CHAPTER 1

INTRODUCTION

Machine learning has experienced explosive growth in the last few decades. It has achieved sufficient maturity to provide efficient techniques for a number of research and engineering fields including machine perception, computer vision, natural language processing, syntactic pattern recognition, and search engines. Machine learning provides a firm theoretical basis upon which to propose new techniques that leverage existing data to extract interesting information or to synthesize more data. It has been widely used in computer animation and related fields, e.g. rendering, modeling, geometry processing, and coloring. Based on these techniques, users can efficiently utilize graphical materials, such as models, images, and motion capture data to generate animations in practice, including virtual reality, video games, animation films, and sport simulations.

In this chapter, we introduce the uses of machine learning methods and concepts in contemporary computer animation and its associated fields. We also provide novel insights by categorizing many existing computer animation techniques into a common learning framework. This not only illuminates how these techniques are related, but also reveals possible ways in which they may be improved. The details of these potential improvements are presented in subsequent chapters.

Modern Machine Learning Techniques and Their Applications in Cartoon Animation Research, First Edition. By Jun Yu and Dacheng Tao

1.1 PERCEPTION

Computer animation has become a prevalent medium for science, art, education, communication, and entertainment. The production of computer animation involves multidisciplinary skills, and the special effects for film are generated by the collaboration of animators, excellent artists, and engineers.

Generally, in animation production, the time-consuming and tedious work is artistic, not technical [111, 244]. This means that the efficiency of the artists is the bottleneck in animation production, due to the fact that computer animation algorithms are eminently reusable, but the data they operate on are often customized and highly stylized. This bottleneck is exemplified by computer-generated film production where the human workload is usually more than 80% artistic (e.g., modeling, texturing, animating, etc.) [79].

To remove the bottleneck in animation data generation, data transformation and modeling techniques have been adopted to synthesize or generalize data. Machine learning techniques are utilized based on the existing data because of two specific merits: structural information extraction and new data synthesizing. Thus, time can be saved in animation production by using machining learning techniques to reuse existing data. Allen et al. [16] proposed a novel technique which generates meshes of new human bodies based on a small set of example meshes through regression.

Even when the computation workload is more important than the human workload, clever synthesis and reuse of data can still often be beneficial. Schödl et al. [245] provided a new type of video production technique which synthesizes videos from a finite set of images. This allows for the rapid generation of convincing videos that do not exhibit discernible noise. Tang [279] et al. proposed a framework for generating video narrative from existing videos in which the user only needs to conduct two steps: selecting the background video and avatars, and setting up the movement and trajectory of avatars. By using this approach, realistic video narratives can be generated from the chosen video clips and the results will be visually pleasing.

In this chapter, recent uses of machine learning techniques in computer animation are presented. Techniques such as manifold learning and regression are reviewed, and we introduce important and popular topics in computer animation and related fields, discussing how machine learning has been leveraged in each. We also provide suggestions for future work on enhancing computer animation through machine learning.

1.2 OVERVIEW OF MACHINE LEARNING TECHNIQUES

In this section, we provide an overview of modern machine learning techniques. A more thorough introduction to machine learning is provided in Ref. [190, 199]. Machine learning [266] is concerned with the problem of how to build computer programs that acquire and utilize information to effectively perform tasks. One successful example is the field of computer vision [266]. Many vision techniques rely on, and are even designed around, machine learning. For example, machine learning

through Bayesian inference has proven extremely powerful at analyzing surveillance video to automatically track humans [357, 352] and label human activities [185].

Unfortunately, the computer animation community has not utilized machine learning as widely as computer vision. Nevertheless, we can expect that integrating machine learning techniques into the computer animation field may create more effective methods. Suppose that users wish to simulate life on the streets of Bei Jing in ancient China, or in a mysterious alien society. They have to generate all the 3D models and textures for the world, the behaviors of animations for the characters. Although tools exist for all of these tasks, the scale of even the most prosaic world can require months or years of labor. An alternative approach is to create these models from existing data, either designed by artists or captured from the world. In this section, we introduce the idea that fitting models from data can be very useful for computer graphics, along with the idea that machine learning can provide powerful tools.

To consider the problem of generating motions for a character in a movie, it is important to realize that the motions can be created procedurally, i.e. by designing algorithms that synthesize motion. The animations can be created "by hand" or captured from an actor in a studio. These "pure data" approaches give the highest-quality motions, but at substantial cost in time and effort of artists or actors. Moreover, there is little flexibility: If it is discovered that the right motions were not captured in the studio, it is necessary to retrack and capture more. The situation is worse for a video game, where all the motions that might conceivably be needed must be captured. To solve this problem, machine learning techniques can be adopted to promise the best of both worlds: Starting with an amount of captured data, we can procedurally synthesize more data in the style of the original. Moreover, we can constrain the synthetic data, for example, according to the requirements of an artist. For such problems, machine learning offers an attractive set of tools for modeling the patterns of data. These data-driven techniques have gained a steadily increasing presence in graphics research. Principal components analysis and basic clustering algorithms are commonly used in SIGGRAPH, the world's premier conference on computer graphics and interactive techniques. Most significantly, the recent proliferation of papers on cartoon and motion synthesis suggests a growing acceptance of learning techniques. In this section, we introduce modern machine learning techniques from which the animations can deeply benefit. Furthermore, we introduce the applications of these techniques in computer animation.

1.2.1 Manifold Learning

Techniques of manifold learning are important and have been popularly applied in computer animation [49, 16, 131, 183]. Manifold learning [313] can be defined as the process of transforming measurements from a high-dimensional space to a low-dimensional subspace through the spectral analysis on specially constructed matrices. It aims to reveal the intrinsic structure of the distribution of measurements in the original high-dimensional space.

Manifold learning has a close relationship with computer animation research. For example, Principal Components Analysis (PCA) [116] is an extremely simple,

linear manifold learning method which is primarily used as a preprocess for high-dimensional problems. PCA has been used as a preprocess in "Style Machines" [49] which addresses the problem of stylistic motion synthesis by learning motion patterns from a highly varied set of motion capture sequences. The human body pose data typically contain 30-40 degrees of-freedom (DOFs); however, the proposed method would learn very slowly on such a large number of DOFs. Applying linear manifold learning is very fast, and the resulting 10-dimensional space maintains the important underlying structure of the motion data. More sophisticated non-linear manifold learning approaches [241, 287, 31, 353] have been developed and applied in the computer animation field. For example, Shin and Lee [253] proposed a low-dimensional motion space constructed by ISOMAP, in which high-dimensional human motion can be effectively visualized, synthesized, edited, parameterized, and interpolated in both spatial and temporal domains. This system allows users to create and edit the motion of animated characters in several ways: The user can sketch and edit a curve on low-dimensional motion space, directly manipulate the character's pose in three-dimensional object space, or specify key poses to create inbetween motions.

Representative manifold learning can be classified into linear and nonlinear manifold learning groups. Principal Component Analysis (PCA) [116] and Linear Discriminant Analysis(LDA) [83] are two typical linear manifold learning techniques. PCA, which is unsupervised, maximizes the mutual information between original high-dimensional Gaussian distributed measurements and projected low-dimensional measurements. LDA finds a projection matrix that maximizes the trace of the between-class scatter matrix and minimizes the trace of the within-class scatter matrix in the projected subspace simultaneously. LDA is supervised because it utilizes class label information. Representative nonlinear manifold learning includes Locally Linear Embedding (LLE) [241], ISOMAP [287], Laplacian Eigenmaps (LE) [31], Hessian Eigenmaps (HLLE) [78], and Local Tangent Space Alignment (LTSA) [353]. Locally Linear Embedding (LLE) [241] uses linear coefficients, which reconstruct a given measurement by its neighbors, to represent the local geometry, and then seeks a low-dimensional embedding in which these coefficients are still suitable for reconstruction. ISOMAP, a variant of MDS, preserves global geodesic distances of all pairs of measurements. LE preserves proximity relationships by manipulations on an undirected weighted graph, which indicates neighbour relations of pairwise measurements. LTSA exploits the local tangent information as a representation of the local geometry, and this local tangent information is then aligned to provide a global coordinate. HLLE obtains the final low-dimensional representations by applying eigen-analysis to a matrix which is built by estimating the Hessian over neighbourhood. All of these algorithms suffer from the out-of-sample problem. One common response to this problem is to apply a linearization procedure to construct explicit maps over new measurements. Examples of this approach include Locality Preserving Projections (LPP) [108], a linearization of LE; neighborhood preserving embedding (NPE) [107], a linearization of LLE; orthogonal neighborhood preserving projections (ONPP) [142], a linearization of LLE with the orthogonal constraint over the projection matrix; and linear local tangent space alignment (LLTSA) [351], a lin-

earization of LTSA. The above analysis shows that all the aforementioned algorithms are designed according to specific intuitions, and solutions are given by optimizing intuitive and pragmatic objectives. That is, these algorithms have been developed based on the experience and knowledge of field experts for their own purposes. As a result, the common properties and intrinsic differences of these algorithms are not completely clear. The framework of "Patch Alignment" [350, 99] is therefore proposed for better understanding the common properties and intrinsic differences in algorithms. This framework consists of two stages: part optimization and whole alignment. The details of this framework will be introduced in Chapter 2.

1.2.2 Semi-supervised Learning

In computer animation, the interaction between computer and artist has already been demonstrated to be an efficient way [120]. Some researchers have explored the close relationship between SSL and computer animation; for example, in character animation, Ikemoto [120] has found what the artist would like for given inputs. Using these observations as training data, the input output mapping functions can be fitted to generalize the training data to novel input. The artist can provide feedback by editing the output. The system uses this feedback to refine its mapping function, and this iterative process continues until the artist is satisfied. This framework has been applied to address important character animation problems. First, sliding foot plants are a common artifact resulting from almost any attempt to modify character motion. Using an artist-trained oracle, it can be demonstrated that the animation sequences can be accurately annotated with foot plant markers. Second, all motion synthesis algorithms sometimes produce character animation that looks unnatural (i.e., contains artifacts or is otherwise distinguishable as synthesized motion by a human observer). It can be demonstrated that these methods can be successfully adopted as the testing parts of hypothesize-and-test motion synthesis algorithms. Furthermore, artists can create compelling character animations by manipulating the details of a character's motion. This process is labor-intensive and repetitive. It has been found that the character animation can be made efficient by generalizing the edits an animator makes on short sequences of training data to other sequences. Another typical example is in cartoon animation. Correspondence construction of objects in keyframes is the pre-condition for inbetweening and coloring in cartoon animation production. Since each frame of an animation consists of multiple layers, objects are complex in terms of shape and structure; therefore, existing shape-matching algorithms, specifically designed for simple structures such as a single closed contour, cannot perform well on objects constructed by multiple contours with an open shape. Yu et al. [338] proposed a semi-supervised learning method for complex object correspondence construction. In particular, the new method constructs local patches for each point on an object and aligns these patches in a new feature space, in which correspondences between objects can be detected by the subsequent clustering. For local patch construction, pairwise constraints, which indicate the corresponding points (must link) or unfitting points (cannot link), are introduced by users to improve the performance of the correspondence construction. This kind of input is conveniently available to animation

software users via user friendly interfaces. Furthermore, the feedback [280, 285] provided by users can enhance the results of correspondence construction.

Based on the above analysis, semi-supervised learning (SSL) [365, 303, 304, 305, 307] could be an appropriate technique for designing novel tools for computer animation. SSL is halfway between supervised and unsupervised learning [306]. Based on the unlabeled data, the algorithm is enhanced by supervision information. Generally, this information will be the targets associated with some of the examples. In this case, the data set $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ will be divided into two parts: The data points $\mathbf{X}_l = [\mathbf{x}_1, \ldots, \mathbf{x}_l]$ with labels $\mathbf{Y}_l = [\mathbf{y}_1, \ldots, \mathbf{y}_l]$ are provided, and the data points $\mathbf{X}_u = [\mathbf{x}_{l+1}, \ldots, \mathbf{x}_{l+u}]$ whose labels are not known. Other forms of partial supervision are possible. For example, there may be constraints such as "these points have (or do not have) the same target." The different setting corresponds to a different view of semi-supervised learning, in which SSL is perceived as unsupervised learning guided by constraints. By contrast, in most other approaches, SSL is assumed to be supervised learning with additional information on the distribution of the data samples. A problem related to SSL is transductive learning [296], which was introduced by Vapnik. In this problem, there is a (labeled) training set and an (unlabeled) test set. The idea of transduction is to perform predictions only for the test points. This is in contrast to inductive learning, where the goal is to output a prediction function which is defined on the entire space. Next, we will introduce several semi-supervised learning methods in detail.

Self-training is a commonly used technique for semi-supervised learning. In self-training, a classifier is first trained with a small amount of labeled data. The classifier is then used to classify the unlabeled data. Typically, the most confident unlabeled points, together with their predicted labels, are added to the training set. Then, the classifier is re-trained and the procedure repeated. This procedure is also called self-teaching or bootstrapping. Yarowsky [332] adopts self-training for word sense disambiguation. Riloff et al. [236] use it to identify subjective nouns. Maeireizo et al. [188] classify dialogues as 'emotional' or 'non-emotional' with a procedure involving two classifiers.

In co-training, several assumptions are made: (1) Features can be classified into two sets; (2) each subfeature set is sufficient to train a good classifier; (3) the two sets are conditionally independent given the class. Initially, two separate classifiers are trained with the labeled data on the two sub-feature sets respectively. Each classifier then classifies the unlabeled data and 'teaches' the other classifier with the few unlabeled examples (and the predicted labels) about which they feel most confident. Each classifier is retrained with the additional training examples given by the other classifier, and the process repeats. Nigam and Ghani [213] perform extensive empirical experiments to compare co-training with generative mixture models and EM. The result shows that co-training gives excellent performance if the conditional independence assumption indeed holds. Jones [128] also used co-training, co-EM and other related methods for information extraction from text. Balcan and Blum [27] show that co-training can be quite effective in an extreme case where only one labeled point is needed to learn the classifier. Zhou et al. [363] give a co-training

algorithm using Canonical Correlation Analysis, which also needs only one labeled point.

In graph-based semi-supervised learning, the nodes in the graph are labeled and unlabeled samples in the dataset, and the edges in the graph represent the similarity of samples. These methods usually assume label smoothness over the graph. In general, graph-based methods can be seen as estimation of a function $\mathbf{f}$ on the graph. Two issues of $\mathbf{f}$ should be satisfied: it should be close to $\mathbf{y}_l$ on the labeled data and it should be smooth on the whole graph. Thus, these two issues can be represented in a regularization framework where the first term is a loss function and the second term is a regularizer. It is more important to construct a good graph than to choose among the methods. Blum and Chawla [45] adopt the graph mincut problem to solve semi-supervised learning. In this method, the positive labels act as sources and the negative labels act as sinks. The objective is to find a minimum set of edges whose removal blocks all flow from the sources to the sinks. The nodes connecting to the sources are then labeled positive, and those to the sinks are labeled negative. The Gaussian random fields and harmonic function method [366] is a continuous relaxation to the difficulty of discrete Markov random fields. It can be viewed as having a quadratic loss function with infinity weight, so that the labeled data are clamped, and a regularizer based on the graph combinatorial Laplacian $\triangle$ can be

$$\sum_{i \in L} (\mathbf{f}_i - \mathbf{y}_i)^2 + \frac{1}{2} \sum_{i,j} \mathbf{w} (\mathbf{f}_i - \mathbf{f}_j)^2 = \sum_{i \in L} (\mathbf{f}_i - \mathbf{y}_i)^2 + \mathbf{f}^{\mathrm{T}} \Delta \mathbf{f}, \tag{1.1}$$

where $\mathbf{f}_i \in \mathbf{R}$ is the key relaxation to Mincut. Zhou et al. [361] proposed the local and global consistency method where the loss function is $\Sigma_{i=1}^{n} (\mathbf{f}_i - \mathbf{y}_i)^2$, and the normalized Laplacian $\mathbf{D}^{-1/2} \triangle \mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2}$ in the regularizer as

$$1/2 \sum_{i,j} \mathbf{w}_{i,j} \left(\mathbf{f}_i / \sqrt{\mathbf{D}_{ii}} - \mathbf{f}_j / \sqrt{\mathbf{D}_{jj}} \right)^2 = \mathbf{f}^{\mathrm{T}} \mathbf{D}^{-1/2} \Delta \mathbf{D}^{-1/2} \mathbf{f} \tag{1.2}$$

The manifold regularization framework [32] employs two regularization terms

$$\frac{1}{l} \sum_{i=1}^{l} V (\mathbf{x}_i, \mathbf{y}_i, \mathbf{f}) + \lambda \|\mathbf{f}\|_K^2 + \beta \|\mathbf{f}\|_I^2 \tag{1.3}$$

where V is an arbitrary loss function, K is a "base kernel", e.g., a linear or RBF kernel. I is a regularization term induced by the labeled and unlabeled data.

In general, there are no explicit rules for choosing hyperparameters for graph-based semi-supervised learning, because it is nontrivial to define an objective function to obtain these hyperparameters. Usually, cross-validation is utilized for parameter selection. However, this grid-search technique tries to select parameters from discrete states in the parameter space, and lacks the ability to approximate the optimal solution. To deal with the parameter configuration problem, an ensemble manifold regularization (EMR) framework [87, 86] is proposed to combine the automatic intrinsic manifold approximation and the semi-supervised classifier learning. By providing a

series of initial guesses of graph Laplacian, the framework learns to combine them to approximate the intrinsic manifold.

Besides, Yu et al. [341] adopt hypergraph [362] to address the problem of parameters selection in graph-based semi-supervised learning. Unlike a graph that has an edge between two vertices, a set of vertices is connected by a hyperedge in a hypergraph. Each hyperedge is assigned a weight. In hypergraph learning, the weights of the hyperedges are empirically set according to certain rules. In practice, a large number of hyperedges will usually be generated, and these hyperedges have different effects. In Ref.[341], an adaptive hypergraph learning method is proposed. For hypergraph construction, the hyperedges are generated based on data samples and their nearest neighbors. Specifically, the size of the neighborhood is varied to produce multiple hyperedges for each sample. Thus, a large set of hyperedges will be generated in this process. This makes the approach much more robust, because the neighborhood size dose not need to be tuned. Since it is difficult to choose a suitable strategy to heuristically weight hyperedges, a principled approach, termed regularized loss minimization, is considered based on statistical learning theory. The loss minimization ensures that the learned weights are optimal for the training set. Since the size of the training set is not large in practice, the regularization item ensures that the learned weights do not overfit to the training samples. This scheme essentially achieves improved classification performance compared to the scheme that uses fixed weights for different hyperedges.

1.2.3 Multiview Learning

Multiview learning has a long history [242]. It has been applied to semi-supervised regression [259, 50, 343] and the more challenging structured output spaces [51]. In computer animation, it is normal to use multiple features from different views to describe the character. For instance, to describe a character well in cartoons, it is elementary to extract a set of visual features to define its color, texture, motion and shape information. A set of vectors can be obtained in different spaces to represent the character. In this case, one possible method is to concatenate these vectors as a new vector. This concatenation is not physically meaningful because each feature has a specific statistical property.

Long et al. [181] proposed a distributed spectral embedding (DSE) to construct a subspace learning with multiple views. To give a multiview datum with n objects having m views, i.e., a set of matrices $\mathbf{X} = \{\mathbf{X}^{(i)} \in \mathbf{R}^{m_i \times n}\}$, each representation $\mathbf{X}^{(i)}$ is a feature matrix from view i. DSE assumes that the low-dimensional representation of each view $\mathbf{X}^{(i)}$ is already known, i.e., $\mathbf{A} = \{\mathbf{A}^{(i)} \in \mathbf{R}^{n \times k_i}\}_{i=1}^{m}, k_i < m_i (1 \leq i \leq m)$. DSE aims to learn a consensus low-dimensional subspace $\mathbf{B} \in \mathbf{R}^{n \times k}$ based on A; the objective function of DSE can be formulated as

$$\mathbf{min}_{\mathbf{B},\mathbf{P}} \sum_{i=1}^{m} \left\| \mathbf{A}^{(i)} - \mathbf{B}\mathbf{P}^{(i)} \right\|^2 \text{s.t. } \mathbf{B}^{\mathbf{T}}\mathbf{B} = \mathbf{I}, \tag{1.4}$$

where $\mathbf{P} = \{\mathbf{P}^{(i)} \in \mathbf{R}^{k \times k_i}\}_{i=1}^{m}$ is a set of mapping matrices. The optimal solution to DSE is given by performing eigendecomposition of the matrix $\mathbf{CC^T}$, $\mathbf{C} = [\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(m)}]$. Furthermore, a novel multiview subspace learning method [324] is proposed based on the patch alignment framework. The details are presented in Chapter 2.

1.2.4 Learning-based Optimization

In this section, we present an overview of mathematical optimization, which will be used in the subsequent contents of this book.

The form of a mathematical optimization problem can be written as

$$\mathbf{min}\ f_0(\mathbf{x}),\ \text{s.t.}\ f_i(\mathbf{x}) \leq b_i,\ \ i = 1, \ldots, m, \tag{1.5}$$

where the vector $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is the optimization variable for the problem. The function $f_0 : \mathbf{R}^n \rightarrow \mathbf{R}$ is the objective function, and the function $f_i : \mathbf{R}^n \rightarrow \mathbf{R}, i = 1, \ldots, m$, are the constraint functions. The constants $b_1, \ldots, b_m$ are the bounds for the constraints. A vector $\mathbf{x}^{'}$ is called a solution of the problem in Eq. (1.5), if it has the smallest objective value among all vectors which satisfy the constraints. In general, the optimization problems can be characterized by particular forms of the objective and constraint functions. Next, we present the optimization methods of least-squares, linear programming and convex optimization.

Least-Squares Problems: The least-squares problem can be formulated as an optimization problem with no constraints and the objective function is formulated by summing the squares of terms of the form $\mathbf{a}_i^{\mathbf{T}}\mathbf{x} - \mathbf{b}_i$:

$$\mathbf{min}\ f_0(\mathbf{x}) = \|\mathbf{Ax} - \mathbf{b}\|_2^2 = \sum_{i=1}^{k}(\mathbf{a}_i^{\mathbf{T}} - \mathbf{b}_i)^2, \tag{1.6}$$

where $\mathbf{A} \in \mathbf{R}^{k \times n}$ (with $k \geq n$), $\mathbf{a}_i^{\mathbf{T}}$ is the rows of $\mathbf{A}$, and the vector $\mathbf{x} \in \mathbf{R}^n$ is the optimization variable. The solution of the least-squares problem, Eq. (1.6), can be reduced to solving a set of linear equations

$$(\mathbf{A^T A}) = \mathbf{A^T b}. \tag{1.7}$$

Thus, the analytical solution is $\mathbf{x} = (\mathbf{A^T A})^{-1}\mathbf{A^T b}$. The least-squares problem can be solved in a time approximately proportional to n^2k with a known constant. In many cases, the least-squares problem can be solved by exploiting some special structure in the coefficient matrix $\mathbf{A}$. For instance, by investigating sparsity, we can usually solve the least-squares problem much faster than order n^2k.

The least-squares problem is the basis for regression analysis, optimal control, and many parameter estimation and data fitting methods. It has a number of statistical interpretations, e.g., as the maximum likelihood estimation of a vector $\mathbf{x}$. It is straightforward to recognize the optimization problem as a least-squares problem. We only need to verify whether the objective function is a quadratic function. Recently,

several standard techniques have been used to increase the flexibility of least-squares in applications. For example, the weighted least-squares cost can be formulated as

$$\sum_{i=1}^{k} \mathbf{w}_i(\mathbf{a}_i^{\mathbf{T}}\mathbf{x} - \mathbf{b}_i)^2, \tag{1.8}$$

where $\mathbf{w}_1, \ldots, \mathbf{w}_k$ are positive. In a statistical setting, weighted least-squares arise in estimation of a vector $\mathbf{x}$, given linear measurements corrupted by errors with unequal variances.

Another technique in least-squares is regularization, in which extra terms are added to the cost function. In the simplest case, a positive multiple of the sum of squares of the variables is added to the cost function:

$$\sum_{i=1}^{k} \mathbf{w}_i(\mathbf{a}_i^{\mathbf{T}}\mathbf{x} - \mathbf{b}_i)^2 + \alpha \sum_{i=1}^{n} \mathbf{x}_i^2, \tag{1.9}$$

where $\alpha > 0$. Here, the extra terms penalize large values of $\mathbf{x}$, and result in a sensible solution in cases when minimizing the first sum only does not. The parameter α is selected by the user to provide the right trade-off.

Linear Programming: Another important group of optimization methods is linear programming, in which the objective function and all constraints are linear.

$$\mathbf{min}\ \mathbf{c}^{\mathbf{T}}\mathbf{x}\ \mathbf{s.t.}\ \mathbf{a}_i^{\mathbf{T}}\mathbf{x} \leq \mathbf{b}_i, i = 1, \ldots, m, \tag{1.10}$$

where the vectors $\mathbf{c}, \mathbf{a}_1, \ldots, \mathbf{a}_m \in \mathbf{R}^n$ and scalars $\mathbf{b}_1, \ldots, \mathbf{b}_m \in \mathbf{R}$ are problem parameters which specify the objective and constraint functions.

There is no simple analytical formula for the solution of a linear program. However, there is a variety of very effective methods for solving them, including Dantzig's simplex method [235]. We can easily solve problems with hundreds of variables and thousands of constraints on a small desktop computer. If the problem is sparse, or has some other exploitable structure, we can often solve problems with tens or hundreds of thousands of variables and constraints, as shown in Ref. [364].

Convex Optimization: The convex optimization problem [267, 127] can be formulated as

$$\mathbf{min}\ f_0(x)\ \mathbf{s.t.}\ f_i(\mathbf{x}) \leq \mathbf{b}_i,\ i = 1, \ldots, m, \tag{1.11}$$

where the functions $f_0, \ldots, f_m : \mathbf{R}^n \to \mathbf{R}$ are convex, and satisfy

$$f_i(\alpha\mathbf{x} + \beta\mathbf{y}) \leq \alpha f_i(\mathbf{x}) + \beta f_i(\mathbf{y}), \tag{1.12}$$

for all $\mathbf{x}, \mathbf{y} \in \mathbf{R}^n$ and all $\alpha, \beta \in \mathbf{R}$ with $\alpha + \beta = 1, \alpha \geq 0, \beta \geq 0$.

The use of convex optimization is very much like using least-squares or linear programming. If a problem can be described as a convex optimization problem, it can be solved efficiently, like solving the least-squares problem, but there are also some important differences. The recognition of a least-squares problem is straightforward, but recognizing a convex function is more difficult. Additionally, there are many more tricks for transforming convex problems than for transforming linear programs.

1.3 RECENT DEVELOPMENTS IN COMPUTER ANIMATION

In this section, we introduce the application of machine learning techniques into computer animation and its related fields. It can be seen that animation has thus far benefited considerably from the application of machine learning [22]. Excellent introductions to the field of computer animation are presented by Parent [218] and Pina et al. [231]. Computer animation is a time-intensive process, and this is true for both the 3D virtual character and 2D cartoon character. As a result, there has long been interest in partially automating animation [218, 231]. Some well-known approaches include keyframing and inverse kinematics, but these techniques still require significant animator input. In the following content, we will present some efficient and effective techniques in automatic animation generation.

1.3.1 Example-Based Motion Reuse

An efficient approach to motion synthesis is to create novel motions using some example motion data, as shown in Figure 1.1 and Figure 1.2. Intuitively, example-based motion synthesis can more easily produce realistic motion than entirely artificial approaches.

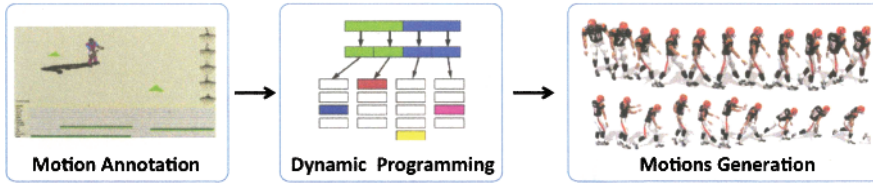

Figure 1.1 Example-based motion synthesis guided by annotations.

1.3.1.1 Introduction of Motion Capture Systems The prerequisite of example-based motion synthesis is obtaining the motion data using specified techniques. These techniques were first proposed in the 1970s. In general, the motion capture can be achieved by adopting specific equipment to record the 3D motion data in real time. In the 1990s, a number of typical motion capture systems have been proposed, the details of which are presented in Table 1.1.

Motion capture systems to date can be classified into the following categories:

- Electronic mechanical system: The advantage of this system is the unconstrained motion capture place, and the avoidance of self-occlusion in the motion capture procedure. The Gypsy5 system from the Animazoo Company [2] can be bought in the market (as shown in Figure 1.3(a)).
- Electromagnetic system: This system adopts multiple sensors to record the joint positions and angles of real objects (as shown in Figure 1.3(b)). The

Figure 1.2 Slide show of example-based motion synthesis. The character transitions from walking to jogging to looking over her shoulder. Example-based techniques have proven quite effective because they synthesize motion that complies with real example motion.

Table 1.1 Motion Capture Systems

Date	Name of Systems
1980-1983	Goniometers [58]
1982-1983	Graphical marionette [88]
1988	Mike the Talking Head [198]
1988	Waldo C. Graphic [300, 301]
1989	Dozo [139]
1991	Mat the Ghost [286]
1992	Mario [238]

deficiencies can be summarized as: First, this system is sensitive to metal objects in a motion capture scene; second, because the sensors are fastened to the actors, some complicated actions cannot be performed; third, the low sampling rates of this system cannot meet the requirements of sports analysis, besides which the cost of the system is very high.

- Optical system: This kind of system has been widely adopted in motion capture. Representative products are Hawk, Eagle [3] from MotionAnalysis and Vicon MX system from the Vicon Company [4]. These systems record motion by sticking markers onto the human body's joints. A group of specialized cameras are adopted to take videos and human motions are then extracted from the videos. In general, this system requires 4-6 cameras. This technique can provide high sampling rates, which can record the motion data for tracking and

3D modeling reconstruction in computer vision. The optical systems are also very expensive.

Figure 1.3 (a) Electronic mechanical motion capture system (b) Electromagnetic motion capture system.

Based on the above elaborations, the available motion capture systems have the disadvantages of high price, complicated operating procedure, and more. In recent years, therefore, scientists and engineers have been exploring novel motion capture devices which are cheap and easy to operate. The video-based motion capture system is an emergent topic, which we will discuss below.

In terms of motion capture devices, video-based motion capture systems can be divided into the single camera system and multiple camera system. The techniques of feature tracking, modeling matching, statistical learning, and silhouette analysis are widely used in video-based motion capture. The following paragraphs offer more detail on these techniques.

Feature Tracking-Based Motion Capture Techniques: Most video-based motion capture techniques are based on a single camera which takes a series of frames for human characters and adopts feature tracking methods to extract the motions. Here, the adopted features are feature points and motion patches. Segen and Pingali [248] proposed a contour-based motion capture system. They adopt the points' position and curvature to match two successive frames' feature points and to estimate the motion parameters. Bregler and Malik [54] proposed a novel vision-based motion capture technique which can recover human motion from single-view video frames with complicated backgrounds. Pekelny and Gotsman [226] proposed a novel motion capture system which acquires the motion of a dynamic piecewise-rigid object using a single depth video camera. This system identifies and tracks the rigid components in each frame while accumulating the geometric information acquired over time, possibly from different viewpoints. The algorithm also reconstructs the dynamic

skeleton of the object and thus can be used for markerless motion capture. Fossati et al. [84] proposed a motion capture approach which combines detection and tracking techniques to achieve robust 3D motion recovery of people seen from arbitrary viewpoints by a single and potentially moving camera. This approach depends on detecting key postures and can be done reliably, using a motion model to infer 3D poses between consecutive detections, finally refining them over the whole sequence using a generative model. Wang et al. [312] proposed a motion capture system built from commodity components, as shown in Figure 1.4. This approach uses one or more webcams and a color shirt to track the upper-body at interactive rates. In their system, a robust color calibration procedure is described to enable color-based tracking to work against cluttered backgrounds and under multiple illuminants.

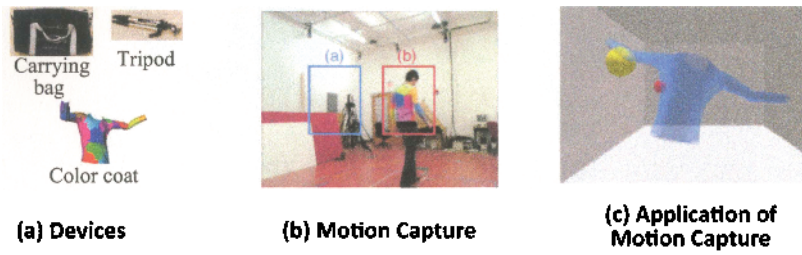

Figure 1.4 (a) A lightweight color-based motion capture system that uses one or two commodity webcams and a color shirt (b) Performance capture in a variety of natural lighting environments, such as the squash court (c) Application of motion capture data in video games.

Model Matching and Silhouette Analysis-based motion capture techniques: In the field of motion capture research, some scientists adopt model matching and silhouette analysis to recover the human motion parameters from the images/videos. The basic idea is to fix a search space for human gesture [281] and build the correspondence between the extracted image features (feature points, silhouette) and the 3D human model. Normally, this procedure is conducted by solving an optimization problem. The details of this procedure are shown in Figure 1.5.

Zhao and Li [354] adopted Genetic Algorithm to match the feature points from the 2D image with an optimal 3D gesture retrieved from 3D human gesture space. However, the matching accuracy of this method depends heavily on the tracking accuracy of the 2D feature points. Thus, satisfactory results cannot be achieved in videos with complicated backgrounds and self-occlusions. Chen et al. [68] proposed a novel method to recover 3D human motion. First, the 3D human gestures are parameterized and projected onto a 2D plane by using different configurations. Then, the silhouette of the character in the video frame is input into the system and its corresponding optimal 3D human gesture can be retrieved. The major deficiency of this approach is that the adopted 3D human model is general and simple, thus it cannot be perfectly matched to the silhouetted extracted from the videos. Micilotta et al. [196] proposed an upper body motion capture system which extracts motion by building correspondence between silhouette and Edge Map; however, this ap-

proach can only capture upper body motion. Agarwal and Triggs [10] presented a learning-based method for recovering a 3D human body pose from single images and monocular image series. This approach does not require an explicit body model and prior labeling of the body parts in the image. Instead, it recovers the pose by direct nonlinear regression against shape descriptor vectors extracted automatically from image silhouettes. For robustness against local silhouette segmentation errors, the silhouette shape is encoded by a histogram-of-shape-contexts descriptors. To handle the problems of depth and labeling information loss, this method is integrated into a novel regressive tracking framework, using dynamics from the previous state estimate together with a learned regression value to disambiguate the pose. Recently, Aguiar et al. [13] proposed a new marker-less approach to capturing human performances from multiview video (as shown in Figure 1.5). This algorithm can jointly reconstruct spatio-temporally coherent geometry, motion and textural surface appearance of actors that perform complex and rapid moves. This algorithm is purely mesh-based and makes as few prior assumptions as possible about the type of subject being tracked; it can even capture performances of people wearing wide apparel, such as a dancer wearing a skirt. In Figure 1.5(c) small-scale time-varying shape detail is recovered by applying model-guided multiview stereo to refine the model surface. Xu et al. [326] presented a method to synthesize plausible video sequences of humans according to user-defined body motions and viewpoints. First, they captured a small database of multiview video sequences of an actor performing various basic motions. They applied a marker-less model-based performance capture approach to the entire database to obtain the pose and geometry of the actor in each database frame. To create novel video sequences of the actor from the database, a user animates a 3D human skeleton with novel motion and viewpoints. This technique then synthesizes a realistic video sequence of the actor performing the specified motion, based only on the initial database. Stoll et al. [269] presented a motion capture approach that incorporates a physically based cloth model to reconstruct the real person in loose apparel from multiview video recordings. This algorithm requires little manual interaction. Without the use of optical markers, this algorithm can reconstruct the skeleton motion and detailed time-varying surface geometry of a real person from a reference video sequence. Wei and Chai [316] provided a novel method to capture physically realistic human motion from monocular video sequences. This approach first computes camera parameters, human skeletal size, and a small number of 3D key poses from video and then uses 2D image measurements at intermediate frames to automatically calculate the "in between" poses. During reconstruction, Newtonian physics, contact constraints, and 2D image measurements are leveraged to simultaneously reconstruct full-body poses, joint torques, and contact forces. Li et al. [171] presented a framework and algorithms for robust geometry and motion reconstruction of complex deforming shapes. This method makes use of a smooth template that provides a crude approximation of the scanned object and serves as a geometric and topological prior for reconstruction. Large-scale motion of the acquired object is recovered using a space-time adaptive, nonrigid registration method. Fine-scale details such as wrinkles and folds can be successfully synthesized with an efficient linear mesh deformation algorithm.

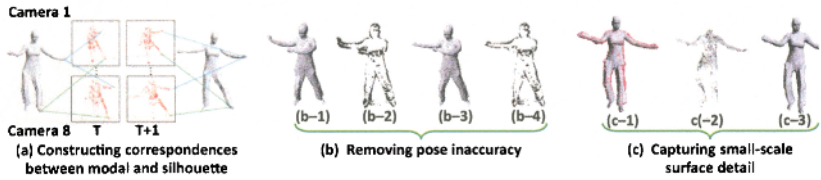

Figure 1.5 (a) 3D correspondences are extracted from corresponding SIFT features in respective input camera views at t and t+1. These 3D correspondences, two of them illustrated by lines, are used to deform the model into a first pose estimate for t+1. (b) Model (b-1) and the silhouette overlap (b-2) after the refinement step; slight pose inaccuracies in the leg and the arms appear black in the silhouette overlap image. (b-3) and (b-4) show that after key vertex optimization, these pose inaccuracies are removed and the model strikes a correct pose. (c) First, deformation constraints from the silhouette contours, shown as red arrows, are estimated in (c-1). Additional deformation handles are extracted from a 3D point cloud that is computed via model-guided multi-view stereo in (c-2). Together, both sets of constraints deform the surface scan to the highly accurate pose shown in (c-3).

Other Motion Capture Techniques: In general, motion capture techniques require the task to be performed in a specified lab or closed stage setting with controlled lighting. Thus, the capture of motions is constrained by the outdoor setting or the traversal of large areas. Recently, T. Shiratori et al. [254] adopted body-mounted cameras to reconstruct the motion of a subject, as shown in Figure 1.6. In this system, outward-looking cameras are attached to the limbs of the subject, and the joint angles and root pose are estimated through nonlinear optimization. The optimization objective function incorporates terms for image matching error and temporal continuity of motion. Structure-from-motion is used to estimate the skeleton structure and to provide initialization for the non-linear optimization procedure. Global motion is estimated and drift is controlled by matching the captured set of videos to reference imagery. For nonrigid object motion capture, Yan and Pollefeys [328] proposed an approach which is based on the modeling of the articulated non-rigid motion as a set of intersecting motion subspaces. Here, a motion subspace is the linear subspace of the trajectories of an object. It can model a rigid or non-rigid motion. The intersection of two motion subspaces of linked parts models the motion of an articulated joint or axis. Bradley et al. [47] described a marker-free approach to capturing garment motion that avoids these downsides. Temporally coherent parameterizations are established between incomplete geometries that are extracted at each timestep with a multiview stereo algorithm. Holes in the geometry are then filled, using a template. This method allows users to capture the geometry and motion of unpatterned off-the-shelf garments made from a range of different fabrics.

In this section, we present various kinds of motion capture systems capable of capturing realistic motions. In the next section, we shall introduce the reuse of motion data in the computer animation field. In general, motion reuse can be separated into three categories: motion retargeting, motion editing and motion synthesis.

Figure 1.6 A novel motion capture system which captures both relative and global motion in natural environments using cameras mounted on the body.

1.3.1.2 Motion Retargeting Motion retargeting can be defined as transferring the captured motion data onto a new character with a different skeleton. This procedure is presented in Figure 1.7.

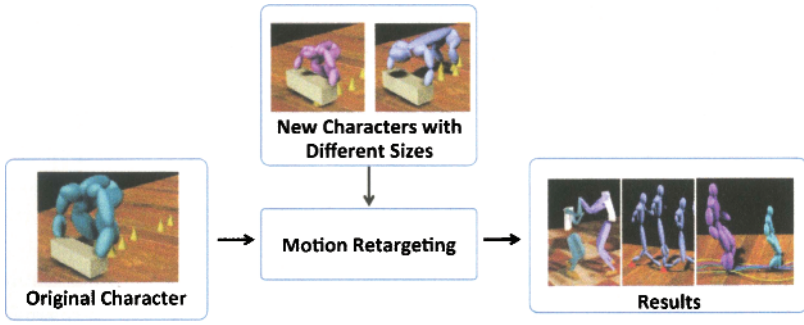

Figure 1.7 Motion of picking up an object is retargeted onto differently sized characters.

In 1997, Hodings and Pollard [113] proposed an automatic method which retargeted existing human motion onto a new character whose skeleton had different length and mass. The presented algorithm adapts the control system to a new character in two stages. First, the control system parameters are scaled based on the sizes, masses, and moments of inertia of the new and original characters. A subset of the parameters is then fine-tuned using a search process based on simulated annealing. In addition to adapting a control system for a new model, this approach can also be used to adapt the control system in an on-line fashion to produce a physically realistic metamorphosis from the original to the new model while the morphing character is performing the behavior. Similarly, Gleicher [91] proposed a constraint-based motion retargeting algorithm. Their focus is on adapting the motion of one articulated figure to another figure having identical structure but different segment lengths. This approach creates adaptations that preserve the desirable qualities of the original motion. In addition, specific features of the motion as constraints are identified and preserved in the retargeting procedure. A time constraints solver computes an adapted motion that reestablishes these constraints while preserving the frequency characteristics of the original signal. To solve the motion retargeting problem be-

tween characters that have different structures, Park and Shin [220] proposed a novel example-based motion retargeting approach. Provided with a set of example motions, this method automatically extracts a small number of representative postures called source key-postures. The animator then creates the corresponding key-postures of the target character, breathing his/her imagination and creativity into the output animation. Exploiting this correspondence, each input posture is cloned frame by frame to the target character to produce an initial animation, which is further adjusted in space and time for retargeting and time warping and then finalized with interactive fine tuning. Hecker et al. [110] provided a real-time motion retargeting system for animating characters whose morphologies are unknown at the time the animation is created. The proposed technique allows describing the motion using familiar posing and key-framing methods. The system records the data in a morphology-independent form, preserving both the animation's structural relationships and its stylistic information. At runtime, the generalized data are applied to specific characters to yield pose goals that are supplied to a robust and efficient inverse kinematics solver.

More complicated motion retargeting tasks have recently been achieved in the research field. For example, Ho et al. [112] presented a novel motion retargeting method which reused the motion involving close interactions between body parts of single or multiple articulated characters, such as dancing, wrestling, and sword fighting, or between characters and a restricted environment, such as getting into a car. In such motions, the implicit spatial relationships between body parts/objects are important for capturing the scene semantics. A simple structure called an interaction mesh is adopted to represent the spatial relationships. The interaction mesh representation is general and applicable to various kinds of close interactions. Yamane et al. [327] proposed an approach for producing animations of nonhumanoid characters from human motion capture data. Characters considered in this work (as shown in Figure 1.8) have proportion and/or topology significantly different from humans, but are expected to convey expressions and emotions through body language that are understandable to human viewers. Keyframing is most commonly used to animate such characters. This method provides an alternative for animating nonhumanoid characters that leverages motion data from a human subject performing in the style of the target character. The method consists of a statistical mapping function learned from a small set of corresponding key poses, and a physics-based optimization process to improve the physical realism.

1.3.1.3 Motion Editing Based on the existing motion capture data, the techniques of motion editing allow users to modify the features of the motion to meet their new requirements. In general, the input of the motion editing system is in single motion clips. In the following section, we shall review these techniques. **Motion Signal Processing:** In 1995, Bruderlin and Williams [55] successfully applied techniques from the image and signal processing domain to designing, modifying, and adapting animated motion. In their opinion, the multiresolution motion filtering, multitarget motion interpolation with dynamic timewarping, waveshaping and motion displacement mapping are well suited to the reuse and adaptation of existing motion data such as joint angles, joint coordinates or higher level motion parameters

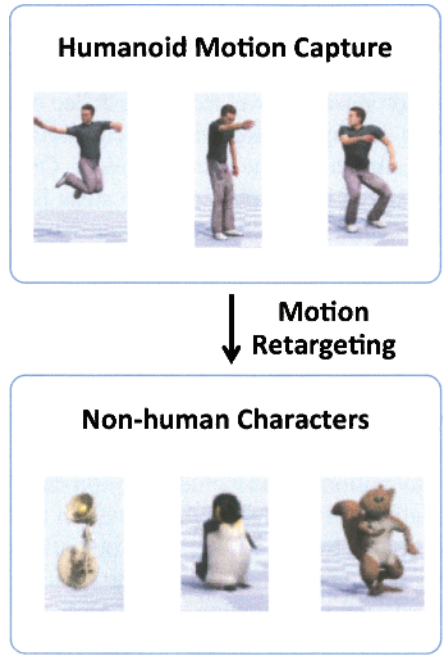

Figure 1.8 Nonhumanoid characters animated using human motion capture data.

of articulated figures with many degrees of freedom. Thus, existing motions can be modified and combined interactively and at a higher level of abstraction than supported by conventional systems. Unuma et al. [295] presented a method for modeling human figure locomotion. In this approach, Fourier expansions of experimental data of actual human behaviors serve as a basis from which the method can interpolate or extrapolate human locomotion. This means, for instance, that transition from a walk to a run is smoothly and realistically performed by the method. Moreover, an individual's character or mood appearing during the human behaviors is also extracted by the method. For example, the method gets "briskness" from the experimental data for a "normal" walk and a "brisk" walk. Then the "brisk" run is generated by the method, using another Fourier expansion of the measured data of running. The superimposition of these human behaviors is shown as an efficient technique for generating rich variations of human locomotion. In addition, step-length, speed, and hip position during locomotion are also modeled, and interactively controlled to get a desired animation.

Keyframe Editing and Motion Deformation: The basic idea of keyframe editing is to put constraints on the keyframes from original motion data. The gestures of these characters in keyframes are deformed according to the specification of users, and the inbetween frames are generated by using an interpolation algorithm. In using these techniques, the results of motion editing are affected by the keyframe selection.

If sufficient and appropriate keyframes are selected, smooth and natural motions can be generated. The motion editing technique proposed in [44] is based on the idea of keyframe editing. In real application, it might be not easy to edit the keyframes; thus, Witkin and Popovic [322] proposed a novel technique called motion warping. Unlike the technique of keyframe editing, motion warping conducts interpolation between the difference of the task motion and the source motion. One typical procedure of the motion warping system can be divided into two steps [221]: First, the inverse kinematics is adopted to achieve the constrained optimization of the keyframes, and interpolation is conducted to obtain the variance. In recent years, Sok et al. [260] proposed an integrated framework for interactive editing of the momentum and external forces in a motion capture sequence. This framework allows users to control the momentum and forces, which provides a powerful and intuitive editing tool for dynamic motions. Neff and Kim [209] proposed a system for editing motion data that is particularly well suited to making stylistic changes. This approach transforms the joint angle representation of animation data into a set of pose parameters more suitable for editing. These motion drives include position data for the wrists, ankles, and center of mass, as well as the rotation of the pelvis. In this system, an animator can interactively edit the motion by performing linear operations on the motion drives or extracted correlations, or by layering additional correlations.

Motion Path Editing: The motion capture data have a specified trajectory (motion path), and path modification is a necessary step for animation production. Michael Gleicher [92] introduced the motion path editing algorithm, which records the trajectory of the skeleton's joints as the character's motion path and adopts B-spline to describe it. After the user has interactively changed the formation of the path, the parameters of the original path curve are adopted to resample and recalculate the transition and rotation parameters for each frame. Thus, the spatial-time constraint of the original motion data can be preserved; meanwhile, the purpose of changing the path formation can be achieved. Recently, Lockwood and Singh [180] presented a system for interactive kinematic editing of motion paths and timing that employs various biomechanical observations to augment and restrict the edited motion. Realistic path manipulations are enforced by restricting user interaction to handles identified along a motion path using motion extrema. An as-rigid-as-possible deformation technique modified specifically for use on motion paths is used to deform the path to satisfy the user-manipulated handle positions. After all motion poses have been adjusted to satisfy the new path, an automatic time-warping step modifies the timing of the new motion to preserve the timing qualities of the original motion.

Spatial-Time-Constraint-Based Motion Editing: The difference between spatial-time-constraint-based motion editing and other motion editing techniques is that this method does not handle isolated frames. Here, the spatial-time constraint means that a group of motion frames are simultaneously processed. In early research work, the spatial-time constraints were initially used to denote the positions of characters in the specified time. The optimal postures of the characters were then obtained in these positions. In the early research work [321, 71], the physical rules are used as constraints of the motion, and the objective function of changing the character's motion is constructed to take the consumption of energy into consideration. Finally,

the minimization of the objective function is conducted to obtain the new motion. In Ref. [93], Gleicher and Litwinowicz made improvements in spatial-time constraint-based motion editing by using a novel objection function, and they provided real-time interactive editing functions in this approach. In recent years, novel motion editing techniques have been proposed to achieve complex motion reuse. Complex acrobatic stunts, such as double or triple flips, can be performed only by highly skilled athletes. On the other hand, simpler tricks, such as single-flip jumps, are relatively easy to master. Majkowska and Faloutsos [189] presented a method for creating complex, multi-flip ballistic motions from simple, single-flip jumps. This approach allows an animator to interact with the system by introducing modifications to a ballistic phase of a motion. This method automatically adjusts motion trajectories to ensure the physical validity of the motion after the modifications. The presented technique is efficient and produces physically valid results without resorting to computationally expensive optimization. Hsu et al. [117] provided a time-warping technique which allows users to modify timing without affecting poses. This technique has many motion editing applications in animation systems, such as refining motions to meet new timing constraints or modifying the action of animated characters. The proposed approach simplifies the whole process by allowing time warps to be guided by a provided reference motion. Given few timing constraints, it computes a warp that both satisfies these constraints and maximizes local timing similarities to the reference. Compared with existing methods, this algorithm is fast enough to incorporate into standard animation workflows. Similar ideas of using time constraints in motion editing can be found in Ref. [168, 121, 205].

1.3.1.4 Motion synthesis The task of motion synthesis is to combine multiple motion clips into novel motion. It is very challenging because it is not obvious how many data may be generalized. For example, an unrealistic animation may be generated by blending two motion clips. In this case, researchers have tried to apply machine learning to make example-based motion synthesis possible.

Motion Transition and Motion Blending: Motion transition seamlessly integrates two motion clips into one long motion. The typical technique for motion transition is verbs and adverbs [240]. Based on machine learning theory, this technique is proposed to create a generative model of motion through regression. The motion clips representing a class of motion (e.g., walking, waving, etc.) are collected by users. These clips are then manually placed in an n-dimensional parameter space determined by the user, and the technique of scattered data interpolation is adopted to synthesize the novel motions.

In such an approach of motion transition, the location of good transition points in the motion stream is critical. Wang and Bodenheimer [308] proposed a method to evaluate the cost function for determining such transition points. A set of optimal weights for the cost function is compared using a constrained least-squares technique. The weights are then evaluated in two ways: first, through a cross-validation study and second, through a medium-scale user study. The cross-validation shows that the optimized weights are robust and work for a wide variety of behaviors. The user

study demonstrates that the optimized weights select more appealing transition points than the original weights.

Motion blending can be defined as averaging multiple motion clips to obtain the novel motion data. The importance of motion blending is that it achieves time registration of the motion clips. The techniques of time warping [222, 223] have been commonly used. In motion transition, the technique of motion blending has been used. Menardais et al. [194] proposed a real-time motion blending algorithm, which achieves smooth motion blending by averaging multiple motions. Furthermore, Wang and Bodenheimer [309] adopted linear blending to determine visually appealing motion transitions.

Motion-Graph-based Motion Synthesis: Motion graph modeling, initially proposed in Ref.[145], consists both of pieces of original motion and automatically generated transitions. Motion can be generated simply by building walks on the graph. A general framework for extracting particular graph walks is presented to meet a user's specification. Details of motion graph are presented in Figure 1.9. In recent years, a group of novel motion synthesis methods have been proposed based on this technique. Gleicher et al. [94] applied this technique to virtual environments and games. In their research, an approach called Snap-Together Motion (STM) preprocesses a corpus of motion capture examples into a set of short clips which can be concatenated to make continuous streams of motion. The result process is a simple graph structure that facilitates efficient planning of character motions. A user-guided process selects "common" character poses and the system automatically synthesizes multiway transitions that connect through these poses. In this manner, well-connected graphs can be constructed to suit a particular application, allowing for practical interactive control without the effort of manually specifying all transitions. Li et al. [336] adopted a statistical technique called a linear dynamic system (LDS) to perform linear function approximation. A graph is constructed to model the transitions that can occur between the LDS's.

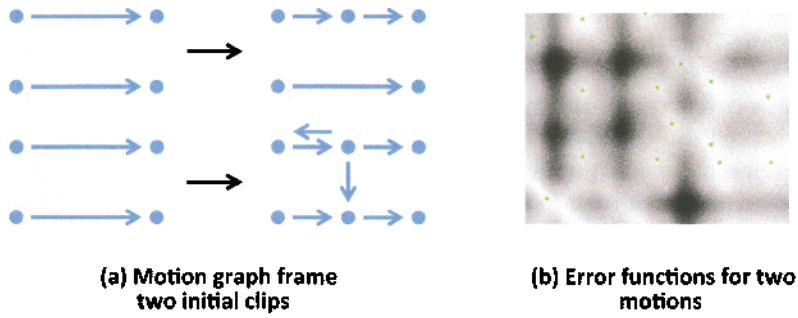

Figure 1.9 (a) Motion graph built frame with two initial clips. A node can be trivially inserted to divide an initial clip into two smaller clips. (b) An example error function for two motions. The entry at *(i; j)* contains the error for making a transition from the *i*th frame of the first motion to the *j*th frame of the second. White values correspond to lower errors and black values to higher errors. The dots represent local minima.

Motion-graph-based methods are popularly used to synthesize long motions by playing back a sequence of existing motion clips. However, motion graphs only support transitions between similar frames. In Ref.[234], an optimization-based graph was introduced to combine continuous constrained optimization with motion-graph-based motion synthesis. The constrained optimization is used to create a vast number of complex realistic-looking transitions in the graph. The graph can then be used to synthesize long motions with nontrivial transitions that for example allow the character to switch its behavior abruptly while retaining motion naturalness. Beaudoin et al. [30] presented a technique called motion-motif graph, which represents clusters of similar motions. Together with their encompassing motion graph, they lend understandable structure to the contents and connectivity of large motion datasets. They can be used in support of motion compression, the removal of redundant motions, and the creation of blend spaces. This research develops a string-based motif-finding algorithm which allows for a user-controlled compromise between motif length and the number of motions in a motif.

The above motion-graph-based approaches have shown great promise for novice users due to their ability to generate long motions and the fully automatic process of motion synthesis. The performance of motion-graph-based approaches, however, relies heavily on selecting a good set of motions to build the graph. The motion set needs to contain enough motions to achieve good connectivity and smooth transitions. At the same time, the motion set needs to be small enough for fast motion synthesis. Manually selecting a good motion set that achieves these requirements is difficult; hence, Zhao et al. [355] proposed an automatic approach to select a good motion set. Here, the motion selection problem is presented as a search for a minimum-size subgraph from a large motion graph representing the motion capture database. This approach especially benefits novice users who desire simple and fully automatic motion synthesis tools. To obtain better motion graphs, Reitsma and Pollard [217] described a method for using task-based metrics to evaluate the capability of a motion graph to create the set of animations required by a particular application. This capability is examined for typical motion graphs across a range of tasks and environments. The results of this approach can be used to evaluate the extent to which a motion graph will fulfill the requirements of a particular application, lessening the risk of the data structure performing poorly at an inopportune moment. The method can also be used to characterize the deficiencies of this technique whose performance will not be sufficient, as well as to evaluate the relative effectiveness of different options for improving those techniques. A similar idea has been applied in Ref.[356].

Subspace Learning-Based Motion Synthesis: The basis of subspace-learning-based motion synthesis is to project the original high-dimensional motion data into a simple low-dimensional subspace. Motion editing and synthesis are conducted in this new space, and the results are projected back to the high-dimensional space. It is a typical integration of machine learning techniques and computer animation. Alex and Muller [15] adopted principle component to represent the animation frames, while Mori and Hoshino [201] used the Independent Component Analysis (ICA) to obtain the sparse gesture subspace for realistic human motion synthesis. First, independent motion features are extracted from the original high-dimensional motion capture data

to construct the ICA subspace. Then, the new human motion data are generated through interpolation in the ICA subspace. Similarly, Glardon et al. [90, 89] adopted PCA to project the motion data into the low-dimensional space. The operations of interpolation/extrapolation are conducted in this new space to obtain new motions with different speed. It can also be reused in new characters with different skeletons. Safonova et al. [243] utilized manifold learning and regression to combine motor controller-based motion synthesis and data-driven synthesis (as shown in Figure 1.10). In this method, a small number of similar motion clips are connected by the user, and dimensionality reduction is then performed on the data using PCA. The synthesis is guided through optimization, seeking useful paths on the manifold. A similar idea can be found in [61].

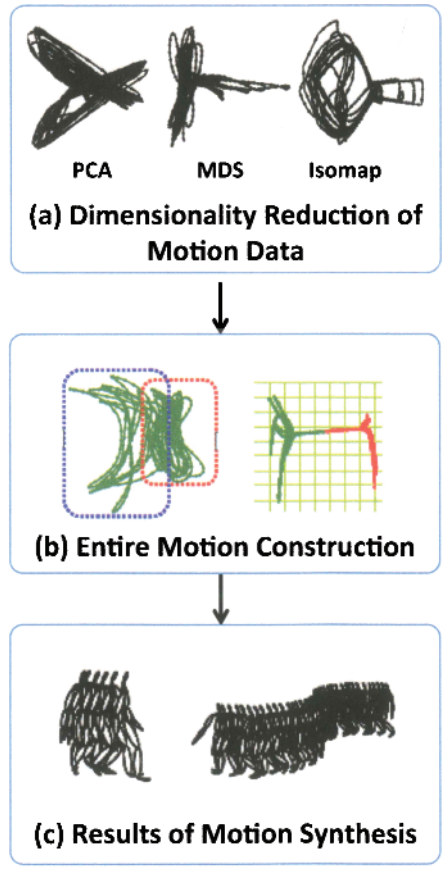

Figure 1.10 Projection of walking motion into low-dimensional space (a) Dimensionality reduction of motion data through PCA, MDS, and Isomap (b) Entire motion construction (c) Results of motion synthesis.

Parameterization-Based Motion Synthesis: Parameterization methods have been adopted by some researchers to represent human gestures. In motion synthesis, the new motions are generated by assigning parameter values. Rose et al. [240] provided an interesting technique which builds on verbs and adverbs. In this method, an unstructured motion data library is automatically processed, segmented, and classified into similar motion classes. Finally, motion synthesis is conducted by scattered data interpolation. Kovar and Gleicher [144] provide an automated method for identifying logically similar motions in a data set and use them to build a continuous and intuitively parameterized space of motions. To find logically similar motions that are numerically dissimilar, this method employs a novel distance metric to find "close" motions and then uses them as intermediaries to find more distant motions. Once a set of related motions has been extracted, they are automatically registered and blending techniques are applied to generate a continuous motion space. This algorithm extends previous work by explicitly constraining blend weights to reasonable values and having a runtime cost that is almost independent of the number of example motions. The scheme of this method is presented in Figure 1.11.

Figure 1.11 Visualization of the accessible locations for varied motions. In the left and right blocks, large red cubes show parameters of example motions; small gray cubes are sampled parameters.

Mukai and Kuriyama [204] proposed a common motion interpolation technique for realistic human animation, which is achieved by blending similar motion samples with weighting functions whose parameters are embedded in an abstract space. This method treats motion interpolations as statistical predictions of missing data in an arbitrarily definable parametric space. A practical technique is then introduced for statistically estimating the correlations between the dissimilarity of motions and the distance in the parametric space. Ahmed et al. [14] presented the employment of motion blending with time-warping for realistic parametric motion generation. This approach allows the animator to define the desired motion using its natural parameters, such as speed. Analysis has also been carried out to investigate the relationship between the walking speed and blending factor to remove the burden of trial and errors from the animator. As a result, a realistic walking motion with the speed specified by the user can be generated. The approaches proposed in Ref.[109] [302] integrated the control of the motion parameters in the motion graph techniques; thus, users can obtain motion data according to the input parameters. A similar idea can be found in Ref.[187] and [179].

Certain techniques [21, 94], based on machine learning techniques, concatenate disjointed segments of motion data to create novel motions. For instance, Arikan

and Forsyth [21] adopted a support vector machine (SVM) to classify motion frames into behavior classes defined by the user, and Lee et al. [158] performed clustering of motion data to speed up motion synthesis. A totally different approach is taken by Lim and Thalmann [174]. An existing motion clip is interactively evolved into a new motion through the guidance of a user; no fitness function is required. In other words, the system produces candidate motion clips and the user specifies those he/she does and does not like. The genetic algorithm uses this feedback to produce the next generation of candidate motions. Through this technique, the user can easily modify the style of an existing animation (e.g., change the mood of a walking motion).

1.3.2 Physically based Computer Animation

Physically based computer animation is about simulating the physics of a system for graphical purposes. The techniques of physically based animation have been widely used in fluid simulations, including water animation, smoke animation, and explosion animation, as well as being successfully applied to character animation. Physically based methods have been used in other interesting fields, such as sound simulation and cloth animation. In this section, the research work of physically based animation is presented according to these three categories.

1.3.2.1 Physically Based Fluid Simulation Pighin et al. [229] proposed a method for modeling incompressible fluid flows in which massless particles are advected through the fluid, and their paths are recorded. The entire fluid volume may then be reconstructed by interpolating these RBFs at a given time. Zheng et al. [360] proposed a fast dual-domain multiple boundary-integral solver, with linear complexity in the fluid domain's boundary. Enhancements are proposed for robust evaluation, noise elimination, acceleration, and parallelization. Wicke et al. [320] proposed a finite element simulation method that addresses the full range of material behavior, from purely elastic to highly plastic, for physical domains that are substantially reshaped by plastic flow, fracture, or large elastic deformations. By using this method, a dynamic meshing algorithm refines the drop while maintaining high-quality tetrahedra. At the narrowest part of the tendril, the mesher creates small, anisotropic tetrahedral where the strain gradient is anisotropic, so that a modest number is adequate. Work hardening causes the tendril to become brittle, whereupon it fractures.

Thurey et al. [292] presented a multiscale approach to mesh-based surface tension flows. In this approach, surface tension flows are simulated using a mesh-based surface representation. Lentine et al. [165] provided a novel algorithm for incompressible flow using only a coarse grid projection. This algorithm scales well to very large grids and large numbers of processors, allowing for high-fidelity simulations that would otherwise be intractable. Yu and Turk [344] proposed a novel surface reconstruction method for particle-based fluid simulators such as Smoothed Particle Hydrodynamics. In particle-based simulations, fluid surfaces are usually defined as a level set of an implicit function. Ando and Tsuruno [17] proposed a particle-based algorithm that preserves thin fluid sheets. This algorithm is effective

in creating thin liquid animations and is capable of producing visually complex thin liquid animations. The scheme of this approach is presented in Figure 1.12.

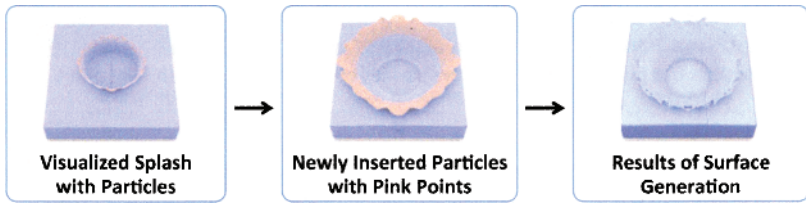

Figure 1.12 Water splash by the particle-based algorithm. The first and second diagrams: A visualized splash with particles. The pink points indicate newly inserted particles. The third diagram: A thin surface generated by anisotropic kernels.

Hong et al. [114] proposed a hybrid of Eulerian grid-based simulation and Lagrangian SPH for the realistic simulation of multiphase fluids, focusing on bubbles. Using this heuristic bubble model, they generated natural-looking computer generated bubbly water and formable objects, as well as both volumetric objects and thin shells. Zhao et al. [359] provided a framework to integrate turbulence to an existing/ongoing flow suitable for graphical controls. Compared to direct field addition, this framework avoids artificial and complex coupling by solving integration inside the NS solvers. Other techniques have been proposed for liquid simulation. Brochu et al. [251] provided a Eulerian liquid simulation framework based on the Voronoi diagram of a potentially unorganized collection of pressure, and they presented a simplified Voronoi Delaunay mesh velocity interpolation scheme and a direct extension of embedded free surfaces and solid boundaries to Voronoi meshes. Wojtan et al. [323] proposed a mesh-based surface tracking method for fluid animation that both preserves fine surface detail and robustly adjusts the topology of the surface in the presence of arbitrarily thin features such as sheets and strands. The interaction between objects and liquid has also been studied in fluid simulation. For example, Mihalef et al. [197] proposed a new Eulerian-Lagrangian method for physics-based simulation of fluid flow, which includes automatic generation of subscale spray and bubbles. The Marker Level Set method is used to provide a simple geometric criterion for free marker generation. Mosher et al. [202] proposed a novel solid/fluid coupling method that treats the coupled system in a fully implicit manner, making it stable for arbitrary time steps, large density ratios, and so on. The procedure is presented in Figure 1.13. Kim et al. [137] provided a novel wavelet method for the simulation of fluids at high spatial resolution. The algorithm enables large- and small-scale detail to be edited separately, allowing high-resolution detail to be added as a post-processing step.

Lenaerts et al. [163] proposed an SPH method for simulating the interesting fluid flowing through a porous material. Rigid and deformable objects are sampled by particles which represent local porosity and permeability distributions at a macroscopic scale. Figure 1.14 shows the simulation results.

Figure 1.13 Left: Many rigid balls with varying densities plunge into a pool of water. Center: Water splashes out of an elastic cloth bag. Right: Cloth pulled out of water.

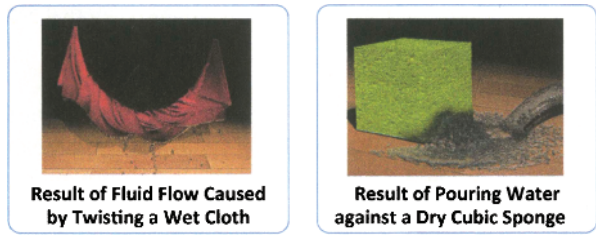

Figure 1.14 Left: Result of fluid flow caused by twisting a wet cloth. Right: Result of pouring water against a dry cubic sponge.

Another popular application of fluid simulation is smoke animation. Robinson-Mosher et al. [239] proposed a novel method for obtaining more accurate tangential velocities for solid fluid coupling. This method works for both rigid and deformable objects as well as both volumetric objects and thin shells.

Narain et al. [208] proposed a novel technique for the animation of turbulent smoke by coupling a procedural turbulence model with a numerical fluid solver to introduce subgrid-scale flow detail. From the large-scale flow simulated by the solver, this models the production and behavior of turbulent energy using a physically motivated energy model, as shown in Figure 1.15.

Nielsen et al. [212] proposed a novel approach to guiding Eulerian-based smoke animations coupled simulations at different grid resolutions. They present a variational formulation that allows smoke animations to adopt low-frequency features from a lower resolution simulation (or nonphysical synthesis) while simultaneously developing higher frequencies. Lentine et al. [164] proposed a novel algorithm (shown in Figure 1.16) for mass and momentum conservation in incompressible flow and designed a new advection method using the basic building blocks used in semi-Lagrangian advection, which is known to work well for inviscid flows, coarse grids, and large time steps, a scenario common in computer graphics.

Nielsen and Christensen [211] proposed an improved mathematical model for Eulerian-based simulations which is better suited to dynamic, time-dependent guid-

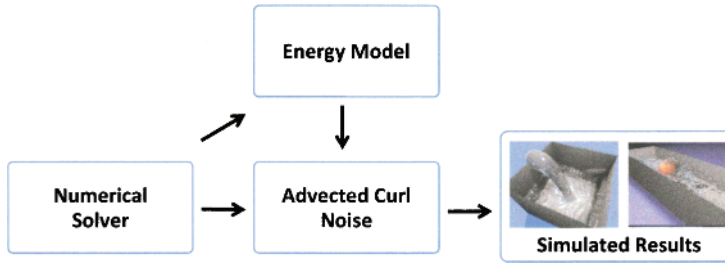

Figure 1.15 The framework of producing turbulent energy using a physically motivated energy model.

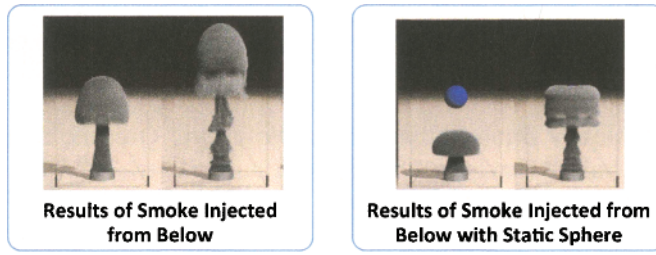

Figure 1.16 Left: Results of smoke injected from below. Right: Results of smoke injected from below with static sphere.

ance of smoke animations through a novel variational coupling of low-and high-resolution simulations. The procedure is shown in Figure 1.17.

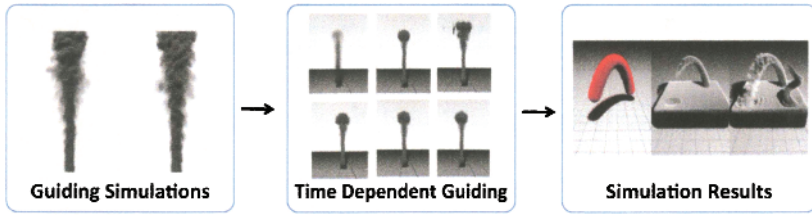

Figure 1.17 The framework of guiding simulations with a user-created flow.

Apart from water and smoke simulation, the flow effects of explosion have been explored in recent researches. Kwatra et al. [147] proposed a novel approach incorporating the ability to handle both the initial states of an explosion (including shock waves) along with the long time behavior of rolling plumes and other incompressible flow effects (as shown in Figure 1.18). In addition, Kawada and Kanai [135] provided

another novel method to procedurally model explosion phenomena by considering physical properties along control paths specified by the user. The intention of the user can be taken into account by this method, and at the same time explosion flows with complex behaviors can be realized by considering the propagations of the pressure and density flow, the fuel combustion and the detonation state to represent the drastic pressure change.

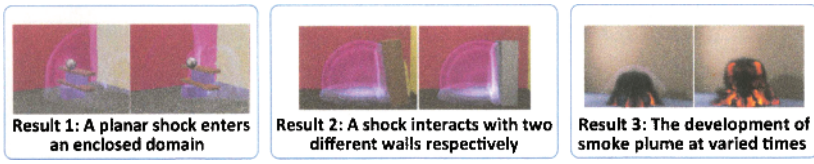

Figure 1.18 Left: A planar shock enters an enclosed domain. Middle: A shock interacts with two different walls respectively. Right: The development of smoke plume at varied times.

1.3.2.2 Physically Based Character Animation Over the decades, physically based techniques have been widely used in character animation. A typical example is NeuroAnimator [97], in which an artificial neural network performs function approximation of a physics system. James and Fatahalian [124] proposed a technique for the animation of deformable objects, as shown in Figure 1.19. This technique takes a tabulated approach to function approximation. The state is represented as the shape of an object, and then n distinct paths through the state space are sampled and stored without modification.

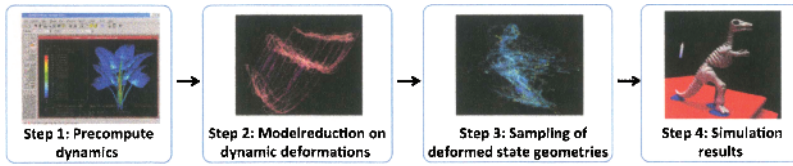

Figure 1.19 Accelerated physically based animation through precomputation.

In motion synthesis and reuse, the physical property provides some specific and useful constraints. Though some physical properties can be used as spatial constraints, more properties are neglected to improve the algorithm's performance. One typical example is that Newton's laws are always neglected. Nevertheless, these properties are important in motion synthesis; for example, Popovic and Witkin [232] proposed a system of physically based motion transformation which preserves the essential physical properties of the motion. By using the spacetime constraints dynamics formulation, this algorithm maintains the realism of the original motion sequence without sacrificing full user control of the editing process. Recently, some researchers have proposed dynamic-based realistic motion synthesis methods; for

example, based on motion capture data, Zordan et al. [367] introduced a novel technique for incorporating unexpected impacts into a motion-capture-driven animation system through the combination of a physical simulation which responds to contact forces and a specialized search routine. Using an actuated dynamic model, this system generates a physics-based response while connecting motion capture segments. This method allows characters to respond to unexpected changes in the environment based on the specific dynamic effects of a given contact while also taking advantage of the realistic movement made available through motion capture. To solve the problem of synthesizing the movements of a responsive virtual character in the event of unexpected perturbations, Ye and Liu [334] devised a fully automatic method which learns a nonlinear probabilistic model of dynamic responses from very few perturbed walking sequences. This model is able to synthesize responses and recovery motions under new perturbations different from those in the training examples. When perturbations occur, a physics-based method is adopted to initiate motion transitions to the most probable response example based on the dynamic states of the character. The effect of the proposed approach is shown in Figure 1.20.

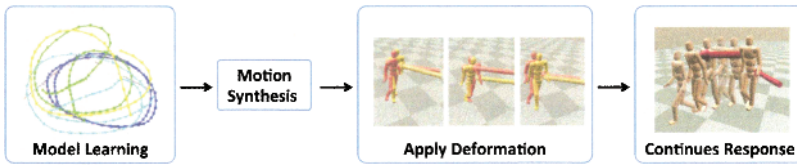

Figure 1.20 The framework of perturbations of motion in different cases. The directions are indicated by the arrows, and the motion of the body is modified by the perturbations.

More related methods have been put forward. Ye and Liu [333] proposed a technique to enhance a kinematically controlled virtual character with a generic class of dynamic responses to small perturbations. This method re-parameterizes the motion degrees of freedom based on joint actuations in the input motion and can create physically responsive motion based on kinematic pose control without explicitly computing the joint actuations (as shown in Figure 1.21). Muico et al. [203] proposed algorithms that construct composite controllers to track multiple trajectories in parallel instead of sequentially switching from one control to another. The composite controllers can blend or transition between different path controllers at arbitrary times according to the current system state. Kwon and Hodgins [149] provided a balancing control algorithm based on a simplified dynamic model: an inverted pendulum on a cart. At runtime, the controller plans a desired motion at every frame based on the current estimate of the pendulum state and a predicted pendulum trajectory. Lee et al. [161] proposed a dynamic controller to physically simulate underactuated three-dimensional full-body biped locomotion. The data-driven controller takes motion capture reference data to reproduce realistic human locomotion through real-time physically based simulation. Lasa et al. [154] presented an approach to the control of physics-based characters based on high-level features

of movement, such as center of mass, angular momentum, and end-effectors. More similar ideas of using the physical properties in motion synthesis can be found in Ref.[177, 9, 178, 252, 162, 148, 310, 123, 182, 122].

Figure 1.21 The framework of controlling virtual character with a generic class of dynamic responses to small perturbations.

1.3.2.3 Other Physically-Based Animation Systems Physically based methods have also been implemented in other interesting research fields. For instance, Cordier and Magnenat-Thalmann [73] proposed a technique in precomputing cloth animation. In this technique, cloth is simulated offline and analyzed with respect to the underlying humanoid. Volino et al. [298] proposed a simple approach to nonlinear tensile stiffness for accurate cloth simulation (as shown in Figure 1.22). This approach proposes a new simulation model that accurately reproduces the nonlinear tensile behavior of cloth materials which remains accurate and robust for very large deformations, while offering a very simple and streamlined computation process suitable for a high-performance simulation system. Kaldor et al. [132] presented a method for approximating penalty-based contact forces in yarn-yarn collisions by computing the exact contact response at one time step, then using a rotated linear force model to approximate forces in nearby deformed configurations. Aguiar et al. [12] proposed a technique for learning clothing models that enables the simultaneous animation of thousands of detailed garments in real time. This surprisingly simple conditional model learns and preserves the key dynamic properties of cloth motion along with folding details. Ozgen et al. [216] proposed a particle-based cloth model where half-derivative viscoelastic elements are included for describing both the internal and external dynamics of the cloth. These elements model the cloth responses to fluid stresses and are also able to emulate the memory-laden behavior of particles in a viscous fluid. The results are shown in Figure 1.23.

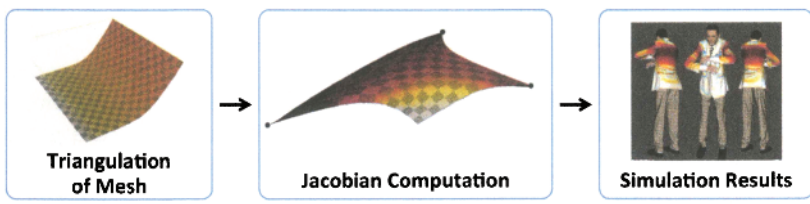

Figure 1.22 The accurate simulation of nonlinear anisotropic cloth materials is required for garment prototyping applications.

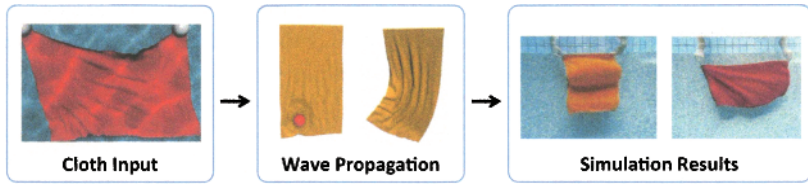

Figure 1.23 The cloth model based on fractional derivatives is able to achieve realistic underwater deformation behavior.

Fire synthesis is another interesting application of physically based simulation. Chadwick and James [64] proposed a practical method for synthesizing plausible fire sounds that are synchronized with physically based fire animations (as shown in Figure 1.24).

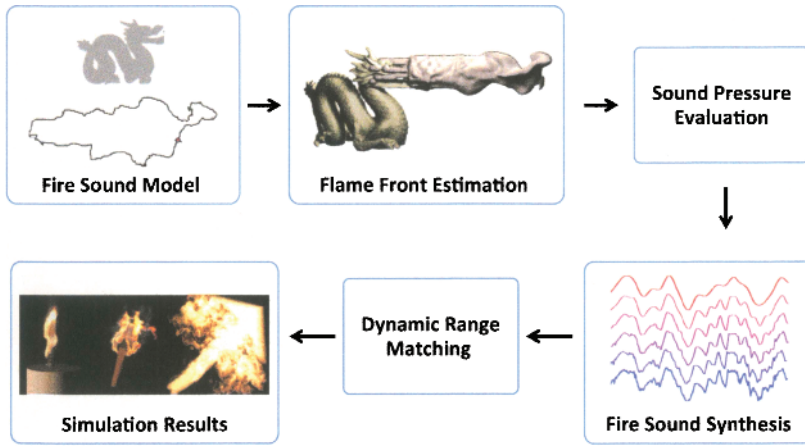

Figure 1.24 The framework of fire sound synthesis which produces the familiar sound of roaring flames synchronized with an underlying low-frequency physically based flame simulation.

1.3.3 Computer-Assisted Cartoon Animation

Cartoon animation is a popular and successful media form in contemporary life. Its creation is usually high-cost and labor-intensive, especially in traditional 2D cartoon production which includes many steps, e.g., keyframing, inbetweening, and painting. On the other hand, given the relatively long history of animation, there is a large-scale "cartoon library" that consists of various animation materials including character design, story board, scenes, and episodes, which is useful for animators and cartoon enthusiasts for effectively creating new animations by reusing and synthesizing.

Many attempts have been made in computer-aided animation, video-based cartoon generation and data driven-based cartoon synthesis to face these opportunities and challenges [67, 143, 131, 105, 195]. The following content examines these three aspects.

1.3.3.1 Computer-Aided Cartoon Animation In a traditional computer-aided cartoon animation system, high-quality auto-inbetweening and auto-coloring is achieved by constructing accurate correspondences between keyframe objects, with which inbetween frames can be generated by interpolating corresponding objects and colors can thus be propagated. Early work on correspondence construction-based inbetweening was proposed in Ref.[82], where a manual correspondence setting for a vectorization process which digitizes traditional animation production to build a "paperless" system was mentioned. Subsequently, Kort [143] provided an interactive cartoon system for automatic inbetweening with a possible solution for the "undetectable" parts noticed in Ref. [82]. Whited et al. [319] proposed the "BetweenIT" system for the user-guided automation of tight inbetweening. They designed a set of user-guided semi-automatic techniques that fit well with current practice and minimize the number of required artist gestures. The performance of this system is shown in Figure 1.25. Baxter et al. [29] proposed a set of algorithms for the compatible embedding of 2D shapes. Such embeddings offer a convenient way to interpolate shapes with complicated structures and detailed features. This system has the advantages of less user input with faster and more robust implementation, which make it ideal for interactive use in practical applications.

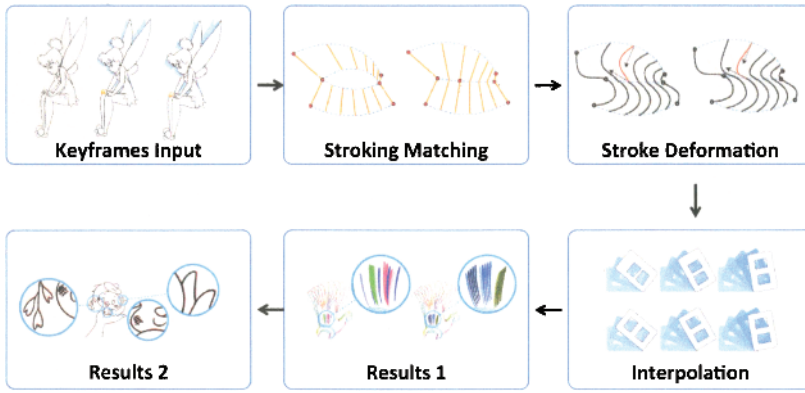

Figure 1.25 The framework for the user-guided automation of tight inbetweening.

Apart from these traditional interpolation-based systems, other computer-aided systems with novel ideas have been proposed for complicated cartoon animation production. Rivers et al. [237] proposed a method to bring cartoon objects and characters into the third dimension by giving them the ability to rotate and be viewed from any angle. Figure 1.26 shows how 2D vector art drawings of a cartoon from different views can be used to generate a novel structure, the 2.5D cartoon model,

which can be used to simulate 3D rotations and generate plausible renderings of the cartoon from any view.

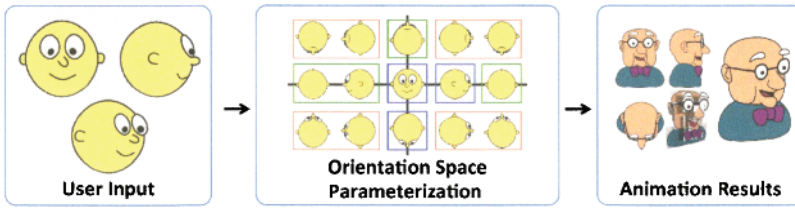

Figure 1.26 The framework of a 2.5D cartoon.(a) Taking the vector art drawings of a cartoon from different views. (b) Parameterizing orientation space with different views. (c) Generating a 2.5D cartoon automatically, which associates each stroke with a 3D position.

In recent years, Sykora and his colleagues have proposed a group of novel techniques [276, 275, 277] to enhance automatic 2D cartoon generation. In automatic depth generation, they proposed a novel interactive approach to cartoon pop-up that enables artists to quickly annotate their hand-made drawings and animations with additional depth information [276]. The scheme of this method is shown in Figure 1.27. In automatic coloring, they provided a novel color-by-example technique [275] which combines image segmentation, patch-based sampling, and probabilistic reasoning. This method is able to automate colorization when new color information is applied on the already designed black-and-white cartoon. The results are shown in Figure 1.28. A "LazyBrush" [277] is provided by their research group. "LazyBrush" is a new interactive tool which is used for painting hand-made cartoon drawings, based on an optimization framework that tries to mimic the behavior of an ideal painting tool as proposed by professional illustrators. In the field of automatic cartoon character generation, the researchers presented a new approach [274] to deformable image registration based on an approach analogous to the dynamic simulation of deformable objects. They adopted a novel geometrically motivated iterative scheme in which point movements are decoupled from shape consistency. By combining locally optimal block matching with as-rigid-as-possible shape regularization, this algorithm allows users to register images undergoing large free-form deformations and appearance variations.

Barnes et al. [28] proposed a novel interface for creating cutout style animations by manipulating paper puppets. The system relies on easy to use components for creating puppets, combined with a natural interface for controlling them. This system is presented in Figure 1.29.

Some researchers are focusing on the effects of light, shade, and smoke. McGuire and Fein [193] proposed an algorithm for rendering animated smoke particle systems in a cartoon style which includes outlines and cel-shading. They combine the renderer with a fast simulator that generates the phenomenology of real smoke but has artistically controllable parameters. Together they produce real-time interactive smoke animations at over 30 fps. Selle et al. [249] proposed a technique for rendering

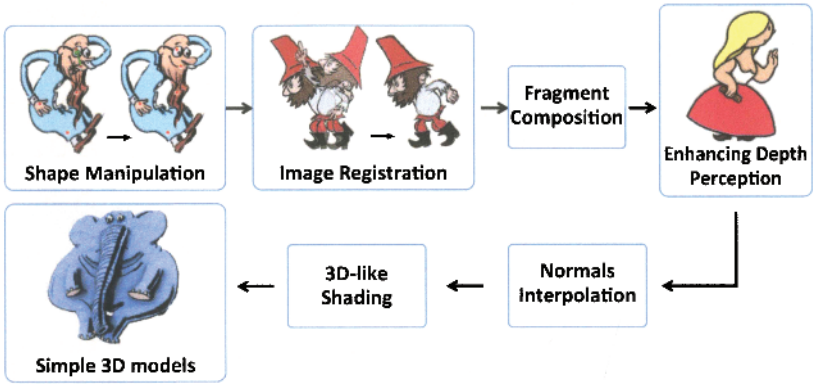

Figure 1.27 Examples of a pop-ups cartoon generating system.

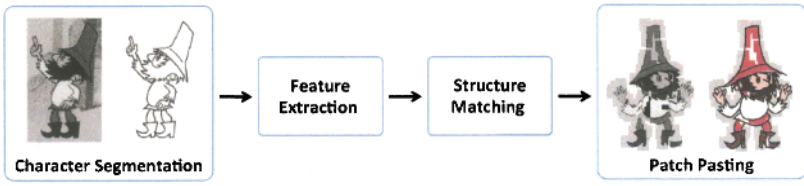

Figure 1.28 The framework of automatic colorization applied on the already designed black and white cartoon.

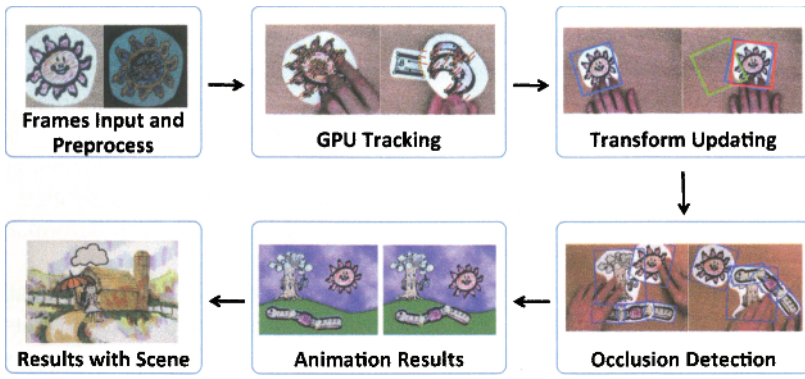

Figure 1.29 The framework of manipulating cutout paper puppets tracked in real time to control an animation.

stylized smoke. The underlying dynamics of the smoke are generated with a standard Navier-Stokes fluid simulator and output in the form of advected marker particles.

The results are shown in Figure 1.30. Anjyo [19] proposed a direct manipulation method that allows users to create and edit stylized highlights and cartoon-shaded areas in realtime, essentially with only click-and-drag operations. This method provides intuitive click-and drag operations for translating and deforming the shaded areas, including rotation, directional scaling, splitting, and squaring of highlights, all without tedious parameter tuning. The results can be found in Figure 1.31.

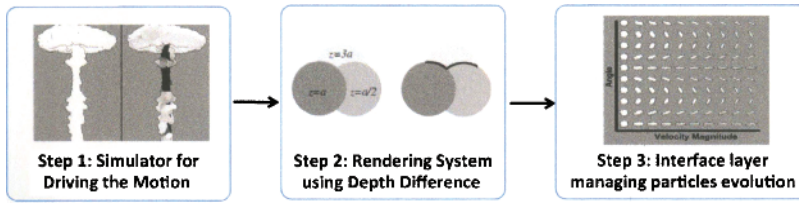

Figure 1.30 The framework of rendering stylized smoke.

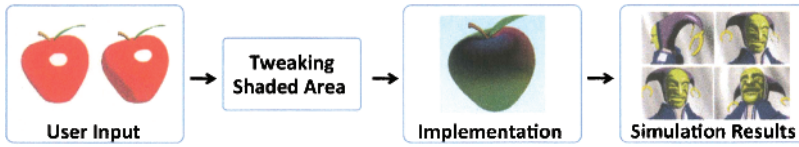

Figure 1.31 Direct manipulation of stylized shading.

1.3.3.2 Video-Based Cartoon Generation The computer-aided systems discussed above are designed to reduce the time cost of cartoon production; however, many skillful interactions are required due to the lack of effective cartoon representation. The quality of the obtained animation mainly relies on traditional manual production methods; therefore, these systems are specifically designed for professionals rather than nonprofessional users who produce cartoons for different purposes, such as fun and education. These drawbacks prompt researchers to provide novel systems which will allow users to quickly and efficiently generate cartoons from existing videos.

Liang et al. [173] proposed a prototype system for generating 3D cartoons from broadcast soccer video. This system takes advantage of computer vision and computer graphics techniques to provide users with a new experience which cannot be obtained from the original video. Collomosse et al. [72] proposed a novel NPR framework for synthesizing nonphotorealistic animations from video sequences. The spatio-temporal approach adopted by the framework enables the users to smoothly vary attributes, such as region or stroke color over time, and to create improved motion estimates of objects in the video. The generated results are presented in Figure 1.32.

Wang et al. [311] proposed a system for transforming an input video into a highly abstracted, spatio-temporally coherent cartoon animation with a range of styles.

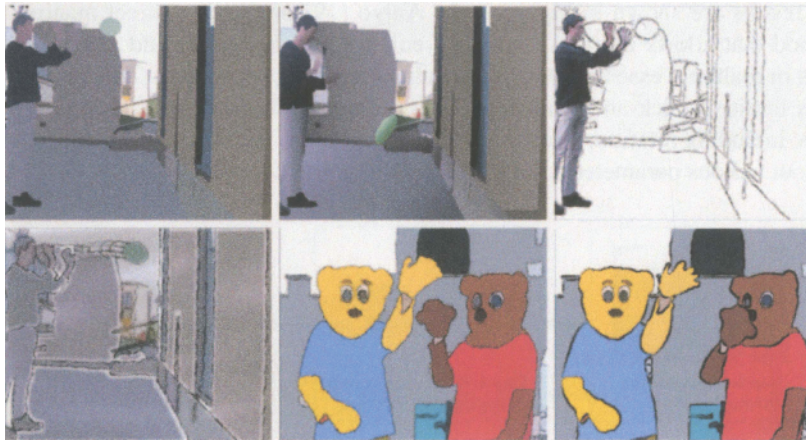

Figure 1.32 Some of the available artistic effects. Top, left to right: Cartoon flat and gradient shaded animations from the BOUNCE sequence. Mixed reality effect where original footage has been selectively matted in to a sketchy line animation. Bottom, left to right: Water color wash effect and cartoon flat shaded bears with sketchy and thick stroke outlines.

In this system, the user simply outlines objects on keyframes. A mean-shift-guided interpolation algorithm is then employed to create three dimensional semantic regions by interpolation between the keyframes while maintaining smooth trajectories along the time dimension. A variety of rendering styles is shown in Figure 1.33.

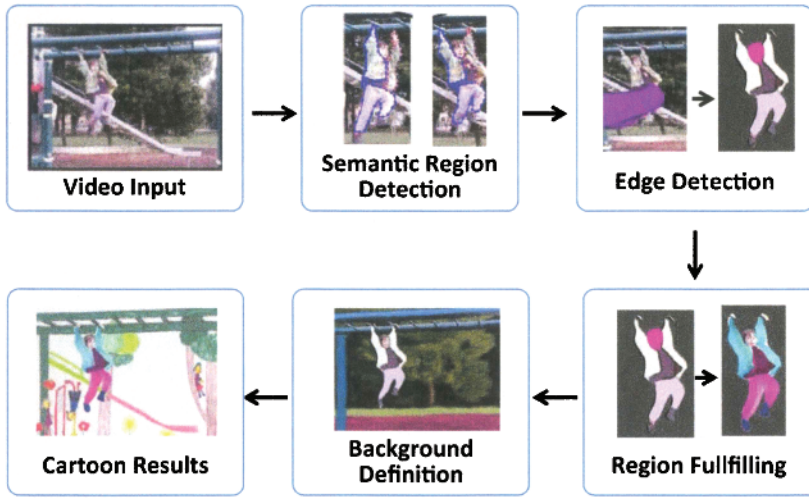

Figure 1.33 Examples created by Video Tooning with different styles.

Hays et al. [106] proposed a technique for transforming images and videos into painterly animations with different artistic styles. They determine and apply motion information from different user-specified sources to static and moving images. These properties that encode the spatio-temporal variations are then used to render (or paint) effects of selected styles to generate images and videos with a painted look. Painterly animations are generated using a mesh of brush stroke objects with dynamic spatio-temporal properties. The rendering results are shown in Figure 1.34.

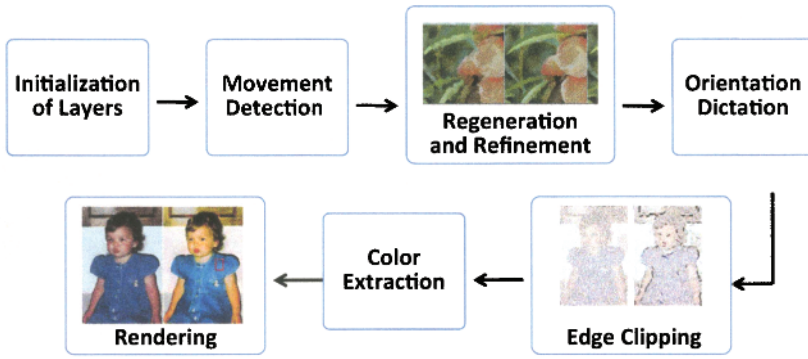

Figure 1.34 The framework of transforming images and videos into painterly animations with different artistic styles.

Lin et al. [175] provided an interactive system that stylizes an input video into a painterly animation. The system consists of two phases. The first is a Video Parsing phase that extracts and labels semantic objects with different material properties (skin, hair, cloth, and so on) in the video, and then establishes robust correspondence between frames for discriminative image features inside each object. The second Painterly Rendering phase performs the stylization based on video semantics and feature correspondence. This system can efficiently generate oil painting animations from the video clips. Similarly, Agarwala [11] proposed an interactive system that allows children and others untrained in cel animation to create two-dimensional cartoons from video streams and images. A cartoon is created in a dialogue with the system. After recording video material, the user sketches contours directly onto the first frame of the video. These sketches initialize a set of spline-based active contours which are relaxed to best fit the image and other aesthetic constraints. Small gaps are closed, and the user can choose colors for the cartoon. The system then uses motion estimation techniques to track these contours through the image sequence. The user remains in the process to edit the cartoon as it progresses. Some results are shown in Figure 1.35.

1.3.3.3 Data-Driven-Based Cartoon Synthesis Data driven approaches have been widely used in cartoon animation; for instance, motion data are applied to drive the cartoon characters to move. Bregler et al. [53] proposed a new technique that captures the motion style of cartoons and retargets the same style to a different domain.

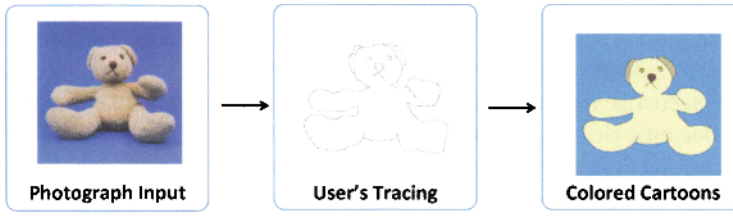

Figure 1.35 Creating cartoon images of toys. (a) The original photographs. (b) The user's tracing. (c) The finished, colored cartoons.

This is done by tracking affine motion and key-shape-based interpolation weights. Figure 1.36 shows how the motion extracted from the original character is retargeted to a new character. Li et al. [172] proposed a system to animate cartoon faces from speech with emotions. This system consists of two components: emotion-driven cartoon animation and speech-driven cartoon animation.

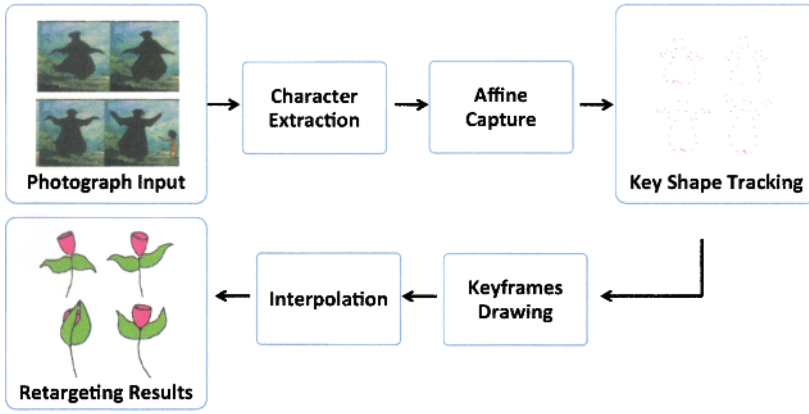

Figure 1.36 Motion of walking cartoon character retargeted to 3D model.

Another group of data-driven approaches in cartoon animation is to reuse the existing cartoon frames [342, 347] to synthesize novel animations. De Juan and Bodenheimer [131] proposed a Cartoon Clip Synthesis system which creates novel cartoon clips from existing cartoon data. This method combines sequences of similar-looking cartoon data into a user-directed sequence. Starting with a small amount of cartoon data, a nonlinear dimensionality reduction method is employed to discover a lower-dimensional structure of the data. The user selects a start and end frame and the system traverses this lower-dimensional manifold to re-sequence the data into a new animation. The performance is presented in Figure 1.37. Haevre et al. [103] proposed a novel method that facilitates the creation of endless animations or video

loops for smaller-scale productions. This approach benefits from several ideas and techniques from cartoon textures, computer-assisted animation, and motion graphs. It combines the re-sequencing of existing material with the automatic generation of new data. Furthermore, the animator can interfere with the animation process at any arbitrary moment.

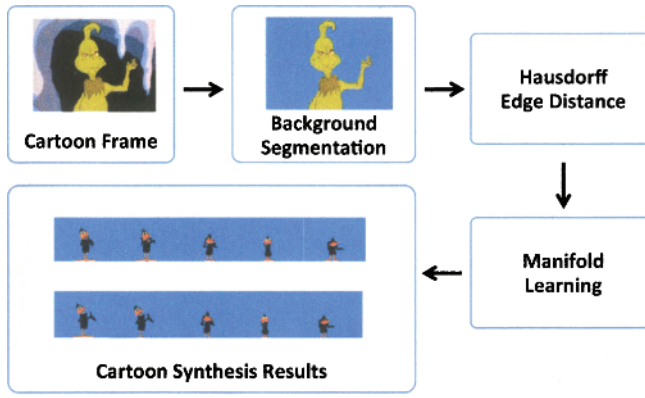

Figure 1.37 The framework of cartoon synthesis using "Cartoon Texture."

The method of cartoon texture performs well for simple cartoon characters, but for characters with complex colors and motion patterns, it fails to generate smooth clips because the edges can encode neither the color information nor the motion pattern. Therefore, Yu et al. [345] provided an efficient technique which combines both the motion difference and edge difference in similarity estimation. This model is controlled by a weight parameter.

In the linear combination method [345], it is not easy for users to determine the weights. In this case, Yu and his colleagues proposed a semi-supervised multiview subspace learning algorithm (semi-MSL) [339] to automatically combine multiple features in cartoon synthesis by using alternative optimization. In this approach, multiple features including color histogram, Hausdorff edge feature and skeleton feature, are adopted to represent cartoon characters. Retrieval-based cartoon synthesis is adopted which requires users to provide an initial character to start the retrieval. The final cartoon synthesis is conducted as an iteration process in which a group of similar cartoon characters is obtained to form new sequences. The results of cartoon synthesis are shown in Figure 1.38. The details of this approach will be introduced in the following chapters.

Cartoon character retrieval is critical for cartoonists to effectively and efficiently make cartoons by reusing existing cartoon data. To successfully achieve these tasks, it is essential to extract visual features to comprehensively represent cartoon characters and accurately estimate dissimilarity between cartoon characters. However, due to the semantic gap, the cartoon retrieval by using these visual features still cannot achieve excellent performance. Since the labeling information has been proven effective

to reduce the semantic gap, Yu et al. [337] introduce a labeling procedure called Interactive Cartoon Labeling (ICL). The labeling information actually reflects userâ^s retrieval purpose. A dimension reduction tool, termed sparse transfer learning (SPA-TL) [294], is adopted to effectively and efficiently encode userâ^s search intention. In particular, SPA-TL exploits two pieces of knowledge data, i.e., the labeling knowledge contained in labeled data and the data distribution knowledge contained in all samples (labeled and unlabeled). Experimental evaluations in cartoon synthesis suggest the effectiveness of the visual features and SPA-TL. The framework is presented in Figure 1.39.

Figure 1.38 The result of retrieval-based cartoon synthesis.

1.3.4 Crowd Animation

In simulation, it is critical to create an interactive, complex, and realistic virtual world in which the user can have an immersive experience during their navigation through the world. As the size and complexity of environments in the virtual world increase, it becomes more necessary to populate them with people, and this is the reason why rendering crowds [37, 331, 76, 77, 24] in realtime is crucial. Generally, crowd animation is composed of three important areas: the realism of group behavior [291], the realism of individual behavior [157] and the integration of both. In the following section, recent developments in this field is reviewed according to these three categories.

1.3.4.1 Realism of Group Behavior Group behavioral realism is mainly targeted for simple 2D visualizations because most of the attention is concentrated on simulating the behavior of the group. The concept is to create virtual characters

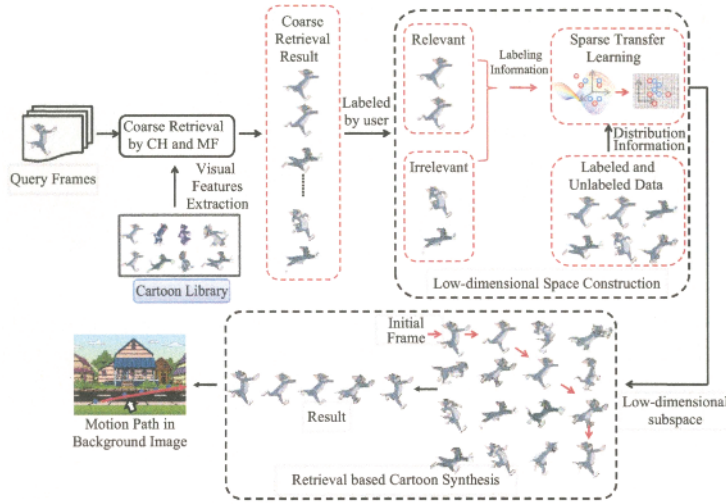

Figure 1.39 The framework of interactive cartoon reusing by sparse transfer learning (SPA-TL).

that are autonomous agents and thus self-animating. A character makes its decisions through a behavioral model: an executable model defining the character's thought process. Collision avoidance and path finding are mainly considered in this research field.

Ondrej et al. [214] proposed a novel vision-based approach of collision avoidance between walkers that fits the requirements of interactive crowd simulation. In imitation of humans, and based on cognitive science results, they detect future collisions as well as their dangerousness from visual stimuli. The simulation results are presented in Figure 1.40. Similarly, Guy et al. [101] proposed a robust algorithm for collision avoidance among multiple agents. This approach extends the notion of velocity obstacles from robotics and formulates the conditions for collision free navigation as a quadratic optimization problem. A discrete optimization method is used efficiently to compute the motion of each agent. It can robustly handle dense scenarios with tens or hundreds of thousands of heterogeneous agents in a few milliseconds. Some simulation results are presented in Figure 1.41. In addition, Lamarche and Donikian [152] provided a real-time crowd model based on continuum dynamics. In this model, a dynamic potential field simultaneously integrates global navigation with moving obstacles such as other people, efficiently solving the motion of large crowds without the need for explicit collision avoidance. Patil et al. [225] proposed a novel approach to direct and control virtual crowds using navigation fields. This method guides one or more agents toward desired goals based on guidance fields. A similar idea can be found in [256]. Paris et al. [219] proposed a method for solving interactions between pedestrians and avoiding inter-collisions. This approach is agent-based and predictive: Each agent perceives surrounding agents and extrapolates their trajec-

tory in order to react to potential collisions. Similar works can be found in Ref. [100, 150, 278].

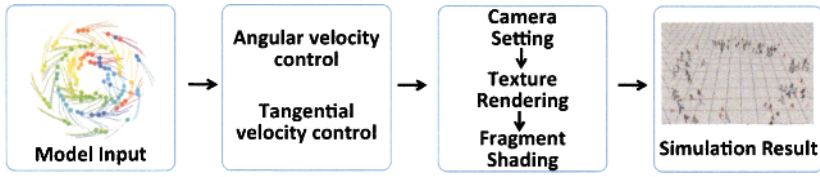

Figure 1.40 Emergent self-organized patterns appear in real crowds of walkers. This simulation displays similar effects by proposing an optic flow-based approach for steering walkers inspired by cognitive science work on human locomotion.

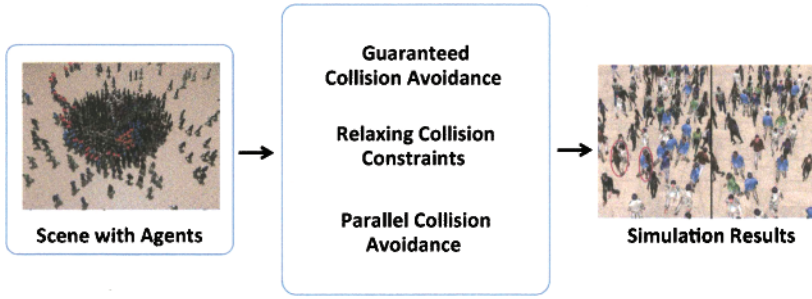

Figure 1.41 Dense Circle Scenario: 1000 agents are arranged uniformly around a circle and move towards their antipodal position. This simulation runs at over 1000 FPS on an Intel 3.14 GHz quad core, and over 8000 FPS on 32 Larrabee cores.

Apart from collision avoidance, path finding is another important issue in achieving behavioral realism in crowd simulation. Lamarche et al. [152] proposed a suitable topological structuring of the geometric environment to allow fast path finding as well as an efficient reactive navigation algorithm for virtual humans evolving inside a crowd. Putting together the high complexity of a realistic environment such as a city, a large number of virtual humans and the real-time constraint requires optimization of each aspect of the animation process. The objective of this work is to reproduce this crucial human activity inside virtual environments. The results of path finding in an indoor and outdoor scene are presented in Figure 1.42. Sung et al. [273] integrated the motion capture data into the crowd simulation offering a highly efficient motion synthesis algorithm that is well suited to animating large numbers of characters. Given constraints that require characters to be in specific poses, positions, and orientations in specified time intervals, this algorithm synthesizes motions that exactly satisfy these constraints. To provide a good initial guess for the search, they adopt a fast path planner based on probabilistic roadmaps to navigate characters through complex environments. Similarly, Lai et al. [151] proposed Group Motion Graphs, a data-

driven animation technique for groups of discrete agents, such as flocks, herds or small crowds. Similar works can be found in Ref.[270].

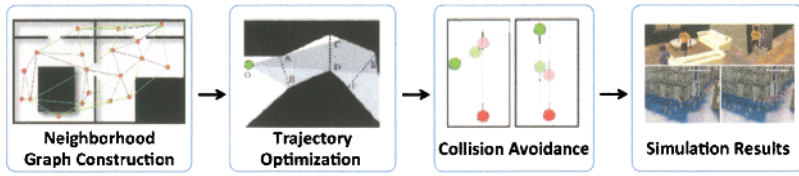

Figure 1.42 The framework of path finding for indoor and outdoor navigation.

Another interesting crowd simulation method imitates real human crowds by video tracking. Lee et al. [160] proposed a novel approach for simulating a crowd of virtual humans by recording the motion of a human crowd from an aerial view with a camcorder. Their method extract the two-dimensional moving trajectories of each individual in the crowd, and learns an agent model from observed trajectories. The agent model decides each agent's actions based on features of the environment and the motion of nearby agents in the crowd. Figure 1.43 shows the simulation results. Zhao et al. [358] proposed a model-based approach to interpret the image observations by multiple partially occluded human hypotheses in a Bayesian framework. Wang et al. [314] proposed a novel unsupervised learning framework to model activities and interactions in crowded and complicated scenes. In this framework, hierarchical Bayesian models are used to connect three elements in visual surveillance: low-level visual features, simple "atomic" activities, and interactions. A crowd synthesis method [129] has also been proposed to blend existing crowd data for the generation of a new crowd animation. The new animation includes an arbitrary number of agents, extends for an arbitrary duration, and yields a natural looking mixture of the input crowd data. Figure 1.44 illustrates the simulation results.

Additionally, Guy et al. [102] proposed a totally novel technique to generate heterogeneous crowd behaviors using personality trait theory. This formulation is based on adopting results of a user study to derive a mapping from crowd simulation parameters to the perceived behaviors of agents in computer-generated crowd simulations. The simulation results are shown in Figure 1.45.

1.3.4.2 Realism of Individual Behavior Creating realistic motion for animated characters in crowds is an important problem in achieving individual realism. The use of motion capture data for animating virtual characters has become a popular technique in recent years. By capturing the movement of a real human and replaying this movement on a virtual character, the resulting motion exhibits a high degree of realism. Lau and Kuffner [157] presented a behavior planning approach to automatically generate realistic motions for animated characters in a crowd. First, the motion clips are organized into a finite-state machine (FSM) of behaviors. Each state of the FSM includes a collection of motions representing a high-level behavior. Given this behavior FSM and a pre-defined environment, this algorithm searches the FSM and

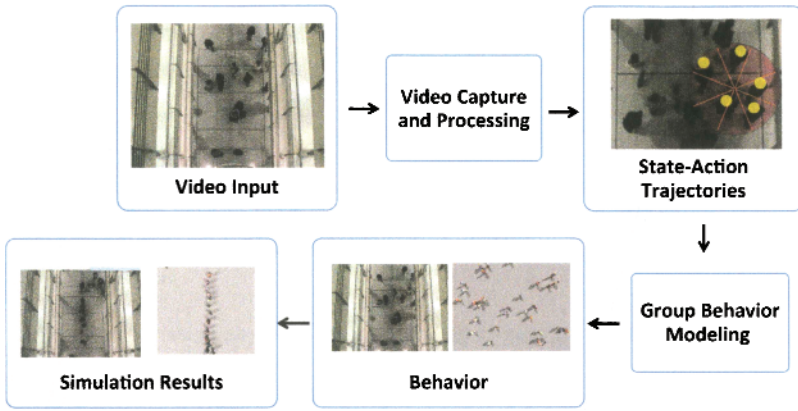

Figure 1.43 Simulation results of learning group behavior from crowd videos.

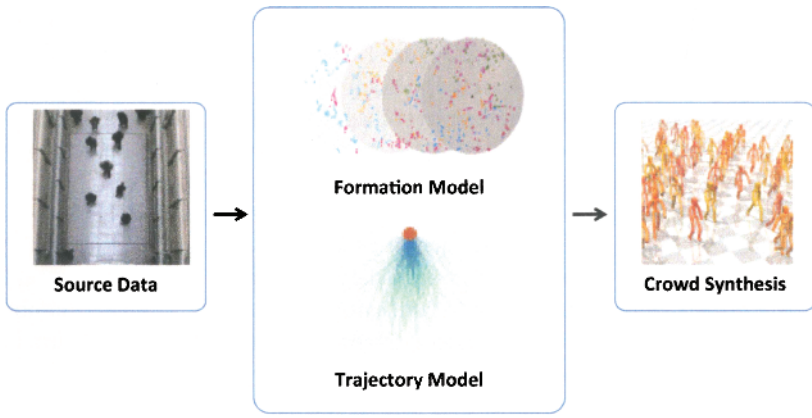

Figure 1.44 Morphable crowd models synthesize virtual crowds of any size and any length from input crowd data. The synthesized crowds can be interpolated to produce a continuous span of intervening crowd styles.

plans for a sequence of behaviors that allows the character to reach a user-specified goal. The simulation results are shown in Figure 1.46. Lerner et al. [167] proposed a data-driven approach for fitting behaviors to simulated pedestrian crowds. This method annotates agent trajectories, generated by any crowd simulator, with action-tags. The aggregate effect of animating the agents according to the tagged trajectories enhances the impression that the agents are interacting with one another and with the environment. Kim et al. [136] proposed a novel motion editing technique that allows the user to manipulate synchronized multiple character motions interactively. This method formulates the interaction among multiple characters as a collection of

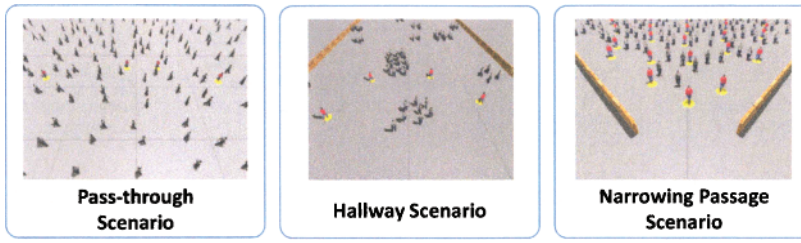

Figure 1.45 Left: Pass-through Scenario. Middle: Hallway Scenario. Right: Narrowing Passage Scenario.

linear constraints and enforces the constraints, while the user directly manipulates the motion of characters in both spatial and temporal domains. The synthesized multiple character motions are presented in Figure 1.47.

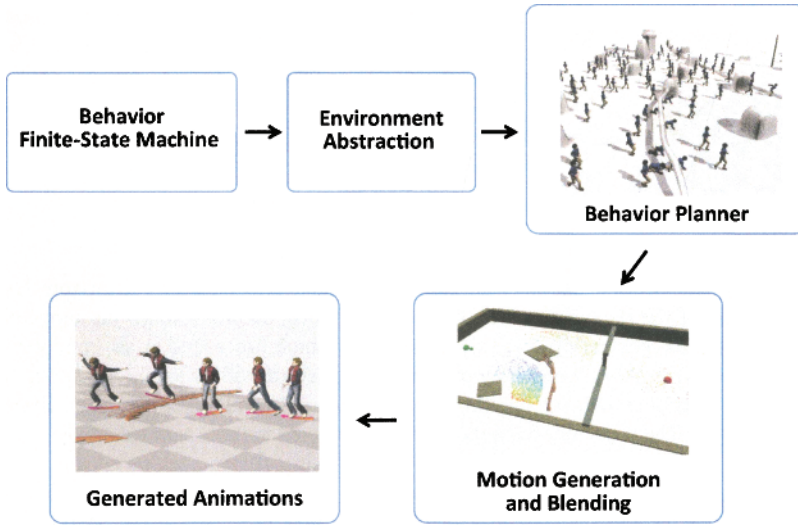

Figure 1.46 The framework of planning behaviors for animated characters navigating in a complex dynamic environment.

In addition to motion capture, the information from video tracking can be used to directly drive the characters. Lerner et al. [166] proposed a novel example-based crowd simulation technique. By learning from real-world examples contained in videos, the autonomous agents in this system display complex natural behaviors that are often missing in crowd simulations. Examples are created from tracked video segments of real pedestrian crowds. During a simulation, autonomous agents search for examples that closely match the situation that they are facing. Trajectories taken

Figure 1.47 The characters carry boxes in relay. The user interactively manipulates the synchronized multicharacter motion to shape a spiral.

by real people in similar situations are copied to the simulated agents, resulting in seemingly natural behaviors. The simulation results are shown in Figure 1.48.

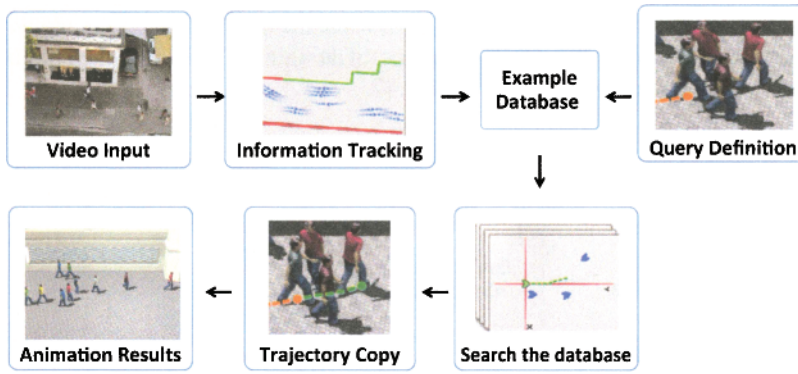

Figure 1.48 Framework of example-based crowd simulation technique. The top row depicts the construction of the database, which takes place during preprocessing: the input video is manually tracked, generating a set of trajectories which are encoded as examples and stored in the database. At runtime (bottom row) the trajectories of the agents are synthesized individually by encoding their surroundings (forming a query) and searching the database for a similar example. The trajectory from the example is copied to the simulated agent.

Sung et al. [272] presented a new approach to controlling the behavior of agents in a crowd. This method is scalable in the sense that increasingly complex crowd behaviors can be created without a corresponding increase in the complexity of the agents. Users can dynamically specify which crowd behaviors happen in various parts of an environment. Ennis et al. [80] investigated the importance of matching audio to body motion for virtual conversing characters. Yeh et al. [335] introduced the concept of composite agents to effectively model complex agent interactions for agent-based crowd simulation. Each composite agent consists of a basic agent that is associated with one or more proxy agents. This formulation allows an agent to exercise influence over other agents greater than that implied by its physical properties.

When simulating crowds, it is inevitable that the models and motions of many virtual characters will be replicated. Therefore, the perceptual impact of this trade-off

should be studied. McDonnell et al. [191] proposed an algorithm to consider the ways in which an impression of variety can be created and the perceptual consequences of certain design choices. They established that replicated models can be masked by color variation, random orientation, and motion. Conversely, the perception of cloned motions remains unaffected by the model on which they are displayed. Other factors that influence the ability to detect clones were examined, such as proximity, model type and characteristic motion. The results are shown in Figure 1.49. The authors also investigated which body parts of virtual characters are most looked at in scenes containing duplicate characters or clones [192].

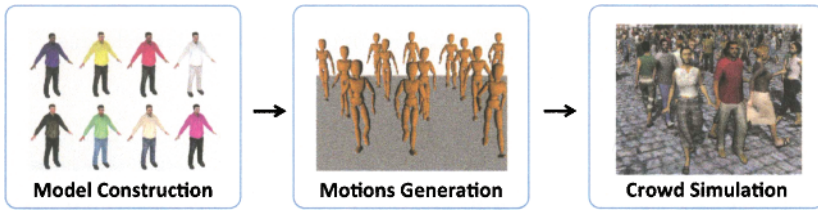

Figure 1.49 Examples of a crowd used in the Appearance Variation Experiment with the maximum number of clones.

1.3.4.3 Integration of Group Realism and Individual Realism Some research works focus on achieving both group behavior realism and individual behavior realism; for instance, large dense crowds show aggregate behavior with reduced individual freedom of movement. Narain et al. [207] presented a novel, scalable approach for simulating such crowds, using dual representation both as discrete agents and as a single continuous system. In the continuous setting, they introduce a novel variational constraint called unilateral incompressibility to model the large-scale behavior of the crowd and accelerate inter-agent collision avoidance in dense scenarios. This method makes it possible to simulate very large, dense crowds composed of up to a hundred thousand agents at near-interactive rates on desktop computers. The simulation results are presented in Figure 1.50. Shao and Terzopoulos [250] proposed another sophisticated human animation system that combines perceptual, behavioral, and cognitive control components, whose major contribution is a comprehensive model of pedestrians as highly-capable individuals. They address the difficult open problem of emulating the rich complexity of real pedestrians in urban environments. In their approach, the comprehensive model features innovations in these components, as well as in their combination, yielding results of unprecedented fidelity and complexity for fully autonomous multi-human simulation in a large urban environment. The experimental results are presented in Figure 1.51. Pelechano et al. [227] proposed a novel system-HiDAC which focuses on the problem of simulating the local motion and global pathfinding behaviors of crowds moving in a natural manner within dynamically changing virtual environments. By applying a combination of psychological and geometrical rules with a social and physical forces model,HiDAC

exhibits a wide variety of emergent behaviors from agent line formation to pushing behavior and its consequences.

Figure 1.50 Some examples of large, dense crowds simulated with this technique.

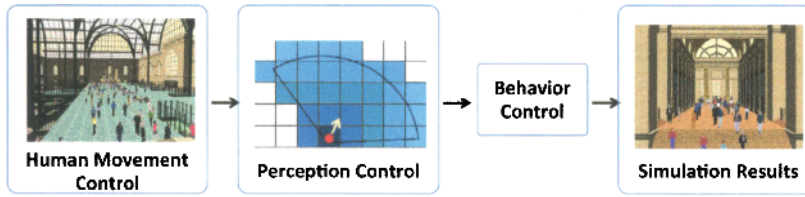

Figure 1.51 A large-scale simulation of a virtual train station populated by self-animated virtual humans.

Recent advances in local methods have significantly improved the collision avoidance behavior of virtual characters. However, existing methods fail to take into account that in real-life pedestrians tend to walk in small groups, consisting mainly of pairs or triples of individuals. Karamouzas and Overmars [133] proposed a novel approach to simulate the walking behavior of such small groups. This model describes how group members interact with each other, with other groups and individuals. In their approach, both the individual and group behavior are considered in crowd simulation. The results are shown in Figure 1.52.

Figure 1.52 Example test-case scenarios. (a) A group has to adapt its formation to pass through a doorway; (b) Group interactions in a confined environment; (c) A faster group overtakes a slower moving group; (d) A group of three agents walking through a narrow corridor.

1.3.5 Facial Animation

Recent interest in facial modeling and animation is spurred by the increasingly frequent appearance of virtual characters in film and video, inexpensive desktop processing power, and the potential for a new 3D immersive communication metaphor for human-computer interaction. According to the processed information and rendering effects, facial animation can be separated into 2D face synthesis and 3D face synthesis [176]. We review recent research developments in face animation from these two aspects.

1.3.5.1 2D Face Synthesis According to the 2D face animation application, a variety of techniques including the statistical method, linear regression, support vector regressor and data-driven method have been applied in 2D face synthesis. The specific 2D face applications are natural face synthesis and cartoon face synthesis. Abboud and Davoine [8] presented a statistical-based method for extracting appearance parameters from a natural image or video sequence for natural face synthesis, which allows reproduction of natural-looking, expressive synthetic faces. This technique was used to perform face synthesis and tracking in video sequences as well as facial expression recognition [342, 264] and control. Leyvand et al. [170] proposed a digital face beautification method based on the optimization of a beauty function modeled by a support vector regressor. Given a new face, this method extracts a set of distances between a variety of facial feature locations which define a point in a high-dimensional "face space". The face space is then searched for a nearby point with a higher predicted attractiveness rating. Once such a point is found, the corresponding facial distances are embedded in the plane and serve as a target to define a 2D warp field which maps the original facial features to their adjusted locations. The simulation results are shown in Figure 1.53.

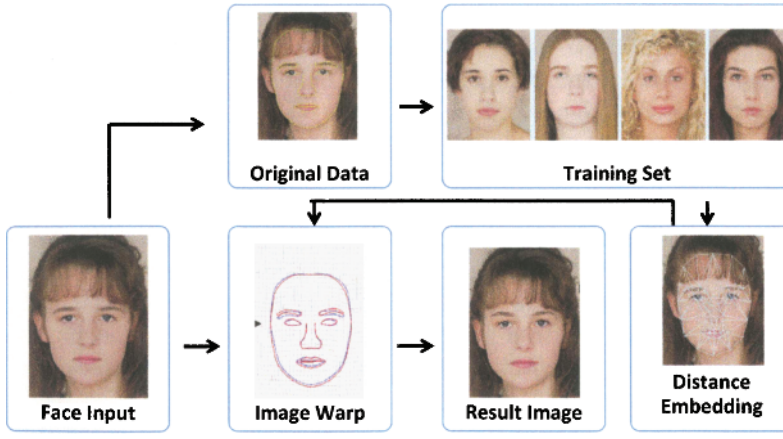

Figure 1.53 The framework of digital face beautification method.

Many systems have been proposed for cartoon face synthesis. Caricature Zone [1] is a cartoon face making system, which has stored many components of human face, such as hair, cheek, etc. Through this system, the user can select various cartoon style components to construct the specified human. UDrawFace Pro [5] adopts one human face and the user provides some interaction; the system then adopts edge detection to extract the facial features and create the cartoon face. Hsu and Jain [118] proposed semantic face graphs derived from a subset of vertices of a 3Dface model to construct cartoon faces for face matching. The cartoon faces are generated in a coarse-to-fine fashion; face detection results are used to coarsely align semantic face graphs with detected faces and interacting snakes are used to finely align face graphs with sensed face images. Chen et al. [66] presented a novel cartoon system called PicToon, which can generate a personalized cartoon face from an input Picture. PicToon is convenient to use and requires little user interaction. It consists of three major components: an image-based Cartoon Generator, an interactive Cartoon Editor for exaggeration, and a speech-driven Cartoon Animator. The framework of this method is shown in Figure 1.54.

Wang et al. [315] propose a novel face photo-sketch synthesis and recognition method using a multiscale Markov Random Fields(MRF) model. This system has three components: (1) given a face photo, synthesizing a sketch drawing; (2) given a face sketch drawing, synthesizing a photo; (3) searching for face photos in the database based on a query sketch drawn by an artist. It has useful applications for both digital entertainment and law enforcement. It is assumed that faces to be studied are in a frontal pose, with normal lighting and neutral expression, and have no occlusions. To synthesize sketch/photo images, the face region is divided into overlapping patches for learning. The size of the patches decides the scale of local face structures to be learned. From a training set which contains photo-sketch pairs, the joint photo-sketch model is learnt at multiple scales using a multiscale MRF model. By transforming a face photo to a sketch (or transforming a sketch to a photo), the difference between photos and sketches is significantly reduced, thus allowing effective matching between the two in face sketch recognition. After the photo-sketch transformation, in principle, most of the proposed face photo recognition approaches can be applied to face sketch recognition in a straightforward way. The synthesized results are presented in Figure 1.55.

Gao et al. [85] propose an automatic sketch photo synthesis and retrieval algorithm based on sparse representation. The proposed sketch-photo synthesis method (SNS-SRE) works at patch level and is composed of two steps: sparse neighbor selection (SNS) for an initial estimate of the pseudo-image (pseudo-sketch or pseudo-photo) and sparse-representation-based enhancement (SRE) for further improving the quality of the synthesized image. SNS can find closely related neighbors adaptively and then generate an initial estimate for the pseudo-image. In SRE, a coupled sparse representation model is first constructed to learn the mapping between sketch patches and photo patches, and a patch-derivative-based sparse representation method is subsequently applied to enhance the quality of the synthesized photos and sketches. Finally, four retrieval modes, namely sketch-based, photo-based, pseudo-sketch-based, and pseudo-photo-based retrieval, are proposed, and a retrieval algorithm is

developed by using sparse representation. The framework of synthesis is presented in Figure 1.56.

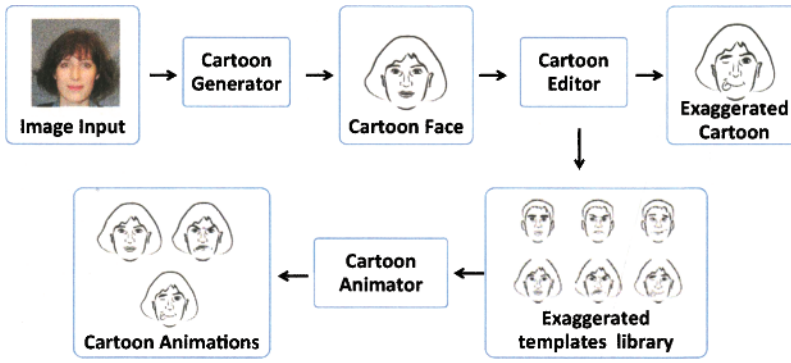

Figure 1.54 Framework of the PicToon System.

A closely related task in facial animation is speech animation. Some notable techniques are designed around machine learning. The goal is to learn how to properly animate a face so that realistic animations can be produced automatically for arbitrary speech. Ideally, the use of real-world examples should result in highly realistic animation. Realism is critical because human observers can easily detect incorrect speech animation. Some of the techniques in this area also propose integrated image-based rendering schemes. Bregler et al. [52] proposed a novel technique which can modify the movie sequence to synchronize the actor's lip motions to the new soundtrack. It uses computer-vision techniques to track points on the speaker's mouth in the training footage, and morphing techniques to combine these mouth gestures into the final video sequence. The new video combines the dynamics of the original actor's articulations with the mannerisms and setting dictated by the background footage. Ezzat et al. [81] provided a novel speech-driven method. It first records a human subject using a video camera as he/she utters a predetermined speech corpus. After processing the corpus automatically, a visual speech module is learned from the data that is capable of synthesizing the human subject's mouth uttering entirely novel utterances that were not recorded in the original video. The synthesized utterance is re-composited onto a background sequence which contains natural head and eye movement. At runtime, the input to the system can be either real audio sequences or synthetic audio produced by a text-to-speech system, as long as they have been phonetically aligned. Voice puppetry [48] is a fully automatic technique for creating novel speech animation from an example. Computer vision is utilized to track the facial features of a human demonstrator who reads a predefined script. This motion data are then learned using a hidden Markov model or HMM which represents the probabilities of transitions between different facial poses and velocities. Cao and colleagues [59] have also presented a method for speech motion editing through automatically discovered parameters. This is made possible through

Figure 1.55 Face sketch synthesis results.

manifold learning. Independent component analysis or ICA (a popular technique in machine learning and statistics [190]) is used to create parameters for the regression of speech motion.

Motion capture data can also be applied to drive the human face. Zhang et al. [348] proposed a geometry-driven facial expression synthesis system. Based on the point positions of a facial expression, this system automatically synthesizes a corresponding expression image that includes photorealistic and natural looking expression details. A technique is developed to infer the missing feature point motions from the tracked

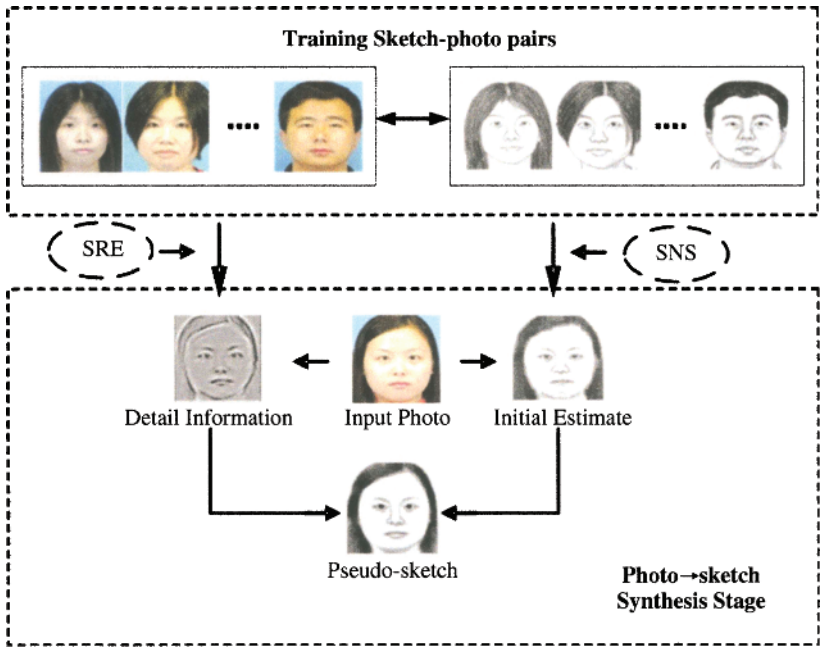

Figure 1.56 Framework of the proposed SNS-SRE image synthesis algorithm.

subset by using an example-based approach. The details of the technique are shown in Figure 1.57.

Figure 1.57 (a) Feature points; (b) Face region subdivision.

1.3.5.2 3D face synthesis Pighin et al. [230] presented a new technique for creating photorealistic textured 3D facial models from photographs of a human subject, and for creating smooth transitions between different facial expressions by morphing between these different models. In this method, a scattered data interpolation technique is used to deform a generic face mesh to fit the particular geometry of the subject's face. Similarly, interpolation methods have been applied in [224, 20]. The results of interpolation is presented in Figure 1.58. The Pose Space Deformation(PSD) method presented by Lewis et al.[169] provides a general framework for example-based interpolation which can be used for blendshape facial animations. In their work, the deformation of a surface (face) is treated as a function of a set of abstract parameters, and a new surface is generated by scattered data interpolations.

Figure 1.58 Linear Interpolation is performed on blend shapes. Left: Neutral pose. Middle: Interpolated shape. Right: "A" mouth shape.

Sifakis et al. [258] proposed an anatomically accurate face model controlled by muscle activations and kinematic bone degrees of freedom. The tissues are endowed with a highly nonlinear constitutive model including controllable anisotropic muscle activations based on fiber directions. A viable solution is proposed to automatically determine muscle activations which track a sparse set of surface landmarks. The scheme of this method is shown in Figure 1.59. Tao et al. [283] proposed a decoupled probabilistic algorithm calledBayesian tensor analysis (BTA). Theoretically, BTA can automatically and suitably determine dimensionality for different modalities of tensor data. With BTA, a collection of 3D faces can be well modeled. Empirical studies on expression retargeting also justify the advantages of BTA. The representation of the tensor face is shown in Figure 1.60.

Data-driven methods have also been widely applied in 3D face synthesis. King and Parent [138] proposed a facial model designed primarily to support animated speech. This facial model takes facial geometry as the input and transforms it into a parametric deformable model. According to this approach, the facial and coarticulation models must first be interactively initialized. The system then automatically creates accurate real-time animated speech from the input text. It is capable of cheaply producing tremendous amounts of animated speech with very low resource requirements. The results are shown in Figure 1.61. Deng and Neumann [75] proposed a novel data-driven animation system for expressive facial animation synthesis

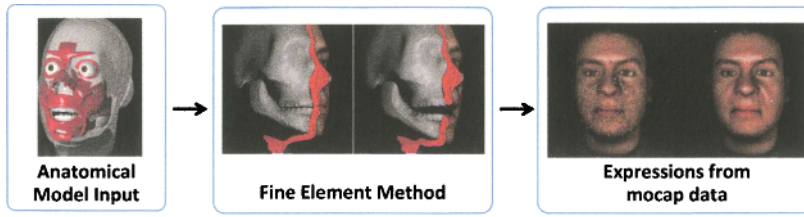

Figure 1.59 Facial expression created by the action of 32 transversely isotropic muscles and simulated on a quasistatic finite element tetrahedral mesh. The original markers are colored red, and the marker positions resulting from the simulation are depicted in green.

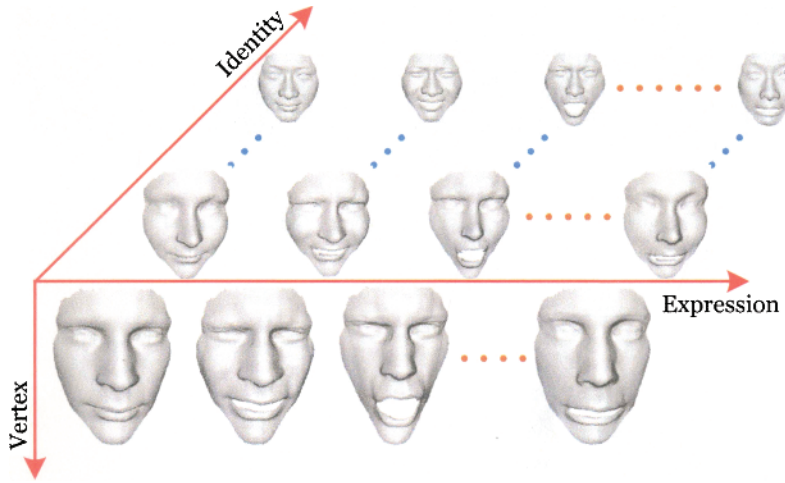

Figure 1.60 A collection of 3D facial data with different identities and different expressions.

and editing. Given novel phoneme-aligned speech input and its emotion modifiers (specifications), this system automatically generates expressive facial animation by concatenating captured motion data while animators establish constraints and goals. Ju and Lee [130] proposed a technique for generating subtle expressive facial gestures (facial expressions and head motion) semi-automatically from motion capture data. This approach is based on Markov random fields that are simulated in two levels. In the lower level, the coordinated movements of facial features are captured, parameterized, and transferred to synthetic faces using basis shapes. The results are shown in Figure 1.62. To achieve real-time motion retargeting, Weise et al. [318] proposed a complete integrated system for live facial puppetry that enables high-resolution real-time facial expression tracking with transfer to another person's face. The system utilizes a real-time structured light scanner that provides dense 3D data and texture. The motion retargeting result is shown in Figure 1.63.

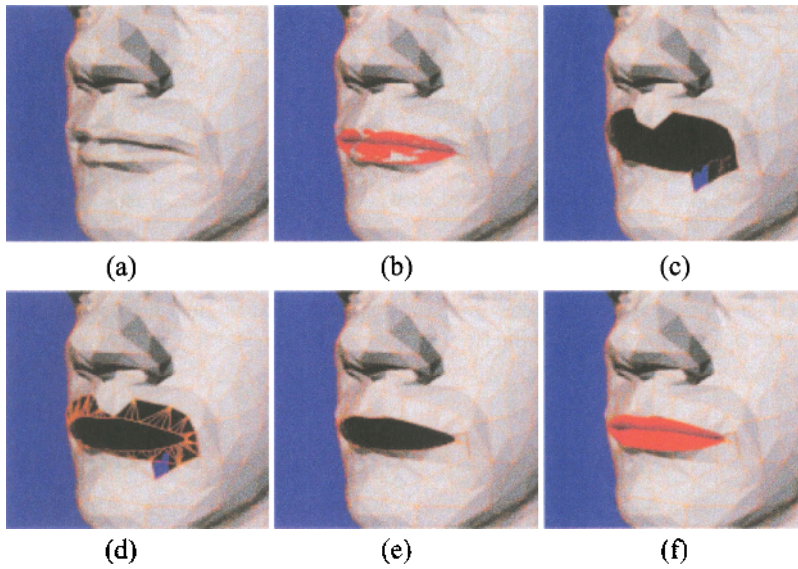

Figure 1.61 The grafting process. (a) The geometry inputting. (b) Lip model fitting. (c) Overlapping triangles removing. (d) The boundary of the lip model and the boundary of the removed triangles are retriangulated. (e) Adding new triangles. (f) The lip model geometry is added to the input geometry.

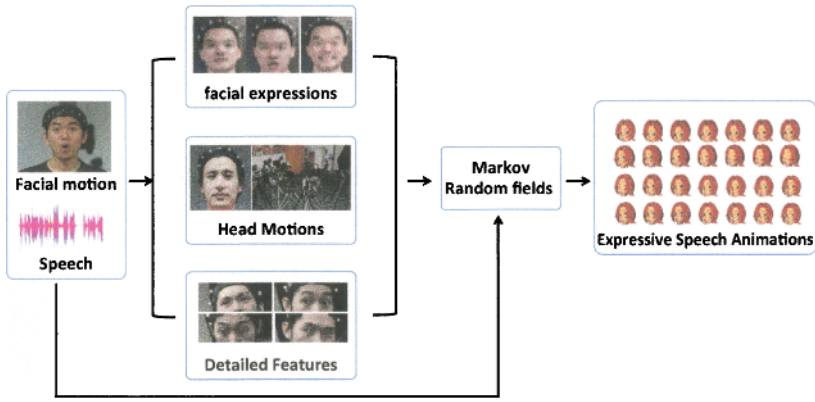

Figure 1.62 Expressive facial expressions and head motions are captured, parameterized, and transferred to the synthetic face.

Lau et al. [156] proposed an intuitive and easy-to-use system for interactively posing 3D facial expressions. The user can model and edit facial expressions by drawing freeform strokes, by specifying distances between facial points, by incre-

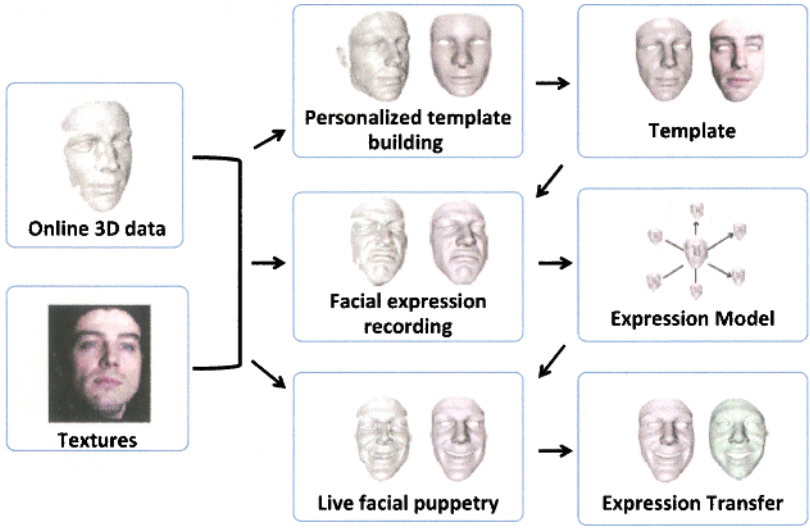

Figure 1.63 Accurate 3D facial expressions can be animated and transferred in real-time from a tracked actor to a different face.

mentally editing curves on the face, or by directly dragging facial points in 2D screen space. This system is shown in Figure 1.64. Na and Jung [206] proposed a novel technique for retargeting captured facial animation to new facial models. Dense motion data are used to express fine motions such as wrinkles. A normal mesh, which is a special multiresolution mesh, is then used to represent the source and target models. A normal mesh is defined by the base mesh and the sequence of its normal offsets. The retargeting consists of two steps: base mesh and detail mesh retargeting. For base mesh retargeting, an example-based technique is used to take advantage of the intuition of the user in specifying the similarity between the source and target expressions. In detail mesh retargeting, the variations of normal offsets in the source mesh are hierarchically computed and transferred to the target mesh.

Reconstruction of a 3D face model from a single 2D face image is fundamentally important for face recognition and animation because the 3D face model is invariant to changes of viewpoint, illumination, background clutter, and occlusions. Given a coupled training set that contains pairs of 2D faces and the corresponding 3D faces, Song et al. [263] train a novel coupled radial basis function network (C-RBF) to recover the 3D face model from a single 2D face image. The C-RBF network explores: (1) The intrinsic representations of 3D face models and those of 2D face images; (2) mappings between a 3D face model and its intrinsic representation; and (3) mappings between a 2D face image and its intrinsic representation. Since a particular face can be reconstructed by its nearest neighbors, it can be assumed that the linear combination coefficients for a particular 2D face image reconstruction are identical to those for the corresponding 3D face model reconstruction. Therefore, a

3D face model can be reconstructed by using a single 2D face image based on the C-RBF network. The whole framework is presented in Figure 1.65.

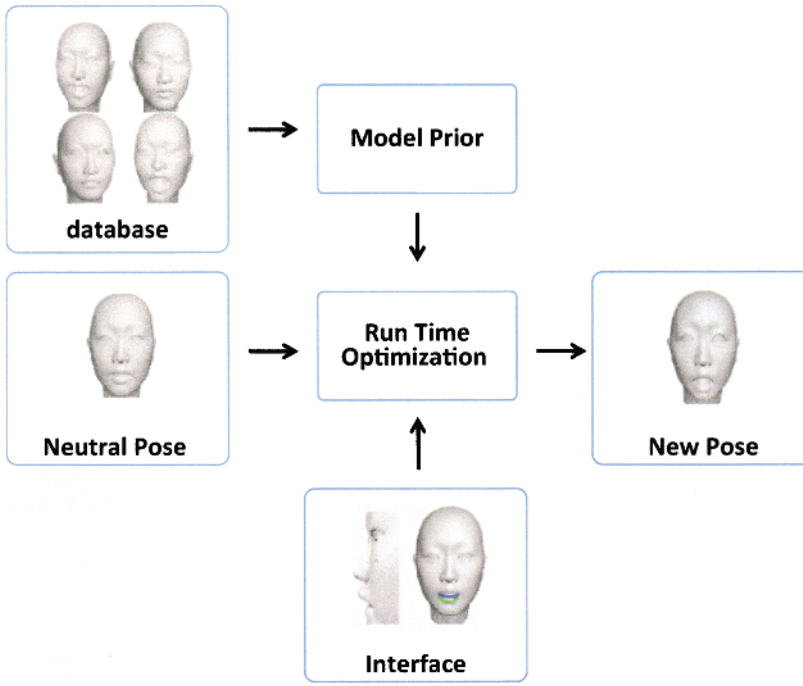

Figure 1.64 The face poser system allows intuitive modeling and editing of 3D facial expressions in real time.

1.4 CHAPTER SUMMARY

Many computer animation techniques are both data and computationally intensive. In this chapter, we have introduced how some machine learning methods or concepts are utilized to help alleviate these bottlenecks in animation techniques. However, machine learning has not as yet been used widely throughout the field of computer animation. We discussed in this chapter how animation research can leverage the machine learning literature to underpin, validate, and develop their proposed methods. A close relationship between computer animation and learning techniques is proposed, which will result in the development of new and enhanced animation techniques. In the following chapters, we shall introduce the emerging machine learning techniques and their applications in computer animation fields.

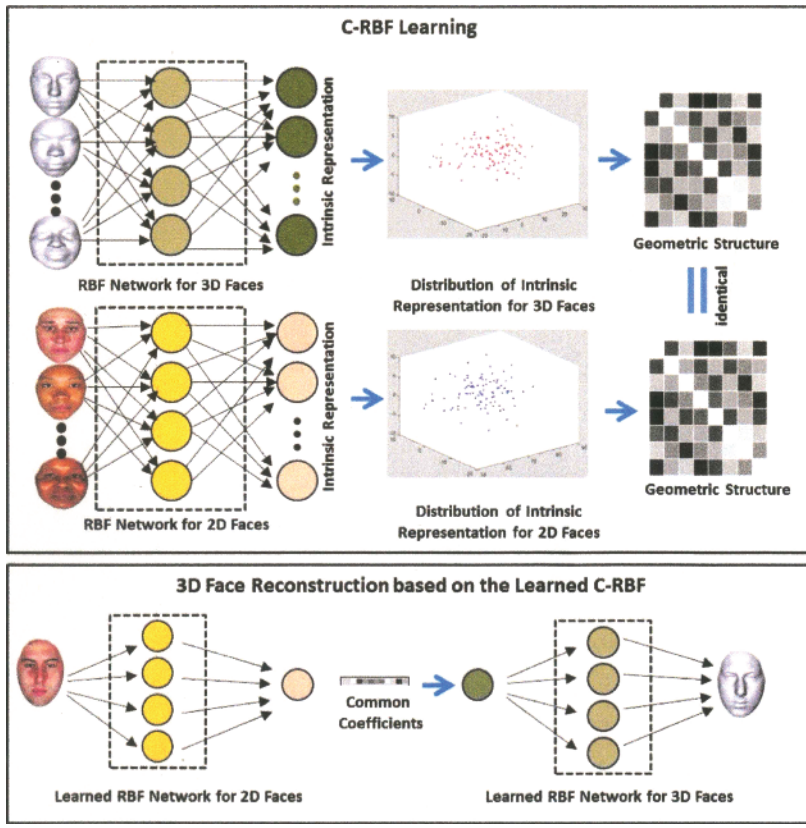

Figure 1.65 Framework of the C-RBF network.