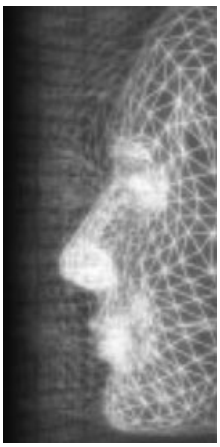
COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds (in press)
Published online in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/cav.176

Analytical inverse kinematics with body posture control



*By Marcelo Kallmann**

This paper presents a novel whole-body analytical inverse kinematics (IK) method integrating collision avoidance and customizable body control for animating reaching tasks in real-time. Whole-body control is achieved with the interpolation of pre-designed key body postures, which are organized as a function of the direction to the goal to be reached. Arm postures are computed by the analytical IK solution for human-like arms and legs, extended with a new simple search method for achieving postures avoiding joint limits and collisions. In addition, a new IK resolution is presented that directly solves for joints parameterized in the swing-and-twist decomposition. The overall method is simple to implement, fast, and accurate, and therefore suitable for interactive applications controlling the hands of characters. The source code of the IK implementation is provided. Copyright © 2007 John Wiley & Sons, Ltd.

Received: 23 August 2006; Revised: 25 February 2007; Accepted: 20 April 2007

KEY WORDS: analytical inverse kinematics; character animation; reaching

Introduction

Inverse kinematics (IK)[1] is a widely used technique for controlling the arms of human-like characters in interactive applications, and is employed in several domains: ergonomics, virtual reality, computer games, animation, etc. For instance, nearly all professional animation packages have built-in IK solvers for manipulating skeleton hierarchies.

This paper proposes a new IK method suitable for reaching tasks performed by autonomous and interactive virtual humans.[2] The basic problem is to pose the character in such a way that its hand reaches an arbitrary position and orientation in space. As the main interest is in interactive applications, the presented method is only based on posture interpolation and analytical solvers, so there is no convergence time required and the achieved solutions are exact.

The method is composed of two phases. The first phase consists of blending pre-designed key body postures, which are organized only as a function of the goal position to be reached, in relation to the shoulder frame. The goal position is described by a two-parameter space based on the swing parameterization of the shoulder. By making use of a planar triangulation in the swing plane, it becomes very efficient to determine the key postures to be blended for a given goal position. Furthermore an intuitive user interface for the design of new body behaviors is achieved.

After the interpolated body posture has been determined, the second phase consists of analytically solving the IK of the arm linkage[3] in order to exactly reach the given goal. For this purpose, I propose a new formulation of the arm IK that directly solves for joints parameterized in the swing-and-twist decomposition,[4] and that includes an automatic search mechanism for determining the swivel angle, handling collision avoidance, and joint limits based on spherical ellipses.

As a result, the overall method is able to achieve a coordinated and continuous whole-body motion as a function of the character's hand target. As there are no numerical methods employed, the system always accurately returns if the goal was reached or otherwise the exact reason of failure: not reachable, violating limits, or in collision. Furthermore, the method proposes a fast and intuitive way of controlling the generated body postures through the design of few example key body postures.

After reviewing related work, I start describing the new formulation of the arm IK, and then I proceed on to the posture interpolation process and the overall proposed method.

*Correspondence to: M. Kallmann, University of California, Merced, 5200 N. Lake Road, Merced, CA 95343, USA.
E-mail: mkallmann@ucmerced.edu

Related Work

The basic IK problem of finding a character pose satisfying given constraints is well known in computer animation.[1] Applications are mainly related to interactive control of characters, and adaptation of motion capture data,[5–8] but several other uses exist, as for example to animate hands.[9]

The problem is usually underdetermined, and therefore additional criteria are required for choosing a solution amidst all possible ones. A popular approach is to use additional constraints,[10,11] for example to be as close as possible to a given reference pose, and then use numerical Jacobian-based methods.[12,13] At first, the ability to add arbitrary constraints is appealing, however such systems can quickly become difficult to control and prone to get stuck in local minima. One alternative is to use analytical IK solutions,[3,14] which are fast and accurate, however not addressing arbitrary linkages or constraints.

Because of its unsurpassed suitability to real-time applications, the analytical IK is adopted in this work and several extensions are proposed here to overcome its limitations.

1. The original analytical solution[3,14] proposes resolution equations based on joints parameterized by Euler angles. Here I present a new set of equations for the IK resolution which is based on quaternion algebra and the swing-and-twist parameterization, more suitable for defining meaningful joint limits.
2. An automatic method that determines the elbow position and the *swivel angle* is presented that also takes into account avoidance of collisions and joint limits.
3. Finally, in order to address arbitrary skeletons, I also propose in this paper to combine the analytical solution with a posture blending mechanism.

Among the extensions listed above, it is with blending of body postures that the proposed method is able to generate full-body postures.

Several methods have been proposed based on blending example motions from databases of designed or motion captured motions.[15–17] In particular, some works have addressed the reaching problem based on example motions.[18–21] The main drawback of such approaches is the required large number of example motions for covering all workspace and the need to tweak interpolation parameters. Furthermore, interpolation of example data alone does not give exact placements of the end-effector at desired locations and additional IK procedures are required to refine the result.

More sophisticate systems include learning mechanisms from given data for better addressing and modeling different styles of motions.[22–24] In particular the work proposed by Grochow *et al*.[22] is able to learn body motion styles from given motion data and a numerical IK solver is able to satisfy given constraints, giving preference to solutions which are closer to the learned styles.

In contrast, this paper proposes an analytical solution for achieving customizable whole-body IK. By combining the analytical IK with a posture blending mechanism it becomes possible to both achieve customizable whole-body postures and efficiently compute exact placements for the character's hand. The aim of this work is to propose a simple method to implement that is robust and suitable for interactive applications. As it is based on analytical solutions and simple interpolation, the proposed method represents the fastest existing approach to whole-body IK. The next section defines the framework on which the new arm IK formulation is presented.

Framework

In the analytical IK formulation, arms and legs are modeled with joint hierarchies (or *linkages*), each consisting of three joints. They are called here as: the base, mid, and end joints. In the arm case these joints are the shoulder, elbow, and wrist; and in the leg case they are the hip, knee, and ankle.

In this work, each joint has a local frame with the $\hat{\mathbf{z}}$ axis lying along the main axis of the corresponding limb, and with the positive direction pointing toward the child joint. For achieving intuitive mappings, the $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ axes can be placed differently in each linkage. For the right arm case (Figure 1) the $\hat{\mathbf{y}}$ axis is vertical with the positive direction pointing upwards and the $\hat{\mathbf{x}}$ axis is obtained with the cross product $\hat{\mathbf{x}} = \hat{\mathbf{y}} \times \hat{\mathbf{z}}$. The IK solution

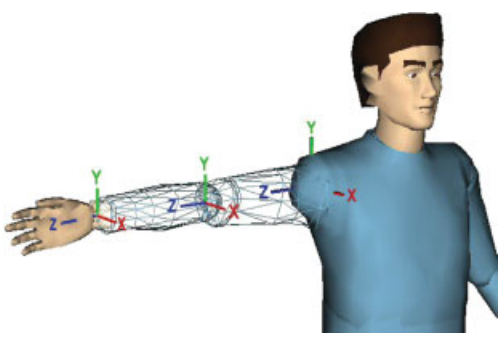


Figure 1. Local frames and the zero-posture of the right arm.

DOI: 10.1002/cav

presented in this paper considers local frames and the zero-posture is placed as in the right arm case of Figure 1.

Parameterization

Each linkage has seven degrees of freedom (DOFs): three in the base joint, and two in the mid and end joints. A linkage configuration can be therefore determined with a set of seven values denoted by vector $(v_0, \ldots, v_6)$. The parameterization of each joint is now specified.

The base is always a ball-and-socket joint with three DOFs and is parameterized with the swing-and-twist decomposition.[4] The swing rotation has the rotation axis always perpendicular to the $\hat{\mathbf{z}}$ axis and is represented with the 2D axis-angle (v_0, v_1) lying in the x–y plane of the base local frame, where (v_0, v_1) is the rotation axis and $\|(v_0, v_1)\|$ is the rotation angle. The twist rotation that follows is simply a rotation around the $\hat{\mathbf{z}}$ axis of angle v_2.

The mid joint has two DOFs and is parameterized with two Euler angles, which are the flexion and the twist rotations of the elbow or knee joint. The flexion is a rotation of angle v_3 around the $\hat{\mathbf{y}}$ axis and the twist is a rotation of angle v_4 around the $\hat{\mathbf{z}}$ axis of the local frame.

Note that in a more anatomical representation the mid twist would be applied to an extra joint placed between the mid and the end joints. The use of such joint is also beneficial for achieving better skinning weights for skinned characters. The presented formulation covers this situation: as the mid twist is applied after the mid flexion, it does not matter if it is applied to the mid joint itself, or to an extra joint between the mid and end joints.

Finally, the end joint is also parameterized by a swing rotation represented with a 2D axis-angle (v_5, v_6) in the x–y plane of the end joint local frame. Therefore, (v_5, v_6) is the rotation axis and $\|(v_5, v_6)\|$ is the rotation angle of the end joint swing rotation.

The next section presents the new formulation of the arm IK, directly solving the IK using the described parameterization based on swing, twists, and Euler angles.

Swing-and-Twist Arm IK

The original implementation of the analytical IK solution[3] was developed for joints parameterized with Euler angles, which suffer from *gimbal lock* when applied to ball-and-socket joints with range of rotations greater than $\pi/2$, as is usually the case for the shoulder and hip joints. In addition, anatomically plausible limits usually cannot be specified with minimum and maximum Euler angles.

Fortunately, the swing-and-twist rotation decomposition (also known as the *exponential map*[4]) is able to address these problems and to define a singularity-free rotation parameterization inside meaningful range limits. In order to avoid conversions between different parameterizations, I present now a new formulation based on the swing-and-twist parameterization.

The new formulation is presented in the next subsections. The first two subsections (swivel angle and mid flexion) summarize results from Reference [3] that serve as starting point to the new formulation, which is introduced in the subsequent subsections.

Swivel Angle

Let point $\mathbf{e} \in \mathbb{R}^3$ and quaternion $\mathbf{q_e} \in \mathbb{S}^3$ be the desired position and orientation of the end joint in the base joint frame coordinates. Note that each arm and leg linkage has seven DOFs, one more DOF than the number of constraints. The extra degree of freedom is parameterized with the so-called *swivel angle* ϕ,[25] which tells how much the mid joint is rotated around the end-base axis. Figure 2 shows the mid joint orbit circles of each linkage.

In order to specify ϕ, a coordinate system is created by defining two unit vectors $\hat{\mathbf{u}}$ and $\hat{\mathbf{v}}$ forming a local

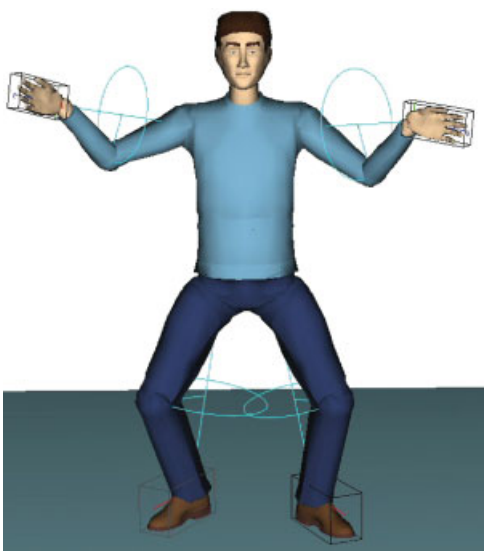


Figure 2. By varying the swivel angle ϕ in each linkage, different positions for the mid joint are obtained along the mid joint orbit circle, all satisfying the goal end joint position and orientation.

DOI: 10.1002/cav

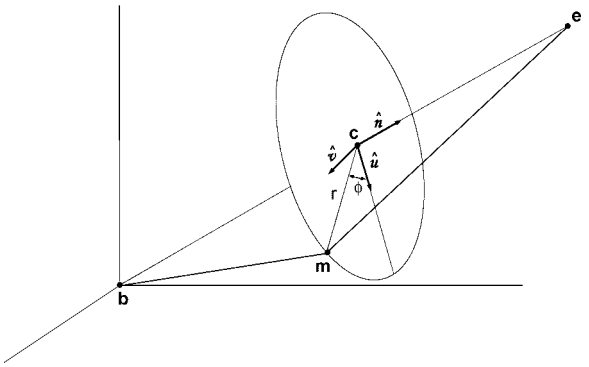


Figure 3. The mid joint orbit circle and the swivel angle ϕ.

coordinate system for the plane containing the orbit circle. Vector $\hat{\mathbf{u}}$ is defined as the projection of $-\hat{\mathbf{y}}$ onto the orbit circle plane, and $\hat{\mathbf{v}}$ is orthogonal to $\hat{\mathbf{u}}$ and the normalized end-base vector $\hat{\mathbf{n}}$:

$$\hat{\mathbf{u}} = \frac{-\hat{\mathbf{y}} + (\hat{\mathbf{y}} \cdot \hat{\mathbf{n}})\hat{\mathbf{n}}}{\left\|-\hat{\mathbf{y}} + (\hat{\mathbf{y}} \cdot \hat{\mathbf{n}})\hat{\mathbf{n}}\right\|}$$

$$\hat{\mathbf{v}} = \hat{\mathbf{u}} \times \hat{\mathbf{n}}$$

$$\hat{\mathbf{n}} = \frac{\mathbf{e}}{\|\mathbf{e}\|}$$

The center $\mathbf{c}$ of the orbit circle, and its radius r, can be derived by trigonometry as follows (see Figure 3):

$$\mathbf{c} = \hat{\mathbf{n}} d_1 \cos\alpha$$

$$r = d_1 \sin\alpha$$

where (see Figure 4):

$$\cos\alpha = \frac{{d_2}^2 - {d_1}^2 - \|\mathbf{e}\|^2}{-2d_1\|\mathbf{e}\|}, \quad \sin\alpha = \frac{d_2 \sin(\pi - v_3)}{\|\mathbf{e}\|}$$

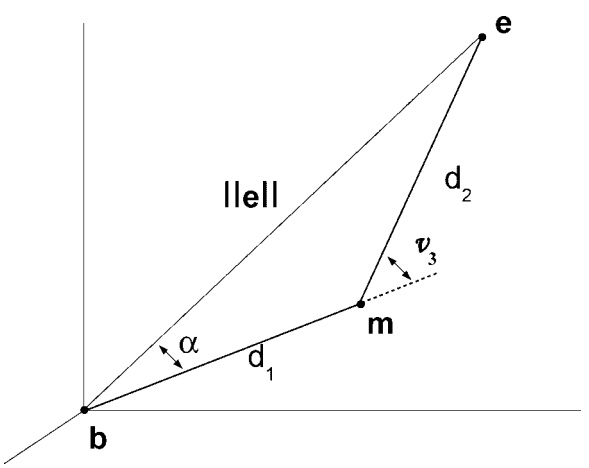


Figure 4. The mid flexion angle v_3.

Once the orbit circle is determined, the mid joint position $\mathbf{m}$ can be directly obtained with:

$$\mathbf{m} = \mathbf{c} + r(\hat{\mathbf{u}}\cos\phi + \hat{\mathbf{v}}\sin\phi) \tag{1}$$

Note that when $\mathbf{e}$ lies exactly on the shoulder frame axis $\hat{\mathbf{y}}$, the orbit circle becomes perpendicular to $\hat{\mathbf{y}}$, and $\hat{\mathbf{y}}$ cannot be projected anymore to the orbit plane for deriving $\hat{\mathbf{u}}$. Although alternatives are possible,[3] such $\mathbf{e}$ position is very unlikely to be anatomically realistic and our system simply returns failure in such cases. However, note that such situation never happens in practice as in the overall method presented in this paper a body behavior can be specified to have the shoulder to move when targets are given near $\hat{\mathbf{y}}$, therefore handling all cases and achieving more realistic results.

Mid Flexion

Given the swivel angle ϕ and the desired position $\mathbf{e}$ and orientation $\mathbf{q_e}$ for the end joint, the corresponding linkage configuration vector $(v_0, \ldots, v_6)$ can be now derived. The first linkage value to be computed is the mid flexion v_3.

Let d_1 be the distance between the base joint position $\mathbf{b}$ and the mid joint position $\mathbf{m}$, and let d_2 be the distance between the mid and end joints. Values d_1 and d_2 are the lengths of the upper and lower limbs of the linkage, and $\|\mathbf{e}\|$ gives the distance between the base and the end joints. No matter the linkage configuration, there is only one mid joint flexion angle v_3 that correctly forms the triangle with sides d_1, d_2, and $\|\mathbf{e}\|$ (see Figure 4); and its value can be determined by using the law of cosines:

$$v_3 = \pi \pm \arccos\left(\frac{d_1^2 + d_2^2 - \|\mathbf{e}\|^2}{2d_1 d_2}\right) \tag{2}$$

Despite the two solutions for v_3 only the one with minus sign is valid in the used parameterization.

Base Swing

Given ϕ, the corresponding mid joint position $\mathbf{m}$ can be computed by Equation (1). The base swing rotation is determined by the rotation needed to bring the upper limb in the zero-posture (lying along the z-axis, see Figure 1) to the mid position $\mathbf{m}$. The axis of rotation is therefore $\hat{\mathbf{z}} \times \mathbf{m}$ and the rotation angle is the angle between $\hat{\mathbf{z}}$ and $\mathbf{m}$. By observing that the axis of rotation is always in the xy-plane of the base local frame, the 2D

DOI: 10.1002/cav

axis-angle representation of the swing rotation becomes straightforward:

$$\begin{pmatrix} v_0 \\ v_1 \end{pmatrix} = \begin{pmatrix} s_x \\ s_y \end{pmatrix}, \quad \mathbf{s} = \frac{\hat{\mathbf{z}} \times \mathbf{m}}{\|\hat{\mathbf{z}} \times \mathbf{m}\|} \arccos \frac{\hat{\mathbf{z}} \cdot \mathbf{m}}{d_1} \tag{3}$$

where s_x and s_y are the x and y components of $\mathbf{s}$.

Base Twist

Let $\mathbf{q}_{(\mathbf{a})}$ be the rotation of angle $\|\mathbf{a}\|$ around axis $\mathbf{a}$ in quaternion format:

$$\mathbf{q}_{(\mathbf{a})} = \left(\cos \frac{\|\mathbf{a}\|}{2}, \frac{\mathbf{a}}{\|\mathbf{a}\|} \sin \frac{\|\mathbf{a}\|}{2} \right)$$

Once the base swing rotation is applied to the base joint, the base twist v_2 will make the linkage to rotate in order for the end joint to reach $\mathbf{e}$.

Let $\mathbf{f}$ be the end joint position obtained when only the mid flexion rotation v_3 is applied to the linkage in the zero-posture:

$$\mathbf{f} = \mathbf{q}_{(0,v_3,0)} \begin{pmatrix} 0 \\ 0 \\ d_2 \end{pmatrix} \mathbf{q}^{-1}_{(0,v_3,0)} + \begin{pmatrix} 0 \\ 0 \\ d_1 \end{pmatrix}$$

Let $\mathbf{g}$ be the goal end position $\mathbf{e}$ rotated by the inverse of the base swing rotation:

$$\mathbf{g} = \mathbf{q}^{-1}_{(v_0,v_1,0)} \mathbf{e}\, \mathbf{q}_{(v_0,v_1,0)}$$

The twist rotation angle v_2 is determined by finding the rotation around $\hat{\mathbf{z}}$ that brings $\mathbf{f}$ to coincide with $\mathbf{g}$. The following equality using a rotation matrix can therefore be written:

$$\begin{pmatrix} \cos v_2 & -\sin v_2 & 0 \\ \sin v_2 & \cos v_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathbf{f} = \mathbf{g}$$

The solution to this equality is obtained with:

$$\arctan v_2 = \frac{f_x g_y - f_y g_x}{f_x g_x + f_y g_y}$$

and v_2 can be correctly extracted with the *four-quadrant inverse tangent* (atan2) function available in programming languages:

$$v_2 = \text{atan2}(f_x g_y - f_y g_x,\ f_x g_x + f_y g_y) \tag{4}$$

Mid Twist and End Swing

Although the mid twist and the end swing rotations are applied to different joints, their combined effect is the same as a single 3-DOF rotation $\mathbf{q}$. I will first find $\mathbf{q}$ here and then decompose it in the twist and swing components.

By composing all computed rotations so far in the correct order, the following identity is obtained:

$$\mathbf{q}_{(v_0,v_1,0)} \mathbf{q}_{(0,0,v_2)} \mathbf{q}_{(0,v_3,0)} \mathbf{q} = \mathbf{q}_{\mathbf{e}}$$

Therefore, $\mathbf{q}$ is obtained with:

$$\mathbf{q} = \left(\mathbf{q}_{(v_0,v_1,0)} \mathbf{q}_{(0,0,v_2)} \mathbf{q}_{(0,v_3,0)} \right)^{-1} \mathbf{q}_{\mathbf{e}}$$

Once $\mathbf{q} = (q_w, (q_x, q_y, q_z))$ is derived, it can be decomposed in a twist rotation v_4 around the z-axis followed by a swing rotation represented by the 2D axis-angle (v_5, v_6).

First, if q_z and q_w are both zero, the orientation is at the singularity of the swing component. This case should never happen as joint limits (described in the next subsections) enforce that rotations are always away from that singularity (which is placed at the $-\hat{\mathbf{z}}$ direction). Therefore the following identity holds:

$$\mathbf{q}_{(0,0,v_4)} \mathbf{q}_{(v_5,v_6,0)} = \mathbf{q}$$

By expanding the equation it is possible to isolate the twist rotation v_4:

$$v_4 = 2 \,\text{atan2}(q_z, q_w) \tag{5}$$

Once the twist component is known, the swing component can be obtained with:

$$\begin{pmatrix} v_5 \\ v_6 \end{pmatrix} = \frac{2\beta}{\sin \beta} \begin{pmatrix} \cos \frac{v_4}{2} & \sin \frac{v_4}{2} \\ -\sin \frac{v_4}{2} & \cos \frac{v_4}{2} \end{pmatrix} \begin{pmatrix} q_x \\ q_y \end{pmatrix} \tag{6}$$

where

$$\beta = \text{atan2}\left(\sqrt{q_x^2 + q_y^2}, \sqrt{q_z^2 + q_w^2} \right)$$

All joint values have now been computed.

Joint Limits

An important property of swing rotations (v_0, v_1) and (v_5, v_6) is that they can be meaningfully limited with spherical polygons or ellipses. A detailed exposition of such limiting curves and of the swing-and-twist

DOI: 10.1002/cav

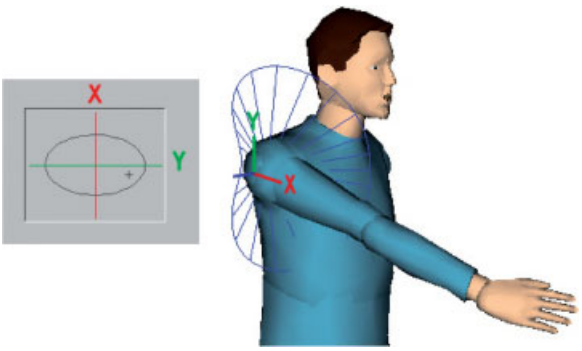


Figure 5. Swing rotations of the shoulder joint are limited by an ellipse defined in the x–y plane of the local frame (left image). The x and y axes are reversed in order to achieve visual correspondence of the selected orientation (the cross mark in the left image) with the arm orientation given by the shoulder swing rotation of the 3D character (right image): both are in the right-lower quadrant. The corresponding limiting surface in 3D has a spherical ellipse boundary (right image).

parameterization is given by Baerlocher.[6,26] In this work I choose to use spherical ellipses for maximum efficiency and robustness.

A spherical ellipse (Figure 5) is defined with an ellipse lying in the x–y plane of the joint local frame. A swing rotation $\mathbf{s} = (s_x, s_y)$ is detected to be valid in respect to its limiting ellipse with semi axes lengths r_x and r_y if the following inequality holds:

$$\left(\frac{s_x}{r_x}\right)^2 + \left(\frac{s_y}{r_y}\right)^2 \leq 1 \tag{7}$$

Figure 5 shows a typical spherical ellipse used to limit the shoulder joint, and its respective ellipse in the x–y plane of the shoulder local frame. It is also worth mentioning that an intuitive modeling interface results from this mapping: all swing rotations can be visualized by moving a pointing device inside the ellipse in the x–y plane.

Joint values v_2, v_3, and v_4 are correctly bounded with minimum and maximum values as usually done with Euler angles.

Automatic Swivel Angle Determination

A simple and fast iterative search procedure is now presented for finding the best ϕ satisfying joint limits and collision constraints. It only requires the ability of testing if the linkage is *valid* at a given configuration, i.e., if it respects joint limits and if it is collision-free.

The procedure starts with a given initial ϕ, e.g. 30°. Then, the linkage configuration obtained with the IK solver is checked for validity. If the configuration is not valid, a quick iterative method is performed for verifying if other ϕ values can lead to a valid posture.

At each iteration, ϕ is incremented and decremented by Δ, and the two new configurations given by the IK solver are checked for validity. If a valid configuration is found, the process successfully stops. Otherwise, if given minimum and maximum ϕ values are reached, failure is returned. Note that the described procedure always start from the initial ϕ, and therefore it will always find the closest solution to this preferred initial value. The elbow will deviate from it when needed, but will return as soon as there are no obstacles or joint limitations. A faster search can be achieved in a greedy fashion, i.e. when Δ increases during the iterations. However, increments cannot grow much to keep the animation continuous.

This procedure was designed with the goal of keeping the mid joint as close as possible to the desired initial value (30° in this example). Note that the initial value can also be determined as a function to the position to be reached, using available sensorimotor transformations from experimental studies.[27]

As the search range is small this simple process is very efficient. For example by setting a range from −15 to 130° for the arms, and correctly adjusting the increments, the whole process can be limited to few validity tests (about less than 40 with acceptable results). Note that both joint limits and collisions are addressed in an unified way. Figure 6 shows an example of an emergent joint coupling achieved, and Figure 7 shows a self-collision avoided. Note that collisions with the environment are also handled in the same way and the same process is also applied to the legs of the character.

The described process searches for possible solutions around the elbow circle and it will always prefer the solution closer to the initial ϕ angle. The method does not need to keep track of previous frames, making it suitable for one-shot computations as required for instance in sampling-based motion planners. In most of the cases the method also produces a continuous animation of the linkage while the end joint is interactively manipulated. However, it is possible to imagine a situation where the elbow is at an awkward configuration due to obstacles,

DOI: 10.1002/cav

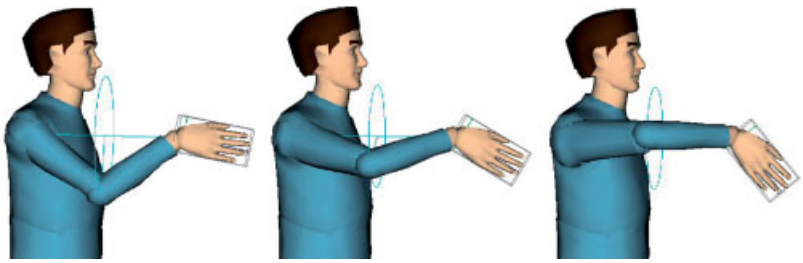


Figure 6. Starting from an orbit angle of 20° *(left image), as the wrist rotates downwards, the elbow rotates to higher positions (center and right images), maintaining the wrist joint values in their valid range.*

and then it suddenly jumps to a better posture because one of the obstacles moved away. If needed, this situation can be avoided with an additional step that continuously updates the current arm configuration until it matches the computed IK solution, and prematurely stopping if any collisions are detected. Alternatively, a search procedure starting at the current arm configuration frame can also be devised.

Note also that the search procedure is mainly required because of the collision-free constraint. If only joint limits are taken into account, it is possible to clamp the non-complying joint to its closest maximum limit, and then recompute ϕ and the joint values of the remaining joints. However, specific analytical derivations for each non-complying joint have to be derived.

Body Control

I describe now the blending mechanism for obtaining customizable full-body behavior. First, pre-designed postures are organized according to target directions to be reached, in relation to the shoulder frame. Without loss of generality, the right hand is considered here to be the end-effector while the left arm is assigned to be part of the body postures used for defining the body behavior.

A body behavior B is defined by a set of pairs in the following form:

$$B = \{(\mathbf{s}_1, \mathbf{p}_1), \ldots, (\mathbf{s}_n, \mathbf{p}_n)\} \tag{8}$$

where $\mathbf{s}_i = (x_i, y_i)$ is a 2D axis-angle representing a swing rotation of the shoulder joint and $\mathbf{p}_i = (p_i^1, \ldots, p_i^r)$ defines a r-DOF key posture of the body behavior, $1 \leq i \leq n$.

The values in postures $\mathbf{p}_i$ are the joint values of all joints being controlled by the body behavior. According to the used character representation, joint values can be of different types: translation values for the root joint, components of a swing axis-angle, twist rotations, or Euler angles. The used joint representation has just to guarantee that no singularities appear in the valid range of the joints, allowing linear interpolation between the values of two postures to be meaningful. Note that Euler

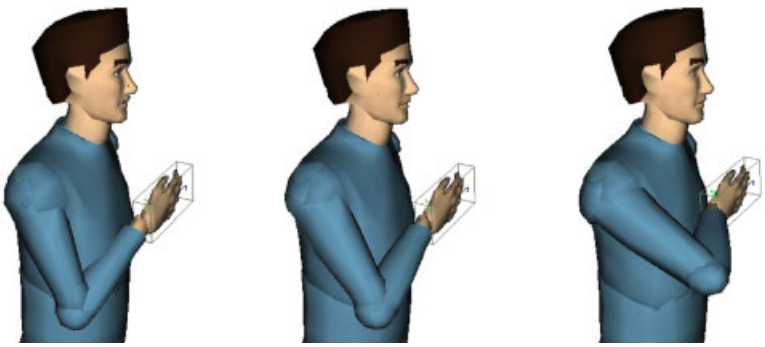


Figure 7. If collisions are detected (left) the orbit angle is updated and a valid posture is eventually found (center). As the hand gets closer to the body, the orbit angle is continuously updated (right).

DOI: 10.1002/cav

angles are only used in 1-DOF and 2-DOF joints. 3-DOF rotations can be interpolated in axis-angle format, or if preferred, the interpolation scheme can be updated to accommodate quaternions.

Postures $\mathbf{p}_i$ may specify values for all joints in the character, or to fewer joints as needed. For example it is common for interactive characters to have dedicated attention controllers for the head; in such cases behavioral key postures would not contain head joint values. If a posture specifies values for the joints of the IK linkages of the right arm or the legs, these joint values will be overridden when the IK is applied to ensure the end-effectors constraints, as explained later on in the next section.

One key advantage of organizing postures in relation to shoulder swings is that the swing rotation can be mapped in a 2D plane, greatly simplifying the user interface of the body behavior design phase. Each swing $\mathbf{s}_i = (x_i, y_i)$ is mapped to a point in the *swing plane*, and the set of all swing points can be easily triangulated,[28] obtaining a well-defined neighborhood connectivity suitable for interpolation.

We use the incremental Delaunay algorithm to interactively maintain the triangulation of the swing points $\mathbf{s}_i$. First, the triangulation is trivially initialized with four vertices lying outside the valid range, which is delimited by an ellipse in the swing plane. These four vertices are however not near the singular point, i.e., their coordinates are not close to $-\pi$ or π. These initial vertices can be selected and their respective postures edited. Later on in the style edition process they are usually hidden in order to simplify the visualization, but they ensure that all points inside the valid swing range are covered by the triangulation (see Figure 8).

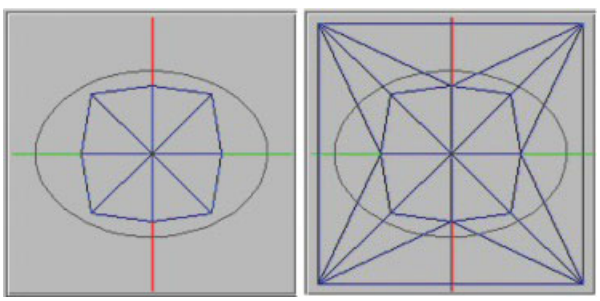


Figure 8. The swing plane. The left image shows the triangulation of nine user-defined key shoulder swings. The right image shows in addition the four vertices used to initialize the body behavior edition. The shown ellipse delimits the valid swings (inside) from the invalid swings (outside).

During the edition process, when a triangulation vertex $\mathbf{s}_i$ is selected in the swing plane, the associated posture $\mathbf{p}_i$ is applied to the character. The designer can then update posture $\mathbf{p}_i$ by directly manipulating the character or its joint values.

When a swing point $\mathbf{s}$ is selected, and $\mathbf{s}$ is not a vertex, the corresponding posture is determined by blending the postures associated with the three vertices of the triangle t containing $\mathbf{s}$. Fortunately, efficient algorithms with expected sublinear time complexity are available for finding t, as the simple oriented walk method.[29]

Once triangle t is found, the postures associated with its vertices can then be blended. Let $\mathbf{s}_i$, $\mathbf{s}_j$, and $\mathbf{s}_k$ be the vertices of t. The barycentric coordinates (α, β, γ) of $\mathbf{s}$ in respect to t can be easily obtained by dividing sub-areas computed with third-order determinants:

$$\alpha = \begin{vmatrix} \mathbf{s}^T & \mathbf{s}_j^T & \mathbf{s}_k^T \\ 1 & 1 & 1 \end{vmatrix} \Bigg/ \begin{vmatrix} \mathbf{s}_i^T & \mathbf{s}_j^T & \mathbf{s}_k^T \\ 1 & 1 & 1 \end{vmatrix}$$

$$\beta = \begin{vmatrix} \mathbf{s}_i^T & \mathbf{s}^T & \mathbf{s}_k^T \\ 1 & 1 & 1 \end{vmatrix} \Bigg/ \begin{vmatrix} \mathbf{s}_i^T & \mathbf{s}_j^T & \mathbf{s}_k^T \\ 1 & 1 & 1 \end{vmatrix}$$

$$\gamma = 1 - \alpha - \beta$$

The barycentric coordinates are used as blending factors so that posture $\mathbf{p} = (p^1, \ldots, p^r)$ relative to the swing $\mathbf{s}$ can then be obtained with:

$$p^m = \alpha p_i^m + \beta p_j^m + \gamma p_k^m, \quad 1 \leq m \leq r$$

Such interpolation method is widely used for data visualization and in other domains. It provides a continuous result, however not achieving C^1 continuity. In the experiments performed in this work, the method has shown to be extremely satisfactory. The main advantages are simplicity and computation efficiency, scaling very well on the number of example postures. It does not require the definition of extra parameters such as the shape of radial basis functions or the search for the k-nearest neighbors. But these alternative interpolation methods can also be employed if smoothness is preferred over performance, still making use of the organization of postures in the triangulated swing plane.

During edition of the style postures, the designer can select any point $\mathbf{s}$ in the valid portion of the swing plane and the corresponding blended posture $\mathbf{p}$ is applied to the character. If desired, a new pair $(\mathbf{s}, \mathbf{p})$ can be added to B, in which case $\mathbf{s}$ is added as a new vertex in the triangulation and posture $\mathbf{p}$ can be edited in order to refine the behavior as exemplified in Figure 9.

DOI: 10.1002/cav

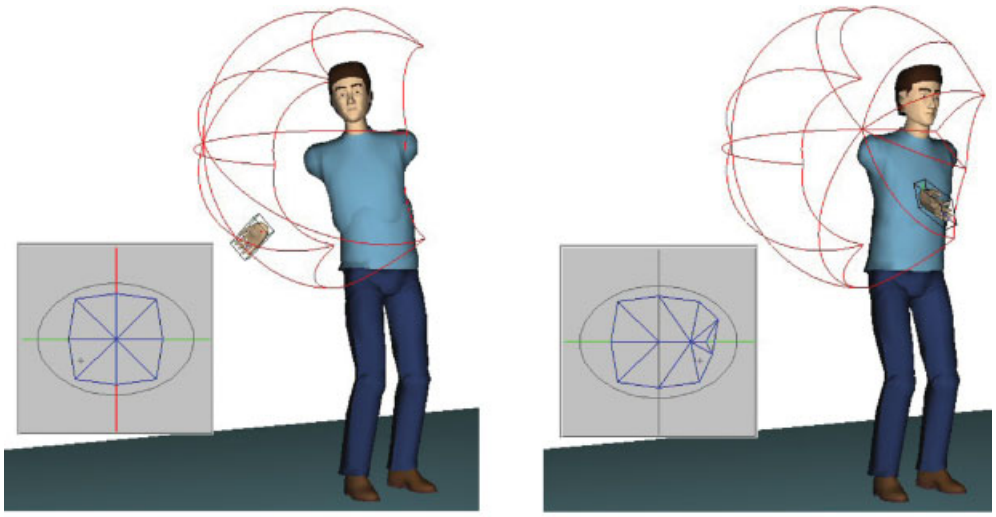


Figure 9. Body behaviors are defined by attaching body postures to vertices in the swing plane. Starting with a simple initial set of vertices (a) new vertices are added as needed (b) until the desired behavior is achieved. The triangulation in the swing plane is also shown projected in the sphere having the shoulder joint as center and the arm length (in full extension) as radius.

The operation for inserting a new vertex in the triangulation consists of linking the new point $\mathbf{s}$ to the vertices of triangle t and then correcting all introduced non-Delaunay edges with a series of *edge flips* until all edges become Delaunay again.[30] Points can be also removed making the style edition process very intuitive.

In Figure 9, only the body parts being influenced by that style edition session are being displayed. When the mouse is dragged on the swing plane, a continuous motion of the character's torso can be visualized according to the interpolation process described above. In the figure, the small black mark in the swing plane corresponds to the shown hand location as determined by the shoulder swing rotation. For a more intuitive mapping, the vertical axis in the swing plane corresponds to the x-axis of Figure 1, and the horizontal axis to the y-axis.

Two-Arms Control

When the simultaneous control of the two arms is required, target positions are specified for both right and left hands simultaneously, requiring individual body control for each arm. Let B^{L} and B^{R} be two behaviors, relative to the left and right arms respectively:

$$B^{\mathrm{L}} = \left\{\left(\mathbf{s}_1^{\mathrm{L}}, \mathbf{p}_1^{\mathrm{L}}\right), \ldots, \left(\mathbf{s}_n^{\mathrm{L}}, \mathbf{p}_n^{\mathrm{L}}\right)\right\},$$

$$B^{\mathrm{R}} = \left\{\left(\mathbf{s}_1^{\mathrm{R}}, \mathbf{p}_1^{\mathrm{R}}\right), \ldots, \left(\mathbf{s}_m^{\mathrm{R}}, \mathbf{p}_m^{\mathrm{R}}\right)\right\}$$

Given target swing rotations $\mathbf{s}^{\mathrm{L}}$ for the left shoulder and $\mathbf{s}^{\mathrm{R}}$ for the right shoulder, the two corresponding body postures $\mathbf{p}^{\mathrm{L}}$ and $\mathbf{p}^{\mathrm{R}}$ are determined independently by the described blending procedure. The final considered body posture can then be defined by a mix of joint values from $\mathbf{p}^{\mathrm{L}}$ and $\mathbf{p}^{\mathrm{R}}$: when values are affecting a same joint they are blended and otherwise just copied. The blending can be a simple average of the values, or more weight can be given to one preferred 'side' (left or right) for simulating a right-handed or left-handed character.

Overall Method

The overall method blends the body behavior posture with a given initial posture according to a user-specified weight variation.

Let $\mathbf{p}^{\mathrm{init}}$ be a given initial (reference) posture of the character, and let B be the body behavior in use. It is assumed that $\mathbf{p}^{\mathrm{init}}$ and the key postures in B have the same number of DOFs.

Let $\mathbf{t} = (\mathbf{p}, \mathbf{q})$ be a given hand target and $\mathbf{p}_{\mathrm{s}}$ be the shoulder joint position in global coordinates. Let now $\mathbf{s}$ be the swing rotation of the $\mathbf{p} - \mathbf{p}_{\mathrm{s}}$ vector, and $d = \|\mathbf{p} - \mathbf{p}_{\mathrm{s}}\|$ the length (note that $\mathbf{s}$ is an axis-angle in the shoulder frame).

DOI: 10.1002/cav

Once $\mathbf{s}$ is determined as described in the previous paragraph, the respective body behavior posture $\mathbf{p}^{\mathrm{B}}$ can be extracted from B. However, the final used body behavior posture $\mathbf{p}^{\mathrm{bh}}$ is computed as a function of distance d:

$$\mathbf{p}^{\mathrm{bh}} = \mathrm{interp}(\mathbf{p}^{\mathrm{init}}, \mathbf{p}^{B}, u), \quad u = f_{\mathrm{weight}}(d)$$

where interp interpolates the two postures appropriately and according to the weight $u \in [0, 1]$, which is given by a weight function f_{weight}.

The final posture $\mathbf{p}^{\mathrm{bh}}$ is therefore an interpolation between the initial posture and the behavior posture, as a function of the target-shoulder vector length, and according to a desired weight function f_{weight}. The idea is that reaching to distant locations requires more body motion, while closer locations require less body motion. Let a be the arm length in full extension. Best results are achieved with f_{weight} having value 0 when the target-shoulder vector length is 0 and rapidly approaching 1 when the target-shoulder vector length approaches a. One example of such a weight function is:

$$f_{\mathrm{weight}}(d) = \begin{cases} 0, & d < 0 \\ \frac{k}{a^2} d^2, & d \in \left[0, \frac{a}{\sqrt{k}}\right] \\ 1, & d > \frac{a}{\sqrt{k}} \end{cases}$$

where k is a value near 1 (and ≤ 1) such as 0.8. This weight function has $f'(0) = 0$, $f(0) = 0$, and $f(a) = k$.

Once the final body posture $\mathbf{p}^{\mathrm{bh}}$ is determined, it is applied to the character and the analytical arm IK is invoked for exactly posing the right arm of the character to the given target $\mathbf{t}$. As $\mathbf{p}^{\mathrm{bh}}$ may affect the translation of the root of the skeleton, the analytical IK is also applied to the legs for ensuring that both feet remain in the same location as in the given initial posture.

Note that the analytical IK applied to the arms and legs can use the collision avoidance search mechanism described in the earlier sections of this paper. The obtained body posture should not result in self-collisions as they were carefully designed by hand. But collisions with obstacles in the environment can be introduced. A simple correction method checks if the body posture is in collision, and if it is, the interpolation parameter u between $\mathbf{p}^{\mathrm{init}}$ and $\mathbf{p}^{\mathrm{B}}$ can be reduced toward 0 until no collisions are reported.

Extended Behaviors

In the described setting, body behaviors do not affect the IK linkages and therefore it is left to the automatic orbit search mechanism to determine the final postures of the linkages. In case more control is desired, additional parameters can be associated to the swing points defining key postures (Equation 8). For example the parameters of the orbit angle search method can be included and interpolated in the same manner as the posture interpolation. In this way, each given swing rotation will have a specific orbit search behavior. This capability was found to be useful for precisely customizing goal orbit angles.

Animating Reaching Motions

Neuroscience research provides several computational models (2/3 power law, Fitts' law, etc.)[31] that can help synthesizing realistic arm motions. The simplest model states that in point-to-point reaching movements, the hand path approximates a straight line, and the tangential velocity resembles a symmetrical bell-shape. These guidelines can be easily implemented by animating the hand of the characters with such a path and applying the IK solver at each frame for obtaining the intermediate postures (see Reference [32] for more details).

Analysis and Results

Examples of postures obtained with different body behaviors are presented in Figure 10. A sequence demonstrating the usefulness of the collision avoidance mechanism integrated with the arm IK is shown in Figure 11.

It is possible to observe that the system is capable of generating a wide range of body behaviors as a function of the location being reached. The simple body control interpolation scheme represents practically no additional computational cost and allows to achieve much more realistic postures, in comparison with using the arm IK alone. However, it is clear that the method is mainly suitable for reaching tasks and cannot achieve arbitrary posture control as other multi-task numerical methods can do.

The proposed method was designed with the goal of being very fast. The employed interpolation method only performs interpolation between three body postures,

DOI: 10.1002/cav

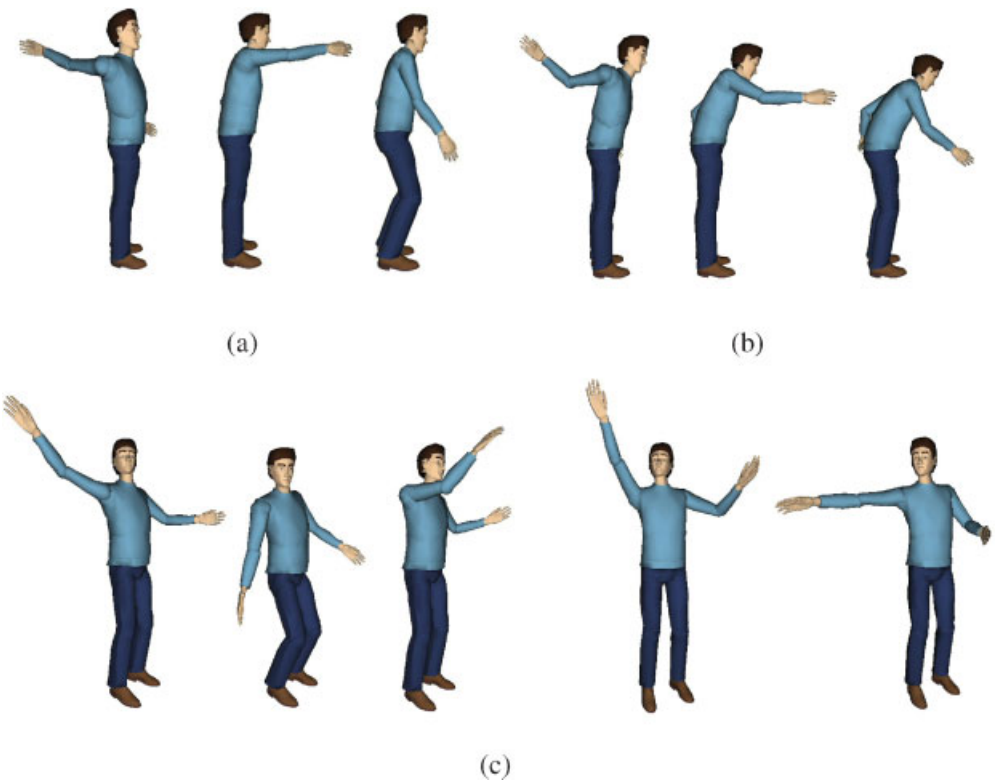


Figure 10. Several whole-body postures determined as a function of the global position and orientation of the right hand of the character. All postures are collision-free and respect joint limits. Knee flexion is achieved according to the used body behavior. The first row shows three postures obtained with a normal behavior (a) and with a 'tired behavior' (b). The second row (c) shows several postures obtained with a behavior that also controls the motion of the head and the left arm, and that is applied to an initial posture with the legs slightly open.

and the used dimensionality reduction allows the parameterization of the data in a two-dimensional plane (the shoulder swing plane). The method has shown to offer enough flexibility for customizing the body behavior in reaching tasks. However, if more control and stylistic results are required, other interpolation methods, for example based on radial basis functions[18] could be integrated.

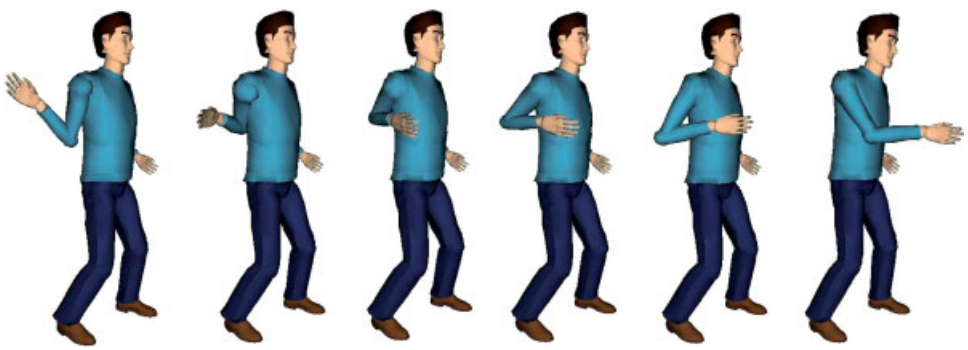


Figure 11. A difficult sequence where the right hand moves from a back to a frontal position passing very close to the body and correctly avoiding collisions.

DOI: 10.1002/cav

The source code of the arm IK and videos demonstrating the method are available at the author's web site.[†]

Performance

Given a target position for the right hand, the complete method takes no more than 10 milliseconds for deriving a posture with collision detection handling 12K triangles. This time depends on how many iterations the orbit angle search takes in the solution of the IK linkages. However note that, as the maximum number of iterations is fixed, the maximum computation time is bounded.

If no collisions are handled the system takes no more than 0.1 milliseconds. This gives an indication of the time spent in collision detection. For demanding applications collision detection can be easily reduced to only critical tests (as detection between the elbow and torso), or even disabled. If no collision detection and no orbit angle search are employed, the system takes about 0.05 milliseconds to derive postures. This includes style posture interpolation and solving the IK linkages. These tests were performed on a Pentium 3.0 GHz.

Extensions

If example key postures are organized spatially, the planar mapping with the shoulder swing plane is lost but it is still possible to interpolate the data using other methods and therefore with much more control of the final body style behavior. The design of such postures may be complex, however such approach could be suitable to represent data automatically extracted from a motion capture database.

Conclusions

This paper presented a new customizable whole-body IK system which is fast, robust, and simple to implement. The main contributions are:

- A new analytical IK formulation based on the swing-and-twist parameterization that handles collision avoidance and joint coupling in a unified way (source code provided).
- The approach of organizing key body postures as a function of the goal direction to reach, simplifying interpolation and allowing for an intuitive way of designing body behaviors.
- The overall approach, which probably represents the fastest available whole-body IK including desired features such as collision avoidance and customizable whole-body behavior.

[†]`http://graphics.ucmerced.edu/`

References

1. Watt A, Watt M. *Advanced Animation and Rendering Techniques*. ACM Press: New York, NY, 1992.
2. Rickel J, Johnson L. Animated agents for procedural training in virtual reality: perception, cognition, and motor control. *Applied Artificial Intelligence* 1999; **13**(4–5): 343–382.
3. Tolani D, Badler N. Real-time inverse kinematics of the human arm. *Presence* 1996; **5**(4): 393–401.
4. Grassia S. Practical parameterization of rotations using the exponential map. *Journal of Graphics Tools* 1998; **3**(3): 29–48.
5. Welman C. *Inverse Kinematics and Geometric Constraints for Articulated Figure Animation*. PhD thesis, Simon Fraser University, 1993.
6. Baerlocher P. *Inverse Kinematics Techniques for the Interactive Posture Control of Articulated Figures*. PhD thesis, Swiss Federal Institute of Technology, EPFL, 2001. Thesis number 2383.
7. Girard M, Maciejewski AA. Computational modeling for the computer animation of legged figures. In *Proceedings of SIGGRAPH'85*, New York, NY, USA, ACM Press, 1985; pp. 263–270.
8. Bodenheimer B, Rose C, Rosenthal S, Pella J. The process of motion capture: dealing with the data. In *Computer Animation and Simulation'97*, Thalmann D, van de Panne M (eds). Springer-Verlag: New York, NY, 1997; 3–18. *In* Eighth Eurographics Computer Animation and Simulation Workshop, Budapest, Hungary, September 2–3, 1997.
9. ElKoura G, Singh K. Handrix: animating the human hand. In *ACM Symposium on Computer Animation*, Faloutsos P, van de Panne M, Terzopoulos D (eds). ACM Press: New York, NY, 2003; 110–119.
10. Zhao J, Badler N. Inverse kinematics positioning using nonlinear programming for highly articulated figures. *ACM Transactions on Graphics* 1994; **13**(4): 313–336.
11. Yamane K, Nakamura Y. Natural motion animation through constraining and deconstraining at will. *IEEE Transactions on Visualization and Computer Graphics* 2003; **9**(3): 352–360.
12. Buss SR, Kim J-S. Selectively damped least squares for inverse kinematics. *Journal of Graphics Tools* 2005; **10**(3): 37–49.
13. Baerlocher P, Boulic R. Task-priority formulations for the kinematic control of highly redundant articulated structures. In *Proceedings of IEEE IROS'98*, Victoria, Canada, November 1998; pp. 323–329.
14. Tolani D, Goswami A, Badler NI. Real-time inverse kinematics techniques for anthropomorphic limbs. *Graphical Models* 2000; **62**(5): 353–388.
15. Arikan O, Forsyth DA. Synthesizing constrained motions from examples. *ACM Transaction on Graphics (Proceedings of SIGGRAPH'02)* 2002; **21**(3): 483–490.

DOI: 10.1002/cav

16. Kovar L, Gleicher M, Pighin FH. Motion graphs. *ACM Transaction on Graphics (Proceedings of SIGGRAPH'02)* 2002; **21**(3): 473–482.
17. Lee J, Chai J, Reitsma P, Hodgins JK, Pollard N. Interactive control of avatars animated with human motion data. *ACM Transaction on Graphics (Proceedings of SIGGRAPH'02)* 2002; **21**(3): 491–500.
18. Rose CF III, Sloan P-PJ, Cohen MF. Artist-directed inverse-kinematics using radial basis function interpolation. *Computer Graphics Forum (Proceedings of Eurographics'01)* 2001; **20**(3): 239–250.
19. Arikan O, Forsyth DA, O'Brien JF. Motion synthesis from annotations. *ACM Transaction on Graphics (Proceedings of SIGGRAPH'03)* 2003; **22**(3): 402–408.
20. Kovar L, Gleicher M. Automated extraction and parameterization of motions in large data sets. *ACM Transaction on Graphics (Proceedings of SIGGRAPH'04)* 2004; **23**(3): 559–568.
21. Wiley DJ, Hahn JK. Interpolation synthesis of articulated figure motion. *IEEE Computer Graphics and Applications* 1997; **17**(6): 39–45.
22. Grochow K, Martin S, Hertzmann A, Popović Z. Style-based inverse kinematics. *ACM Transactions on Graphics (Proceedings of SIGGRAPH'04)* 2004; **23**(3): 522–531.
23. Brand M, Hertzmann A. Style machines. *In SIGGPAPH'OO: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press: Addison-Wesley Publishing Co.: New York, NY, USA, 2000; 183–192.
24. Li Y, Wang T-S, Shum H-Y. Motion texture: a two-level statistical model for character motion synthesis. *ACM Transactions on Graphics (Proceedings of SIGGRAPH'02)* 2002; **21**(3): 465–472.
25. Korein JU. *A Geometric Investigation of Reach*. The MIT Press: Cambridge, MA, 1985.
26. Baerlocher P, Boulic R. Parametrization and range of motion of the ball-and-socket joint. In *Proceedings of the AVATARS conference* Lausanne, Switzerland, 2000.
27. Soechting JF, Flanders M. Sensorimotor representations for pointing to targets in three-dimensional space. *Journal of Neurophysiology* 1989; **62**(2): 582–594.
28. Preparata FP, Shamos MI. *Computational Geometry—An Introduction*. Springer-Verlag, New York, NY, 1985.
29. Devillers O, Pion S, Teillaud M. Walking in a triangulation. In *ACM Symposium on Computational Geometry*, 2001; pp. 106–114.
30. Guibas L, Stolfi J. Primitives for the manipulation of general subdivisions and the computation of voronoi diagrams. *ACM Transactions on Graphics* 1985; **4**(2): 75–123.
31. Schaal S. Arm and hand movement control. In *The Handbook of Brain Theory and Neural Networks* (2nd edn), Arbib M (ed.). The MIT Press: Cambridge, MA, 2002; 110–113.
32. Kallmann M. Scalable solutions for interactive virtual humans that can manipulate objects. In *Proceedings of the Artificial Intelligence and Interactive Digital Entertainment (AIIDE'05)*, Marina del Rey, CA, 1–3 June 2005; pp. 69–74.

Authors' biographies:

Marcelo Kallmann received his Ph.D. from the Swiss Federal Institute of Technology (EPFL) in 2001. Currently he is Assistant Professor and Founding Faculty in the School of Engineering of the University of California, Merced. He is also affiliated as Adjunct Assistant Professor to the Computer Science Department of the University of Southern California (USC). His main areas of research include robotics, motion planning, computer animation and geometric modeling.

DOI: 10.1002/cav