Forward Kinematics and Friends

Robert Bridson

September 21, 2011

Forward Kinematics refers to a direct approach of animating a character or any other model as a hierarchy, where each part of the model is positioned relative to its parent in the hierarchy—with the world itself being the parent of the root. This chapter will cover Forward Kinematics alongside related techniques for producing character animations, and other topics in directly animating parts of a scene.

1 Jointed Models

The human body is spectacularly complex, especially in motion. Just typing these words involves a delicate choreography of a multitude of muscle fibres (not as cleanly segmented into distinct muscles as you might expect from simple anatomical illustrations) tugging on tendons that transmit the forces to dozens of bones as they slide over each other as well as fat and fascia layers, all deforming according to laws of elasticity, with cartilage and ligaments linking bones together at irregularly shaped rolling joints. While researchers are making progress in leaps and bounds towards capturing all of this biomechanical glory—which is probably necessary to reproduce truly realistic motion—we can get very far indeed with much simpler models.

The simplest model in common use is to abstract the human body into a stick figure, in essence: a set of rigid segments (sometimes called "bones" but not necessarily corresponding to actual real bones in the body) connected at idealized *joints*.

1.1 Coordinate Systems

The first concept to grapple is what a rigid object, such as a Forward Kinematics bone, really is. We might first identify it with the geometry of the bone, but it turns out that's almost irrelevant in this case: ultimately we want to render our characters with a full fleshy volume, covered in softly

deforming skin, not as abstract stick figures or skeletons! (How to do that comes later in this chapter.) A representative shape like a box of roughly the right size may well be useful for the animator until we get to the final body, but what we really care about is the *local coordinate system* of the bone.

A coordinate system, or *frame*, encodes an origin and a set of basis vectors (three of them, for 3D), and provides a way to numerically specify positions with coordinates. For animation we generally assume that *world space* is our default for interpreting coordinates. Every other coordinate system should then be specified directly or indirectly in terms of world space: we need to be able to figure out the coordinates of the frame's origin in world space, and also the frame's basis vectors expressed in world space coordinates. This notion of coordinate systems should be familiar to you from earlier work in computer graphics—in particular, on the rendering side you should have looked at camera space, possibly Normalized Device Coordinates space, and screen space in relation to world space.

A local coordinate system for a rigid object is one that is "attached" to the object, so that the local coordinates of the geometry of the object don't change even as the object moves through world space. A rigid frame should furthermore always have *orthonormal* basis vectors: perpendicular to each other and unit length. Obviously if the rigid object moves through world space, its local coordinate system has to be changing with respect to world space too.

A rigid coordinate system is closely tied to the notion of *rigid transformation*: a transformation that converts coordinates in one frame to coordinates in another. Saying a frame is rigid is equivalent to saying that the transformation from world space to the frame is rigid, which is equivalent to saying the distance between points is preserved under the transformation. Note that a rigid frame exists on its own, but a rigid transformation always has to specify from which frame and to which frame it's doing the transformation—they are not identical concepts, even though it's necessary to use a rigid transformation to define a rigid frame in practice (specifying

how it relates to some other, already defined, coordinate system such as world space).

There are many possible representations for a rigid transformation. One with which you should be familiar is a 4×4 matrix, for use with homogenous coordinates. However, not just any 4 matrix is a rigid transformation; anything with a "perspective divide" is right out, for one, so we really only want to look at matrices of the form

$$M = \begin{pmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{1}$$

where I stick with the convention that transformation matrices multiply column vectors of coordinates on the left, with the homogenous coordinate being the fourth coordinate. This form of matrix also encompasses scalings, which change distances, and thus are not rigid. For rigidity we have an additional constraint that the leading 3×3 submatrix is an orthogonal matrix, i.e.

$$\begin{pmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{pmatrix}^T \begin{pmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{2}$$

Taking into account symmetry of the last product, this boils down to six constraint equations on M. With twelve variables in M and six equations they must satisfy, we are left with six ($12 - 6 = 6$) *degrees of freedom* for specifying a rigid trasnformation.[1]

Having so many additional constraints makes for a very awkward representation. Generally people only use the matrix form once the transformation is known, as it leads to very fast calculations via matrix multiplication,

[1] Actually, there is usually one more subtle restriction for physically reasonable motion—we disallow reflections, only allowing rotations, which means $\det M = 1$ instead of $\det M = -1$. Notwithstanding this, there are still six degrees of freedom.

but do not use this representation when actually specifying the transformation, and especially avoid it when trying to animate a rigid transformation. Doing the obvious interpolation between two matrices that represent rigid transformations, making a motion curve for each entry, almost certainly won't produce rigid transformations for intermediate values. For example, the average between the identity and a 90° rotation around the z axis gives a non-rigid transformation:

$$\frac{1}{2}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} + \frac{1}{2}\begin{pmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ -\frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (3)$$

Motion curves are useless with this representation.

A much nicer representation is to represent the transformation as a sequence (in some order) of a translation and three rotations around a set of axes. The rotation angles are called *Euler angles* in this case. The three coordinates of the translation vector together with the three rotation angles gives six degrees of freedom directly, with no constraints. The Euler angles themselves can be animated with motion curves, producing smooth rotational motion that at all times gives a rigid transformation.

Euler angles have their own problem, however, sometimes called "gimbal lock": for some special orientations, infinitely many different sets of rotations map to the same orientation. In a later chapter when we discuss rigid body physics, we will look at this in more depth, and cure it with yet another representation.

1.2 Joints

A joint fundamentally is just a parameterized rigid transformation, specifying how the local coordinate system of one bone is related to the local coordinate system of a connected bone. We can characterize joints in terms of how many parameters—actual degrees of freedom—they offer.

Of course, a joint with no parameters isn't really a joint: the two bones would be permanently welded together. However, it could still be convenient to think of it as a joint, since this naturally lets us introduce another coordinate system for a part of the body.

A joint with one rotation degree of freedom, around a given axis (specified as fixed in either one of the coordinate systems), is called a *revolute joint*. Knees, elbows and knuckes are commonly modeled as such. For simplicity, we usually require that the origin of one of the coordinate systems lies on the axis of rotation (in some sense the "location" of the joint), and that the axis of rotation is one of the coordinate axes of that coordinate system, say the x axis. Say bone A and bone B are connected by such a joint. The full transformation $T^{B\leftarrow A}$ from coordinates in A's frame to B's frame can be constructed as a product $R(\theta)T_0$. The rigid transformation T_0 that applies to the A-coordinates first is just the fixed transformation for the "neutral" or "rest" pose where the joint angle is zero: T_0 is responsible for ensuring the origin of B lies at the joint location (so the coordinates of the joint location in B get mapped to the origin $(0, 0, 0)$ in A) as well as any rotation needed to ensure that the axis of rotation lines up with B's x-axis. The parameterized rotation $R(\theta)$ is then just a simple rotation around the x axis in B, expressed in matrix form as:

$$R(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{4}$$

The inverse of this product gives the transformation $T^{A\leftarrow B}$ from coordinates in B's frame back to coordinates in A's frame of course.

A joint with one translation degree of freedome, along a fixed axis (specified as fixed in either one of the coordinate systems), is called a *prismatic joint*. It's quite common in robots, but rarely used for animating creatures. Again this would typically be set up as the product of a transformation which arranges for one coordinate system to have its origin on the axis of

sliding at the "neutral" point (where the sliding parameter is zero) and its x-axis lined up with the sliding axis, together with a transformation which simply translates along x.

A joint with a full three degrees of freedom in rotation around a fixed centre (no translation), is quite common, and is called either a *spherical joint* or a *ball-and-socket joint* (because it can be most easily created in real life as a spherical ball encased in a spherical socket). This is fairly general purpose for things like hips, the base of each finger (though one of the axes of rotation, around the length of the finger, is highly constrained), wrists, ankles, and sometimes parts of the spine. Specifying this with transformations proceeds as with the revolute joint, except instead of just one rotation around x we would have a sequence of three rotations around different axes. A common choice is to go through x, y, and z axes in that order. The total transformation is then something like:

$$\begin{aligned} T^{B\leftarrow A} &= R_z(\theta_z)\, R_y(\theta_y)\, R_x(\theta_x)\, T_0 \\ &= \begin{pmatrix} \cos(\theta_z) & -\sin(\theta_z) & 0 & 0 \\ \sin(\theta_z) & \cos(\theta_z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ &\quad \cdot \begin{pmatrix} \cos(\theta_y) & 0 & \sin(\theta_y) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta_y) & 0 & \cos(\theta_y) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ &\quad \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta_x) & -\sin(\theta_x) & 0 \\ 0 & \sin(\theta_x) & \cos(\theta_x) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} T_0 \end{aligned} \tag{5}$$

A joint with all six degrees of freedom allowed—any rotation, any translation—is also commonly used in animation for the most complex of the connections, like shoulders, the neck, and the lower jaw. To the transformation of

the spherical joint you would typically multiply in a translation after the rotations.

Further definitions and formal notation have been adopted in robotics, where these joints have an accurate mechanical basis, including the observation that any joint with more than one degree of freedom can be thought of as an assembly of several one-degree-of-freedom joints together. However, we won't go down this path for animation.

1.3 Modeling Error

One issue to be keenly aware of is that this representation—rigid bones with simple joints between them—is **not** a great model of real biomechanics.

Even a joint like the hip, which is pretty close to a ball-and-socket joint, doesn't have exact spherical symmetry, and also has particular (and irregularly shaped) limits on the rotation that's possible.

A joint like the knee or elbow is even further removed from a revolute joint: instead of rotating around a single axis, the top of the lower leg (forearm) rolls over the upper leg (upper arm) along a not-quite-circular "track". Though it's arguably still one degree of freedom, there's simultaneous translation and rotation.

The motion of the shoulder is a particularly tricky case. Real shoulders have a lot of freedome of movement, with very complicated limits. Often it's best for a rig to leave shoulders as full six degree of freedome joints, where all translations and rotations are possible. The same holds true of the lower jaw.

The rotation of the hand around the axis of the forearm is even worse. The biomechanical reality is that there are two bones in the forearm that can twist around each other in different ways to effect a net rotation of the

hand—but this twist takes place over the length of the forearm, not at a particular joint. The forearm simply isn't rigid: in a sense, it's one long joint and the rotation is interpolated along its length.

Finally, the spine is another odd case. Most animation models can't be bothered to track a separate bone for every vertebra, even though there may be slight movement possible between each and every one in reality. Modeling the spine with a smaller number of bones—or likewise simplifying any part of the skeleton (which offically has more than 200 separate bones)—incurs error obviously.

2 Tree Representation

As well as deciding how many bones are needed for a model, and what sorts of joints exist between them, Forward Kinematics needs a notion of hierarchy. One bone (rigid frame) should be the root, specified with a full six degrees of freedom relative to world space. The bones directly connected to the root by joints have frames described by the parameterized joint transformations from the root to their frames, and so on recursively. The model must be structured as a tree (where nodes are bones/frames and edges are joints) for this to make sense: in particular, cycles are not permissible.

Typically the hips are chosen as the root for humans: this tradition goes back to hand animation where the hips are generally thought to be the most fundamental part of a motion to get right first.

Since the joints only encode the transformation between two adjacent bones in the tree, a tree traversal is required to evaluate the transformations from world space to any specific bone. The transformations are multiplied together in a path through the tree.