A Comparative Study of Incremental Algorithms for Computing the Inverse Kinematics of Redundant Articulated Systems

Received 9th April 2010; accepted 26th May 2010.

Abdelak Moussaoui, Rafaa Otmani, Alain Pruski

Abstract:

The field of control of redundant articulated mechanical systems requires the use of algorithms to compute the inverse kinematics. The fields of animation, virtual reality or game in particular, are very interested in these algorithms. We propose in this paper a comparison between several algorithms of incremental type. The considered application concerns the accessibility evaluation of an environment used by a handicapped person (an apartment, a house, an institution…). The physical disability involves a particular characteristic of the human articulated structure that gives rise to constraints that must be taken into account in computing the inverse kinematics.

Keywords: *inverse kinematics, accessibility, physical disability.*

1. Introduction

The field of robotics has been the first that was interested in the problem of inverse kinematics. The articulated systems have long been made up of six degrees of freedom and their controls were based on linearization methods, which gave solutions consistent with expectations. Currently the humanoid robots with high redundancy require more computing time. On the other hand, the animation of avatars requires the use of the same types of computing. We note that the type of algorithm is highly dependent on the type of application envisaged. In this paper we contribute to propose algorithms for computing the inverse kinematics with constraints related to an application in the field of accessibility assessment. We analyse the relevance of each in a specific context.

2. Background

The comparative study that we present in this paper concerns algorithms to determine the inverse kinematics of redundant articulated systems with very high number degrees of freedom. These systems are highly non-linear and require special methods of resolution. The principle is to calculate the value of joint variables such as:

$$\Theta = f^{-1}[(X])$$

With Θ the joint variables and [X] the constraint vector.

Many methods are proposed in the literature that we can classify into three categories:

- Analytical methods;
- Linearization methods;
- Optimization methods.

Each method has advantages and drawbacks. In general each depends by the application that you want to manage. The selection criteria are mostly the computing time and/or the accuracy obtained. A redundant system has no single solution and other criteria are frequently used to converge toward a particular solution.

The analytical methods are used when the number of variables is not too important. [1] and [2] have developed methods adapted to a humanoid articulated structure at the arm level. This structure has led to a method of solving the inverse kinematics specific for obtaining a solution quickly. By cons, it is linked to this particular structure and cannot extend to other structures.

Linearization methods are commonly used when the problem is complex and non-linear. The method is to approach the solution by successive increments considering that the system is linear around the operating point. The problem is solved by inverting a Jacobian matrix that is usually singular implying to calculate its inverse by the pseudo inverse techniques whose computational cost is important. [3]

The optimization methods are the most interesting and often used when the number of variables is important if we wish to obtain solutions meeting certain criteria. The principle involves formulating the problem as a cost function minimisation problem. Many approaches have been developed which include the gradient descent with special adaptations [4], [5] and genetic algorithms [6]. These algorithms are effective but can often lead to local minima. They can be very fast especially using BFGS-type methods, which approximate calculation of the inverse matrix.

The work presented in this paper concerns algorithms that are part of the class of optimization methods.

3. Context

3.1. Introduction

The work presented in this article is made in the context of a specific application that is to test the accessibility manipulation (Fig. 1) of a living place for a disabled person. For this purpose the articulated system consists of a humanoid-type structure that moves with a mobile base that models a walker or a wheelchair. The displacement device is important since we must consider the physical placement of the articulated structure with its mobile base in the environment. This application requires taking into account several parameters:

- The number of degrees of freedom taking into account the human trunk, one arm and the mobility system which is 24 degrees of freedom, as will see later;
- In essence, the human articulated system has limitations that we consider;
- We want verify only if a solution exists. We do not take into account other characteristics such as comfort or energy expended in computing the solution. We believe that if a solution exists then the person can reach the considered point;
- The accessibility evaluation is conducted in statics. We do not consider the motion type (e.g. the wheelchair non holonomy is not taken into account). Only the admissible geometric placement, which is to say without intersection with the physical environment, is considered;
- The required computing accuracy is not very important since we can consider that the compliance of the human body compensates for errors.

Fig. 1. Accessibility of the environment with a wheelchair.

3.2. Modelling

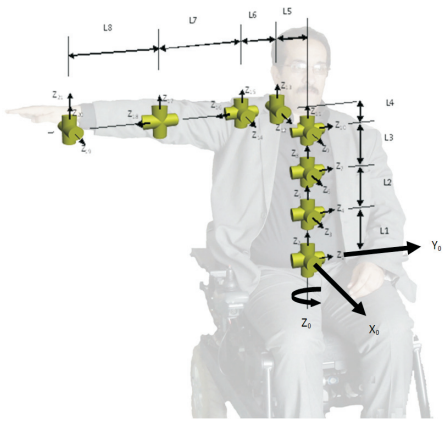

Fig. 2. The used kinematic joint structure.

The incremental algorithms that we study in this article require knowledge of the direct kinematics model. The system is described by a biomechanical model of the human being with limits in the range of joint movement. These are taken into account in the algorithm of the inverse kinematics, as we will detail later. The model we use is that proposed by [7] from which we extracted a model with 21 degrees of freedom from waist toward the tip of his right hand. We believe that mobility is achieved by a rectangular base corresponding to a wheelchair. The motion device is modelled using three degrees of freedom: a rotation and two translations. The frame position of the root structure is situated at the height of the waist of a seated person and the area swept by the wheelchair is a rectangle. Figure 2 shows the kinematics chain of the global user with his mobile base.

The mathematical model is established from the multiplication of Denavit-Hartenberg matrices [8] whose prototype is given below:

$$DH_i = \begin{bmatrix} \cos\Theta_i & -\cos\alpha_i \sin\Theta_i & \sin\alpha_i \sin\Theta_i & a_i \cos\Theta_i \\ \sin\Theta_i & \cos\alpha_i \cos\Theta_i & -\sin\alpha_i \cos\Theta_i & a_i \sin\Theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

And for n joints:

$$DH_{0,n} = DH_0 ... DH_i ... DH_n \tag{2}$$

For any vector V in the frame n in homogeneous coordinates, we will have in the R_0 world frame:

$$V_{R_0} = DH_{0,n} V_{R_n} \tag{3}$$

Two of the three algorithms that we analyse, work in an iterative manner that is to say that the joints are modified one after the other in order to verify if the end-effector approaches the goal. This sequential aspect of the algorithm allows to speeds up the computation of the direct model without having to reconsider all the joints. When we change one variable, only the variable corresponding matrix is modified.

We can write DH_0, n by considering two matrices.

$$DH_{0,n} = DH_{0,i-1} * DH_{i,n} \tag{4}$$

If we want to change the matrix corresponding to the variable i we can write that

$$DH_{0,n}^{q+1} = DH_{0,i-1}^{q} * DH_i^{q+1} * (DH_i)_q^{-1} * DH_{i,n}^{q} = \\ = DH_{0,i}^{q+1} * DH_{i+1,n}^{q+1} \tag{5}$$

or

$$DH_{0,n}^{q+1} = DH_{0,i}^{q} * (DH_i)_q^{-1} * DH_i^{q+1} * DH_{i+1n}^{q} = \\ = DH_{0,i-1}^{q+1} * DH_{i,n}^{q+1} \tag{6}$$

With q for the iteration computing. This method requires only three matrix multiplications instead of n. The inverse DH^{-1} is directly given by the following expression:

$$DH_{i-1} = \begin{bmatrix} \cos\Theta_i & \sin\Theta_i & 0 & -a_i \\ -\cos\alpha_i \sin\Theta_i & \cos\alpha_i \cos\Theta_i & \sin\alpha_i & -d_i \sin\alpha_i \\ \sin\alpha_i \sin\Theta_i & -\sin\alpha_i \cos\Theta_i & \cos\alpha_i & -d_i \cos\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

The two equations (5) and (6) are used, the first when the variables are changed sequentially from 0 to n (from

root to tip) and the second when the variables are changed from n to 0 (root to tip). The proposed algorithms use the two equations.

3.3. The algorithms

We propose in this section three algorithms whose performance depends on usage. All these algorithms are of incremental type with optimising a cost function, which corresponds to the error between the point to be reached, and the current point. This point is defined by the position of the hand and the orientation of the two vectors X and Z (Fig. 3). The error is the sum of the differences between the projections of current vectors X_{21} and Z_{21} and the desired vector in world frame.

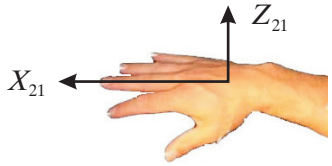

Fig. 3. Definition of vectors used in defining orientation.

Thus we determine:

$$\varepsilon = w_0 * (\text{Current Position - Desired Position}) + w_1 * (\text{Current Orientation - Desired Orientation}) \quad (8)$$

And generally: $$\varepsilon = |f([\Theta]) - [X]| \quad (9)$$

With $f([\Theta])$ the instantaneous position and $[X]$ the desired position and orientation.

3.3.1. Algorithm Cyclic Coordinate Descent (CCD)

This algorithm has been developed over many years and the following paper [9] details the fundamental principles. It is based on an incremental computing of the variables from the system end toward the root by minimizing the error between the desired point (and direction) and the current point (and direction). Some drawbacks are well known as an inhomogeneous variables adaptation in the chain joints and a slow convergence near the target. Where no limits imposed on variables, then the algorithm converges without local minima. It is possible to impose limits on the joints as we do in our application, in that case local minima may occur.

- 1. Initialise randomly the joint variables Θ_i
- 2. Do
  - For each variable Θ_i from $i=n$ to 1
    - Find the optimum value of Θ_i that minimise the error ε
- 4. While Stop Conditions not verified

The joint limits are taken into account by checking if the optimum value of Θ_i is included in the admissible limits. Otherwise, the calculated value Θ_i is not taken into account and we move to the next variable. In Phase 2 of the algorithm, it is necessary to compute the direct model in particular to obtain the error between the goal and that current point corresponding to the cost function ε. In computing the loop, only one variable is changed thereby applying equation (6).

3.3.2. Incremental approximation algorithm (IAA)

The principle of this algorithm is to modify each value of variables Θ_i from root to tip in order to minimize the magnitude of the cost function ε. Unlike the CCD algorithm, the value of the increment Inc applied to each variable is not calculated but imposed.

The resulted value of Θ is preserved if it is within the allowed range. The increment Inc is calculated for each joint i as :

$$\mathrm{Inc(i)} = (\mathrm{Max(i)} - \mathrm{Min(i)}) * \mathrm{IncrementRate} \quad (10)$$

with $\mathrm{Max(i)}$ and $\mathrm{Min(i)}$ the minimum and maximum limits of the joint i. IncrementRate can adjust the speed of the algorithm convergence. The parameters $\mathrm{Inc(i)}$ is very important in two aspects of its sign and amplitude that contribute to the speed of convergence. In the methods of gradient descent as Newton-Raphson, gradient matrices and the inverse of the Hessian fulfil these roles. The optimization of these values can accelerate the convergence. In our case we modify the basic algorithm by storing the sign of Inc. For each variable i we use the same sign at the next iteration. Convergence is rapid initially and then the variation becomes smaller with the proximity of the solution. We propose a modification of the algorithm by adjusting the step of the increment $\mathrm{Inc(i)}$ depending on the magnitude of the cost function $\Delta\varepsilon$ in a non-linearly manner as in equation (11). Other adaptation functions could be applied:

$$\text{if } (\Delta\varepsilon = 0) \text{ then } \mathrm{IncrementRate} = \mathrm{IncrementRate}/2 \quad (11)$$

A linear adjustment does not improve the speed of convergence. If the increment $\mathrm{Inc(i)}$ is sufficiently large, the distance change is rapidly becoming zero around the solution. We use the cancellation of De to decrease the value of the increment $\mathrm{Inc(i)}$. This algorithm is adaptable to any articulated structure. In this case, it is equation (5), which is used to calculate the direct model.

- 1. Initialise randomly the joint variables Θ_i
- 2. Do
  - 2.1. Define the increment $\mathrm{Inc\,(i)}$
  - 2.2. Do for each variable Θ_i
    - 2.2.1. $\Theta_i = \Theta_i + \mathrm{Inc\,(i)}$
      - Compute the distance between Current Solution and Goal such as $\varepsilon = f([\Theta]) - [X]$
      - if $(\Delta\varepsilon) < 0$ then keep Θ_i
      - Else $\Theta_i = \Theta_i \; 2 * \mathrm{Inc\,(i)}$
        - Compute $\varepsilon = f([\Theta]) - [X]$
        - if $(\Delta\varepsilon < 0)$ then keep Θ_i
          Else $\Theta_i = \Theta_i + \mathrm{Inc\,(i)}$
          (keep the original value)
- 3. While Stop Conditions not verified

3.3.3. The Random algorithm Approximation Algorithm (RAA)

Unlike the other two algorithms, this one treats all variables simultaneously. In a range of values $\mathrm{Inc(i)}$ defined by equation (10), we choose randomly increments that are added to the current values of variables. If the

choice lowers the cost function ε then the setof new values is maintained if not it is rejected. The procedure is repeated until the stop conditions are satisfied. In this case, it is equation (2), which is used to determine the model because all variables are modified during each iteration. The definition of the range increments $\mathrm{Inc(i)}$ is performed as previously by a non-linearly adjustment of IncrementRate (11) by varying the cost function.

- 1. Initialise randomly the joint variables Θ_i
- 2. Do
  - 2.1. Define the increment $\mathrm{Inc\,(i)}$
  - 2.2. For all variable Θ_i $\mathrm{Do}\ \Theta_i = \Theta_i + \mathrm{Rand(Inc\,(i))}$ ($\mathrm{Rand(X)}$ returns a random value in the range X)
  - 2.3. Compute the distance between Current Solution and Goal such as $\Delta\varepsilon = \mathrm{f([\Theta])\text{-}[X]}$
  - 2.4. if $(\Delta\varepsilon) < 0)$ then keep Θ_i Else keep the original value of Θ_i
- 3. While Stop Conditions not verified

When a set of values reduces the cost function then the same set of increment $\mathrm{Inc(i)}$ is applied at the next iteration. This accelerates the convergence.

3.4 Comparative analysis of the algorithms based on their application.

3.4.1. Introduction

The algorithms presented above have different behaviours depending on the type of use and constraints related to the application. The performance study of each algorithm is performed on a PC of Pentium 4 CPU running at 3.4 GHz. The algorithms stop conditions consists of two elements:

- A minimum value of the error ε (cost function) equal to 1 with $w_0 = 1$ and $w_1 = 5000$ from equation (8)
- A maximum number of iterations equal to 1000.

The RAA and IAA algorithms require imposing a rate value IncrementRate we placed in 0.015. This value, defined empirically, lead to the best results. It is linked to the size of the components of the articulated structure, which are defined in appendix.

The three algorithms show no local minimum when the joint variables are not restricted. In our case we limit the amplitude of the joints based on physical abilities of the person. Thus in some cases it is possible that the algorithms do not find a solution even if it exist. This is due to bad choice of joint variables values in initialisation during algorithms phase 1. We give our results in a success rate on a set of 10,000 trials. We consider that the algorithm has found a solution if the error is less than a minimum allowed value ε.

In Figure 4 we see that the algorithms converge quickly during the first iterations and take longer then. This overall behaviour remains the same but according to the initial value of variables and the goal to achieve, the algorithms speed may vary. Our application requires the accessibility verification in many the environment points requiring for each target point in space, to calculate the inverse kinematics. The comparison is performed on a statistical study on a large number of trials (10,000). We can compare the number of iterations between the CCD algorithm and IAA since the same number of matrices products is performed at each iteration, it is to say $3 * n$ (where n is the number of dof). By cons, RAA algorithm requires n multiplications per iteration, which is lower than the other two algorithms. A larger number of iterations is necessary in order to find the solution and the computing time will be longer. We consider that an iteration is achieved when all variables of the articulated structure has been treated.

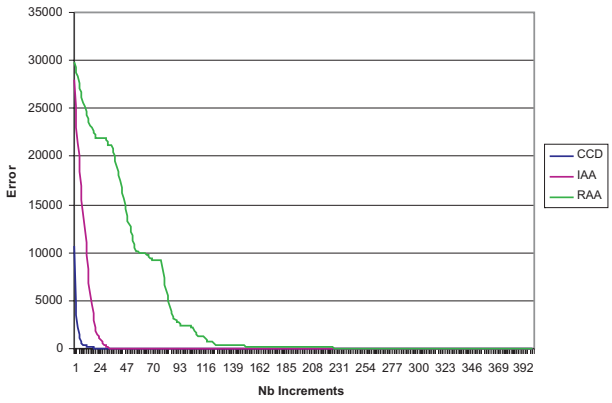

Fig. 4. Global behaviour of algorithms.

3.4.2. Comparative analysis of the computing of the inverse kinematics without mobility.

We propose to decompose the analysis such as we do not consider the mobility at the first time, without the degree of freedom in rotation Z_0 and without the two degrees of freedom in translation X and Y. The system has 21 degrees of freedom corresponding to a human joint structure from the waist to a hand. The target choice is achieved as follows. The joint variables values are initialised randomly

Table 1. Comparison between the described algorithms.

Algorithms	CCD		IAA		RAA	
	Average iteration number	Average Computing time in ms	Average iteration number	Average Computing time in ms	Average iteration number	Average Computing time in ms
Position $w_0=1$ and $w_1=0$	12	2.7	18	7	131	6.9
Position and orientation $w_0=1$ and $w_1=5000$	211	60.9	146	46.0	4770	301

that allows to determine associated position and orientation. It is these values that we target to reach.

Table 1 shows the comparative elements between the algorithms. We note that the CCD algorithm is the faster when the position is imposed. When we impose both the position and the orientation, the algorithm IAA is slightly more efficient. The results are averages on 10,000 trials. We note that the RAA algorithm is the least. Sometimes, a local minimum may occur and a solution is not available. In this case we start the algorithm again with other initial values given in phase 1. For that reason all solutions are given with 100% success.

3.4.3. Comparative analysis for the computing of the inverse kinematics of articulated structure based on mobile.

In this case, it is necessary to take into account both the additional degrees of freedom and the bulk of the mobile base. We assume that the mobile base moves along the floor, which allows us to define the mobility area by a polygon E that we call, envelop polygon. Some obstacles create exclusion areas in which the base cannot moves. All possible positions inside the Envelop polygon is called configuration polygon C. The shape of the polygon C is related to the orientation Z_0 of the mobile. It corresponds to the Minkowski difference of which the reader will find details in [10]. The configuration polygon is determined by scanning the envelope polygon for any value of the Z_0 variable since it is dependent on the orientation of the mobile (Fig. 5).

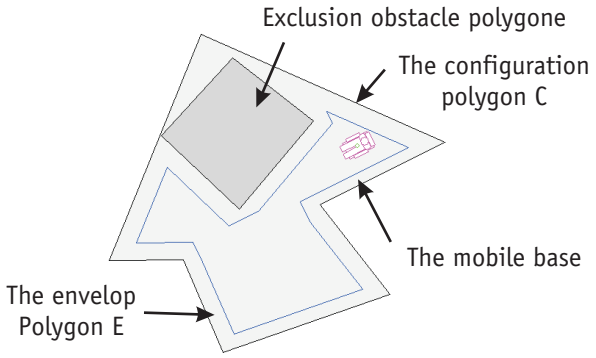

Fig. 5. Different polygons definition.

Now, the problem is to find a solution to the following problem

$$f([\Theta])=[X]-\phi(x,y,z) \quad (12)$$

It is to check whether there is a set of variables Θ such as $f([\Theta])$ may reach the point $[X]$ whereas the mobility space $\phi(x,y,z)$. The position to achieve does not longer correspond to a point but a surface. Thus the cost function becomes

$$\varepsilon=|f([\Theta])-[X]+\phi(x,y,z)| \quad (13)$$

We can write that:

$$\varepsilon = w_0*(\text{Current Position - Desired Surface}) + w_1*(\text{Current Orientation - Desired Orientation}) \quad (14)$$

Factor w_1 of the cost function remains unchanged but the other part requires the distance computation between a point and a surface. This is easy to calculate. The CCD algorithm requires a point to reach and not a surface to find the optimal variables value at each iteration. Thus this method is not applicable to the CCD algorithm.

We note that the inverse kinematics computing is significantly faster than for the fixed base. The reason comes from the number of potential points to reach is more important since we have to compute the distance from one point to a surface. The algorithm must compute, for each value of the mobile base orientation, the related configuration polygon. In order to accelerate the computing, we have pre compute these polygons. Adding the mobile base accelerates the computing.

3.4.4. Comparative analysis for the computation of the inverse kinematics of articulated structure on fixed base with small variations

In our application, which is to verify the existence of a solution in the case of accessibility, we test a set of points in the environment. We must check for each environment point if a solution exists. The environment points are achieved by taking each target point as situated in the neighbourhood of the previous one. Thus, we do not achieve a random joint variables initialisation as defined in the algorithms. We maintain the joint variables values as previously. We initialise the joint variables randomly only at the beginning of the process of computing when the first application of the algorithm. To model this application we perform the computation of a set of target points situated on a circle consisting of 200 points.

As we saw earlier, the application of the method that takes into account the mobility can not be applied to the CCD algorithm where the shaded area in Table 3. We see that the various algorithms require only a few iterations to achieve the goal where the weak computing time.

Table 2. Comparison between the algorithms IAA and RAA by considering the bulk of the mobile base.

	IAA		RAA	
	Average iteration number	Average Computing time in ms	Average iteration number	Average Computing time in ms
Position $w_0=1$ and $w_1=0$	11.3	0.7	26	1.6
Position and orientation $w_0=1$ and $w_1=5000$	35.9	15.7	2028	125

Table 3. Comparison between the algorithms if a weak variation of the objective point is performed and if the algorithms step 1 of variable initialisation is not done. Two cases are considered: when the base is fixed and when it moves.

	Algorithms	CCD		IAA		RAA	
		Average iteration number	Average Computing time in ms	Average iteration number	Average Computing time in ms	Average iteration number	Average Computing time in ms
Fixed reference	Position $w_0=1$ and $w_1=0$	1.4	0.3	1.8	0.7	39	2.1
	Position and orientation $w_0=1$ and $w_1=5000$	8.9	2.2	6.2	2.2	369	17.1
Reference on mobile base	Position $w_0=1$ and $w_1=0$			0.7	0.5	80	5.2
	Position and orientation $w_0=1$ and $w_1=5000$			4.2	1.7	132	20

In the case of mobile base, we find, as in Table 2, the increasing speed of execution with a limited number of iterations. The execution speed on our computer becomes less than a millisecond for a point target for the algorithm IAA. We recall that the error allowed is 1 for equation (14).

4. Conclusion

In this article, we compare three algorithms for computing the joint variable values in the case of the inverse kinematics of an articulated mechanical structure. The context of the use of these algorithms consists to check the accessibility of living environments for persons with disabilities. This application requires taking into account the constraints that have been detailed. The computation time is a key criterion that led the work conducted in this area. The CCD algorithm, well known in the field of animated avatars, seems particularly helpful when the base is fixed and when the target point is defined by a position. When we introduce the orientation constraint on the target point then the algorithm IAA we propose is slightly faster for a higher accuracy. The difference in results is not really significant and does not lead to a clear choice of best algorithm to use. When we consider that the mechanical structure is based on a mobile base so we get a greater difference in the results. The proposed algorithm takes into account the area swept by the mobile base is not adaptable to the CCD algorithm. It requires knowledge of a goal point for compute the joint variable values. Especially for the calculation of scalar and vector products. The proposed algorithm considers the point to reach is materialized by a surface and the number of potential points to reach is more important which has the effect of reducing the execution time and the number of iterations. We have tried to take the nearest point from the polygon solution to the articulated system. No significant improvements are noted and for the position and orientation constraint the results are lower in terms of iteration number.

AUTHORS
Abdelak Moussaoui, Rafaa Otmani, Alain Pruski* - Lasc, ISEA, 7 rue Marconi, University of Metz, France, tel. + 33 3 87 31 52 81. E-mails: {abdelhak.moussaoui, otmani, alain.pruski}@univ-metz.fr.
* Corresponding author

Appendix

Table 4. The Denavit-Hartenberg table of the considered kinematic chain.

	Θ	D	α	a
0	PI/2	0	PI/2	0
1	PI/2	0	PI/2	0
2	PI/2	0	PI/2	0
3	PI/2	L1	PI/2	0
4	PI/2	0	PI/2	0
5	PI/2	0	PI/2	0
6	PI/2	L2	PI/2	0
7	PI/2	0	PI/2	0
8	PI/2	0	PI/2	0
9	PI/2	L3	PI/2	0
10	PI/2	0	PI/2	0
11	PI/2	0	PI/2	0
12	-PI/2	L4	PI/2	L5
13	0	0	PI/2	0
14	0	0	- PI/2	L6
15	0	0	PI/2	0
16	PI/2	0	PI/2	0
17	0	L7	- PI/2	0
18	0	0	PI/2	0
19	PI/2	L8	PI/2	0
20	PI/2	0	PI/2	0
21	0	0	0	0

L1=10; L2=10; L3=10; L4=5; L5=10; L6=10; L7=30; L8=30;

References

[1] Tolani D., Goswami A., Badler N., "Real-Time Inverse Kinematics Techniques for Anthropomorphic Limbs", *Graphical Models*, vol. 62, 2000, pp. 353-388.

[2] Kallmann M., "Analytical inverse kinematics with body posture control", *Computer animation and virtual worlds*, 2008, 19, pp. 79-91.

[3] Baerlocher P., *Inverse kinematics techniques for the interactive posture control of articulated figures*. Federal school of Lausanne Swaziland, PhD Thesis, 2001.

[4] Mukundan R., A fast Inverse Kinematics Solution for an n-link Joint Chain, *5th Int. Conf. on Information Technology and Applications*, 2008, pp. 349-354.

[5] Muller-Cajar R., Mukundan R., "Triangulation: A new algorithm for Inverse Kinematics", Proc. *Image and Vision Computing*, New Zealand, 2007, pp. 181-186.

[6] Abdel-Malek K., Yu W., Yang J., Nebel K., "A mathematical method for ergonomic-based design placement", *Int. Journal of Industrial Ergonomics*, 2004, pp. 375-394.

[7] Yang J., Pitarch E.P., "*Digital Human Modeling and Virtual Reality for FCS*", Technical Report n° VSR-04.02; the University of Iowa, Contract/PR NO.DAAE07-03-D-L003/0001, 2004.

[8] Denavit J., Hartenberg R.S., "A Kinematic Notation for Lower Pair Mechanisms Based on Matrices", *Journal of Applied Mechanics*, vol . 77, 1955, pp. 215-221.

[9] Wang L.T., Chen C.C., "A combined Optimization Method for Solving the Inverse Kinematics Problem of Mechanical Manipulators", *IEEE Transactions of Robotics and Automation*, vol. 7, no. 4, 1991, pp. 489-499.

[10] Lozano-Perez T., "Spatial planning: a configuration Space Approach"., *IEEE Trans. on Computer*, vol. C 32, n°2, 1983, pp. 108-120.