CHAPTER

Motion Capture 6

Creating physically realistic motion using techniques involving key frames, forward kinematics, or inverse kinematics is a daunting task. In the previous chapters, techniques to move articulated linkages are covered, but they still require a large amount of animator talent in order to make an object move as in the real world. There is a certain quality about physically correct motion that we are all aware of, and it only takes a small deviation from that quality to make what is supposed to be realistic movement quite distracting. As a result, it is often easier to record movement of a real object and map it onto a synthetic object than it is for the animator to craft realistic-looking motion. The recording and mapping of real motion to a synthetic object is called *motion capture*, or *mocap* for short.

Motion capture involves sensing, digitizing, and recording of an object's motion [6]. Often, the term specifically refers to capturing human motion, but some very creative animation can be accomplished by capturing the motion of structures other than animal life. However, the discussion in this chapter focuses on motion capture of a human figure.

In motion capture, a (human) subject's movement is recorded. It should be noted that locating and extracting a figure's motion directly from raw (unadulterated) video is extremely difficult and is the subject of current research. As a result, the figure whose motion is to be captured (sometimes referred to as the *talent*) is typically instrumented in some way so that positions of key feature points can be easily detected and recorded. This chapter is concerned with how this raw data is converted into data usable for animation.

There are various technologies that can be applied in this instrumentation and recording process. If the recordings are multiple two-dimensional images, then camera calibration is necessary in order to reconstruct the three-dimensional coordinates of joints and other strategic positions of the figure. Usually, joint angles are extracted from the position data and a synthetic figure, whose body dimensions match those of the mocap talent, can then be fitted to the data and animated using the extracted joint angles. In some cases, the mocap data must be edited or combined with other motion before it can be used in the final animation. These topics are discussed in the following sections.

6.1 Motion capture technologies

There are primarily two approaches to this instrumentation[1]: *electromagnetic sensors* and *optical markers*. Electromagnetic tracking, also simply called *magnetic tracking*, uses sensors placed at the joints that transmit their positions and orientations back to a central processor to record their

[1]There are several other technologies used to capture motion, including electromechanical suits, fiber-optic sensors, and digital armatures [6]. However, electromagnetic sensors and (passive) optical markers are by far the most commonly used technologies for capturing full-body motion.

movements. While theoretically accurate, magnetic tracking systems require an environment devoid of magnetic field distortions. To transmit their information, the sensors have to use either cables or wireless transmission to communicate with the central processor. The former requires that the subject be "tethered" with some kind of cabling harness. The latter requires that the subject also carry a power source such as a battery pack. The advantage of electromagnetic sensors is that the three-dimensional position and orientation of each sensor can be recorded and displayed in real time (with some latency). The drawbacks relate to the range and accuracy of the magnetic field and the restricted movement resulting from the instrumentation required.

Optical markers, on the other hand, have a much larger range, and the performers only have to wear reflective markers on their clothes (see Figure 6.1, Color Plate 2). The optical approach does not provide real-time feedback; however, and the data from optical systems are error prone and noisy. Optical marker systems use video technology to record images of the subject in motion. Because orientation information is not directly generated, more markers are required than with magnetic trackers. Some combination of joint and mid-segment markers is used. While industrial strength systems may use eight or more fast (up to 120 frames per second) infrared cameras, basic optical motion control can be implemented with consumer-grade video technology. Because the optical approach can be low cost, at least in some situations, it is the approach that is considered here.

6.2 Processing the images

The objective of motion control is to reconstruct the three-dimensional motion of a physical object and apply it to a synthetic model. With optical systems, three major tasks need to be undertaken. First, the two-dimensional images need to be processed so that the markers can be located, identified, and correlated in the multiple video streams. Second, the three-dimensional locations of the markers need to be reconstructed from their two-dimensional locations. Third, the three-dimensional marker locations must be constrained to a model of the physical system whose motion is being captured (e.g., a

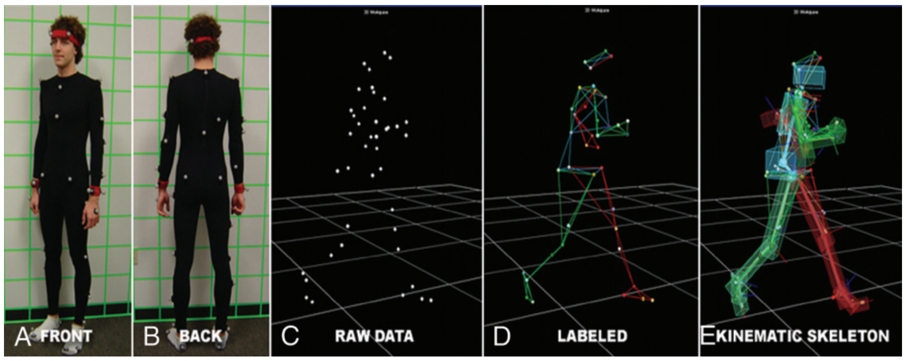


FIGURE 6.1

Images showing progression of optical motion capture session: (a) and (b) optical marker placement, (c) three-dimensional reconstruction positions of marker, (d) identified markers, (e) applied to skeleton.

(Image courtesy of Brian Windsor; data courtesy of Brent Haley.)

triangulated surface model of the performer). The first requires some basic image-processing techniques, simple logic, usually some user input, and sometimes a bit of luck. The second requires camera calibration and enough care to overcome numerical inaccuracies. The third requires satisfying constraints between relative marker positions.

Optical markers can be fashioned from table tennis balls and coated to make them stand out in the video imagery. They can be attached to the figure using Velcro strips or some other suitable method. Colored tape can also be used. The markers are usually positioned close to joints since these are the structures of interest in animating a figure. The difference between the position of the marker and that of the joint is one source of error in motion capture systems. This is further complicated if the marker moves relative to the joint during the motion of the figure. Once the video is digitized, it is simply a matter of scanning each video image for evidence of the optical markers. If the background image is static, then it can be subtracted out to simplify the processing. Finding the position of a single marker in a video frame in which the marker is visible is the easy part. This step gets messier the more markers there are, the more often some of the markers are occluded, the more often the markers overlap in an image, and the more the markers change position relative to one another. With multiple-marker systems, the task is not only to isolate the occurrence of a marker in a video frame but also to track a specific marker over a number of frames even when the marker may not always be visible.

Once all of the visible markers have been extracted from the video, each individual marker must be tracked over the video sequence. Sometimes this can be done with simple domain knowledge. For example, if the motion is constrained to be normal walking, then the ankle markers (or foot markers, if present) can be assumed to always be the lowest markers, and they can be assumed to always be within some small distance from the bottom of the image. Frame-to-frame coherence can be employed to track markers by making use of the position of a marker in a previous frame and knowing something about the limits of its velocity and acceleration. For example, knowing that the markers are on a walking figure and knowing something about the camera position relative to the figure, one can estimate the maximum number of pixels that a marker might travel from one frame to the next and thus help track it over time.

Unfortunately, one of the realities of optical motion capture systems is that periodically one or more of the markers are occluded. In situations in which several markers are used, this can cause problems in successfully tracking a marker throughout the sequence. Some simple heuristics can be used to track markers that drop from view for a few frames and that do not change their velocity much over that time. But these heuristics are not foolproof (and is, of course, why they are called heuristics). The result of failed heuristics can be markers swapping paths in mid-sequence or the introduction of a new marker when, in fact, a marker is simply reappearing again. Marker swapping can happen even when markers are not occluded. If markers pass within a small distance of each other they may swap paths because of numerical inaccuracies of the system. Sometimes these problems can be resolved when the three-dimensional positions of markers are constructed. At other times user intervention is required to resolve ambiguous cases. See Figure 6.1 for an example of mocap data processing.

As a side note, there are optical systems that use active markers. The markers are light-emitting diodes (LEDs) that flash their own unique identification code. There is no chance of marker swapping in this case, but this system has its own limitations. The LEDs are not very bright and cannot be used in bright environments. Because each marker has to take the time to flash its own ID, the system captures the motion at a relatively slow rate. Finally, there is a certain delay between the measurements of markers, so the positions of each marker are not recorded at exactly the same moment, which may present problems in animating the synthetic model.

A constant problem with motion capture systems is noise. Noise can arise from the physical system; the markers can move relative to their initial positioning and the faster the performer moves, the more the markers can swing and reposition themselves. Noise also arises from the sampling process; the markers are sampled in time and space, and errors can be introduced in all dimensions. A typical error might result in inaccurate positioning of a feature point by half a centimeter. For some animations, this can be a significant error.

To deal with the noise, the user can condition the data before they are used in the reconstruction process. Data points that are radically inconsistent with typical values can be thrown out, and the rest can be filtered. The objective is to smooth the data without removing any useful features. A simple weighted average of adjacent values can be used to smooth the curve. The number of adjacent values to use and their weights are a function of the desired smoothness. Generally, this must be selected by the user.

6.3 Camera calibration

Before the three-dimensional position of a marker can be reconstructed, it is necessary to know the locations and orientations of cameras in world space as well as the intrinsic properties of the cameras such as focal length, image center, and aspect ratio [8].

A simple pinhole camera model is used for the calibration. This is an idealized model that does not accurately represent certain optical effects often present in real cameras, but it is usually sufficient for computer graphics and image-processing applications. The pinhole model defines the basic projective geometry used to describe the imaging of a point in three-space. For example, the camera's coordinate system is defined with the origin at the center of projection and the plane of projection at a focal-length distance along the positive z-axis, which is pointed toward the camera's center of interest (Figure 6.2). Equivalently, the projection plane could be a focal length along the negative z-axis on the other side of the center of projection from the center of interest; this would produce an inverted image, but the mathematics would be the same.

The image of a point is formed by projecting a ray from the point to the center of projection (Figure 6.3). The image of the point is formed on the image (projection) plane where this ray intersects the plane. The equations for this point, as should be familiar to the reader, are formed by similar triangles. Camera calibration is performed by recording a number of image space points whose world

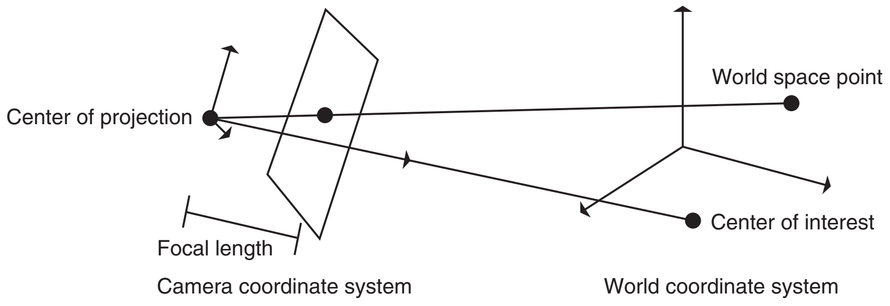


FIGURE 6.2

Camera model.

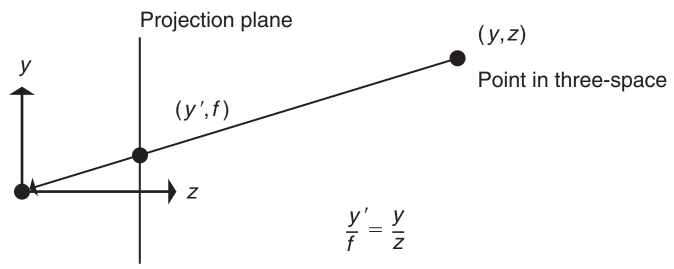


FIGURE 6.3

Y-Z projection of a world space point onto the image plane in the camera coordinate system.

space locations are known. These pairs can be used to create an overdetermined set of linear equations that can be solved using a least-squares solution method. This is only the most basic computation needed to fully calibrate a camera with respect to its intrinsic properties and its environment. In the most general case, there are nonlinearities in the lens, focal length, camera tilt, and other parameters that need to be determined. See Appendix B.11 for further discussion.

6.4 Three-dimensional position reconstruction

To reconstruct the three-dimensional coordinates of a marker, the user must locate its position in at least two views relative to known camera positions, C_1 and C_2. In the simplest case, this requires two cameras to be set up to record marker positions (Figure 6.4). The greater the orthogonality of the two views, the better the chance for an accurate reconstruction.

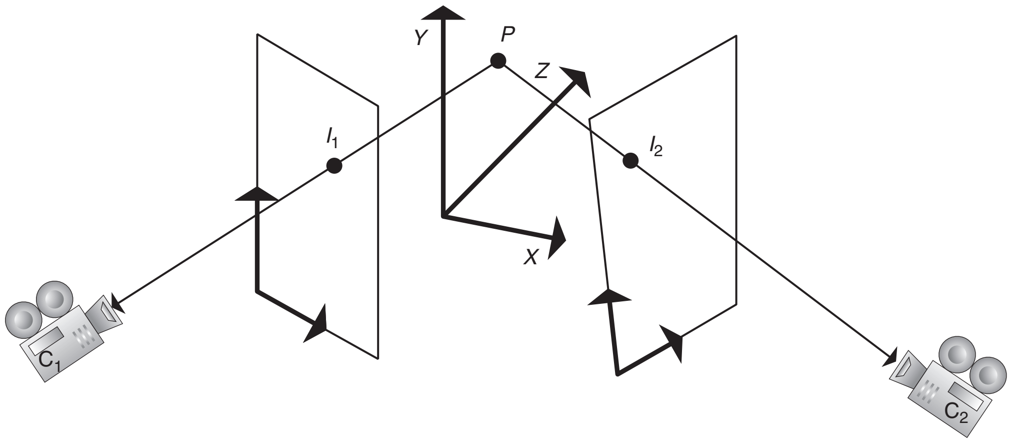


FIGURE 6.4

Two-camera view of a point.

If the position and orientation of each camera are known with respect to the global coordinate system, along with the position of the image plane relative to the camera, then the images of the point to be reconstructed (I_1, I_2) can be used to locate the point, P, in three-space (Figure 6.4). Using the location of a camera, the relative location of the image plane, and a given pixel location on the image plane, the user can compute the position of that pixel in world coordinates. Once that is known, a vector from the camera through the pixel can be constructed in world space for each camera (Eqs. 6.1 and 6.2).

$$C_1 + k_1(I_1 - C_1) = P \tag{6.1}$$

$$C_2 + k_2(I_2 - C_2) = P \tag{6.2}$$

By setting these equations equal to each other, $\mathrm{C}_1 + k_1\ (I_1\ C_1) = C_2 + k_2\ (I_2\ C_2)$. In three-space, this represents three equations with two unknowns, k_1 and k_2, which can be easily solved—in an ideal world. Noise tends to complicate the ideal world. In practice, these two equations will not exactly intersect, although if the noise in the system is not excessive, they will come close. So, in practice, the points of closest encounter must be found on each line. This requires that a P_1 and a P_2 be found such that P_1 is on the line from Camera 1, P_2 is on the line from Camera 2, and $P_2 P_1$ is perpendicular to each of the two lines (Eqs. 6.3 and 6.4).

$$(P_2 - P_1) \cdot (I_1 - C_1) = 0 \tag{6.3}$$

$$(P_2 - P_1) \cdot (I_2 - C_2) = 0 \tag{6.4}$$

See Appendix B.2.6 on solving for P_1 and P_2. Once the points P_1 and P_2 have been calculated, the midpoint of the chord between the two points can be used as the location of the reconstructed point. In the case of multiple markers in which marker identification and tracking have not been fully established for all of the markers in all of the images, the distance between P_1 and P_2 can be used as a test for correlation between I_1 and I_2. If the distance between P_1 and P_2 is too great, then this indicates that I_1 and I_2 are probably not images of the same marker and a different pairing needs to be tried. Smoothing can also be performed on the reconstructed three-dimensional paths of the markers to further reduce the effects of noise on the system.

6.4.1 Multiple markers

The number and positioning of markers on a human figure depend on the intended use of the captured motion. A simple configuration of markers for digitizing gross human figure motion might require only 14 markers (3 per limb and 2 for positioning the head). For more accurate recordings of human motion, markers need to be added to the elbows, knees, chest, hands, toes, ankles, and spine (see Figure 6.5). Menache [6] suggests an addition of three per foot for some applications. Also refer back to Figure 6.1, which shows a sample marker set on mocap talent.

6.4.2 Multiple cameras

As the number of markers increases and the complexity of the motion becomes more involved, there is greater chance for marker occlusion. To reconstruct the three-dimensional position of a marker, the system must detect and identify the marker in at least two images. As a result, a typical system may have eight cameras simultaneously taking images. These sequences need to be synchronized either

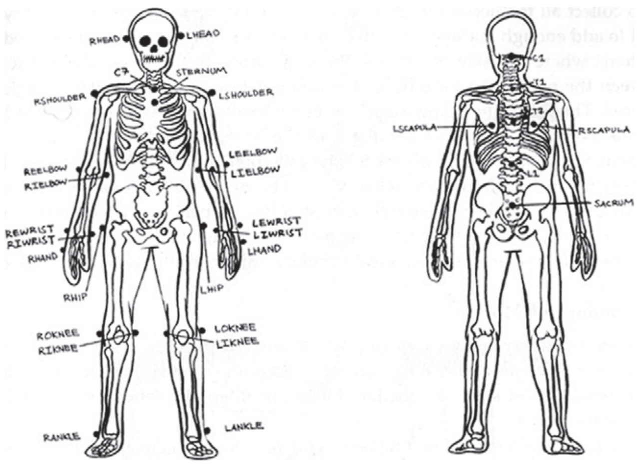


FIGURE 6.5

Complete marker set [6].

automatically or manually. This can be done manually, for example, by noting the frame in each sequence when a particular heel strike occurs. However, with manually synchronized cameras the images could be half a frame off from each other.

6.5 Fitting to the skeleton

Once the motion of the individual markers looks smooth and reasonable, the next step is to attach them to the underlying skeletal structure that is to be controlled by the digitized motion. In a straightforward approach, the position of each marker in each frame is used to absolutely position a specific joint of the skeleton. As a first approximation to controlling the motion of the skeleton, this works fine. However, on closer inspection, problems often exist. The problem with using the markers to directly specify position is that, because of noise, smoothing, and general inaccuracies, distances between joints of the skeleton will not be precisely maintained over time. This change in bone length can be significant. Length changes of 10-20 percent are common. In many cases, this is not acceptable. For example, this can cause *foot-sliding* of the skeleton (also known as *skating*). Inverse kinematics used to lock the foot to the ground can counteract this skating.

One source of the problem is that the markers are located not at the joints of the performers, but outside the joints at the surface. This means that the point being digitized is displaced from the joint itself. While a constant distance is maintained on the performer, for example, between the knee joint and the hip joint, it is not the case that a constant distance is maintained between a point to the side of the knee joint and a point to the side of the hip joint.

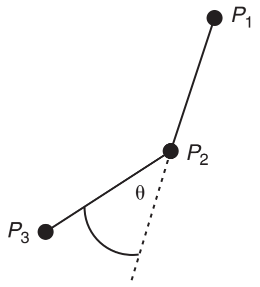
The one obvious correction that can be made is logically relocating the digitized positions so that they correspond more accurately to the positions of the joints. This can be done by using marker information to calculate the joint position. The distance from the marker to the actual joint can be determined easily by visual inspection. However, the problem with locating the joint relative to a marker is that there is no orientation information directly associated with the marker. This means that, given a marker location and the relative distance from the marker to the joint, the user does not know in which direction to apply the displacement in order to locate the joint.

One solution is to put markers on both sides of the joint. With two marker locations, the joint can be interpolated as the midpoint of the chord between the two markers. While effective for joints that lend themselves to this approach, the approach does not work for joints that are complex or more inaccessible (such as the hip, shoulder, and spine), and it doubles the number of markers that must be processed.

A little geometry can be used to calculate the displacement of the joint from the marker. A plane formed by three markers can be used to calculate a normal to a plane of three consecutive joints, and this normal can be used to displace the joint location. Consider the elbow. If there are two markers at the wrist (commonly used to digitize the forearm rotation) the position of the true wrist can be interpolated between them. Then the wrist-elbow-shoulder markers can be used to calculate a normal to the plane formed by those markers. Then the true elbow position is calculated by offsetting from the elbow marker in the direction of the normal by the amount measured from the performer. By recalculating the normal every frame, the user can easily maintain an accurate elbow position throughout the performance. In most cases, this technique is very effective. A problem with the technique is that when the arm straightens out the wrist-elbow-shoulder become (nearly) collinear. Usually, the normal can be interpolated during these periods of congruity from accurately computed normals on either side of the interval. This approach keeps limb lengths much closer to being constant.

Now that the digitized joint positions are more consistent with the skeleton to be articulated, they can be used to control the skeleton. To avoid absolute positioning in space and further limb-length changes, one typically uses the digitized positions to calculate joint rotations. For example, in a skeletal hierarchy, if the positions of three consecutive joints have been recorded for a specific frame, then a third of the points in the hierarchy is used to calculate the rotation of that limb relative to the limb represented by the first two points (Figure 6.6).

After posing the model using the calculated joint angles, it might still be the case that, because of inaccuracies in the digitization process, feature points of the model violate certain constraints such as avoiding floor penetration. The potential for problems is particularly high for end effectors, such as the hands or feet, which must be precisely positioned. Often, these must be independently positioned, and then the joints higher up the hierarchy (e.g., knee and hip) must be adjusted to compensate for any change to the end effector position.



$$\cos(\theta) = \frac{(P_3 - P_2)\ \ (P_2 - P_1)}{|P_3 - P_2|\,|P_2 - P_1|}$$

FIGURE 6.6

One-DOF rotation joint.

6.6 Output from motion capture systems

As the previous discussion has outlined, there is a lot of processing that must be done in order to convert the raw video data from the cameras to joint angles for a figure. Fortunately, motion capture software is available to enable the user to set up the marker set hierarchy, calibrate cameras, and automate as much of the marker processing as possible as well as provide interactive tools when user input is required to resolve ambiguities. The result can be used to animate a figure or saved to a file for later use.

Various de facto standard file formats exist for motion capture data. Two of the more common formats are the Acclaim two-file format (.asf/.asm) and the Biovision (a group of motion capture studios) file format (.bvh). While the various formats differ in their specifics, they all have the same basic information: the static information describing the joint hierarchy and the time-varying information containing the values for each degree of freedom (DOF) for each joint.

The joint hierarchy is described by (1) defining the joints including the number and type of each DOF, the order that DOF parameters are given, and joint limits, and (2) defining the hierarchical connectivity including the location of joint relative to previous joint. Ignoring header information and other detail not related to geometry, the static joint hierarchy information would look something like the following:

```
root
  joint Chest offset 0 5.21 0
    3 DOFs: Zrotation, Yrotation, Xrotation
    Limits: (−180,180) (−90,90) (0, 270)
    joint Neck
      offset 0 5.45 0
      3 DOFs: Zrotation, Yrotation, Xrotation
  ...
  joint LeftUpperLeg
    offset 3.91 0.0 0.0
    3 DOFs: Xrotation, Yrotation, Zrotation
    joint LeftLowerLeg
      offset 0.0 8.1 0.0
      1 DOF: Xrotation
  ...
```

In this simplified example, joint names are just for user identification, indentation indicates child–parent relationship, the offset is the location of the current joint relative to its parent, and the DOF line indicates the number of DOFs for that joint and the order they appear in the data. Optionally, joint limits can be specified.

The time-varying data records the reconstructed joint angles over some number of frames and might look something like the following:

```
total frames: 300
−2.4 40.2 3.2 2.8 22.0 −23.0 ...    −12.4 40.2 3.2 2.8 ...
−2.8 42.1 3.4 2.8 22.1 −23.2 ...    −12.2 42.3 3.3 3.8 ...
0.2 48.2 3.5 3.9 30.2 −25.2 ...  −14.4 45.8 3.1 3.0 ...
2.4 50.4 3.3 5.0 35.2 −23.3 ...  −20.2 44.9 3.3 2.8 ...
1.4 51.6 3.2 8.8 35.3 −20.2 ...  −22.4 45.2 3.7 2.8 ...
...
```

where each line is one time sample of the motion, the number of values on a line corresponds to the total DOFs of the figure, and they are given in the order they appear in the hierarchy information (the specific numbers in the previous example are completely random).

Motion capture (typically ASCII) files containing this information are the final output of a motion capture session. Many are made available on the Web for free download. It's not a difficult exercise, for example, to write a C/C++ program to read in a specific file format and visualize the motion by animating a stick figure using OpenGL.

6.7 Manipulating motion capture data

The data generated by the motion capture process are directly usable for animating a synthetic figure whose dimensions match those of the captured subject. Even then, as noted earlier, noise and numerical inaccuracies can disturb the resulting motion. In addition, it is not unusual for error-free captured motion to not quite satisfy the desires of the animator when finally incorporated into an animation. In these cases, the motion must be recaptured (an expensive and time-consuming process), or it must be manipulated to better satisfy the needs of the animator. There are various ways to manipulate mocap data. Some are straightforward applications of standard signal processing techniques while others are active areas of research. These techniques hold the promise of enabling motion libraries to be constructed along with algorithms for editing the motions, mapping the motions to new characters, and combining the motions into novel motions and/or longer motion sequences.

6.7.1 Processing the signals

The values of an individual parameter of the captured motion (e.g., a joint angle) can be thought of, and dealt with, as a time-varying signal. The signal can be decomposed into frequencies, time-warped, interpolated with other signals, and so forth. The challenge for the animator in using these techniques is in understanding that, while the original signal represents physically correct motion, there is nothing to guarantee physical correctness once the signal has been modified.

Motion signal processing considers how frequency components capture various qualities of the motion [1]. The lower frequencies of the signal are considered to represent the base activity (e.g., walking) while the higher frequencies are the idiosyncratic movements of the specific individual (e.g., walking with a limp). Frequency bands can be extracted, manipulated, and reassembled to allow the user to modify certain qualities of the signal while leaving others undisturbed. In order to do this, the signal is successively convolved with expanded versions of a filter kernel (e.g., 1/16, 1/4, 3/8, 1/4, 1/16). The frequency bands are generated by differencing the convolutions. Gains of each band are adjusted by the user and can be summed to reconstruct a motion signal.

Motion warping warps the signal in order to satisfy user-supplied key-frame-like constraints [9]. The original signal is $\theta(t)$ and the key frames are given as a set of (θ_i, t_i) pairs. In addition, there is a set of time warp constraints (t_j, t'_j). The warping procedure first creates a time warp mapping $t = g(t')$ by interpolating through the time warp constraints and then creates a motion curve warping, $\theta(t) = f(\theta, t)$. This function is created by scaling and/or translating the original curve to satisfy the key-frame constraints, $\theta'(t) = a(t)\theta(t) + b(t)$ whether to scale or translate can be user-specified. Once the functions $a(t)$

and $b(t)$ for the key frames are set, the functions $a(t)$ and $b(t)$ can be interpolated throughout the time span of interest. Combined, these functions establish a function $\theta(t')$ that satisfies the key-frame and time warp constraints by warping the original function.

6.7.2 Retargeting the motion

What happens if the synthetic character doesn't match the dimensions (e.g., limb length) of the captured subject? Does the motion have to be recaptured with a new subject that does match the synthetic figure better? Or does the synthetic figure have to be redesigned to match the dimensions of the captured subject? One solution is to map the motion onto the mismatched synthetic character and then modify it to satisfy important constraints.

This is referred to as *motion retargeting* [2]. Important constraints include such things as avoiding foot penetration of the floor, avoiding self-penetration, not letting feet slide on the floor when walking, and so forth. A new motion is constructed, as close to the original motion as possible, while enforcing the constraints. Finding the new motion is formulated as a space–time, nonlinear constrained optimization problem. This technique is beyond the scope of this book and the interested reader is encouraged to consult the work by Michael Gleicher (e.g., [2]).

6.7.3 Combining motions

Motions are usually captured in segments whose durations last a few minutes each. Often, a longer sequence of activity is needed in order to animate a specific action. The ability to assemble motion segments into longer actions makes motion capture a much more useful tool for the animator. The simplest, and least aesthetically pleasing, way to combine motion segments is to start and stop each segment in a neutral position such as standing. Motion segments can then be easily combined with the annoying attribute of the figure coming to the neutral position between each segment.

More natural transitions between segments are possible by blending the end of one segment into the beginning of the next one. Such transitions may look awkward unless the portions to be blended are similar. Similarity can be defined as the sum of the differences over all DOFs over the time interval of the blend. Both motion signal processing and motion warping can be used to isolate, overlap, and then blend two motion segments together. Automatic techniques to identify the best subsegments for transitions is the subject of current research [4].

In order to animate a longer activity, motion segments can be strung together to form a longer activity. Recent research has produced techniques such as motion graphs that identify good transitions between segments in a motion database [3] [5] [7]. When the system is confronted with a request for a motion task, the sequence of motions that satisfies the task is constructed from the segments in the motion capture database. Preprocessing can identify good points of transition among the motion segments and can cluster similar motion segments to reduce search time. Allowing minor modifications to the motion segments, such as small changes in duration and distance traveled, can help improve the usefulness of specific segments. Finally, selecting among alternative solutions usually means evaluating a motion sequence based on obstacle avoidance, distance traveled, time to completion, and so forth.

6.8 Chapter summary

Motion capture is a very powerful and useful tool. It will never replace the results produced by a skilled animator, but its role in animation will expand and increase as motion libraries are built and the techniques to modify, combine, and adapt motion capture data become more sophisticated. Current research involves extracting motion capture data from markerless video. This has the potential to free the subject from instrumentation constraints and make motion capture even more useful.

References

[1] Bruderlin A, Williams L. Motion Signal Processing. In: Cook R, editor. Proceedings of SIGGRAPH 95, Computer Graphics, Annual Conference Series. Los Angeles, Calif.: Addison-Wesley; August 1995. p. 97–104. Reading, Massachusetts. ISBN 0-201-84776-0.

[2] Gleicher M. Retargeting Motion to New Characters. In: Cohen M, editor. Proceedings of SIGGRAPH 98, Computer Graphics, Annual Conference Series. Orlando, Fla.: Addison-Wesley; July 1998. p. 33–42. ISBN 0-89791-999-8.

[3] Kovar L, Gleicher M, Pighin F. Motion Graphs, In: Proceedings of SIGGRAPH 2002, Computer Graphics, Annual Conference Series. San Antonio, Tex.; July 23–26,; 2002. p. 473–82.

[4] Kovar L, Gleicher M. Flexible Automatic Motion Blending with Registration Curves. In: Breen D, Lin M, editors. Symposium on Computer Animation. San Diego, Ca.: Eurographics Association; 2002. p. 214–24.

[5] Lee J, Chai J, Reitsma P, Hodgins J, Pollard N. Interactive Control of Avatars Animated with Human Motion Data, In: Computer Graphics, Proceedings of SIGGRAPH 2002, Annual Conference Series. San Antonio, Tex.; July 23–26, 2002. p. 491–500.

[6] Menache A. Understanding Motion Capture for Computer Animation and Video Games. New York: Morgan Kaufmann; 2000.

[7] Reitsma P, Pollard N. Evaluating Motion Graphs for Character Navigation. In: Symposium on Computer Animation. Grenoble, France. p. 89–98.

[8] Tuceryan M, Greer D, Whitaker R, Breen D, Crampton C, Rose E, et al. Calibration Requirements and Procedures for a Monitor-Based Augmented Reality System. IEEE Trans Vis Comput Graph September 1995;1 (3):255–73. ISSN 1077-2626.

[9] Witkin A, Popovic Z. Motion Warping. In: Cook R, editor. Computer Graphics, Proceedings of SIGGRAPH 95, Annual Conference Series. Los Angeles, Calif: Addison–Wesley; August 1995. p. 105–8. Reading, Massachusetts. ISBN 0-201-84776-0.