

# Chapter 5

## Fatigue Exploitation in an Inverse Kinematics Framework

### 5.1 Introduction

As was seen in the related work chapter, there are several available techniques to generate computer animations. In particular, Inverse Kinematics was presented as a technique in which the animator only has to specify the goal to reach and the Inverse Kinematics engine solves the succession of joint angles needed to achieve it. This technique allows an interactive manipulation of complex structures. We propose a fatigue model to be applied to an articulated figure representing a human body so that it is an adequate technique for such a complex structure.

Since the time dimension is not explicitly handled in Inverse Kinematics, the convergence loop is often interpreted as the progressive enforcement of the constraints over time. The present chapter integrates time as an explicit variable in an Inverse Kinematics framework in such a way that fatigue evolution over time can be exploited. Fatigue is then applied to postures optimization and characterization.

## 5.2 Inverse Kinematics

In Inverse Kinematics, the animator generally specifies end-effector and positions constraints (tasks), and the computer solves for the joint configuration needed to achieve the desired task/s. Tasks are usually expressed in Cartesian space. End-effector is the part of the articulated structure that is controlled, for example, the hand in Figure 5-1. On the contrary, in Forward Kinematics, the animator specifies angles and the computer finds the end-effector position.

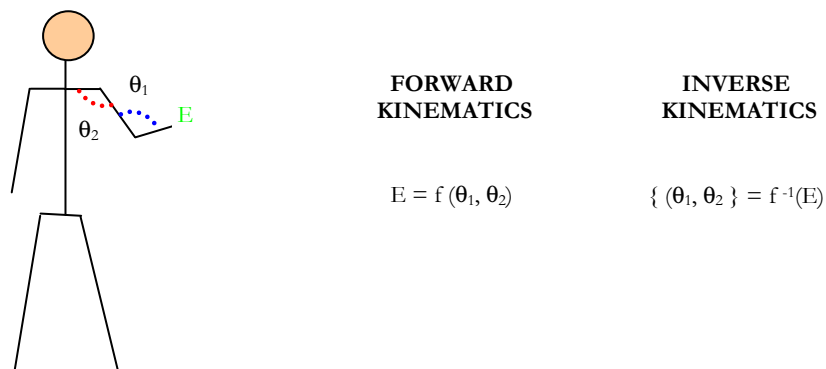


Figure 5-1. Forward vs. Inverse Kinematics

In analytic Inverse Kinematics (AIK), methods are adequate for simple structures. These methods are commonly used in the robotic field [Cra86]. Based on an earlier work [Kor82], IKAN provides an analytical method to solve generalized Inverse Kinematics problems on a human arm or leg. Its major limitation is that the system separates the articulated structure in several kinematics chains for which an analytical approach is applicable [Tol00].

One of our motivation for choosing numeric Inverse Kinematics is to provide synergistic solutions where all joints contribute to achieve all constraints. Numerical methods use equation solving or optimization techniques to obtain a more general solution.

Analytic solutions are not adequate for complex articulated structures. Hence, numerical methods are used to solve the Inverse Kinematics problem.

In the fatigue exploitation, we have used an Inverse Kinematics engine based on the well-known resolved motion-rate control from robotics [Whi69].

This technique does a linearization of a non-linear problem using the Jacobian Matrix  $J$ , which relates small changes of the joint coordinates  $\Delta q$  to small changes of task coordinates  $\Delta x$ .

$$\Delta x = J \Delta q$$

In fact, the Inverse Kinematics problem tries to find a solution to the “inverse” linear system:

$$\Delta q = J^+ \Delta x \quad (1)$$

where  $J^+$  represents the pseudoinverse of the Jacobian [Bou71].

The linear system is solved and the process repeated until the system converges to a solution.

The control of human postures needs to combine several tasks or constraints. For example, if we want to reach a point in the space, we will need a center of mass constraint to maintain the balance, a look-at constraint to gaze at the goal and a foot constraint to keep the foot in the floor. Therefore, the formulation given in (1) has to manage multiple tasks and they should be ordered by priority, in the example, the balance constraint has a higher priority than the look-at constraint.

It is important to select a strategy to solve the multiple tasks problem. One solution, the so-called weighting strategy, is to find a trade-off solution where no task is achieved exactly but each residual error is minimized [Bad87]. Another solution is to sort the tasks by order of priority. When they can be achieved simultaneously, all are satisfied, otherwise the task with higher priority reaches its goal while the residual error of the other tasks is minimized, without perturbing the achieved one.

Table 5-1 resumes several task-priority formulations, describing the author, year and main characteristics. Our system uses Baerlocher’s approach.

Although in the next section redundancy will be used for other optimization purposes, it has been efficiently exploited by all task-priority formulations. Considering that the primary task has higher priority level than the secondary task, in a task priority strategy, a

task with lower priority is executed only if it does not conflict with the higher priority task. This is done by projecting the secondary task onto the null space of the primary task.

YEAR/AUTHOR	FORMULATION CHARACTERISTICS
1981/Nakamura	Algorithmic singularities occur, then Damped-Least-Squared-Inverse is used and secondary task tracking is less accurate [Nak86]
1991/Siciliano	Recursive extension of [Nak86] to $n$ tasks [Sic91]
1994/97 Chiaverini	Algorithmic singularities do not occur but secondary task tracking introduces greater error [Chi94][Chi97]
1998/Baerlocher	Improve the speed of algorithm introduced in [Sic91]  Propose an incremental formula for the null space projector evaluation [Bae98]
2001/Youngjin Choi	Utilize the weighted pseudoinverse in place of the pseudoinverse  Bring the smaller error for the secondary task comparing to previous methods  Algorithmic singularities do not occur [Cho01]

Table 5-1. Task-priority formulations

Next section describes in more detail the problem of redundancy and its exploitation in several optimization scenarios.

### 5.2.1 Redundancy

An articulated structure is redundant when there are more degrees of freedom than constraints to be satisfied; we face an underdetermined problem as the number of degrees of freedom of the human body exceeds the number of task coordinates.

Methods based on the resolved motion-rate control allow to exploit the redundancy problem in several ways. The formulation due to Liégeois [Lie77] uses the gradient vector of a performance criterion,  $z$ , in the homogeneous part of the solution to the inverse kinematics problem:

$$\Delta q = J^+ \Delta x + P_{N(J)} z$$

where  $\Delta x$  is a known desired task increment,  $\Delta q$  is the unknown increment of joint coordinates,  $J^+$  is the pseudo-inverse Jacobian,  $P_{N(J)} = (I - J^+ J)$  is the orthogonal projection operator on the null space of the Jacobian and  $z$  represents an additional optimization term in joint space.

An application of optimization is to keep the joint angles as close as possible to some desired values. Liégeois proposed to exploit the homogeneous part of the solution to achieve joint limit avoidance. The function to minimize was the squared norm of the difference between the current and the mid-range posture:

$$z = -2(q - q_m)$$

where  $q$  represents the current posture and  $q_m$  the mid-range posture.

Another researcher used it to avoid singularity [Kle84]. Singular configurations are undesirable joint configurations characterized by the Jacobian matrix losing its rank, and inducing extremely large joint velocities for small end-effector changes. Maciejewski demonstrated the use of Singular Value Decomposition (SVD) to detect near-singular configurations and apply damping measures to joint velocities as compensation for the discontinuities produced by the pseudoinverse method [Mac90].

Redundancy has also been applied to obstacle or collision avoidance [Bou86] [Esp85] and maximum comfort [Bru87].

Maciejewski attached an instantaneous repulsive velocity to the manipulator point closest to the obstacles [Mac85] [Mac89]. Then, it is defined a secondary end effector with

Jacobian  $J_s$  and pseudo-inverse  $J_s^+$ . As the main task has influence on the secondary end effector, it is needed to subtract it from the repulsive velocity in order to compensate it in the final solution. The following formulation describes Maciejewski's proposal:

$$\Delta q = J^+ \Delta x + [J_s(I - J^+ J)]^+ (\Delta x_s - J_s J^+ \Delta x)$$

In conclusion, besides primary tasks (i.e. cartesian constraints), the Inverse Kinematics technique allows the execution of secondary tasks (usually expressed in joint space) thanks to the redundancy of the articulated figure. The secondary task has the lowest priority level due to it is projected in the null space of the primary one. For this reason, we do not use the optimization approach for fatigue minimization. In the next section, we present the hard constraint concept which guarantees the highest priority for constraints.

### 5.2.2 Hard Constraints

In addition to Inverse Kinematics tasks (i.e. end effector position/orientation), it is possible to ensure the satisfaction of others equality and inequality constraints with the highest priority, in other words hard constraints. Our approach uses the constraining property of the task-priority solution described in [Bae01] for postures optimization. The author applied hard constraints to joint limits and joints couplings. Let us describe how equality and inequality constraints are integrated in the initialization phase of the task-priority solution. A set of  $s$  linear equality and inequality constraints are expressed as follows:

$$c_i^T q = b_i, \quad i=1 \dots r$$

$$c_i^T q \leq b_i, \quad i=r+1 \dots s$$

where  $q$  is a  $n$ -dimensional vector of joint coordinates,  $c_i$  represents a vector of dimension  $n$  and  $b_i$  is a scalar. In the following,  $C$  represents the Jacobian of all active constraints, it gathers every  $c_i$ .

Without using equality and inequality constraints, in the initialization phase of the task-priority algorithm the solution vector is set to zero, and the projection matrix to the identity:

$$1) \Delta q_0 = 0$$

$$2) P = I_n$$

Using equality and inequality constraints, the algorithm modifies the initialization stage as follows:

$$1) \Delta q_0 = C^+(b - Cq)$$

$$2) P_{N(J_0)} = P_{N(C)} = I_n - C^+C, \text{ where } C = [c_1 \dots c_r]^T \text{ and } b = [b_1 \dots b_r]^T$$

Inequality constraints are managed using the active set method. The active set method proceeds by partitioning inequality constraints in two sets, active (or sufficiently close to be supposed active for this iteration) and inactive. The inactive constraints are ignored for the iteration. The active set for this iteration is sometimes called the working set.

Initially, the method proceeds by including only equality constraints in the working set, then it is solved for  $\Delta q$ . The new state  $q + \Delta q$  is calculated and it is checked whether there is any violated inequality constraint. These are then introduced in the working set (by converting them into equality constraints). This conversion is done by the active set method which assumes that an inequality constraint that lies on the boundary of the constraint acts like an equality constraint. The new state is selected by moving on the surface defined by the working set. With this new working set, a new  $\Delta q$  is computed and the process is repeated until all constraints are satisfied.

### 5.3 Fatigue Model Exploitation

#### 5.3.1 Overview

In this research, the use of the fatigue model is twofold. On one hand, to search for postures where fatigue is minimized, that is, it is used in postures optimization. On the other hand, to identify postures or reachable spaces using the fatigue physiological factor. Both are described in the next sections.

#### 5.3.2 Postures Optimization

A previous research on Inverse Kinematics proposed a torque minimization due to gravity torques; it was based on geometric properties [Bou97]. A more recent proposal

included both gravity and external forces. Its purpose was to converge to postures with a minimum amount of joint torques. Both approaches exploited structure redundancy to solve the optimization problem.

Our fatigue minimization method is based on the constraint approach described in section 5.2.2. We introduce a hysteresis activation pattern for each half-joint to set a fatigue reduction constraint whenever necessary. They can be named “hard constraints” as they have to be ensured with a higher priority than all other tasks.

Figure 5-2 describes the constraint hysteresis activation pattern. When a half-joint fatigue level is above the fatigue threshold, the joint variation is constrained to reduce the half-joint torque by a small increment compatible with the corresponding time increment (hypothesis of slow motion). Then the other tasks are achieved in the null space of that constraint. The constraint exists until the half-joint recovery level is reached (second line of arrows in Figure 5-2), moment in which the fatigue minimization process is deactivated.

The hysteresis activation pattern forces a minimal duration for the recovery by setting a lower threshold for de-activating constraints; this reflects better human behavior compared to a single activation/deactivation threshold [Mcn02].

Our approach achieves fatigue minimization exploiting active and passive torques at joint level. We have utilized a factor, named “muscular tonus” (see Figure 5-3 ), which represents the proportion of active torque that is being used in the fatigue reduction process.

Tonus is a value normalized between 0 and 1. When tonus is 1, active torque is completely exploited in the fatigue minimization constraint. In the opposite case, when tonus is 0, active torque is not included and only passive torque is being exploited. The control of the tonus is introduced in order to produce a wider range of valid postures.

The fatigue reduction constraint produces a joint variation that reduces the active torque of the most fatigued half-joint and increases the passive torque produced by tendons and ligaments.



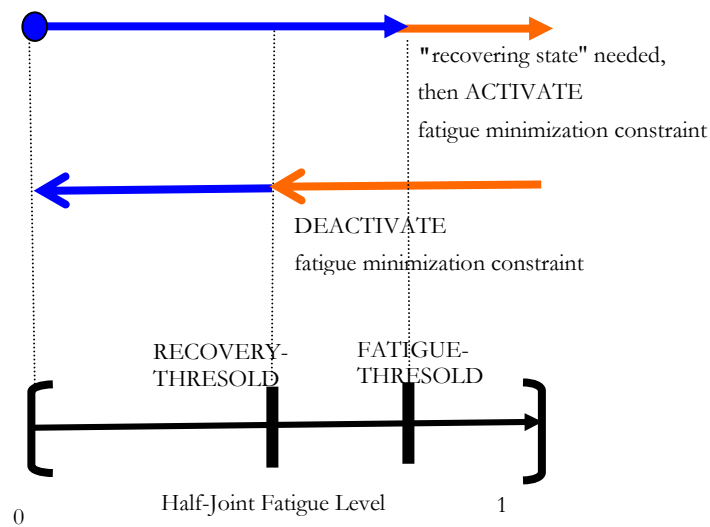


Figure 5-2. Constraint hysteresis activation pattern

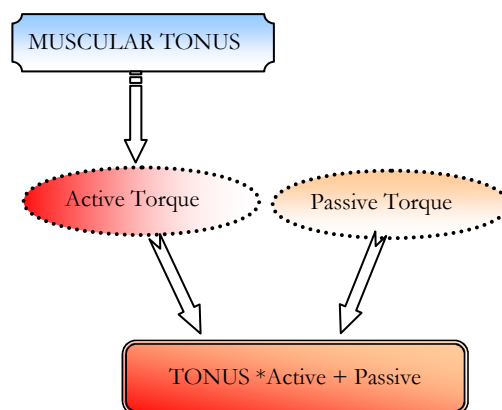


Figure 5-3. Tonus determines the influence of the active torque

### Tonus Factor Contribution

The tonus factor used as described above allows the generation of a wider range of postures as its exploitation leads to solutions not achieved only using active torque. It allows to generate a greater potential of psychologically marked poses.

An example is the passive torque achieved at joint limits by people standing; poses adopted in such conditions look more slouch.

### Active Torque

In the case of active torque, torque reduction is calculated as follows: If the joint  $l$  is fatigued, its torque  $\tau_{active\ l}$  has to be minimized. We compute the partial derivative of  $\tau_{active\ l}$  with respect to all joints:

$$Vq_j = \frac{\partial \tau_{active\ l}}{\partial q_j}, \forall j$$

A vector storing these partial derivatives indicates how much torque  $\tau_{active\ l}$  changes for a small variation of each joint. Then, the vector  $Vq$  stores the partial derivatives:

$$Vq = (Vq_0, \dots, Vq_{n-1}), \text{ where } n \text{ is the number of joints.}$$

To compute  $Vq_j$ , we need the Jacobians  $J_{T_i}$  associated with the external forces  $f_i$  and the gravity Jacobian  $J_G$  associated with the weight  $w$ . This is the expression of the coordinate  $Vq_j$  corresponding to joint  $j$ :

$$Vq_j = \sum_i^{ne} J_{T_i\_l} \cdot (f_i \times a_j) + J_{G\_l} (w \times a_j)$$

where  $ne$  is the number of external forces,  $J_{T_i\_l}$  is the column  $l$  of  $J_{T_i}$ ,  $J_{G\_l}$  is the column  $l$  of  $J_G$  associated with the weight  $w$  and  $a_j$  represents the unit axis of rotation of joint  $j$ . We need only the column  $l$  of these Jacobians because it corresponds to the torque  $\tau_{active\ l}$  to minimize. The entire Jacobians  $J_{T_i}$  and  $J_G$  were used in the study mentioned in the previous section [Bar01].

In fact, what we want is to construct an inequality constraint of the form:  $a^T q < b$ . Recall the definition of hyperplane H:

$$H = \left\{ q \in R^n / a^T q = b, b \in R \right\}, \text{ where } a^T \text{ represents the normal of the hyperplane H. A}$$

hyperplane divides space into upper and lower halfspaces:

$$H_L = \left\{ q \in R^n / a^T q < b, b \in R \right\}$$

$$H_U = \left\{ q \in R^n / a^T q > b, b \in R \right\}$$

We will see the construction of the inequality constraint with an example in 2D, where

the hyperplane represents a line.

As the gradient vector described above,  $Vq$ , is the direction of maximum increase of the function and what we want is to minimize the torque, we take its opposite. We set a maximum speed for the joint variation, for example in the arm case study the maximum value is 1.2 degree/second.

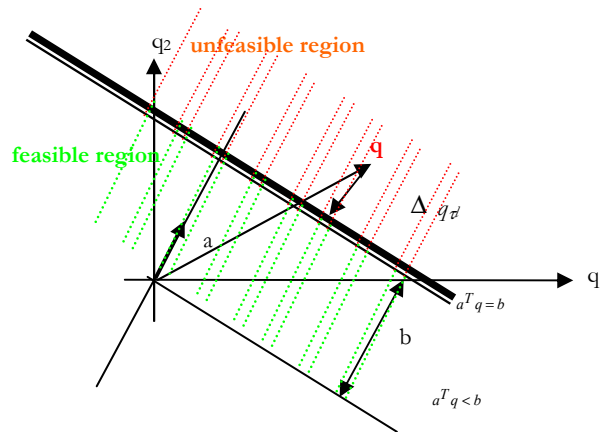


Figure 5-4. Example of hyperplane in 2D

The joint variation needed for the minimization is calculated as follows:

$\Delta q_d = \alpha \cdot \tau_{active_l} Vq$ , where  $\alpha$  is a negative scalar,  $\tau_{active_l}$  is the torque of the most fatigued joint and  $Vq$  is the gradient vector indicating how much  $\tau_{active_l}$  changes for a small variation of each joint.

As can be seen in Figure 5-4,  $q$  is out of the feasible region, then the  $\Delta q_d$ , needed to drive it to the feasible region, has opposite direction to  $a^T$ :

$a^T = - \text{normalized}(\Delta q_d)$ , note that  $a^T$  is the gradient vector that represents the hyperplane.

As  $q + \Delta q_d$  is on the hyperplane, its product by  $a^T$  gives the scalar  $b$  :

$$b = a^T \cdot (q + \Delta q_d)$$

Therefore, we have already defined an inequality constraint that drives the  $q$  vector

towards a direction of minimum torque.

### Passive Torque

In the case of passive torque, we use a model derived by Riener [Rie99]. Functions  $k$  and  $h$  in Figure 5-5 and Figure 5-6 represent the passive moment of knee and hip joints respectively, where  $\theta_A, \theta_K, \theta_H$  are ankle, knee and hip angles in degrees. Similar functions can be derived for other joints.

$$k(\theta_A, \theta_K, \theta_H) = \exp(1.800 - 0.0460\theta_A - 0.0352\theta_K + 0.0217\theta_H) - \exp(-3.971 - 0.0004\theta_A + 0.0495\theta_K - 0.0128\theta_H) - 4.820 + M_K^*$$

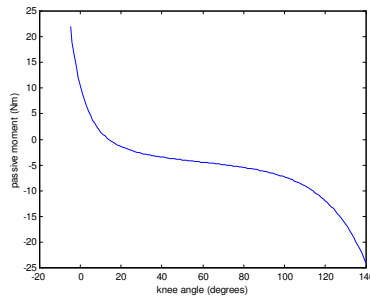


Figure 5-5. Graphic representation of function  $k(\theta_A, \theta_K, \theta_H)$

$$h(\theta_K, \theta_H) = \exp(1.4655 - 0.0034\theta_K - 0.0750\theta_H) - \exp(1.3403 - 0.0226\theta_K + 0.0305\theta_H) + 8.072$$

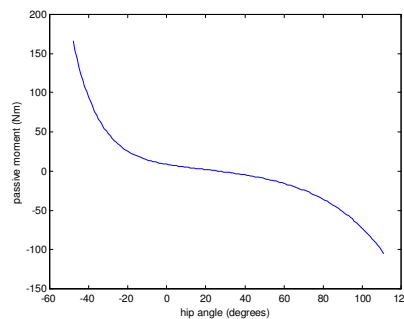


Figure 5-6. Graphic representation of function  $h(\theta_K, \theta_H)$ 

The term to calculate the joint variation needed to find passive torque of the most fatigued joint is calculated by means of the partial derivative of the previous functions, that is, their Jacobian.

Table 5-2 shows the partial derivatives of function  $h$ . Then, a gradient vector  $V_p$ , storing the partial derivatives of function  $h$ , along with  $V_q$  is used in the creation of the inequality constraint.

<p><i>Partial derivative respect to <math>\theta_K</math> :</i></p> $\frac{\partial h}{\partial \theta_H} = -0.0750 \exp(1.4655 - 0.0034\theta_K - 0.0750\theta_H) - 0.0305 \exp(1.3403 - 0.0226\theta_K + 0.0305\theta_H)$
<p><i>Partial derivative respect to <math>\theta_H</math> :</i></p> $\frac{\partial h}{\partial \theta_K} = -0.0034 \exp(1.4655 - 0.0034\theta_K - 0.0750\theta_H) - 0.0226 \exp(1.3403 - 0.0226\theta_K + 0.0305\theta_H)$

Table 5-2. Partial derivatives of function  $h$  in Figure 5-6

The fatigue minimization as described before, accomplishes the minimization of the most fatigued joint reducing active and passive torques coming from external forces, tendons and ligaments.

The following chapter presents the case studies <lifting in sagittal plane> and <the contraposto>. Both cases exploit the fatigue minimization approach as described in this section.

#### 5.4 Reachable Space Evaluation for Postures Characterization

At a higher level than posture optimization, postures characterization is achieved thanks to the combination of fatigue assessment at joint level and the generation of reachable spaces. In this way, a feature that identifies a posture is the induced fatigue while

the posture is sustained.

When a subject maintains a posture during certain time while, i.e. carrying an external load; after a fatigue assessment, the posture can be classified inside a range of fatigue indices. Consequently, the reachable space using this posture or similar ones can also be characterized by the fatigue factor. It is possible to appreciate that a region of the reachability space is reached with less fatigued postures than other regions. In the following, we introduce the concept of reachability and describe how spaces of reachability are constructed.

Human movement and, in particular, human arm motions play an important role in studies of the human body. Reaching is a daily life activity. We frequently reach a glass of water, a door handle, a book on a shelf, etc.

Computer generated reachable spaces allow the analysis of the human body and its environment. We can generate and compare reachable spaces in different conditions (sitting, standing,...).

Our motivation is due to the necessity of systems that help to manage data relative to the most frequent tasks involved in human activity. In particular, we aim to use fatigue data to characterize postures and reachable spaces in several reaching strategies.

#### **5.4.1 Construction of a Reachable Volume for Different Strategies**

Depending on the type of reaching that we want to generate, we establish a set of constraints that characterize the task. Therefore, a set of constraints has to be defined in order to generate the reachable volume corresponding to each strategy [Rod03c] [Rod03d].

The distance between the virtual human and the volume to approximate determines a near or far reaching. The height of the volume with respect to the virtual human also determines differences in the sort of reaching. A very low volume situated at feet level will allow studying crouch reaching.

Table 5-3 shows the set of Inverse Kinematics constraints that define three different strategies.

The performance of a standing reach needs some forward bending of the trunk, thus

requiring postural stability provided by a controlled center of mass. Other constraints are a positional constraint applied to the body part, which does the reach, and a look-at constraint that makes the subject gaze at the target. All these constraints define what we call a direct near reach.

If the goal is so low that the subject needs to crouch, in addition to those constraints previously mentioned, flexing legs and a change on the root of motion are also required.

Finally, for seated reach, in which the person has to reach an object starting from a seated posture, the center of mass needs not be controlled, but it is necessary to change the location of the motion flow root. The root of the motion is set to the thigh because the hip is highly involved in the movement. If this joint was not included in the joint chain, wrong postures and smaller reachable spaces would be generated.

Figure 5-7 describes the process of labeling and storing a reachable volume. Given the set of constraints that define a strategy, a mechanism queries the Inverse Kinematics engine, asking for reachability.

An initial voxel is specified and the reachability query is applied to its eight vertexes. The initial voxel should be chosen large enough so that it is bigger than the reachable space. In this way it will force a voxel decomposition. Otherwise, the decomposition might stop too early with a crude approximation.

The Inverse Kinematics engine replies telling whether the voxel is reachable or unreachable. We say that a voxel is reachable when the strategy is adequate to reach its eight vertexes.

An unreachable voxel, on the other hand, is entirely made of unreachable points. When a voxel has a mixture of reachable and unreachable parts, it is divided into eight new, smaller child voxels and the same process is applied to each of them.

A detailed description of the algorithm followed to construct a reachable volume can be found in Figure C.1 of appendix C.

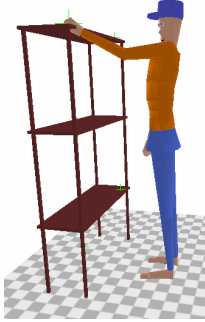
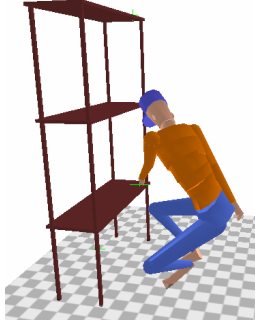
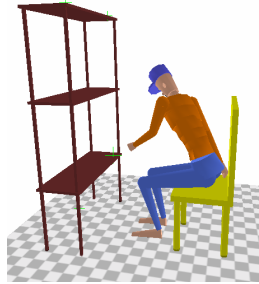
SET OF CONSTRAINTS	STRATEGY
<p>Center of mass control                      Positional constraint (i.e. hand)                      Look-at constraint                      Motion flow root (pelvis)</p>	<p>STANDING REACH:                       DIRECT</p> 
<p>Center of mass control                      Position constraint (i.e. hand)                      Looking constraint                      Flexing legs                      Change root of motion (i.e. foot)</p>	<p>STANDING REACH:                       CROUCH</p> 
<p>Position constraint (i.e. hand)                      Looking constraint                      Change root of motion (i.e. foot)</p>	<p>SEATED REACH</p> 

Table 5-3. Constraints that define reaching strategies



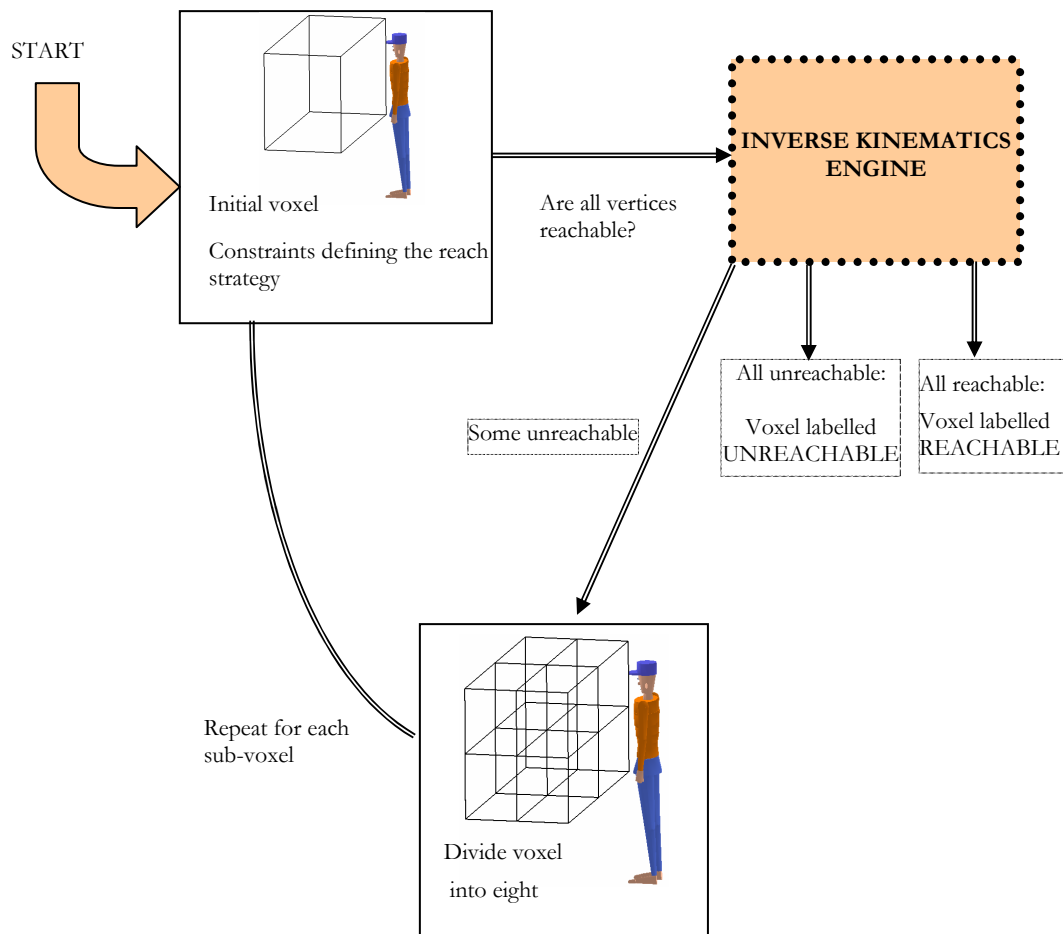


Figure 5-7. Generation of a reachable volume

Figure 5-8 shows reachable volumes generated with different strategies. Reach space (in red) shown in Figure 5-8.A was generated by a simulation with a direct strategy but using as kinematics chain only the arm of the articulated figure.

Figure 5-8.B displays the reachable space (in green) generated by a simulation with a direct strategy but using the upper body as kinematics chain. Reachable space in (B) is higher than in (A) because the clavicle adds extra degrees of freedom. This space is also bigger in front and side views due to the contribution of the spine.

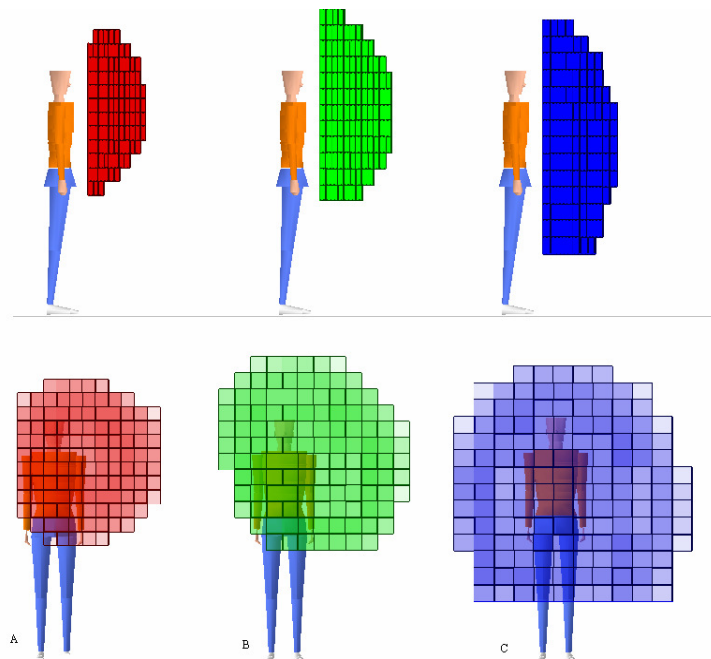


Figure 5-8. Three reachable spaces: (A) Direct using only the arm (B) Direct using the upper body (C) Direct using the entire hierarchy and controlling balance

Reachable space (in blue) on Figure 5-8.C is generated by a simulation with a direct strategy using as kinematics chain all the hierarchy and controlling the center of mass. In this case some voxels in the upper and front regions are not reachable anymore, compared to case (B). Also compared to case (B), in case (C) lower positions are reachable due to the contribution of the hip joint that in case (B) was not included.

#### 5.4.2 Adding Fatigue Data to the Reachability Volume

In the storing of reachability data along with fatigue data, the Inverse Kinematics engine and the fatigue module play an important role. Figure 5-9 shows the mechanism followed to store information about fatigue in reachable points. Numbers drawn over arrows indicate the order of processes. The Inverse Kinematics module receives information about the point that want to be classified as reachable or unreachable, and a set of tasks that define the strategy to do the reaching. Then, the Inverse Kinematics engine gives information about reachability. Only for reachable points the fatigue assessment

process is activated.

The Inverse Kinematics engine iterates towards the goal defined by the reachable point. The achieved posture is maintained during certain time in order to assess how fatiguing the posture is. During steps 3 and 4 a high interaction between the two modules is needed as the fatigue module needs updated data about posture in order to calculate joint fatigue level. In fact, these steps are performed repeatedly until the time established for the simulation is reached. Finally, the fatigue module returns a fatigue value used to feature the point as reachable with a determined fatigue value. This fatigue value has been computed for a predefined duration and external load. It is shown on Figure 6-9 of chapter 6 which shows results on reachable spaces including fatigue data.

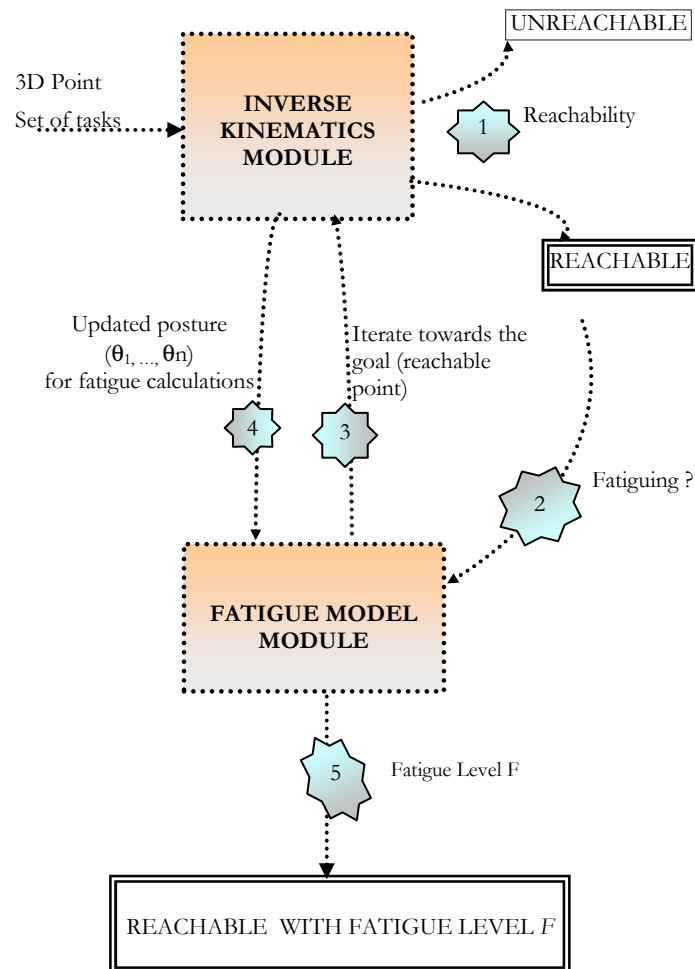


Figure 5-9. Featuring reachable points with fatigue data

## 5.5 Summary

In this chapter, we have shown how the fatigue model handles the time dimension within an Inverse Kinematics optimization loop. We have presented this technique as adequate for the interactive management of complex articulated figures, in particular, virtual humans.

Fatigue has been exploited in an Inverse Kinematics framework and it has been applied to postures optimization and characterization.

In posture optimization, a mechanism for fatigue minimization, named constraint hysteresis activation pattern, is used to adjust fatigued postures.

The fatigue model has also been used to store information of fatigue in reachability trees. We have described the process of construction of a reachable volume for different strategies. In addition, we have shown the interaction between the fatigue and reachability modules in order to store fatigue data in reachable volumes.