specific case constrains, we have concluded that the optimal region is a space solution where Φ_4 is zero.

Because it sometimes is preferred to control the direction of the end-effector, special attention is given to IK module. This module accepts optional parameters like the value of Φ_5, or other restrictions to middle links.

The grasping motion itself can be improved using different techniques. The most used approach is to define a set of primitives that will guide the grasp module to achieve a desired direction and orientation. Although this research hasn't yet been conducted on this field, the possibility of achieving a certain target with restrictions on the end-effector's position has to be maintained. Sometimes this implies setting up fixed values or boundaries to Φ_5. It has been decided to include this case in our software; that is why the program has been split into 2 parts: one where the user doesn't set up Φ_5 and one where the user chooses a value for this angle. In this way the architecture can be further developed and the IK function can be limited to solving angle configuration for just 2 links (Fig. 6).

6. Research issues and further development

There are still some issues opened to research while implementing the solution described above.

While most tasks can be easily satisfied by this simple approach, but the limitations imposed on joint 3 reflect in the narrowing of the dextrous workspace (the positions which can be reached by the end-effector with arbitrary direction). It hasn't been taken in consideration any dynamical factors (which are resolved by each motor encoder).

Optimizing the trajectory for achieving the lowest consumed energy would grant the robot a higher flexibility by increasing its autonomy, a highly desirable aspect. Collision avoidance should seldom be performed without moving the robot base. Both needs described above can be satisfied by including joint 3 in the kinematic chain and reconsidering the IK approach.

Further development will conducted into integrating higher lever grasping structures (like Barett robotic hand) with PowerCube platform. A driver for Player/Stage/Gazebo (P/S/G) environment should also be created.

The arm controller will be further integrated within a shell that will also support vision computation, speech recognition, scene mapping creation and other features that will improve human-robot interaction, as our goal is to combine research results from these fields within a cognitive robot.

7. References

Bertram, D., et. al.: An Integrated Approach to Inverse Kinematics and Path Planning for Redundant Manipulators. ICSEUK, Karlsruhe, Germany. In *Proc. of IEEE Int. Conf. on Robotics and Automation* (2006), p. 1874-1879. http://www.kuffner.org/james/papers/

Hadi, M., Sukhan Lee: Joint Limit Analysis and Elbow Movement Minimization for Redundant Manipulators Using Closed Form Method. Sungkyunkwan University, In *Proc. of International Conference on Intelligent Computing*, ICIC 2005, Hefei, China (2005), p. 423-432. http://www.springerlink.com/content/8x4e8q4etmam6rwg/

Kang, T.G.: *Solving Inverse Kinematics Constraint Problems for Highly Articulated Models*. Master thesis, University of Waterloo, Ontario, Canada, 2000. Available at: http://www.cs.uwaterloo.ca/research/tr/

Kopicki, M.: *Path Planning in Robot Arm Control.* The University of Birmingham, 2007, PowerPoint presentation. http://www.cs.bham.ac.uk/~msk/IRLab/

LaValle, S.M.: *Planning Algorithms.* Cambridge University Press, Cambridge, USA, 2006. http://planning.cs.uiuc.edu/

Oiama, E., et al.: *Inverse Kinematics Learning for Robotic Arms with Fewer Degrees of Freedom by Modular Neural Network Systems.* NIAIST, Ibaraki, Japan, 2005. http://www.macdorman.com/kfm/

Rotenberg, S.: *Inverse Kinematics*. UCSD, 2005. PowerPoint presentation. http://graphics.ucsd.edu/courses/cse169_w05/

Spong, M.W., Hutchinson, S., Vidyasagar, M.: *Robot Modeling and Control*. John Wiley & Sons, 2005. http://www4.cs.umanitoba.ca/~jacky/

Zoppi, M.: *Effective Backward Kinematics for a 6R Painting Robot*. ASME Design Engineering Technical Conferences and Computer and Information in Engineering Conference DECT02, Montreal, Canada, 2002. http://www.dimec.unige.it/PMAR/pages/download/papers/