More Motion Capture* in Games — Can We Make Example-Based Approaches Scale?

Michael Gleicher

University of Wisconsin - Madison, Madison WI 53706, USA
gleicher@cs.wisc.edu
http://cs.wisc.edu/~gleicher

Abstract. Synthesis-by-Example (SBE) approaches have been successful at animating characters in both research and practice (games). These approaches assemble motions from pre-recorded examples, usually motion captured or keyframed. To date, the methods have relied on a small set of simple, generic building blocks for assembling the motions. To meet the increasing demands for better character movement and control in games, the approaches will need to evolve. An obvious path to address these challenges, employing increasingly large collections of examples, is enabled by recent research. However, scaling up the number of examples is unlikely to sufficiently scale up the quality of the character animation. Methods that make better use of examples will be required.

1 Introduction

Human (or human-like) characters are important in many types of computer games and interactive simulations. Often, the movements of these characters are created by an example-based approach where pre-recorded clips of movement, either from motion capture or keyframing, are assembled as needed. While these *Synthesis-By-Example* (SBE) approaches have been extremely successful in research and practice, future applications (e.g. improved games) will have increased demands. In this paper, we consider how the (SBE) approaches may evolve to meet these new needs.

Creating the movements for game characters is difficult. Animating human characters is difficult in general: human movement is incredibly complex, diverse, and subtle. People can do many things in many ways. Even simple, everyday actions like walking involve the complex coordination of many degrees of freedom, and involve an amazing degree of subtlety. Movements can convey a large amount of information in this subtlety: from watching someone walk, we can often get a sense of their mood, their personality, their intent, etc.

Interactivity (e.g. in games) provides another set of challenges. While many games include scripted "cut-scenes," most game play involves a player's action (or other unknown factors) and therefore cannot be determined ahead of time. Characters' movement must be responsive to the unfolding situation in the game,

* Or keyframe animated motion.

A. Egges, A. Kamphuis, and M. Overmars (Eds.): MIG 2008, LNCS 5277, pp. 82–93, 2008.

whether it is under the direct control of the player or otherwise. Games often involve longer durations requiring large amounts of animation.

There are two main strategies for creating game character motion: algorithmic (model-based) synthesis and synthesis by example (SBE). The key ideas contrast: algorithmic synthesis focuses on creating specialized techniques based on an understanding of movement, while SBE methods employ generic algorithms that avoid understanding the movement, treating it as generic data that is combined by simple, generic procedures. In practice, a spectrum of approaches blends these extremes.

Much of the success of current game character animation stems from the use of SBE approaches. As will be discussed in Section 2, examples provide a number of advantages for animation creation. SBE approaches (§3) have been successful in research and practice as they preserve these advantages. However, future games will need better character animation (§4). Improving the results of an SBE approach typically means using a larger set of examples, but while the enabling technologies for this are in place, this strategy is unlikely to scale (§5). Instead, we are likely to need to improve the methods used in SBE, some first steps in this direction are discussed in Section 6.

2 Why Examples?

The central idea of example-based, or data-driven, approaches to motion is that the movement is obtained from some outside source (like capturing the motion of an actor or having an animator create keyframes). The movement data is just data: a set of measurements that can be replayed. There is no "model" of the movement to explain why a particular set of data is the desired motion. All of the complexity and subtlety is stored in the data.

Examples avoid the need to model the movements to be created. High quality movement can be created without understanding the complexities and subtleties. Given enough time and effort, it may be possible to model a particular motion algorithmically. It is unclear how well the effort for any particular model applies to other motions. Modeling motion is difficult to scale to large repertoires or diversity of styles.

In contrast, example-based approaches readily scale to diverse sets of movement - all that is required is to obtain more examples. An actor (or a keyframe animator) is capable of an amazingly wide range of actions and styles, and can quickly produce many examples of movement. More significantly, example creation provides artistic control. Motion capture involves a partnership and communication between the actor, the director, and (to a lesser degree) the technologist. The actor and director can work together to create the necessary movement, without having to figure out how to explain it in sufficiently concrete terms that it can be encoded algorithmically. The beauty of example-based approaches is that they put creative control over the motion into the hands of the artistic team.

3 Synthesis by Example

By themselves, examples provide only the specific motions that are recorded. Synthesis-by-Example approaches create new motions based on a collection of pre-recorded examples.

Any synthesis approach involves some amount of algorithmic process, and usually data. We use the term *Synthesis-By-Example* (SBE) to refer to approaches that attempt to stay close to the model-free, data-driven concept. The spirit of such approaches is that they tend to be model-free: motions are just sets of numbers that are combined using simple, generic algorithms. The specifics and complexities of different movements come from the data. In contrast, *Model-Based* approaches encode more of the movement algorithmically. SBE approaches maintain the advantages of examples: movements are primarily specified through the examples, allowing for a diverse range of movements to be created, and for these movements to be specified by the artistic team.

Synthesis-By-Example approaches generally combine example motions in a few basic ways. These basic operations, such as blending or sequencing, are not necessarily simple: finding a motion that is "halfway in-between" running and sitting, or a transition between walking and a handstand, can be as difficult as creating the initial examples themselves. However, the space of motions is generally smooth: that is, small changes to valid motions are likely to be valid motions as well. Therefore, if motions are similar, combining them is easy. Basic mathematical operations, such as blending or sequencing, provide reasonable motions when provided with appropriate data.

Synthesis-by-Example methods generally use simple techniques for combining motions, but apply them only where they are likely to lead acceptable results. Almost all SBE is based on the same building blocks: motions are blended (generally by linear combinations of individual parameters), concatenated (often with transitions in between), layered (per-channel adjustments are phased-in and -out), and transformed (positioned spatially and temporally).

Much of the limitations of SBE approaches stem from the simplicity of the building blocks: because the basic techniques are only likely to work on appropriate example motions, their applicability is limited. Since the techniques generally only apply to create small changes to motions (e.g. to transition or blend between similar motions, or to make small adjustments to an example), the achievable results must be similar to the source examples. Phrased differently: without a model to provide an understanding of why an example is desirable, a method must be conservative in how it deviates from the example since it cannot know what in the example must be preserved.

While SBE methods all share the same basic building blocks (e.g. blending, concatenation) for combining motions, they differ in how these building blocks are used. These combination methods are only one of three aspects of an SBE method. Additionally there must be *preparation* to determine what data can be

combined and *control* to determine how to use the combinations to assemble the necessary motions. There is much more diversity in these other aspects of SBE.

3.1 Synthesis-by-Example in Games

The use of pre-recorded pieces that are assembled to create interactive animations in games certainly predates the use of motion capture. However, the demands of motion capture pushed the techniques. With motion capture data (or 3D animated characters in general), the pieces needed to be assembled more carefully to maintain visual quality (or at least avoid objectionable artifacts). As the early pioneers developed the use of motion capture in games, they also had to develop the basic foundations of the synthesis-by-example approaches to use these results. Unfortunately, there is little record of these early developments.[1]

By the mid-1990s, synthesis by example systems were applying motion capture to characters in games[2]. Transition graphs were planned out in advance, and manually constructed. Tools for supporting graph design were in use at least far back enough to be used for games released in early 1996 [5], although the first published description of a graph construction tool seems to be much later [6]. Even in this early era, blending was used both to create transitions as well as to create more precise control (e.g. blending left and right to create gradated turns), although most early blending was pre-computed because it was considered too costly to be done at run time[5].

Modern game characters have evolved from their early origins. However, they are still (often) built from the same basic SBE building blocks. Modern characters usually have a discrete set of actions, where each action has a continuous parameterization. For example, a character might be able to punch and kick to various locations and walk with various turning curvatures and speeds. Such characters still employ a transition graph to specify which actions can follow a given one, but the graphs describe parametric ranges of movements, rather than specific examples. The motions for the parametric actions might be created by any number of methods, such as blending examples together or making layered adjustments to a single example.

The key to practical application of synthesis by example approaches has been the careful planning and manual labor in preparing the data such that the examples work together[1,7]. Developers carefully choose what movements to acquire (either capture or hand animate), carefully plan these movements so that they can transition or blend as needed, carefully choose the blends and transitions to facilitate the necessary control over the character, and carefully perform the movements to create examples that can be connected.

[1] Published accounts of the early motion capture, such as [1] or [2,3,4], generally focus on the development of the capture hardware, the application of data to character models, or film applications. Most of my knowledge of this "folklore" of motion editing history comes from conversations with the pioneers. In particular, Mark Schafer has graciously provided me with some specifics of the history at Acclaim.

[2] It is difficult to pinpoint the exact origins, since games used motion capture for cut-scenes and promotional animations as well.

3.2 Synthesis-by-Example in Research

Synthesis-by-Example approaches have evolved differently in research than in practice. The same basic building blocks of blending, layering, and concatenation are used. The introduction of these methods to the research community (for example, blending [8,9,10,11] or layering [12,13]), often came after they had already been deployed in practice. While there are examples of efforts to provide improved building blocks, such as better transition generation [14] or alternate blending schemes [15], most work on synthesis-by-example approaches has focused on making use of the same basic building blocks for combining motions as used in practice.

Where research has deviated from practice is in the application of automation and more advanced algorithms for the preparation of the examples and the creation of control strategies. For example, the "motion graph" approaches [16,17,18] (and their successors) distinguish themselves from their predecessors by automatically searching through a repository of motions to find potential transitions to introduce. This opportunistic graph construction avoids the labor of identifying transitions and offers the potential for reducing the planning effort. However, the unstructured nature of the resulting graphs makes the control problem more complex. Research has addressed this challenge with a wide range of search strategies.

Reducing the need for planning and manual labor through automation and clever control strategies is a common theme in SBE research. For example, [19] shows how the alignments required for blending can be determined automatically. This automation not only saves labor, but allows for the creation of blends that would be impractical to create manually. In an extension of the approach, [20] shows how a database of motions can be searched to find appropriate examples to blend and provides an automatic way to build the control strategy that maps parameters to blending weights. Again, this not only significantly reduces the amount of effort required to create parametric actions, but also extends the range of where blending can be applied. The ease with which parametric actions can be built allows for easy experimentation, which often leads to surprising examples.

Another aspect of SBE that has been automated in research is the design of controls (choosing which examples to combine and how). The search strategies demanded by unstructured graphs have lead to entirely new control paradigms. For interactive control, pre-computed search [21,22] and optimal control [23] methods automate control mappings even in situations where the example collection is not carefully planned. Automatic blend parameterizations [20] allow for precise control of blends even when the example set is irregular.

4 The Needs for Better Game Animation

While character animation in games is quite advanced, improvement is still important. In current games, the quality of the movement of characters already lag other aspects, particularly the visual appearance of the environments. Advances in interactive rendering that exploit increasing hardware resources (GPUs) are

widening the gap. If nothing else, movement quality must improve so that characters don't look out of place in the well-rendered scenes.

Characters with more convincing movement can better add to creating compelling visuals, and provide game designers with more flexibility to create a wider range of compelling experiences for players. However, to provide for this range of future games, the technology to animate character must provide:

More quality - the characters' movements should meet the designer's goals for the visual style that creates the experience. For example, if a game's design is meant to be realistic, the characters in the game should move realistically.

More actions - characters should have richer repertoires, and ultimately be able to do (at least) the range of things that actors can do.

More styles - characters should be able to do these things in the entire range of ways that people do them, as these differences are often important.

More subtlety - the differences in the ways that things are done can often be quite subtle. Conveying these subtleties is important for communicating things such as mood, intent and personality. Current games generally rely on other means (such as narration) to convey this.

More situated - the characters' movements must relate correctly to their surroundings, otherwise the illusion of the character inhabiting its world is broken. This requires a degree of precision in motions: if connections (such as contacts) are not exact, they are a very visible reminder.

More responsiveness - characters should respond quickly to their control (e.g. a player's commands or other occurances in their environment).

While the current technologies allow for excellence in some of these categories, this excellence usually comes at the expense of other attributes. For example, it is possible to make a character with very high quality movement if its range of actions is very small, or is unresponsive - the cut scenes in games often have very good motion.

5 Using More Examples

The results of SBE can be improved by using larger sets of examples. Larger sets of examples help improve many of the aspects of character animation:

More actions and styles - most obviously, having a wider range of examples is the primary (possibly the only) way to extend the range of actions and styles the character is capable of.

More quality, subtlety and artistic control - more examples means that whatever motion is going to be created is more likely to be close to an example. Deviation from the examples is the source of loss of quality (because the examples are given by the artistic team and express their goals).

To see how increased example sets can help, consider a simple example: a character walks up to a bookcase and grabs a book. With a single example, an SBE approach might use IK to reposition the hand to reach different

places on the book shelf, and then to propagate these changes to neighboring frames. If many examples are provided (i.e. having examples of a person reaching to multiple locations), the other variability in the movements can be captured. For example, a person might move differently to read a higher object, and their eyes need to find the book before they can grab it. While it may be possible to algorithmically encode these details, each would need to be identified, understood, and implemented.

More responsiveness - a larger example set provides more options for synthesis methods to create motions, so it is more likely that a choice will be readily available when a change is necessary.

Fortunately, technological improvements have facilitated obtaining and using larger example sets. The equipment and software for motion capture and post-processing makes obtaining more examples practical. The wider availability of the equipment reduces the limitations in the amount of capture possible. At runtime, data storage is becoming more plentiful and motion data is very compact relative to other assets like sounds and textures.

Automation in the authoring tools, as described in Section 3.2 are better able to scale to larger number of examples. Automatic tools not only reduce the amount of labor to process the data, they also reduce the required planning and can provide results that cannot be achieved manually, such as very complex blend spaces, accurate blend parameterizations, or near-optimal control.

5.1 Why Example Sets Cannot Scale

One possible stumbling block for increased motion usage is performance. While modern games (especially consoles) have considerable processing resources, they are often constrained in the amount of bandwidth available to access examples. Also, motion synthesis is only one aspect of a game whose computing resources are growing. Increased computational abilities have raised player expectations about rendering quality, visual complexity, and artificial intelligence, all of which demand increasing amounts of processing resources. To retain practicality, the memory bandwidth requirements of SBE methods needs to be considered more thoroughly as example sizes scale.

A more challenging issue is unintended variation in the examples. Some of this comes from limitations in the combination processes. For example, in the bookcase scenario above, in order for blending to work, the character must always initiate the reaching motion with a step on the same foot. More subtle variabilities may not cause failures, but instead have unintended consequences. For example, imagine a walking character created by combining footsteps found in an example database. While the large example set may provide a diverse range of steering and speeds allowing for a very controllable character, every footstep has its own story. For example, on any given example the actor might be more or less tired or distracted, have a more or less clear idea of where they are going, or may have stumbled or twitched. As SBE chooses different examples for each

step, it may mix these stories[3]. The degree of quality assurance to insure the regularity of the examples may preclude large example sets.

All of the variabilities could be viewed positively: all of the differences between motions might become parametrically controllable. There are several reasons why such an approach is unlikely to scale. First, existing methods for automation are not good at identifying and parameterizing the more subtle variability. Second, the different parameters aren't necessarily orthogonal. Third, tradeoffs between differences are difficult to compare (is it better to pick an example that is similar in tiredness, personality, or position?). Fourth, as the number of parameters grows, the space of possibilities grows exponentially. This leads to increased demands on the number of examples required to adequately sample the space and the methods for controlling within it.

The inability of SBE approaches to deal with high dimensional parameter spaces is likely to be the ultimate limitation on its scalability. To create truly expressive and responsive characters, a myriad of properties need to be controlled for any particular action. In games, it is easy to see a slippery slope of wanting more and more parameters to be controlled: a walking character should be able to turn, vary its speed, step up/down on obstacles, have varying levels of injury, have varying levels of intensity/focus, ...

The curse of dimensionality is a final limiting factor of the standard SBE approaches. While automation might help the methods scale to larger example sets, it cannot help them scale enough. Example sets would need to grow exponentially.

6 Scaling SBE Methods

The previous section argued that increasing the size of the example sets used in SBE methods is unlikely to scale to the needs of future applications. Similar arguments have been made by several others in this workshop (c.f. [24,25,26]).

The power of example-based methods to allow for artistic collaboration to specify desired movement properties by example means that the SBE approach is unlikely to go away. Purely algorithmic approaches, in some sense the antithesis of example-based ones, still must somehow engage collaboration with the artists, designers, and directors who provide the vision of the movement requirements. While Perlin has shown great progress in creating parameterized algorithmic controllers at this workshop [25], it is unclear how well this approach will scale to large repetoires, movement styles, or ranges of visual style (including realism). It takes a very expert programmer to understand movement well enough to devise algorithmic synthesis processes, and the need to make these flexible enough to meet an artists' stylistic wishes even further complicates the problem.

I believe that the future of technology for animated characters in interactive systems lies as a hybrid of synthesis-by-example and algorithmic approaches.

[3] Sometimes, coherence in the variability can lead to unusual outcomes. In one capture shoot, left turns were captured in the morning and right turns at the end of the day, yielding a character that looked tired whenever they turned right.

There are two different ways in which the "pure" approaches may mix: using more sophisticated methods for combining examples and using collections of data to derive algorithmic controllers. Examples of both paths can be seen in the literature. Popović's presentation at this workshop [26] provides a particularly compelling example of how an algorithmic controller might be derived from examples.

Here, I provide two brief examples from our group's work[4] that illustrate these hybrid approaches. In the first example, the standard set of methods for combining examples is extended, providing a mechanism for scaling to a broader repertoire without a commensurate expansion in the example set. In the second example, an algorithmic synthesis procedure is derived from data. While the strategies are quite different, they both represent attempts to provide more scalable SBE approaches.

6.1 Splicing Actions

Different parts of a character might perform different actions simultaneously. For example, a character might wave, carry a box, or stare in a particular direction at the same time that they walk, stand or sit. This creates a potential combinatorial explosion of possible things a character might do (i.e. stare to the left while standing, carrying a box, tapping the left foot). When limited to the traditional mechanisms for combining examples, examples are needed for each combination.

Being able to partition the parts of a character and provide independent examples (or motion synthesis methods) for each avoids the need for all combinations. For example, if we could consider the upper and lower body separately, we could have a set of example upper body actions (e.g. wave, salute, carry a box, hold a coffee cup) and lower body actions (e.g. walk and run with various turns and speeds) without having examples of all n^2 combinations. *Splicing* methods assemble movements for a character from independent sources of movement for each part.

Adding splicing to the set of methods used to create SBE offers a mechanism for greatly reducing the number of examples needed. However, when multiple actions are performed simultaneously, they do interact (e.g. carrying a heavy box changes the way one walks). Creating these couplings can be challenging. Simple splicing methods do not provide the proper couplings, and therefore often look wrong. However, splicing is so useful that these simple splicing methods are commonly used in practice in games despite the quality problems.

The diversity and complexity of couplings between body parts suggests that a general solution for splicing may be illusive. To date, researchers have focused on creating splicing methods for specific parts and situations, such as hands [28]. In [29] we presented a method for splicing upper body motions onto lower bodies in the specific case where both examples come from locomotion. The method works by specifically identifying important types of couplings, including posture, coordinated timing, and spatial alignment, and taking specific steps to make sure that each coupling is properly established in the result.

[4] The research described in these sections is part of the Ph. D. thesis research [27] of my former student, Rachel Heck.

Splicing can serve as a building block for synthesis-by-example, along side the more usual blending and concatenation. By building SBE approaches with a richer set of building blocks, there is a potential to achieve greater performance (in terms of result quality, repetoire range, range of stles, etc.) without an explosion of examples. However, our splicing method lacks the simplicity, genericness, and broad applicability of the "pure" SBE approaches. In a sense, our splicing technique shows a mixture of a model-based and example-based approach: understanding of a specific class of motions was used to create an algorithmic synthesis method that relies on data.

6.2 Gaze Control

It is important to be able to control the gaze direction of an animated character. To an observer, shifts in the location or direction where a character is looking might not only indicate a shift in attention but can also convey a person's goal before they act on it.

Effective control of the gaze direction is complicated. The gaze direction is determined by the orientation of head, as well as the eyes. To look in a particular direction, a person might adjust their torso and neck (in order to orient the head), as well as move their eyes. The timing and coordination of these movement are also complicated, as the eyes can move much more rapidly than the head, leading to a progression where they move first, and usually overshoot the target. The specifics of the movements depend on the direction, the size of the change, and even the individual and their mood (e.g. different people have different ranges of motions and preferences, and may react differently if they are tired or scared).

Gaze control would be very difficult with standard SBE approaches. It adds at least 2 new parameters to any movement (the direction of gaze). Providing good control would require not only sufficient examples to allow for the range of gaze directions, but also to allow for the range of gaze timings (i.e. a character might look in a particular direction at a particular instant). The number of examples required to create such a diverse space of possibilities would be prohibitive.

We have developed a technique for controlling the gaze of an animated character, described in Chapter 5 of [27]. Given the character's motion and a gaze target (a direction at a particular time), the motion is adapted to meet the gaze target. The technique uses a specifically designed model of gaze motions, built from an understanding of the psychological and physiological principles involved. In many ways, the technique shows the traditional process of algorithmic synthesis development: where a programmer gained an understanding of a particular movement and encoded this understanding into an algorithmic process with appropriate controllability. In this case, the algorithmic synthesis produces a change to an existing motion (that is added by layering).

Our gaze technique also employs an example-based approach to achieve individual and/or mood/style variability. Motion capture data of an actor performing a number of examples of gaze movement is used to generate a set of parameters that are used by the gaze controller. Effectively, the algorithmic

model is built from example data. The gaze technique exemplifies the broader goal of deriving algorithmic control is from example data.

7 The Future

A great actor provides a director/producer with an amazing range of possible actions and movement styles. Computer animated characters extend this flexibility in other ways, such as providing for different visual styles (e.g. realistic, cartoony) or responsiveness tradeoffs. The greater range of character animation makes more tools available for designers to create better interactive experiences. The technology to drive future game characters will need to be more like a great actor, providing game developers with a powerful and expressive component to create better games.

These future game characters will require technology beyond what is currently available. Examples are still likely to be useful, as they enable a designer or director to specify the movements that they want, as well as providing an effective way to create the necessary diversity of actions and styles. However, current synthesis-by-example approaches with larger sets of examples are unlikely to scale to meet to meet the challenges. New methods that can make more use out of a compact set of examples will be required.

Acknowledgements

The research work performed in our group represents the efforts of a long list of talented students, including Lucas Kovar, Rachel Heck (now Rose), Alex Mohr, Andrew Selle, Mankyu Sung, and Hyun Joon Shin.

References

1. Menache, A.: Understanding Motion Capture for Computer Animation and Video Games. Morgan Kaufmann, San Francisco (1999)
2. Elson, M., Sturman, D., Dyer, S., Trager, W., Schafer, M.: Character motion systems (SIGGRAPH, Course Notes) (1994)
3. Sturman, D.: A brief history of motion capture for computer character animation. SIGGRAPH Hypergraph Web Page (1994)
4. Trager, W.: A practical approach to motion capture: Acclaim's optical motion capture system. SIGGRAPH HyperGraph Web Page (1994)
5. Schafer, M.: Personal Communication (July 2008)
6. Mizuguchi, M., Buchanan, J., Calvert, T.: Data driven motion transitions for interactive games. In: Eurographics 2001 Short Presentations (September 2001)
7. Kines, M.: Planning and directing motion capture for games. Game Developer Magazine (1998); Also in GDC 1998 and Gamasutra
8. Perlin, K.: Real time responsive animation with personality. IEEE Transactions on Visualization and Computer Graphics 1(1), 5–15 (1995)

9. Guo, S., Roberge, J.: A high-level control mechanism for human locomotion based on parametric frame space interpolation. In: Proc. of Eurographics Workshop on Computer Animation and Simulation 1996, pp. 95–107 (August 1996)
10. Wiley, D., Hahn, J.: Interpolation synthesis of articulated figure motion. IEEE Computer Graphics and Application 17(6), 39–45 (1997)
11. Rose, C., Cohen, M., Bodenheimer, B.: Verbs and adverbs: multidimensional motion interpolation. IEEE Computer Graphics and Application 18(5), 32–40 (1998)
12. Witkin, A.P., Popović, Z.: Motion Warping. In: Proc. SIGGRAPH 1995, pp. 105–108 (August 1995)
13. Bruderlin, A., Williams, L.: Motion signal processing. In: Proc. ACM SIGGRAPH 1995. Annual Conference Series, pp. 97–104 (1995)
14. Rose, C., Guenter, B., Bodenheimer, B., Cohen, M.F.: Efficient generation of motion transitions using spacetime constraints. In: Proc. SIGGRAPH 1996, pp. 147–154 (1996)
15. Mukai, T., Kuriyama, S.: Geostatistical motion interpolation. ACM Transactions on Graphics 24(3), 1062–1070 (2005)
16. Arikan, O., Forsyth, D.A.: Synthesizing Constrained Motions from Examples. ACM Transactions on Graphics 21(3), 483–490 (2002)
17. Kovar, L., Gleicher, M., Pighin, F.: Motion graphs. ACM Transactions on Graphics 21(3), 473–482 (2002)
18. Lee, J., Chai, J., Reitsma, P., Hodgins, J., Pollard, N.: Interactive control of avatars animated with human motion data. ACM Trans. on Graph. 21(3), 491–500 (2002)
19. Kovar, L., Gleicher, M.: Flexible automatic motion blending with registration curves. In: Proceedings of the Symposium on Computer Animation (July 2003)
20. Kovar, L., Gleicher, M.: Automated extraction and parameterization of motions in large data sets. ACM Transactions on Graphics 23(3), 559–568 (2004)
21. Lee, J., Lee, K.H.: Precomputing avatar behavior from human motion data. In: SCA 2004: Proc. of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 79–87 (2004)
22. Lau, M., Kuffner, J.J.: Precomputed search trees: Planning for interactive goal-driven animation. In: 2006 ACM SIGGRAPH / Eurographics Symposium on Computer Animation, pp. 299–308 (September 2006)
23. Treuille, A., Lee, Y., Popović, Z.: Near-optimal character animation with continuous control. ACM Trans. Graph. 26(3), 7 (2007)
24. Thalmann, D.: Motion modeling: Can we get rid of motion capture? In: Egges, A., Kamphuis, A., Overmars, M. (eds.) Motion in Games. LNCS. Springer, Heidelberg (2008)
25. Perlin, K., Seidman, G.: Autonomous digital actors. In: Egges, A., Kamphuis, A., Overmars, M. (eds.) Motion in Games. LNCS. Springer, Heidelberg (2008)
26. Popović, Z.: Towards robust dynamic controllers for high-fidelity character locomotion. In: Egges, A., Kamphuis, A., Overmars, M. (eds.) Motion in Games. LNCS. Springer, Heidelberg (2008)
27. Heck, R.: Automated Authoring of Quality Human Motion for Interactive Environments. PhD thesis, Dept. of Comp Sci., University of Wisconsin (2007)
28. Majkowska, A., Zordan, V., Faloutsos, P.: Automatic splicing for hand and body animations. In: Proc. of the Symposium on Computer Animation (SCA) (2006)
29. Heck, R., Kovar, L., Gleicher, M.: Splicing upper-body actions with locomotion. Computer Graphics Forum 25(3) (2006); Proc. Eurographics