The Visual Computer, submitted May 2002, reviewed march 2003, revised December 2003 Pre-Print Version

An Inverse Kinematic Architecture Enforcing an Arbitrary Number of Strict Priority Levels

Paolo Baerlocher[1], Ronan Boulic[2]

1 : NEKO Entertainment, 12-16 rue de Vincennes, F-93100 Montreuil, France
2 : VRlab, EPFL CH-1015 Lausanne, Switzerland

Abstract

An efficient Inverse Kinematics solver is a key element in applications targeting the on-line or off-line postural control of complex articulated figures. In the present paper we progressively describe the strategic components of a very general and robust IK architecture. We then present an efficient recursive algorithm enforcing an arbitrary number of strict priorities to arbitrate the fulfillment of conflicting constraints. Due to its local nature, the moderate cost of the solution allows this architecture to run within an interactive environment. The algorithm is illustrated on the postural control of complex articulated figures.

Categories and subject Descriptors : I.3.7[Three Dimensional Graphics and Realism]: Animation

1. Introduction

The production of believable animations is an expensive process. For some classes of animation with high self-interaction and/or interaction with the environment, automated techniques are required to propose solutions to the intricate problem of finding the most believable posture for a set of conflicting constraints. A large number of approaches, both off-line and on-line, have been described up to now. In the present paper we focus on IK postural control as one essential component of the motion manipulation pipeline. While it does not solve the whole animation design problem, we advocate reconsidering its potential in the light of its efficiency. The present article progressively describes a robust architecture handling complex cases - including in singular context.

Our approach deals with general tree-structured articulated systems relying on a minimal set of coordinates, i.e. including only mechanical degrees of freedom between adjacent bodies. The corresponding space is called the joint space (as used in Robotics). The major criterion behind this choice is performance. For generality reasons, the IK technology we exploit requires matrix inversions; thus it is important to minimize the dimension of the controlled system. An additional aspect to master for believability is to produce postures within each joint's anatomic range of motion. Thus we integrate a clamping algorithm within our architecture.

The second essential requirement for our IK architecture is the causal nature of its solution - depending only on the present and past states - allowing its use in an on-the-fly fashion. Within that framework, it is possible to add and remove any number of constraints at any time. However, constraints are seldom orthogonal, leading to conflicts regarding to their fulfillment. One major novelty of the present paper is to describe an efficient resolution scheme enforcing an arbitrary number of priority levels among constraints. Figure 1 illustrates a postural control application based on this architecture (details in section 4).

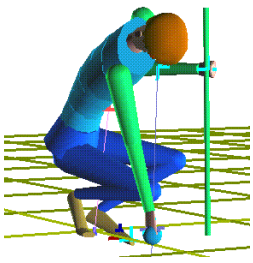


Figure 1: *Squatting posture combining four levels of conflicting constraints.*

The key argument of this paper is the following: the performance of an IK algorithm must be

examined together with the quality of the convergence. At first sight, the computing cost of our architecture might appear too high, but this is the price to pay to guarantee essential properties for the fast convergence towards constraints enforcement. We show in the results section that our IK solver can interactively handle the postural control of a human-like structure with forty degrees of freedom and a fifteen dimensional set of constraints.

The next section briefly covers the State of the Art in Inverse Kinematics. The architecture of our Multiple Priority Levels IK is presented in section 3 with a special focus on performance analysis. Then, section 4 illustrates the potential of this architecture within a testbed application dedicated to interactive postural control. Section 5 concludes.

2. Previous Work

2.1 Trends in the animation production process

The goal of any animation technique is to generate believable motion. Technically, the simplest solution is to smoothly interpolate key postures over time to generate the full set of frames required for an animation, the animator being free to adjust the key postures if the animation is not satisfying. This is known as the keyframing technique[1]. Although very tedious, this technique is still popular among animators because they can master the creative process up to the finest details of the joint trajectories. However, the time and skill required to produce convincing animations become too important to animate a large variety of characters. In addition, it is an off-line process. The production cost problem can be partly solved by capturing and retargeting motions to different articulated figures[2,3,4,5,6,7]. The requirement of on-line techniques is necessary for the control of autonomous characters; it asks for causal methods relying only on the present and past states of the system. Few approaches presently display this property[8].

Among the numerous sub-problems lying within this short overview of the animation production challenge, we have chosen to focus on the Inverse Kinematics component. This component is responsible for proposing a believable postural state for a given set of constraints. Whether this proposed state is adopted depends on other aspects of the control strategy such as the dynamics of the system, but this is beyond the scope of the present paper.

2.2 Background in Inverse Kinematics (IK)

Ideally, an analytic formulation should express the relation between the constraints and the joint variables. General closed forms do not exist, but, when considered independently, the limbs of 3D characters can exploit a robust and fast analytical solution[4,9,10].

In the most general case, IK can be formulated as a constrained optimization problem and thus solved with nonlinear programming methods. For real-time or interactive applications, local optimization methods are preferred to the much more expensive global optimization methods even if there is a risk of getting stuck in a local optimum of the objective function. In computer graphics, Zhao[11] use an optimization method for the manipulation of an articulated figure: a nonlinear function (describing the degree of satisfaction of all constraints) is minimized under a set of linear equality and inequality constraints describing joint limits. An application of this method for the manipulation of articulated figures is integrated within the Jack system[12].

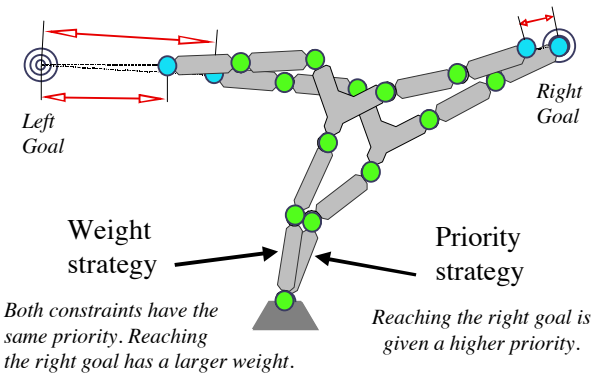


Figure 2: Two strategies for solving conflicting constraints

Among the qualities of any method, it is highly desirable to simultaneously manage multiple constraints. Very often, some constraints cannot be satisfied at the same time, whereas they can be so separately. This conflicting situation is solved with either of the two following strategies (Figure 2):

- weights can be assigned to each task to define their relative importance and a compromise is found that however exactly satisfies none of the tasks.
- tasks are optimized at different priority levels. In this mode, every task is satisfied as much as possible but without affecting the satisfaction of more important ones.

For the positioning and animation of articulated figures, the weighting strategy is the most frequent:

some typical examples are given by Badler et al. [11,12,13]. In the field of Robotics however, researchers have developed *task-priority strategies* to precisely arbitrate conflicts by establishing a clear priority order among the tasks[14,15,16,17,18]. In this family of approaches, the architecture of the optimization solver is simpler but allows to build the projection operator on the Jacobian Null space if the inverse has the required properties. The partition of the joint variation space into N(J) and its complementary offers the potential for two levels of priority; this was first exploited by Liégeois[14].

One of the first applications of this strategy in Computer Animation is due to Girard and Maciejewski[19]: the feet of multi-legged figures are constrained to remain in contact with the floor, while a kinematic locomotion model controls the coordination of the legs. Later, Maciejewski[20] highlighted the ill-conditioning of the Jacobian matrix near singular configurations and its frequent occurence in computer animation (whenever there are unreachable goals). The proposed damped least squares inverse exploits the Singular Value Decomposition[21] (SVD); instead of inverting small singular values, the method proposes to build a smooth function converging to zero when the singular value tends towards zero. Such a property has allowed the use of a two-priority-levels IK architecture for off-line motion retargeting[22]. The idea is to let the user specify temporary modification of the trajectory of some end effectors (in fact any part of the articulated structure). These are the high priority constraints while the original joint space trajectory is realized as much as possible at the lowest priority level. Apart from being off-line, this approach handles with difficulty the problem of continuity in the case of multiple effectors sharing a common subset of joints. Another work, restricted to posture optimization, proposed a multiple-priority-levels architecture for combining end effector control and center of mass position control[23] (i.e. to ensure the static balance of the figure). However, the number of strictly independent priority levels is limited to two[24]. A similar approach is used by Yamane *et al.* for the interactive edition of postures[25].

In the following section, we first show how to build robust projection operators for an arbitrary number of priority levels.

3. The Multiple Priority Levels IK

The present section first explains the most elementary IK architecture, then moves on to two priority levels and finally generalizes to *p* priority levels. In what follows we use the terms *constraint* and *task* interchangeably. The term *effector* designates a frame (attached to one of the bodies or to the center of mass of the articulated structure) to which we apply a constraint (either of position, orientation or both). In the next subsections, the key elements of the IK architecture will be illustrated on simple case studies in parallel with the outlining of the general equations.

3.1 The two priority levels architecture

Figure 3 shows the simplest redundant case with a two degrees of freedom chain $\{\theta_1,\theta_2\}$ and a one-dimensional constraint along the x dimension for the chain tip, i.e. the *effector*. The equation

$$x = f(\theta) \qquad (1)$$

giving the tip x coordinate as a function of the current posture is non linear. It is easy to establish such a function in the general case but, as pointed out in section 2.2, its inversion is possible only in specific, non redundant contexts.

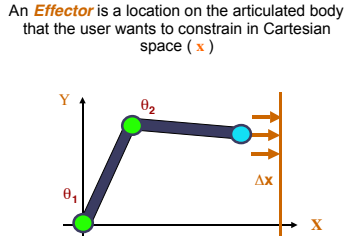


Figure 3 : *Simplest redundant case*

In the present example, the system is redundant; thus the IK control scheme is based on a linearization of equation (1). The resulting Jacobian matrix $\boldsymbol{J}$ is inverted to map the desired constraint variation $\boldsymbol{Dx}$ to a corresponding posture variation $\boldsymbol{Dq}$. We use the pseudo-inverse, noted $\boldsymbol{J}^{+}$, as it ensures desirable properties to build powerful projection operators. Among them, the norm of the solution mapped by $\boldsymbol{J}^{+}$ is minimal, i.e. it is the smallest posture variation realizing the desired constraint variation (Figure 4).

The validity of this operation is limited to small variations. Consequently, the initial gap between the tip position and the desired x value is broken down into small increments. The IK algorithm proceeds iteratively by looping through the linearization process for each update of the chain state. The construction of the Jacobian is an easy step for classical position or orientation constraints[1], so the major cost in this algorithm is the Jacobian inversion (in $O(m.n^2)$ where m is the dimension of the

constraint space and n is the dimension of the joint space). However, on the whole, it compares favorably with the IK algorithm based on the Jacobian transpose[26] because the quality of the convergence is much higher.

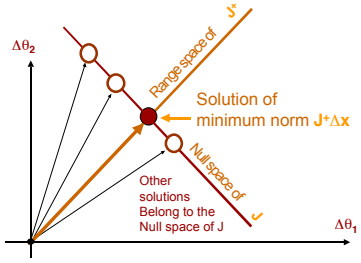


Figure 4: *The Range space of J^+ and the Null space of J are orthogonal.*

By definition, the Null space of the Jacobian, noted N(*J*), is mapped by *J* onto the null vector in the constraint variation space. Rephrased from a practical point of view, this means that a variation vector belonging to N(*J*) has no effect on the constraint. As already mentioned, Liégeois[14] proposed a two-priority-levels architecture by constructing a projection operator onto N(*J*) (Figure 5). This allows the optimization of an additional criterion expressed as a posture variation ***Dα***. The general form is:

$$\Delta\theta = J^{+}\Delta x + P_{N(J)}\Delta\alpha \qquad (2)$$

$$P_{N(J)} = I_n - J^{+}J \qquad (3)$$

with

Dq	**n**-dimensional posture variation
Dx	**m**-dimensional high priority constraints
J	**m** x **n** Jacobian matrix
J^{+}	**n** x **m** pseudo-inverse of ***J***
$P_{N(J)}$	**n** x **n** projection operator on **N(*J*)**
I_n	**n** x **n** identity matrix.
Dα	**n**-dimensional posture variation.

3.2 Handling Singularities

A problem with equation (2) is its instability around a singularity (e.g. in **Figure 3** when the x target value is about to become unreachable for the chain tip). More precisely, the norm of the first term grows to infinity in the immediate vicinity of the singularity. The Singular Value Decomposition[21] clearly highlights this problem and its solution.

The SVD of a **m**x**n** Jacobian matrix *J* of rank *r* is :

$$J = \sum_{i=1}^{r} \sigma_i u_i v_i^T \qquad (4)$$

where σ_i are the singular values (strictly positive), $\{u_i\}$ and $\{v_i\}$ are the basis respectively spanning the Range space of *J* and the complementary space of N(J). The expression of the pseudo inverse J^+ shows the strong influence of any small singular values, thus explaining the instability of the solution *around* the singularity:

$$J^{+} = \sum_{i=1}^{r} \frac{1}{\sigma_i} v_i u_i^T \qquad (5)$$

The solution[17,20] is to introduce a damping factor λ transforming the ill-behaved inverse term in (5) into a damped term converging smoothly to zero when the singular value becomes small:

$$J^{+\lambda} = \sum_{i=1}^{r} \frac{\sigma_i}{\sigma_i^2 + \lambda^2} v_i u_i^T \qquad (6)$$

When λ is zero, we obtain expression (5). By construction, the damped least-squares inverse $J^{+\lambda}$ trades the quality of the constraints satisfaction for an upper bound to the solution. A high value of λ guarantees a stable but slow convergence. The choice of a suitable value for λ thus is one of the subtle difficulties to yield a good compromise between robustness and efficiency. The low-cost under-optimal formula described by Maciejewski[27] gives good results.

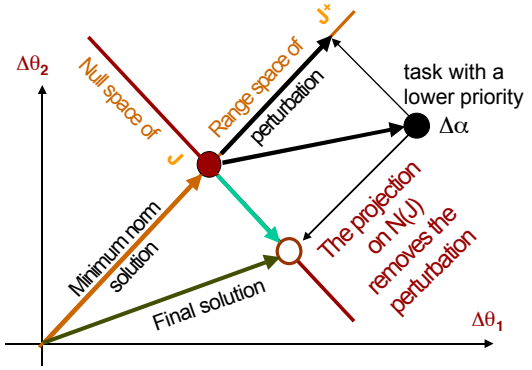


Figure 5: *The perturbation induced by the lower priority task Δα is removed, resulting in the homogeneous solution belonging to N(J).*

A question remains regarding the computation of the projection operator $P_{N(J)}$. It is a frequent mistake to use $J^{+\lambda}$ for its construction as this kind of inverse lacks various vital properties[24]. Instead, it must be

built according to equation (3), i.e. based on the peudo-inverse J^+. A low-cost evaluation is given by combining equations (4) and (5) which leads to:

$$P_{N(J)} = I_n - \sum_{i=1}^{r} v_i v_i^T \qquad (7)$$

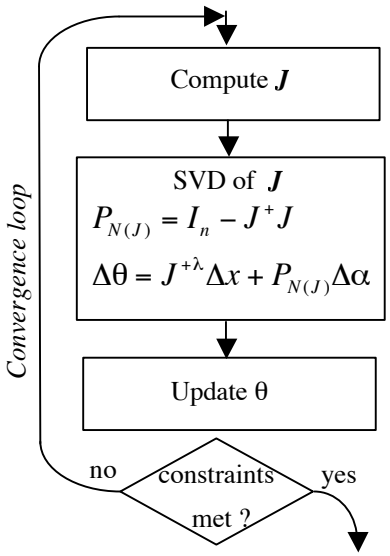


Figure 6: *Structure of the simplest IK algorithm*

A frequent criticism found in the literature about IK is its high computational cost. Indeed, one SVD is required per iteration but the efficiency and the smoothness of the convergence is guaranteed. In addition two points are worth noting to improve performance:

- a trimmed-down version of the SVD, the *thin* SVD[21], provides sufficient elements to compute expressions (6) and (7) at a lower cost.
- In the control of highly articulated figures, the dimension of the constraint space is usually much smaller than the dimension of the joint space (m << n). In this context, the thin SVD of J^T provides the necessary basis for our needs while being faster to compute than the thin SVD of J. A significant speed-up factor of $(n/m)^{3/4}$ can be obtained this way[27].

Figure 6 provides an overview of the IK convergence loop integrating the previously mentioned concepts. Its structure is now refined to take into account joint limits.

3.3 Handling joint limits

In Figure 6, the resulting joint variation is simply added to the current joint state. In real life, a joint value is generally bounded to a validity domain of the form [Min, Max]. A trivial approach to enforce the corresponding inequality constraints is to *clamp* the joint on the limit value whenever the IK update moves it beyond its validity domain.

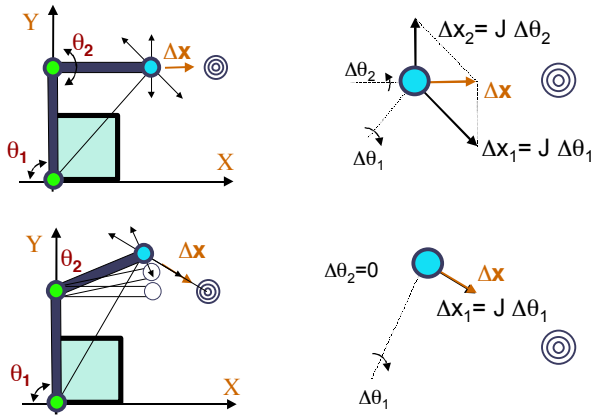


Figure 7*: Pathologic convergence with simple joint clamping in a singular context (top row: initial state, bottom row: final state)*

Unfortunately, this technique converges to a non-optimal final state[29] in singular contexts. Figure 7 illustrates the phenomenon on a simple chain when joint θ_1 reaches its limit (against the cube). This time, the task is two-dimensional for clarity : the tip has to reach a point on the plane. The drawing indicates the directions of the instantaneous effector variations induced by joint variations. IK provides the smallest joint variation ensuring that a desired Δx is the vector sum of such effector variations. The top right corner shows the solution found for the desired constraint variation Δx. However, as clamping leads to ignoring the $\Delta\theta_1$ variation, the forearm moves up over the next few iterations (bottom left corner) until the $\Delta\theta_2$ contribution is reduced to zero (bottom right corner).

The correct handling of joint limits requires an internal loop that checks and removes the joints that reach their limits (Figure 8). A small adjustment is also applied to the projector computation to make the new algorithm structure more readable: first a projector $P_{N(J_0)}$ is initialized with the identity matrix and all the joints are set to the free state, then we enter the clamping loop. After the IK solution evaluation, only the joints in the free state are examined. If a limit violation is detected for joint i, its value θ_i is clamped *on* the limit value θ_{Li}. There might be a small joint variation between the previous and the clamped joint value: we call it the clamping variation $\Delta\theta_{Ci}$. The constraint Δx has to be compensated for this clamping variation by subtracting the vector $J_i.\Delta\theta_{Ci}$ from its current value

(where $\underline{J_i}$ is the Jacobian column vector of joint i partial derivatives). Then joint i is removed from the problem by zeroing the Jacobian column $\underline{J_i}$ and the i diagonal term of $P_{N(J_0)}$. Finally its state is changed to locked and the clamping loop continues until no free joint violates its limits (Figure 8). The algorithm ensures the constraint minimization for an arbitrary number of simultaneous clamped joints.

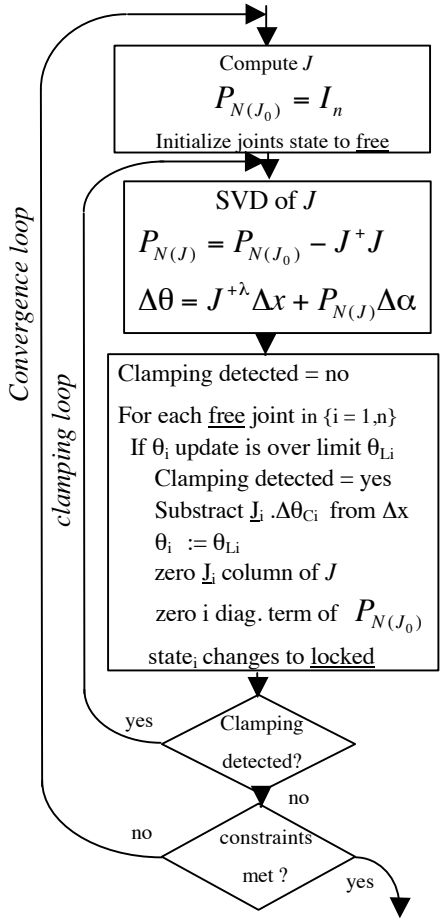


Figure 8: *IK loop including the management of inequality constraints (joint limits)*

3.4 Handling two conflicting constraints

Regarding the choice of the $\Delta\alpha$ term, it is often preferred to control a second end effector directly in Cartesian space rather than minimize a criterion expressed in joint space. In such a context, the two-priority-levels architecture has to be completed for two reasons:

- The solution of the high-priority constraint has some influence on the fulfillment of the low-priority constraint.
- The low-priority constraint can be best optimized if it knows how the high priority constraints restricts its potential (i.e. its Range).

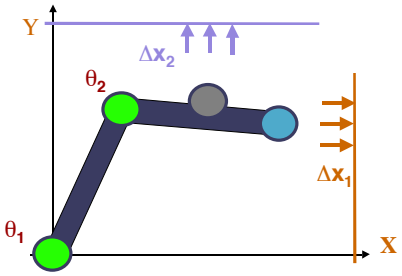


Figure 9: ***The simplest redundant case with two conflicting tasks***

We can still illustrate this important case in a simple 2D drawing (Figure 9). An additional effector, at mid-segment on the forearm, is associated to a second one-dimensional constraint with lowest priority: it has to reach a desired Y coordinate (the corresponding linearized task is noted Δx_2 for generality). Both tasks cannot be satisfied simultaneously. Note too that a posture variation realizing Δx_1 may have some influence on Δx_2.

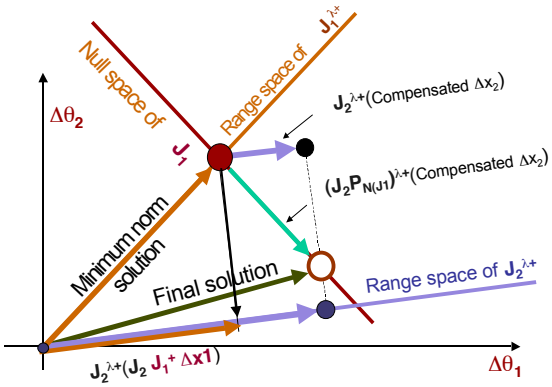


Figure 10: *The compensation and the restriction of J_2 allow the exact realization of both constraints*

The linearization leads to the construction of two Jacobians, respectively J_1 and J_2. Two mechanisms are used to build the low-priority solution[15,17] (Figure 10):

- Compensate the influence of the high priority task by removing its contribution to the secondary task: $J_2(J_1^+\Delta x_1)$.
- Restrict the low-priority Jacobian J_2 to the Range of N(J_1): $J_2 P_{N(J_1)}$. By construction, its pseudo-inverse maps the desired variation Δx_2 onto N(J_1), thus it is unnecessary to again apply the Null space projector $P_{N(J_1)}$.

The general expression takes into account the singularities, among which we find the additional *algorithmic singularity* that appears when the low-priority task can no longer be achieved when projected onto N(J_1) :

$$\Delta\theta = J_1^{+\lambda_1}\Delta x_1 + \left[J_2 P_{N(J_1)}\right]^{+\lambda_2}(\Delta x_2 - J_2(J_1^{+\lambda_1}\Delta x_1)) \quad (8)$$

3.5 Generalizing to *p* priority levels

The human structure being highly redundant, there exists a large solution space for the realization of more than two tasks. In the following the index *i* denotes the priority, 1 being the priority of highest rank and *p* being the total number of priority levels. The first general framework was described by Siciliano[18]. Our approach is based on this architecture. Nevertheless, we describe it with our more efficient evaluation of the projection operators. We also make some small adjustments to the clamping loop: now the constraint compensation is integrated in the first iteration of the priority loop. The priority management exploits a new Jacobian matrix, called the *Augmented* Jacobian and noted J_i^A. It piles up all the individual task Jacobians J_i from level one to level *i*. The structure appears in Figure 11. Note the initialization of a partial solution vector $\Delta\theta_0$ to zero and the priority index *i* to one. Then we enter the priority loop that adds the contribution of each priority level, from highest to lowest, into the partial solution vector $\Delta\theta_i$. After the loop, we add the criterion optimization term.

The contribution of each priority level *i* mimics formula (8) :

- $\Delta\hat{x}_i$ is the compensated task: the influence of the higher priority levels, from *1* to *i-1*, is removed;
- $\tilde{J}_i$ is the restricted Jacobian J_i to N(J_{i-1}^A).

The projection operator $P_{N(J_i^A)}$ is also updated for the next priority level *i*+1 or for the criterion optimization. When expressed as a function of the number of priority levels *p*, our recursive formulation has a linear computational cost compared to the quadratic cost of the previous formulation[18] (see the Appendix for a demonstration of our recursive formula). Its use results in a speed-up factor of roughly *(p+1)/2*, as shown by Baerlocher[28].

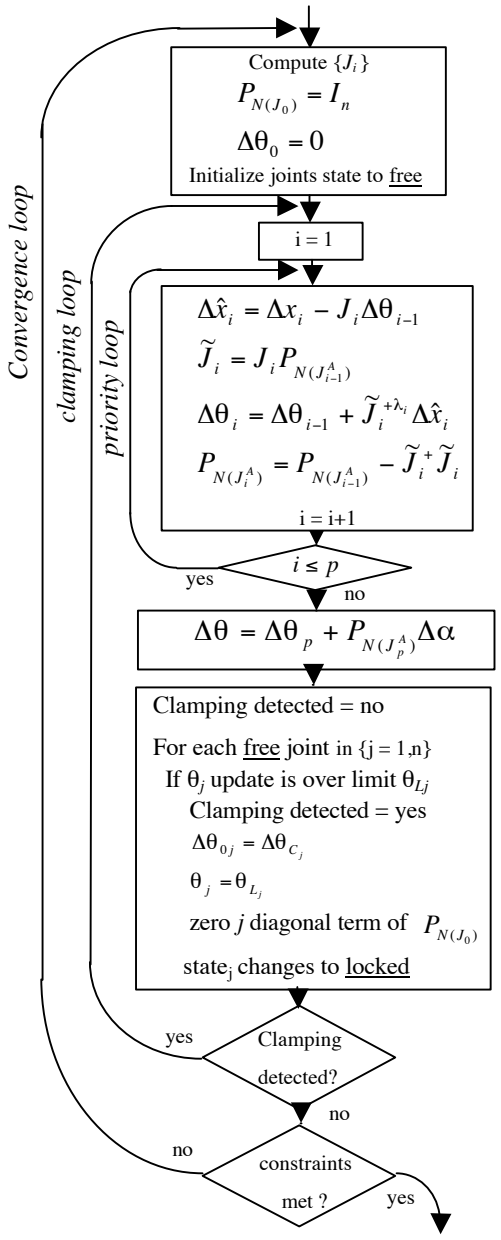


Figure 11: *General structure of the Multiple Priority Levels IK solver*

In its multiple-priority-levels version, the clamping algorithm nicely integrates with the priority

loop owing to the $\Delta\theta_0$ vector. This vector serves for the compensation of the highest priority constraints Δx_1 and is integrated within the partial solution vector $\Delta\theta_1$. Thus its effect is propagated to the lower priority levels by the priority loop.

3.6 Performance evaluation

Figure 12 compares measurements of the priority loop cost for Siciliano's versions and ours (in black). In this comparison, each additional priority level adds a five dimensional constraint space (for a fifty-dimensional joint space). Measurements were carried out on an SGI Octane with an R10000 processor (195 MHz).

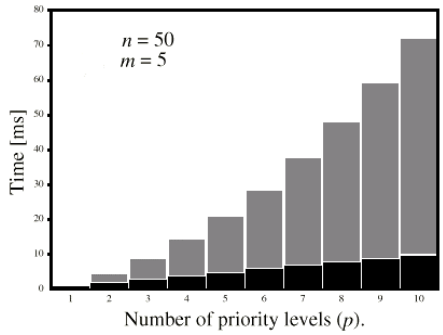


Figure 12: *Comparison of computing costs: $O(p^2)$ in gray vs $O(p)$in black*

Other tests have also compared the computing costs when a given number of constraints are gathered on one or more priority levels. Figure 13 shows a cost reduction when the constraints are distributed on several priority levels. This is due to the fact that the cost of the *transpose* Jacobian SVD (see section 3.2) is a function of m^2. Thus splitting m into p priority levels leads to a cost of $O(p.n.(m/p)^2) \sim O(n.m^2/p)$. This advantage is quickly compensated by the additional p-1 projector updates.

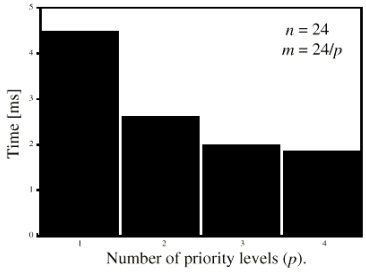


Figure 13: *Effect of introducing multiple priority levels for a problem of constant dimension*

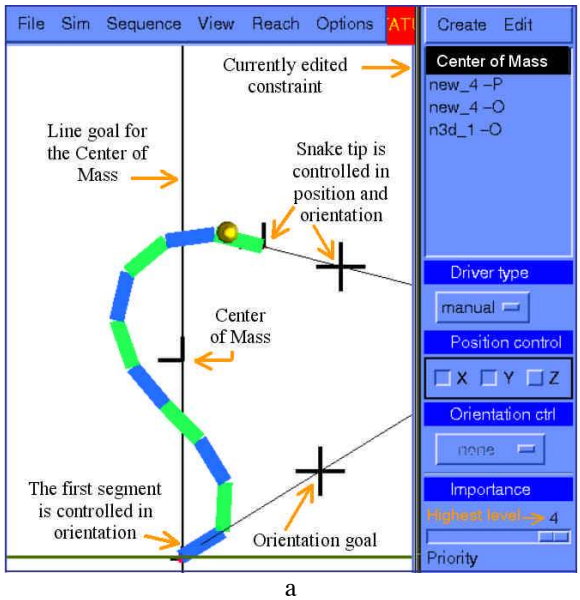
4. Interactive Optimization Framework

The efficiency and robustness of the Multiple Priority Levels IK algorithm have been thoroughly tested with an interactive optimization framework where any aspect of the optimization context can change while the convergence runs in the background. The user is able to add and remove end effectors, as well as change their relative priorities, at any time (multiple tasks can also share the same priority level). Similarly the lowest level cost function is activated or deactivated whenever necessary.

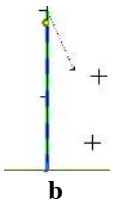
The interactive test application producing the present results runs on an SGI ONYX on one MIPS R10000 processor (195 MHz). In the following 2D and 3D examples, we compare the convergence of both weighted IK and prioritized IK. The first two illustrations demonstrate the quantitative and qualitative superiority of the prioritized strategy.

4.1 A simple articulated chain

The posture of a simple twenty degree of freedom snake-like chain is optimized to minimize a continuity cost function under four Cartesian constraints (Figure 14). The continuity cost function is simply the sum of the squared joint angles which is null when the chain is a straight line.



a



b

Figure 14: *The snake-like chain (a); the center of mass has to reach the vertical line passing through the first segment base; the other constraints are set on the first segment and the snake tip. The initial posture and the goals (b)*

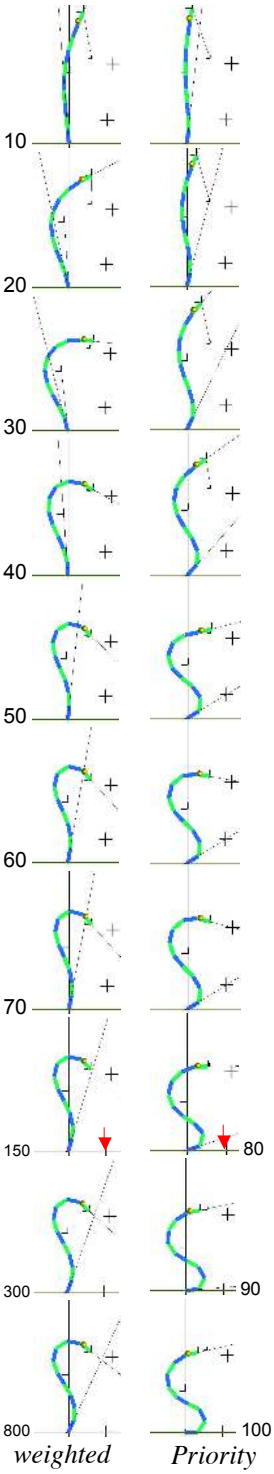


Figure 15: *Convergence comparison*

Both strategies are compared with the same set of four constraints (center of mass position, base orientation, tip position and orientation), the same initial posture and the same goals (Figure 14b). In the weighted strategy, all constraints have the same priority. In the priority strategy, the decreasing priority order is: Center of Mass position, base orientation, tip position, tip orientation (note that the tip position and orientation constraints can have distinct priority levels).

Figure 15 first shows the convergence of both strategies every ten steps up to 70 iterations.

The weighted approach is shown in the left column; its convergence leads to a compromise where only half of the constraints are met after 70 iterations. It is close to a local minima due to the two conflicting orientation constraints. As a consequence, convergence is very slow.

On the other hand, the prioritized IK preserves the fulfillment of the center of mass constraint during the entire convergence. The tip position and the base orientation goals are reached after 50 iterations. The lowest priority tip orientation constraint is met after 70 iterations.

In a second stage, the snake base orientation goal is changed leading to a singular context because some constraints cannot be met. However, the hierarchy of the constraints is still preserved. Thus the center of mass goal and the base orientation goals are still fulfilled. The last three images stress the effectiveness of the error minimization on the tip position and orientation (final posture after 30 iterations). The weighted approach is even slower in this second context; over 600 iterations are not sufficient to converge to a posture with a similarly small error norm.

At that stage it is interesting to note that enforcing a hierarchy of constraint priorities may help to avoid some classes of local minima. This can be explained by the fact that each priority layer introduces additional limitations in the solution space. For this reason the highest priority constraint has an important role in the determination of the convergence direction. Our examples stress the usefulness of the center of mass constraint; more tests should be performed to derive general guidelines for systematically obtaining believable human postures.

4.2 A human articulated structure

In the following examples, we use a simplified human model with forty-two degrees of freedom (no hand mobility). Each degree of freedom appears as a gray square in the top left corner of Figures 17 to 19; it may temporarily show up in yellow when reaching its range limit.

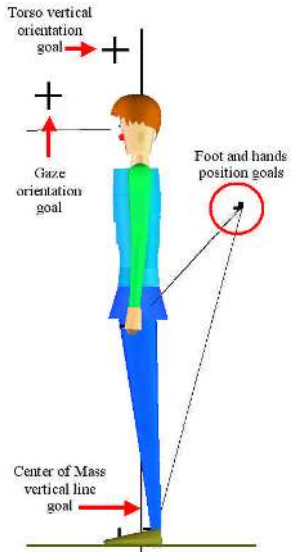


Figure 16: *Six constraint goals for editing a dance posture: the center of mass must remain on a vertical line passing through the right foot, the gaze must focus on a cross, a vertical line attached to the torso must pass through another cross, both hands and the left foot must reach goal positions.*

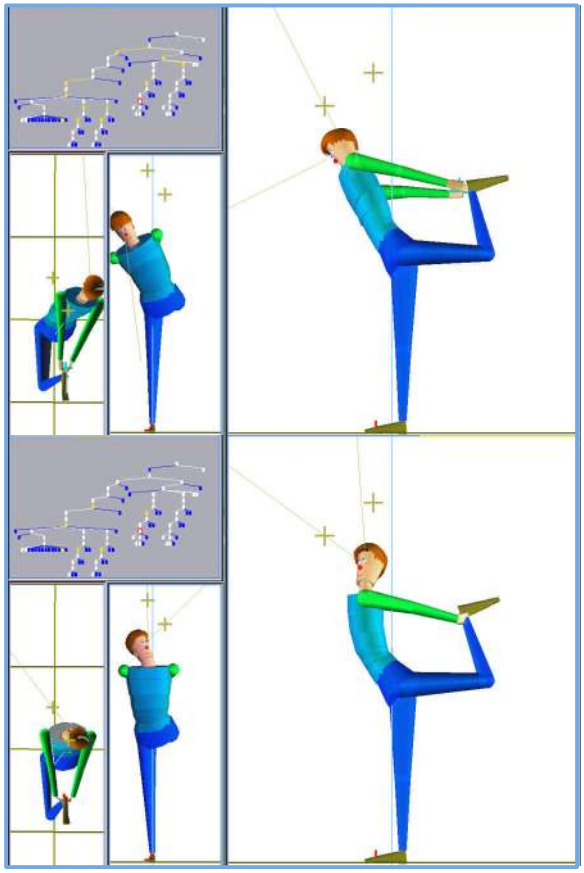


Figure 17: *Convergence of the weighted approach after 50 and 100 iterations. Note the intermediate unbalanced posture and the final compromise solution for the torso orientation.*

The mass distribution is best visible on Figure 16 over the segment volumes, given a constant homogeneous density. The articulated figure is rooted at the right foot, which remains fixed in a world coordinate system (the choice of the root is interactive).
It is important to note that all the joints participate in the satisfaction of all the constraints as opposed to approaches relying on skeleton partitioning with body segments dedicated to specialized tasks (e.g. : arm for reaching, neck for the gaze, torso for balance, etc...). By construction, our architecture fosters the convergence towards synergistic postures, e.g. the gaze constraint is distributed over the whole body. If needed, we can select and associate a sub-set of joints for a given task, but this is not the case in the following examples.

4.2.1 Convergence comparison
First, we compare the convergence of both approaches on the same set of constraints with the same set of goals, as illustrated on Figure 16.

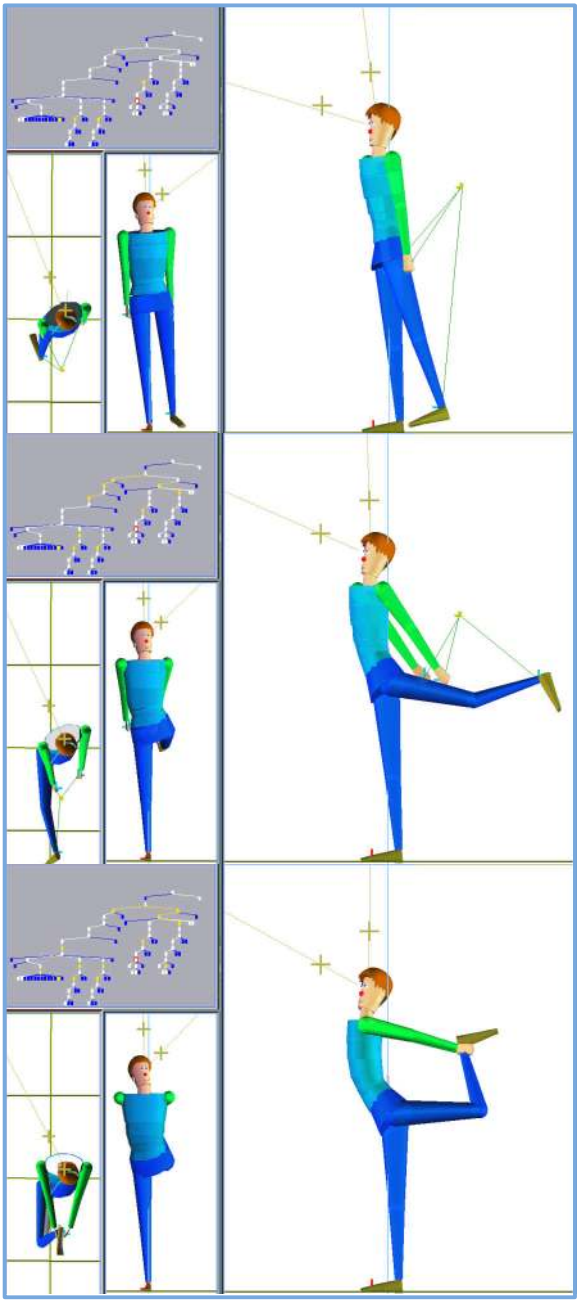


Figure 18: *Convergence of the prioritized IK successively after 5, 35 and 75 iterations.*

The character has to achieve a balanced dance posture requiring a high suppleness. As a consequence, some joints like the left hip, the spine or the neck may reach their range limit during the convergence (indicated with the yellow colored squares in the top left corner). For that reason, the clamping loop is the key for obtaining a correct and stable convergence. As it is exploited for both approaches, the only difference in the convergence

behavior comes from the presence or absence of priority levels associated to the constraints.
Figure 17 shows the weighted IK approach where all constraints are handled at the same priority level. The intermediate image, after 50 iterations, still betrays an unbalanced posture. It is also difficult to predict which constraint will be met first. Finally, the torso vertical orientation constraint is only partially met after 100 iterations.
Figure 18 confirms the enforcement of the hierarchy of prioritized constraints during the convergence together with its smaller computational cost. The center of mass constraint is satisfied right at the beginning as it has the highest priority. The torso and gaze constraints have distinct decreasing priority levels, themselves being higher than the hands and left foot constraints (all sharing the same low priority level). The middle and last images, after respectively 35 and 75 iterations, reflect the priority hierarchy. The final posture better corresponds to a classic dance posture.

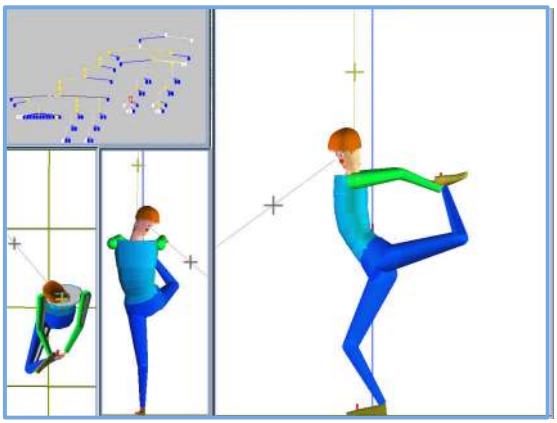


Figure 19: *Lowering the gaze constraint's goal leads to flexing the right leg.*

From a qualitative point of view, it is essential to be able to enforce visually important constraints both at first *and* in a synergistic fashion. A sequential enforcement of prioritized constraints would be a waste of computing resources and would lead to the most primitive robotic-like behavior. On the contrary, we want to offer an optimization scheme where constraints can be activated on the fly with time varying goals and/or priority levels. Our proposition goes in this direction as illustrated in Figure 19 where the gaze goal is changed to a lower position. As a consequence, the whole posture is adjusted synergistically so that the gaze constraint is met but without perturbating the higher priority constraints (center of mass and torso). This explains why the leg is flexed: the neck and spine degrees of freedom having reached their range limit, they are removed by the clamping loop, so only right leg mobility remains to allow the gaze to pass through the lowered cross.

4.2.2 Complex constrained postures

The second 3D case study deals with reachability issues. Table 1 lists all the end effectors and the associated constraints activated permanently or temporarily during that session. In addition, the lowest priority term $\Delta\alpha$ (expressed in the joint space) was activated from time to time. For the present case study we have exploited the following cost functions to globally adjust the posture: leg-flexion, leg-extension or leg-symmetry.

End effector	Purpose of the constraint	rank
Left toe	maintain contact with the floor (position)	1
Eyes	orienting towards the right hand (lookat-orientation)	1
Left hand center	holding onto the vertical bar (position and orientation)	1
Center of mass	projecting on a point or a line between the feet	2
Right hand center	reaching the sphere (Figure 1) or interactively following a target (Figure 20) (position)	3
Pelvis, elbow, knee	temporarily locally readjust the posture (position)	4
Optimization term $\Delta\alpha$	temporarily globally readjust the posture.	5

Table 1: *Constraints with their rank (rank is analog to the i index in Figure 11, rank 1 represents the highest priority.*

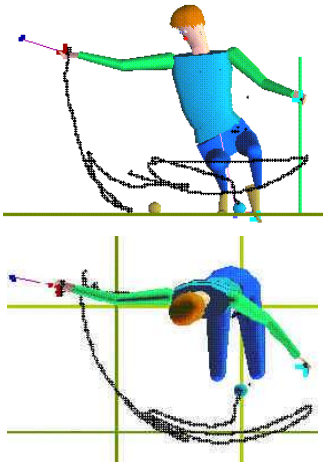


Figure 20: *Interactive optimization with four priority levels and cost function optimization.*

The initial posture is standing up. The posture in Figure 1 is obtained from a first convergence stage compliant with the specification of Table 1. Then, starting from that posture, we explore the right hand reach potential under the same set of constraints. The resulting trace appears in Figure 20. This example stresses the potential of our architecture for ergonomic evaluations.

The last illustration (Figure 21) illustrates the possibility to apply constraints to accessories such as an umbrella, work tools, swords etc.... In this specific case the umbrella handle orientation is constrained to remain vertical.

5. Discussion and Conclusion

There are numerous methods to solve the inverse kinematics problem. Each has its own advantages and drawbacks. We have retained the criteria of generality, robustness with respect to singularities and efficiency.

Compared to simpler analytical methods, the price to pay for this generality are a higher computational cost and complexity of the resolution methods (due to their iterative nature), and a lower reliability since the convergence towards a globally optimal solution is not always guaranteed. However, our coherent handling of joint limits and singularities ensures a stable convergence. Our contribution to complexity reduction allows our IK solver to run at interactive rates for the manipulation of complex articulated structures.

Regarding the specific case of human-like character positioning, realistic results are obtained if a sufficiently detailed model is provided (e.g. with joint limits). It is therefore vital to have good data and models for the joint limits. This modelling task requires an additional effort but it is a rewarding one as such data guide the convergence towards more believable postures.

A specific difficulty has to be mentioned for the human limbs: it sometimes happens that, while interacting, the legs or arms converge to the fully extended posture (a locally singular configuration). Our experience is that such a fully extended posture has a tendency to perdure unless we explicitly act on it through the lowest level optimization term $\Delta\alpha$ (hence the purpose of our leg-flexion, leg-extension cost functions). This behavior is due to the first order nature of our architecture; we have to introduce other mechanisms to overcome locally locked postures. Once this is done, we plan to explore the potential of the architecture for motion retargeting.

Acknowledgements

The authors would like to thank Lorna Herda for proofreading. This work has been realized with the support of the Swiss National Foundation of Research under the grants N°2000-053809 and N°20-61999.00 .

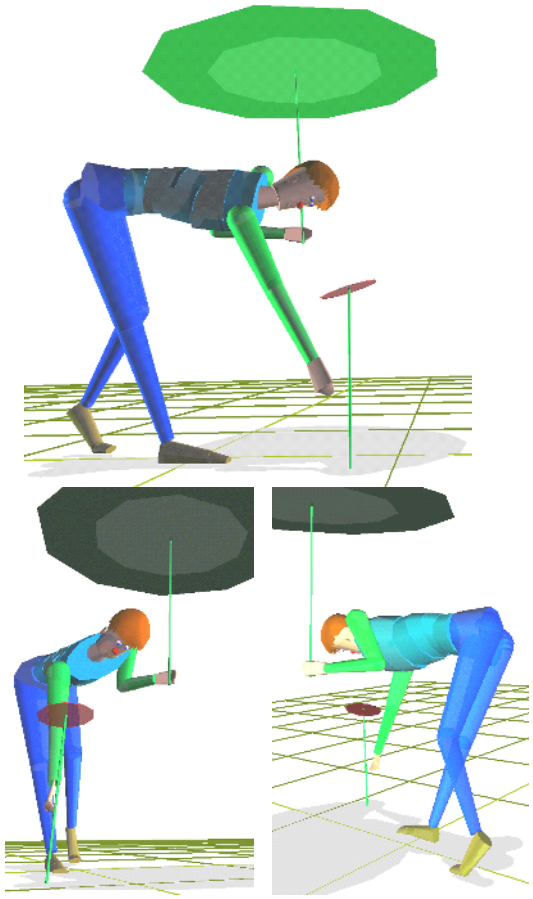


Figure 21: *A combined set of constraints involving balance, reach, gaze while holding the umbrella vertically (with four priority levels).*

Appendix

Proposition: Given an ***m***x***n*** matrix J_i^A partitioned as $\begin{bmatrix} J_{i-1}^A \\ \hline J_i \end{bmatrix}$, the following identity holds:

$$I_n - J_i^{A+} J_i^A = (I_n - J_{i-1}^{A+} J_{i-1}^A) - \widetilde{J}_i^+ \widetilde{J}_i$$

where $\widetilde{J}_i = J_i (I_n - J_{i-1}^{A+} J_{i-1}^A)$

Demonstration: A result due to R. Cline[30] yields the pseudo-inverse of a partitioned matrix. It is applied to J_i^A :

$$J_i^{A+} = (J_{i-1}^{A+} - T_i\ J_i\ J_{i-1}^{A+}, T_i)$$

where $T_i = \widetilde{J}_i^+ + X(I - \widetilde{J}_i\ \widetilde{J}_i^+)$ and X is a complex term not useful here.

The identity can now be established:

$$I_n - J_i^{A+} J_i^A = I_n - (J_{i-1}^{A+} - T_i\ J_i\ J_{i-1}^{A+}, T_i) \begin{bmatrix} J_{i-1}^A \\ J_i \end{bmatrix}$$
$$= I_n - J_{i-1}^{A+} J_{i-1}^A - T_i\ J_i\ (I_n - J_{i-1}^{A+} J_{i-1}^A)$$
$$= I_n - J_{i-1}^{A+} J_{i-1}^A - T_i\ \widetilde{J}_i$$
$$= I_n - J_{i-1}^{A+} J_{i-1}^A - (\widetilde{J}_i^+ + X(I - \widetilde{J}_i\ \widetilde{J}_i^+))\widetilde{J}_i$$
$$= (I_n - J_{i-1}^{A+} J_{i-1}^A) - \widetilde{J}_i^+ \widetilde{J}_i$$

In the last step, the pseudo-inverse property $JJ^+J = J$ is used.

References

1. A. Watt, M. Watt, "Advanced Animation and Rendering Techniques", Addison-Wesley, ACM Press, 1992
2. J.K. Hodgins, N.S. Pollard, "Adapting Simulated Behaviors for New Characters", Proc. of SIGGRAPH'97, pp 153-162, Los Angeles, 1997
3. R. Bindiganavale R., N. Badler, "Motion Abstraction and Mapping with Spatial Constraints", Proc of Captech98, LNAI 1537, pp 70-82, Springer-Verlag Berlin Heidelberg
4. J. Lee and S.Y. Shin, "A Hierarchical Approach to Interactive Motion Editing for Human-Like Figures", Proc. of SIGGRAPH'99, Los Angeles
5. Z. Popovic and A. Witkin, "Physically Based Motion Transformation", Proc. of SIGGRAPH'99, Los Angeles
6. S. Tak, Song O-Y., Ko H.-S.,"Motion Balance Filtering", proc. of Eurographics'2000, Computer Graphics Forum, vol 19(3), Blakwell publishers 2000
7. M. Gleicher "Comparing Constraint-based Motion Editing methods", Graphical Models 63, 107-134, Academic Press, The Journal of Vizualisation and Computer Animation, 9, 65-94 (1998)
8. K.-J. Choi, H.-S. Ko,"Online motion retargeting". Journal of Visualization and Computer Animation 11(5): 223-235 (2000)
9. J.U. Korein, "A Geometric Investigation of Reach", The MIT Press, Cambridge, 1985
10. D. Tolani, A. Goswami, N. Badler, "Real-time inverse kinematics techniques for anthropomorphic arms", Graphical Models, Vol. 62, pp. 353 - 388, 2000
11. J. Zhao, N. Badler, "Inverse Kinematics Positioning using Nonlinear Programming for Highly Articulated Figures", ACM Transactions on Graphics, Vol. 13, No. 4, pp. 313 - 336, Oct. 1994
12. C.B. Phillips, N Badler," Interactive Behaviors for Bipedal Articulated Figures" Computer Graphics 25 (4), pp 359-362, July 1991
13. N.I. Badler, K.H. Manoochehri, G. Walters, "Articulated Figure Positioning by Multiple Constraints", IEEE Computer Graphics & Applications, Vol. 7, No. 6, pp. 28 - 38, June 1987
14. Liégeois A., "Automatic Supervisory Control of the Configuration and Behavior of Multibody Mechanisms", IEEE Transaction on Systems, Man and Cybernetics, Vol. SMC-7, N°12, 1977, pp 868-871
15. H. Hanafusa, T. Yoshikawa, Y. Nakamura, "Analysis and Control of Articulated Robot with Redundancy", IFAC, 8th Triennal World Congress, Vol. 4, pp. 1927 - 1932, 1981
16. Maciejewski A.A., Klein C. A., "Obstacle Avoidance for Kinematically Redundant Manipulators in Dynamically Varying Environments", The International Journal of Robotics Research, pp109-117, Vol. 4. N°3, 1985
17. Y. Nakamura, H. Hanafusa, "Inverse Kinematic Solutions with Singularity Robustness for Robot Manipulator Control", Journal of Dynamic Systems, Meas., and Control, Vol. 108, pp. 163 - 171, Sept. 1986
18. Siciliano B. and Slotine J-J. "A General Framework for Managing Multiple Tasks in Highly Redundant Robotic Systems", Proc. of ICAR'91, Vol 2, pp 1211-1215, ISBN 0-7803-0078-5, 1991
19. M. Girard, A.A. Maciejewski, "Computational modeling for the computer animation of legged figures", Proc. of SIGGRAPH'85, Computer Graphics, Vol 19, pp 263-270
20. A.A: Maciejewski, "Dealing with the ill-Conditioned Equations of Motion for Articulated Figures", IEEE CGA, Vol. 10, n°3, pp 63-71, 1990
21. W. Press, S. Teukolsky, W. T. Vetterling, B. P. Flannery, "Numerical Recipes in C", second edition, Cambridge University Press, ISBN 0 521 43108 5, pp59-70, 1992
22. J.-S. Monzani, P. Baerlocher, R. Boulic, D. Thalmann, "Using an Intermediate Skeleton and Inverse Kinematics for Motion Retargeting", Proc. Eurographics 2000, Interlaken August 2000
23. R. Boulic, R. Mas, D. Thalmann, "A Robust Approach for the Center of Mass Position Control

with Inverse Kinetics", Journal of Computers and Graphics, Vol. 20, No. 5, pp. 693 - 701, 1996.

24. P. Baerlocher, R. Boulic., "Task-Priority Formulations for the Kinematic Control of Highly Redundant Articulated Structures", Proc. of IEEE IROS 98, Victoria, pp. 323-329, Oct. 1998.
25. K. Yamane, Y. Nakamura, "Synergetic CG Choreography through Constraining and Deconstraining at Will", Proc. of IEEE ICRA 2002, Vol. 1, pp. 855-862, Washington D.C., U.S.A., May, 2002.
26. C. Welman, "Inverse Kinematics and Geometric Constraints for Articulated Figure Manipulation", Master Thesis, Simon Fraser University, 1993.
27. A.A. Maciejewski, C.A. Klein, "Numerical Filtering for the Operation of Robotic Manipulators through Kinematically Singular Configurations", Journal of Robotic Systems, Vol. 5 N°6, pp. 527-552, 1988
28. P. Baerlocher, "Inverse Kinematics Techniques for the Interactive Posture Control of Articulated Figures", PhD Thesis N°2383, Swiss Federal Institute of Technology, Lausanne, EPFL, Switzerland, April 2001
29. R. Boulic, R. Mas, D. Thalmann, "Interactive Identification of the Center of Mass Reachable Space for an Articulated Manipulator", *Proc. of International Conference of Advanced Robotics ICAR '97*, Monterey, pp. 589-594, July 1997.
30. R.E. Cline, " Representation for the Generalized Inverse of a Partitioned Matrix ", Journal of Soc. Industrial Applied Mathematics, XII, pp. 588-600, 1964.