Chapter 7

Discussion

Realistic human motion is difficult to animate — not only is the motion itself intrinsically complicated, but human observers are quite adept at discerning even subtle flaws. *Controllably* generating realistic motion is even more challenging, because one needs an entire repertoire of motion to draw from, along with control mechanisms for creating motions with desired properties. Data-driven synthesis is a powerful method for solving both of these problems: a discrete set of high-quality examples is converted into a generative model that can produce a wide range of new motions, and a high-level interface makes it possible to generate motions with specific properties. However, data-driven synthesis can only reliably produce motion that is similar to the original examples, and expressive motion models therefore require large data sets. While motion capture provides a practical means of obtaining large collections of examples, previous work has been limited to manual techniques that become awkward and laborious when applied to large data sets. This dissertation has introduced automated methods for building motion models, greatly reducing the time and labor needed to convert a finite set of examples into a richer object that can controllably synthesize new, realistic motion. We have focused on methods for building and using two popular models that offer complementary forms of control: motion graphs, which control sequences of actions, and motion blends, which control individual actions. Four primary contributions have been made:

1. **Efficient and accurate enforcement of kinematic end effector constraints.** Chapter 3 presented a new algorithm for precisely enforcing kinematic end effector constraints without introducing discontinuities into the original motion, regardless of how many constraints there are or when they turn on and off. Combined with methods for inferring constraints in synthesized motion from the original data, this algorithm is a crucial component of our graph-based and blending-based synthesis models, as it allows important interactions between the character and the environment to be automatically preserved.

2. **Automated construction and use of motion graphs.** Chapter 4 showed how motion graphs can be automatically constructed by identifying places where motions are similar and synthesizing special transition motions at the points. Search algorithms were then used to provide a high-level interface for extracting particular motions from the resulting graph, and this technique was demonstrated on the important problem of directing very general kinds of motion down arbitrary paths.

3. **Automated motion blending.** Chapter 5 introduced registration curves, a data structure that summarizes relationships between the timing, local coordinate frame, and constraint state of arbitrarily many motions. Registration curves make it possible to blend motions with no user intervention beyond specification of the blend weights themselves, providing an automated alternative for systems that use manual blending methods to create transitions, interpolations, and continuously controllable motion.

4. **Automated construction of parameterized motions.** Building off of the technology in Chapter 5, Chapter 6 provided automated techniques for building parameterized motions, which are an important application of motion blending. First, a search algorithm was introduced for automatically extracting from a data set motion segments that are similar to a query, greatly simplifying the task of collecting a set of related example motions. Second, a new method was provided for parameterizing the space of interpolations; this method is more efficient and scalable than those used previously and does not require the example motions to span an *a priori* range of variation.

We have discussed motion graphs and motion blends separately, but they could be used together for additional flexibility. For example, motion blending could be used to create a sampling of different variations of a motion in order to supply a motion graph with additional source material. A clear direction for future work is the development of tools for automatically constructing hybrid graph/blending synthesis models, such as a graph where each edge is a parameterized motion rather than a static motion clip. Indeed, this particular model has been investigated previously in the Verbs and Adverbs system of Rose et al. [76]. In this work each parameterized motion was constructed manually (example motions were manually cropped and time-aligned for blending, and parameterizations relied on a priori knowledge of the accessible range of variation), and the graph structure itself was also directly defined by the user (linear-blend transitions were added at specified frames). We believe the methods of this dissertation could be extended to considerably automate this process. Moreover, we believe more general models could be developed where, for example, the set of allowable transitions depends not just on what kind action has been taken but also on what the properties (parameters) of that action were.

The remainder of this chapter discusses applications of our models, limitations of our methods, and avenues for future work.

7.1 Applications

Our motion models are appropriate for a variety of applications. For movie special effects, blending-based synthesis could be used to adjust the motion of human or human-like characters (e.g., aliens or monsters) so they better interact with the environment or with other actors. For crowd animation, graph-based synthesis could be used to generate motions that follow predefined trajectories constructed so as to avoid inter-character collisions. For games, graph-based synthesis could be used for general animation of non-player characters (where some control delay would be better tolerated), and blending-based synthesis could be used to adjust on the fly actions that depend upon unpredictable circumstances (e.g., picking up an object that could be dropped anywhere within a region). More generally, both graph-based and blending-based synthesis are appropriate

for rapid construction of sets of motions for movies, commercials, scripted training simulations, or any other form of visualization.

One application for which our models need further development is the online animation of user-controlled characters, such as in a game. Not only does this application require motion to be continually generated and adaptable to changing circumstances, but this motion must also be highly responsive to user control. Supplying this responsiveness is challenging because one must be able to switch between actions on a very short timescale (a quarter of a second or less is typical for a game). This requires considerably more careful organization of the available motions than the methods used in Chapter 4, as one needs to guarantee that a large number of actions are possible at effectively any point in time (see Gleicher et al. [33] for some initial work in this direction). Moreover, motion quality is more difficult to preserve in highly responsive models. Real people cannot instantly change their motion, and momentum and balance constraints prevent people from arbitrarily switching between different actions. At the very least, this means that mechanisms must exist for selecting a tradeoff between realism and responsiveness. A preferable (but possibly more challenging) approach is to develop tools for specially designing motions such that rapid transitions can be made realistically. This would likely require example motions to be captured more carefully, and may even require a rethinking of how such capture can best be carried out.

Our technical presentation has implicitly assumed that the acquisition of a data set is independent of model construction, but our techniques are sufficiently fast that they could be used in conjunction with a live motion capture session. A performer's motion could be recorded for a few minutes, and within a comparable amount of time one could already be resequencing motion with a motion graph, experimenting with different blends, or assessing the range of variation provided by a parameterized motion. Based on this, one could then decide whether existing motions need to be performed differently or whether new motions need to be captured. This quick feedback loop would simplify the process of creating motion models tuned to the requirements of a specific application.

7.2 Limits to Automation

While our methods are automated, they are not fully automatic. In general, we rely on user intervention whenever the meaning of a motion is important. For example, if a user wants to restrict the style of motion generated from a motion graph, then it is up to them to add the appropriate descriptive labels to the original data. Similarly, the user must tell our system what aspect of a motion is the appropriate basis for a parameterization; it cannot, for instance, automatically infer that the heel position at full extension is the most important part of a kick. We believe that this sort of user intervention is ultimately unavoidable, since computationally achieving high-level understanding of something as subtle as human motion is likely to remain an unsolved problem for the foreseeable future. However, while user intervention probably cannot be eliminated, it likely can be reduced, possibly through the application of machine learning algorithms that extrapolate from information that has already been supplied by the user [6].

A second place where we rely on user intervention is when determining how much a motion model can modify the original data. Allowing larger changes gives the model more freedom at the expense of placing looser guarantees on motion quality. This tradeoff is difficult to balance automatically because the levels of fidelity and control needed in a motion model depend upon the intended application. At the same time, our mechanisms for selecting this tradeoff are relatively simple. For motion graphs, the tradeoff is controlled by the distance thresholds used when deciding where to place transitions. For parameterized motions, it is controlled by the search threshold (which effectively determines how different the closest examples can be) and by the amount of extrapolation allowed during blending. In the data sets used in this dissertation (which are large by current standards), once these values were set the time needed to complete model construction was on the order of tens of seconds, providing quick feedback for users attempting to adjust these values.

7.3 Limits to Data Acquisition

An important limitation of our models is that they may require prohibitive amounts of input data when motions have a large number of fundamental degrees of freedom that must all be controlled. For example, the path synthesis problem addressed in Chapter 4 is effectively two dimensional, since for control purposes the only relevant property of a clip is the change in position in the floor plane. However, one might want to be able to control many different features of a motion. For instance, for walking one might want to control the speed, curvature, and step length of the walk cycle; the mood and gender of the character; and logically independent actions of the upper body like carrying objects of different sizes and weights. If each of these attributes are to be independently controlled, then it is necessary to have enough data that one property can be varied while everything else is held constant. With our current methods, this necessitates a combinatorial explosion in the amount of input data, since one must effectively have a captured motion for every possible combination of motion features. We believe that this problem can be mitigated by developing mechanisms that identify "independent" features of captured motion and layer them into a composite motion.

We have focused on reducing the amount of labor needed to build motion models from a motion capture data set, but we have not addressed the labor that goes into acquiring the data set itself. Current methods for cleaning raw marker data and fitting it onto a skeleton can still require a significant amount of user intervention, and while there are no published accounts of exactly how much labor is typically needed, anecdotal evidence suggests that a minute of raw motion data can take hours or even days to fully process. Except for some early work on marker-to-skeleton data conversion [11, 67], this problem has largely been ignored by the research community. In the short term, our methods could always be applied to imperfectly cleaned data to give a rough sense of what sorts of new motion can be synthesized, and some our methods (such as identifying transition locations or sets of similar motion segments) can be applied directly to marker locations. As large data sets become increasingly common, however, it will become necessary to drastically reduce the time needed to process marker data into high-quality skeletal motion.

7.4 Incorporating Other Techniques

Our models do not guarantee that synthesized motions are physically accurate, which may result in characters that appear out of balance or that perform motions which are beyond the strength limits of real humans. Methods exist for adjusting motions to better satisfy physical laws [47, 86], and any one of these could be applied to our synthesized motions as a postprocess, much as we currently apply the algorithm of Chapter 3 to enforce kinematic end effector constraints. At present, however, methods for enforcing physical constraints are not sufficiently reliable to be used in an automatic setting. Moreover, the true physical constraints on synthesized motion are unknown. For example, the mass distribution of an actor is typically not acquired at capture time, and it is unclear how the physical accuracy of a motion is affected when it is fit onto a rigid skeleton. Incorporation of physics constraints into data-driven synthesis hence remains an important area for future work.

In general the proportions of an animated character will not match those of the live performer, especially if multiple actors are used for a single data set. Existing retargeting methods [31, 54] use optimization algorithms to adapt captured skeletal motions to different bodies while preserving contact constraints and avoiding self-intersection. These methods could trivially be applied to the captured motions prior to model construction, so as to tailor the models to the character(s) of interest.

7.5 Generalizing Current Models

This dissertation has used synthesis models that are simplified in several ways. One important simplification is that we only treat full-body motion, ignoring finer structures like the face and hands. With current technology, the motion of these body parts is typically captured independently of the rest of the body. In real life, however, facial and hand motion are not truly independent of full-body motion. For example, a character's gestures are typically tightly correlated with facial expression and hand configuration. In future work we plan to develop models that can synthesize facial and hand motion along with full-body motion while preserving relationships between these different types of movement.

An additional simplification used in this dissertation is that every degree of freedom in the body is synthesized simultaneously; there is no logical decoupling between, for example, the upper and lower bodies. To illustrate, imagine that we have a motion of someone walking and a motion of someone waving while standing still. It seems natural to transfer the wave onto the walk so they occur in concert, and yet the models developed in this thesis provide no mechanism for accomplishing this. Executing this style of motion combination is nontrivial, and in particular simply attaching the upper body of one motion onto the lower body of another will not yield a realistic result, because subtle but important correlations between the movement of different body parts are ignored. Providing support for combining motions of different body parts onto a single body is also left for future work.