

Style Translation for Human Motion

Eugene Hsu¹Kari Pulli^{1,2}Jovan Popović¹¹Massachusetts Institute of Technology²Nokia Research Center

Figure 1: Our style translation system transforms a normal walk (TOP) into a sneaky crouch (MIDDLE) and a sideways shuffle (BOTTOM).

Abstract

Style translation is the process of transforming an input motion into a new style while preserving its original content. This problem is motivated by the needs of interactive applications, which require rapid processing of captured performances. Our solution learns to translate by analyzing differences between performances of the same content in input and output styles. It relies on a novel correspondence algorithm to align motions, and a linear time-invariant model to represent stylistic differences. Once the model is estimated with system identification, our system is capable of translating streaming input with simple linear operations at each frame.

CR Categories: I.3.7 [Computer Graphics]: Three Dimensional Graphics and Realism—Animation

Keywords: Human Simulation, Data Mining, Machine Learning

1 Introduction

Style is a vital component of character animation. In the context of human speech, the delivery of a phrase greatly affects its meaning. The same can be said for human motion. Even for basic actions such as locomotion, the difference between a graceful strut and a defeated limp has a large impact on the tone of the final animation. Consequently, many applications of human animation, such as feature films and video games, often require large data sets that contain many possible combinations of actions and styles. Building such data sets places a great burden on the actors that perform the motions and the technicians that process them.

Style translation alleviates these issues by enabling rapid transformation of human motion into different styles, while preserving content. This allows numerous applications. A database of normal locomotions, for instance, could be translated into crouching and limping styles while retaining subtle content variations such as turns and pauses. Our system can also be used to extend the capabilities of techniques that rearrange motion capture clips to generate novel content. Additionally, our style translation method can quickly process streaming motion data, allowing its use in interactive applications that demand time and space efficiency.

It is often difficult to describe the desired translation procedurally; thus, an important objective of this work is to infer the appropriate translations from examples. A style translation model is trained by providing matching motions in an input and an output style. To learn the translation from a normal to a limping gait, for instance, one might pair three steps of a normal gait with three steps of a limping one. Their content is identical, but their style comprises the spatial and temporal variations. In practice, such examples can be easily selected from longer performances.

The relationship between an input style and an output style is described by a linear time-invariant (LTI) model [Ljung 1999]. Unlike previous techniques, which interpret content with discrete states or constraints (e.g., [Brand and Hertzmann 2000; Giese and Poggio 2000; Kovar et al. 2002a; Li et al. 2002]), the LTI model describes the translation without an explicit content model. With this deliberate choice, style translation preserves details in the original motion, such as lifting a leg to avoid an obstacle. These details might otherwise disappear with a discrete content interpretation.

Learning a translation model requires the correct alignment of matching motions. In particular, it is important to identify which parts of the motion in the input style map to which parts of the motion in the output style. Previous alignment approaches apply a time warp that minimizes the numerical differences between individual poses. This works well for identical or similar styles; however, two motions in different styles typically contain very different poses. Instead of relying on heuristic or manual alignment, we propose iterative motion warping (IMW), which automatically computes dense correspondences between stylistically different motions (§3). The aligned motions are used to estimate the translation model with system identification techniques.

After estimation, a translation model can receive motions in the input style and produce translated motions in the output style (§4). Without affecting style, our motion representation removes global position and orientation and exposes time-invariant stylistic differ-

ences. Dense correspondences allow the input-output relationship to be described at the timing rate of the input, thus guaranteeing perfect synchrony. The combined usage of LTI models and existing postprocessing techniques allow our system to translate with constant latency and memory usage.

There are situations in which specific details of new input motions are not explored adequately by provided examples, yielding poorly-trained translation models. We describe a simple and effective heuristic that compensates for this issue. Additionally, since our model does not account for kinematic interaction with the environment, the raw output may contain small visual artifacts, such as feet sliding on the ground. We demonstrate an automatic annotation method that can be used to correct such artifacts.

2 Related Work

The intent of style translation is most similar to the goal of a style machine [Brand and Hertzmann 2000], a general parametric representation for encoding style and content in example motions. After training, a style machine can generate random motions in various styles, or apply such styles to novel inputs. Similar tasks could also be performed with non-parametric methods [Giese and Poggio 2000] by blending between the examples. Unlike our approach, however, both methods explicitly model the content in motion, either by storing examples verbatim, or by learning the parameters of a generative model. As a result, such models often require explicit frame correspondences to be solved during the translation process. We describe a more compact and more efficient translation method that circumvents the need for explicit content modeling.

Our solution draws its main inspiration from retargeting methods, which preserve content, but transform motion to adjust for skeleton size [Gleicher 1998], kinematic constraints [Bruderlin and Williams 1995; Witkin and Popović 1995], or physics [Popović and Witkin 1999]. Style, however, is often difficult to encode in the procedural forms required by these methods. This was addressed by Amaya et al. [1996], who proposed a framework to discover emotional transformations from data. This work inspired us to model style by examining differences between example motions.

Statistical analysis of input-output relationships is a fundamental tool in nearly all areas of science and engineering. In computer graphics, such relationships have been used to translate styles of line drawings [Freeman et al. 2003], complete image analogies [Hertzmann et al. 2001], and so on. Our method is the result of the comprehensive application of these principles to style translation.

2.1 Content Modeling

Database approaches assemble short subsequences to synthesize motions that match user specifications, such as a desired path on the ground or a list of actions to perform [Kovar et al. 2002a; Lee et al. 2002; Arikan et al. 2003; Arikan and Forsyth 2002]. As a result, the output will have the style of the examples in the database, but the style cannot be applied to new motions. Pullen and Bregler resolve this problem by retaining low-frequency components of the input and adding high-frequency texture from the database [2002]. This approach works well when style variations are contained in the high-frequency spectrum. However, many types of stylistic variations, such as those between normal, limping, and crouching gaits, will occur in all bands of the spectrum.

Blending and morphing approaches extrapolate from motion sequences with linear combinations [Rose et al. 1998; Giese and Poggio 2000; Ilg and Giese 2002; Kovar and Gleicher 2003]. Such techniques allow for flexible synthesis of stylized motion, often with intuitive and tunable parameters. Furthermore, they can be used effectively for control from intuitive inputs [Park et al. 2004]. Applying these approaches to our problem is more difficult, as style

translation is a task of control by performance, not by constraints or parameters. As a result, such methods would have to infer the appropriate controls from input, thus adding a level of indirection that can mask detail and add algorithmic complexity.

Parametric approaches postulate a generative model of motion. Li et al. use linear dynamic systems [2002], while Brand and Hertzmann rely on a hidden Markov model [2000]. Such general models of content would be needed to enhance a novice dance performance by removing mistakes, such as stumbles. In our work, we consider those stumbles as part of the content, rather than the style. Furthermore, if the translation were to map the input onto a sequence of discrete states, subtle details of the input motion could be lost.

2.2 Retargeting

Retargeting methods [Bruderlin and Williams 1995; Witkin and Popović 1995] have been shown to yield compelling and nuanced motions in interactive applications [Lee and Shin 1999; Shin et al. 2001]. In particular, the frequency filtering approach of Bruderlin and Williams [1995] demonstrates that certain stylistic qualities, such as exaggeration and nervousness, can be manipulated by adjusting frequency bands with a visual equalizer interface. Unuma et al. also propose frequency control of style [1995]. These methods rely on procedural definitions of desired properties, or on explicit user input. While procedural formulations could be derived [Perlin 1995], it is often quicker and easier to define styles by performance.

Style-based inverse kinematics uses such an approach to infer a model of natural human poses [Grochow et al. 2004]. This method was primarily designed to simplify the process of creating static human poses, but was also shown to be capable of generating stylistically consistent animations. Although the model forgoes temporal modeling, which is essential for effective style translation, its results emphasize that a full model of motion content is not required for compelling results.

Our technique bears conceptual similarities to the framework proposed by Amaya et al. for emotional transformations [1996]. They also build models of stylistic variation that, after a correspondence process, are able to translate structure and content. However, they take a significantly different algorithmic approach. These differences are based on assumptions about the provided examples. While they aim to build translation models that can, for instance, transfer the quality of anger to a kick when provided with an example of an angry drinking motion, we assume that the examples have greater coverage of the desired space of translations. As a result, we are able to employ more powerful translation models.

2.3 Implicit Mapping

Implicit mapping approaches examine the relationship between two signals to learn the mapping between them. Dontcheva et al. learn the mapping between a low-dimensional performance and a high-dimensional animation [2003]. De La Torre and Black determine the mapping between two high-dimensional motions by modeling dynamics in a common low-dimensional linear subspace [2001]. Jebara and Pentland use Gaussian mixtures to model past-future interaction relationships [1999]. In principle, the same learning techniques (asymmetric coupled component analysis, canonical correlation analysis, and conditional expectation maximization) could be applied to style translation. We chose to use LTI models because their importance has led to development of robust and powerful estimation methods [Ljung 1999; Van Overschee and De Moor 1996]. Furthermore, their extensive use in control theory and modeling of physical processes makes it easier to improve style translation with better control and more general models. Still, the success of any learning method is highly dependent on proper correspondence, which is a major contribution of our work.

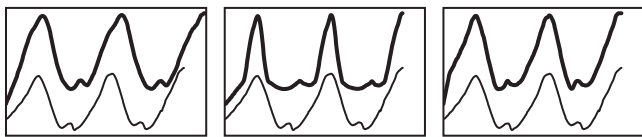


Figure 2: (LEFT) Hip-angle curves from normal and crouching walks. (CENTER) DTW warps the top curve to the bottom curve. Notice the excessive widening of valleys and the mismatched local extrema. (RIGHT) IMW gives a more natural solution.

3 Correspondence

Proper correspondences are vital to building style translation models, as well as many other motion-processing algorithms such as blending and linear analysis. However, most existing techniques are challenged when applied to placing *stylistically different* motions into correspondence, as they rely on the premise that stylistically similar frames are numerically similar as well. This can lead to undesirable artifacts, as shown in Figure 2. Kovar and Gleicher noted this issue as well, and presented a solution that can be used for large, well-sampled data sets [2004]. Other techniques compute sparse correspondences by relying on annotations such as footplants (e.g., [Park et al. 2002]). In this section, we present a novel correspondence algorithm that has no such requirements.

Our approach is fundamentally motivated by motion warping [Witkin and Popović 1995], which demonstrates that many variations in motion (stylistic and otherwise) can be modeled by smooth spatial distortions of individual degrees of freedom. Applied to correspondence, this principle yields an intuitive objective: given two motions, apply spatial and temporal warps to the first to bring it as close as possible to the second. This is subject to the soft constraint of smoothness in the spatial warp and the hard constraint of monotonicity in the temporal warp.

In the following discussion, we focus on motions in one dimension and return to the multidimensional case in §3.3. Given two motions \mathbf{u} and \mathbf{y} , our iterative motion warping algorithm (IMW) finds a correspondence that minimizes the following objective function:

$$E(\mathbf{a}, \mathbf{b}, \mathbf{W}) \equiv \|\mathbf{W}(\mathbf{U}\mathbf{a} + \mathbf{b}) - \mathbf{y}\|^2 + \|\mathbf{F}\mathbf{a}\|^2 + \|\mathbf{G}\mathbf{b}\|^2. \quad (1)$$

Here, \mathbf{U} is $\text{diag}(\mathbf{u})$, and $\mathbf{U}\mathbf{a} + \mathbf{b}$ performs a spatial warp using the *scale vector* \mathbf{a} and the *offset vector* \mathbf{b} . The *warp matrix* \mathbf{W} performs a nonuniform time warp, which is constrained to be monotonically increasing. The particular structure of this matrix will be developed in the following section. The terms $\|\mathbf{F}\mathbf{a}\|^2$ and $\|\mathbf{G}\mathbf{b}\|^2$ measure the smoothness of \mathbf{a} and \mathbf{b} , respectively. More specifically, \mathbf{F} and \mathbf{G} provide weighted finite-difference approximations to the first derivatives of \mathbf{a} and \mathbf{b} , where the weights are chosen to balance their contribution to the objective.

IMW minimizes E by coordinate descent; specifically, it alternates global optimizations of \mathbf{W} and $\{\mathbf{a}, \mathbf{b}\}$ until convergence. The discrete global optimization of \mathbf{W} is performed by an algorithm that is conceptually similar to dynamic time warping (DTW) [Rabiner and Juang 1993]. Our formulation differs algorithmically to enable interoperability with the continuous optimization of \mathbf{a} and \mathbf{b} . We detail these alternating *time-warp* and *space-warp* stages in §3.1 and §3.2, and describe their combination in §3.3.

3.1 Time Warp

In this section, we describe the structure and optimization of the warp matrix \mathbf{W} , independently of the scale and offset vectors \mathbf{a} and \mathbf{b} . We call this component the time-warp stage. The optimization is performed using dynamic programming, in a manner very similar to approximate string matching. This problem is defined as follows.

Given two strings $\mathbf{p} = [p_1 \dots p_m]$ and $\mathbf{q} = [q_1 \dots q_n]$ (where p_i and q_j belong to the same alphabet), find an optimal *edit* ϕ of \mathbf{p} to minimize its difference from \mathbf{q} . Formally stated,

$$\min_{\phi} \sum_j c(\phi_j(\mathbf{p}), q_j), \quad (2)$$

where $\phi_j(\mathbf{p})$ is the j th character of the edited \mathbf{p} , and c is a cost function, typically defined to be 0 if its arguments are identical and 1 otherwise. Generally, the edit ϕ is allowed to perform insertions, deletions, and so on. We restrict it to repetition: it can replace a single character α with α^r , $1 \leq r \leq k$, where k is the *repetition limit*. Although this objective differs from those typically given for approximate string matching, its optimization can be performed with a minor modification of the dynamic programming recurrences.

This can be used for one-dimensional motion correspondence with several observations. Given two motions \mathbf{p} and \mathbf{q} , represented as column vectors, each element can be considered a character that belongs to the alphabet of reals. A reasonable choice of the cost function c , then, would be the squared difference. Furthermore, the edit function ϕ can be fully parameterized by an integer *repetition vector* \mathbf{r} , where r_i indicates the number of times to repeat p_i . This enables a linear algebraic representation of the objective:

$$\min_{\mathbf{R}} \|\mathbf{R}\mathbf{p} - \mathbf{q}\|^2 \equiv \min_{\mathbf{r}} \left\| \begin{bmatrix} \mathbf{1}_{r_1} & & & \\ & \mathbf{1}_{r_2} & & \\ & & \dots & \\ & & & \mathbf{1}_{r_m} \end{bmatrix} \mathbf{p} - \mathbf{q} \right\|^2. \quad (3)$$

Here, $\mathbf{1}_\alpha$ denotes a column vector of α ones. We call \mathbf{R} the *repetition matrix*; intuitively, it is the result of applying \mathbf{r} to the rows of the identity matrix \mathbf{I}_m . The constraints of approximate string matching can be rewritten as $r_i \in \{1, \dots, k\}$ and $\sum_i r_i = n$.

As described, the algorithm lengthens \mathbf{p} to compute the correspondence with \mathbf{q} . It can be applied in more general situations by first lengthening \mathbf{q} by s repetitions of each element. Setting the repetition limit k to s^2 emulates the slope constraints of the standard dynamic time warping algorithm, effectively constraining the slope of the warp to be between $1/s$ and s .

We now return to the IMW objective function in Equation (1). After lengthening \mathbf{y} , we define $\mathbf{p} \equiv \mathbf{U}\mathbf{a} + \mathbf{b}$ and $\mathbf{q} \equiv \mathbf{y}$. It follows from Equation (3) that the described time-warp algorithm performs the desired minimization, giving \mathbf{W} as the repetition matrix \mathbf{R} .

3.2 Space Warp

The result in the previous section provides a global optimization of $E(\mathbf{a}, \mathbf{b}, \mathbf{W})$ with \mathbf{a} and \mathbf{b} fixed. In the space-warp stage, we optimize E with \mathbf{W} fixed. Since the error function is quadratic in \mathbf{a} and \mathbf{b} , a unique global optimum is found by zeroing the first derivative of E with respect to \mathbf{a} and \mathbf{b} . This yields the following normal equations, which can be solved efficiently using Gaussian elimination:

$$\begin{bmatrix} \mathbf{U}^T \mathbf{W}^T \mathbf{W} \mathbf{U} + \mathbf{F}^T \mathbf{F} & \mathbf{U}^T \mathbf{W}^T \mathbf{W} \\ \mathbf{W}^T \mathbf{W} \mathbf{U} & \mathbf{W}^T \mathbf{W} + \mathbf{G}^T \mathbf{G} \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} = \begin{bmatrix} \mathbf{U}^T \mathbf{W}^T \mathbf{y} \\ \mathbf{W}^T \mathbf{y} \end{bmatrix}. \quad (4)$$

3.3 Full Algorithm

The developments of the previous sections describe the steps of iterative motion warping for the local minimization of E . We begin by uniformly lengthening \mathbf{y} by the slope limit s , as prescribed by the time-warp stage. After initializing $\mathbf{a} = \mathbf{1}$ and $\mathbf{b} = \mathbf{0}$, the time-warp and space-warp stages are alternated. Convergence follows from the following two observations: since E only contains quadratic terms, it is lower bounded; since each iteration performs a global optimization, the value of E cannot increase. In this context, it becomes clear why it was necessary to produce \mathbf{W} in the

time-warp stage: without it, the two stages would be unable to cooperate through a common objective function, thus precluding any guarantees of convergence.

So far, we have only considered motions in single dimensions. One could use the algorithm as described on individual degrees of freedom. However, this is undesirable for many applications, including ours. For instance, when placing locomotions into correspondence, it is important that the legs are synchronized in the same time frame. Thus, it is often preferable to couple the time warp over all degrees of freedom. This can be accomplished as follows. Since the time-warp algorithm only depends on having a cost function, it is replaced by the squared norm of the frame difference. The space-warp stage is unmodified and used independently for each degree of freedom. Correctness follows by parity of reasoning.

3.4 Usage

Iterative motion warping is useful for a variety of techniques that use motion data. Of course, as it is somewhat unconventional, it is important to explain how one can substitute it for a standard dynamic time warping algorithm. Given a solution $\{\mathbf{a}, \mathbf{b}, \mathbf{W}\}$, it is unnecessary to retain \mathbf{a} and \mathbf{b} , as they are just auxiliary terms in the optimization. The warp matrix \mathbf{W} alone encodes the desired correspondence. Since it is a repetition matrix, we can extract its repetition vector \mathbf{w} : this can be cached from the time-warp stage, or it can be recomputed as $\text{diag}(\mathbf{W}^T \mathbf{W})$. Recall, however, that this vector encodes a correspondence of \mathbf{u} to the s -lengthened version of \mathbf{y} . To bring it back to normal time, \mathbf{w} is divided by the slope limit s . The resulting \mathbf{w} encodes the *differential time warp*, named as such since *integrating* it yields the familiar time-warp curve. Note that \mathbf{w} is essentially the dense equivalent of Park et al.’s incremental time-warp parameterization [2002].

4 Style Translation

Given a sequence of input frames $\{\mathbf{u}_t\}$ in the input style, our translation system computes a sequence of frames $\{\mathbf{y}_t\}$ in the output style. In addition, translation assumes that the input frames are not available as a single batch, but instead arrive one at a time. Our method uses an LTI model to approximate the relationship between the input and output styles. Once its parameters are estimated from the training data, the model translates new motions with simple linear transformations. The output retains the content and the detail of the original input, but differs in its style of execution. This process is illustrated in Figure 3.

The output motion can contain artifacts, either due to a small training set, or due to interactions between the character and the environment. These can be corrected by a simple but effective postprocess that improves generalization and annotates motions for footskate cleanup and other possible applications.

4.1 Representation

Before modeling, the motions, which are typically encoded in the form of trajectories for each joint angle, are converted into a form that exposes the relevant features and meets the technical assumptions of our model. First, our representation untangles the spatial and temporal variation between matched motions. Second, it uses differential (relative, instead of absolute) parameters. This is commonly done in statistical time-series modeling to meet the stationarity assumptions of LTI models [Brockwell and Davis 2002].

Time. Iterative motion warping (§3) finds a differential parameterization \mathbf{w} of timing differences between motions. This is used to warp examples in the output style to the time frame of their paired

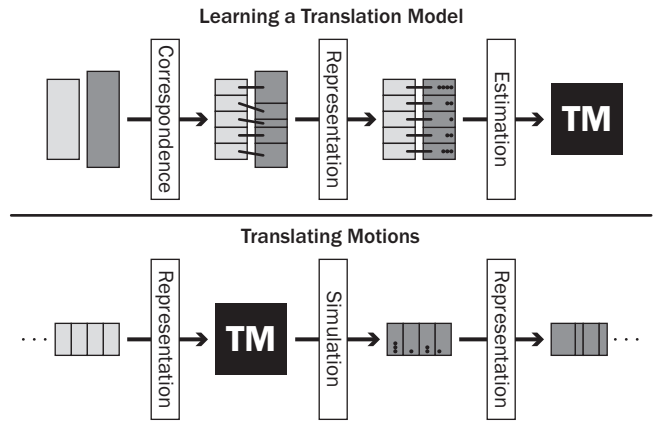


Figure 3: An overview of our style translation system. (TOP) To build a model, correspondences are computed for matching example input-output pairs. During representation, the output is warped to the input, and timing information is added. These frames are used to estimate the translation model. (BOTTOM) New frames of input are properly represented and then simulated by the model, yielding output frames. The representation is inverted for synthesis.

examples in the input style. After this is performed, however, it is necessary to encode the warp so that it can be reversed during synthesis. This is accomplished by attaching elements of \mathbf{w} to their corresponding frames. Since all of these values are positive, they correspond to a monotonically increasing timeline. However, there is no guarantee that the estimated LTI model will produce positive values during synthesis. To remedy this, we store the logarithm of these values instead, which can be freely manipulated by our model. Monotonicity is guaranteed during the inversion of this parameterization because the exponential produces strictly positive values.

Space. Except for the root joint, all joint angles are converted to Cartesian parameters with the exponential-map parameterization [Grassia 1998]. This mapping ensures proper manipulation of non-linear joint-angle quantities by our linear model. The root joint is treated specially because it encodes a transformation with respect to a fixed global coordinate system. As such, it is not invariant to ground-plane translation and vertical-axis rotation. We resolve these problems by encoding the root position in the ground plane and the orientation around the vertical axis relative to its position and orientation in the previous frame. The remaining degrees of freedom, such as height, are preserved. This form is identical to the representation used by Kovar and Gleicher for coordinate-frame alignment [2003]. In addition, we normalize all degrees of freedom to have zero mean and unit variance across all frames. This normalization is performed independently for input and output motions.

4.2 Translation Model

Our style translation system uses a linear time-invariant model to explain the relationship between two motions:

$$\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t + \mathbf{e}_t \quad (5)$$

$$\mathbf{y}_t = \mathbf{C}\mathbf{x}_t + \mathbf{D}\mathbf{u}_t + \mathbf{f}_t \quad (6)$$

These difference equations describe the mapping between an *input* \mathbf{u}_t and an *output* \mathbf{y}_t . The continuous vector \mathbf{x}_t is the *state* of the system, and \mathbf{e}_t and \mathbf{f}_t are *white noise* terms that account for measurement errors. The model parameters determine the mapping between the input and the output; they consist of the dimension of the

state space, the constant *system matrices* $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$, and the constant *covariance matrices* of the noise terms. In style translation, the input \mathbf{u}_t is a d -dimensional vector representing the degrees of freedom for the input frame t . Similarly, the output \mathbf{y}_t is the $(d+1)$ -dimensional vector for the corresponding frame in the output style. The added dimension is the additional time-warp parameter (§4.1).

Our model constrains the input-output mapping to a specific form; namely, it assumes that each joint is independent of the others. This is motivated by the principles that guided the design of motion warping techniques [Witkin and Popović 1995; Bruderlin and Williams 1995], which independently transform joint-angle curves to meet user-specified constraints. In these applications, the aim was to preserve the content of the input motion, despite transformations needed to comply with the *constraints*. In our application, the content must also be preserved, despite transformations needed to change the *style*. Similar applications of LTI models for motions [Li et al. 2002] and dynamic textures [Soatto et al. 2001] have modeled degrees of freedom dependently. Such a decision is desirable when deriving generative models, as they do, but this is not our objective. Indeed, the necessary dimensionality reduction would mask detail and introduce unwanted joint correlations. The independence assumption circumvents these issues while allowing all desired dependencies to be translated from the input.

Algorithmically, the benefit of this decision is that it constrains the system matrices to a specific block structure, which allows us to decompose the large parameter estimation task into a set of much smaller ones. In other words, individual sets of system matrices can be independently estimated for each joint and then combined. By doing so, the number of free model parameters is reduced by an order of magnitude, thus simplifying estimation from small data sets. This enables the use of powerful parameter estimation techniques which automatically infer the dimension of the state space.

One detail that we have omitted so far is the handling of the timing parameter in the output; unlike the joints, there are no directly corresponding degrees of freedom in the input. The appropriate degrees of freedom were chosen by experimentation. We found that the best results were achieved by using the differential position and orientation of the root.

4.3 Estimation

Estimation uses example motions to infer the model parameters: the dimension of the state space, the system matrices, and optionally (because they are not used in translation) the noise covariances. This is a classic system identification task with many standard and robust algorithms [Ljung 1999]. We use the N4SID algorithm [Van Overschee and De Moor 1996] and its implementation in the MATLAB system identification toolbox. Subspace methods like N4SID consist of two steps. First, the input-output data is used to estimate the optimal state sequence. Remarkably, this can be accomplished *without* knowing the system matrices. Second, the system matrices are estimated from the optimal state sequence by solving linear least-squares equations.

For our application, the main advantage of subspace algorithms is ease of use. The only user parameter is the dimensionality of the state space. Even so, the choice can be accomplished by inspecting the singular values of a particular matrix. This process is similar to choosing the dimensionality of the subspace obtained by principal component analysis. Automated heuristics are available; those in MATLAB's N4SID implementation allow estimation of translation models without any user-specified parameters.

System identification routines can provide stable estimates of the dynamical system matrix \mathbf{A} . Practically, this ensures that the translation model recovers quickly from any undesirable state, as the influence of such states will decay exponentially.

4.4 Usage

Using the estimated parameters, our method translates the input by feeding it through a simulation of the model. This process requires setting the initial state \mathbf{x}_0 , which is unknown for an arbitrary input motion. To resolve this issue, we set the initial condition to the steady state $\mathbf{x}_0 = \mathbf{0}$ and rely on the stability of \mathbf{A} to decay any resulting artifacts. This choice can produce visible artifacts, but they quickly disappear after several frames. Depending on the application, these frames can either be ignored or computed heuristically using backward extrapolation.

Given the initial condition, the simulation proceeds in a straightforward manner. As each input frame arrives, it is converted to the proper representation (§4.1): differential root transformations are computed, and degrees of freedom are normalized using the stored input means and variances. This is fed into the LTI model, which produces an output. The steps of the representational transformation are then inverted in the expected manner: reverse the normalization (from stored output means and variances), accumulate the root transformations, and exponentiate the log-differential time-warp parameter.

It is important to emphasize that these operations can be performed in a streaming manner. In other words, each frame of input is translated with no lag or lookahead. Furthermore, the memory requirements are constant, as the state vector compactly represents all relevant information about previous frames. This observation may suggest that style translation can be used in online applications, such as performance-driven animation. Recall, however, that the output of the LTI model includes a time-warp parameter. The application of this parameter would disrupt the synchrony between the performer and the output, thus making the method unsuitable for fully online operation.

This is unfortunately a conceptual limitation of using the time warp. Consider the task of translating an input style of slow limping to an output style of fast walking. In this case, a proper style translation, regardless of algorithm, would attempt to output gait cycles *faster* than they are received. In other words, a live performer would be incapable of keeping up with the translation! In practice, style translation might be used for online applications by simply omitting the application of the time-warp parameter. While this adversely affects the quality of the results, it may be a reasonable compromise if the input and output styles are similarly timed.

4.5 Postprocessing

Given a good training set, our model outputs high-quality motions, but may violate kinematic constraints imposed by the environment. Usually, the most visible artifact is footskate, which can be corrected by existing methods if the footplants are annotated [Kovar et al. 2002b]. In this section, we describe an efficient classifier that learns footplant annotations from labeled examples.

A finite training set cannot fully account for the unexpected during a live performance. For these applications, we describe a heuristic that detects unexpected events and modifies the output to ensure that they are handled gracefully.

Annotation. We use multiple discriminant analysis (MDA) to build a footplant classifier from labeled output examples. For a c -class problem, MDA finds the $(c-1)$ -dimensional projection of labeled feature vectors that maximizes class separation [Duda et al. 2000]. We apply MDA to the estimated state sequences $\{\mathbf{x}_t\}$ of the examples, which are computed during the estimation process (§4.3). Specifically, independent one-dimensional MDA projections are computed for each foot, using classes defined by binary footplant annotations. The projected points in each class are then

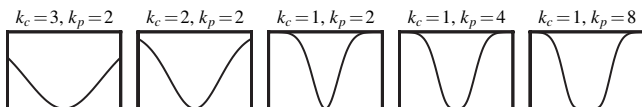


Figure 4: The effect of k_c and k_p on the shape of the blend function. All plots are shown on the same scale and centered at the mean.

modeled by univariate Gaussians, yielding point decision boundaries which are stored in the style translation model, along with the MDA projections, for later use. New footplant classifications are then performed during usage (§4.4) by dot products of the current state \mathbf{x}_t with the stored MDA projections, followed by comparison with the stored decision boundaries. Note that this technique makes the naive and incorrect assumption that footplants are independent. Despite this fact, it achieves good results in practice.

While our usage of MDA is limited to footplant detection, it can be applied to other annotation tasks as well. Furthermore, it does not add any lag or lookahead to the translation process, as it only uses the current state of the LTI model. Note, however, that the application of postprocessing techniques that use these annotations will often introduce latency; in the case of footskate cleanup, the added latency is constant.

Heuristic Generalization. Given a motion outside the space explored in the training examples, our core translation method may output motions with additional artifacts. The universal solution, of course, is to expand the training set, but this is often very inconvenient. We describe an extension that uses a simple heuristic to detect such events and compensates by blending the original input into the output of the LTI model when such events occur. This approach reasons that, if the translation cannot both preserve the content and translate the style, then it should prioritize the preservation of the content.

The detection step uses univariate Gaussians (per degree of freedom) to describe the space of input configurations that can be translated effectively. Recall that these were also used to normalize each degree of freedom (§4.1). Given a new value θ_i for degree of freedom i , the (Gauss) error function provides a reasonable heuristic estimate of translation quality:

$$z(\theta_i, \mu_i, \sigma_i) \equiv \operatorname{erf}\left(\frac{|\theta_i - \mu_i|}{\sigma_i \sqrt{2}}\right) = \int_{-|\theta_i|}^{|\theta_i|} p(x; \mu_i, \sigma_i^2) dx. \quad (7)$$

In this expression, the mean μ_i and the variance σ_i^2 define the Gaussian distribution for the i th degree of freedom. The unit-valued error function $z \in [0, 1)$ is then used to compute a blending weight:

$$b_i(k_c, k_p) \equiv z(\theta_i, \mu_i, k_c \sigma_i)^{k_p}. \quad (8)$$

Here, k_c defines the *confidence* that the translation will perform well on unexpected motions, and k_p controls the *preference* of the blend in ambiguous situations. This is illustrated in Figure 4. We compute the final blend value as $b_* \equiv \max_i b_i(k_c, k_p)$, which weights the blend of the original input into the output of the LTI model.

5 Results

In all our evaluations, we acquired data in a motion capture lab and processed it with standard commercial tools. All motions were recorded at 120 frames per second and mapped to a skeleton with 35 degrees of freedom.

5.1 Correspondence

In this section, we evaluate iterative motion warping, independently of style translation. Our data set contained various stylized locomotions: *normal*, *limp*, *crouch*, *sneak*, *flap*, *proud*, *march*, *catwalk*, and *inverse*. Each style was performed at three speeds: *slow*, *medium*, and *fast*. A short gait cycle was extracted from each motion. Note that IMW, like DTW, requires matched motions to have similar start and end frames.

Using IMW, we aligned the extracted motions with the *normal medium* walk, with a slope limit of $s = 2$. We weighted the smoothness of the scaling and offset terms at 100, but found that these values could be chosen quite flexibly. For comparison, we also performed alignment with our time-warp algorithm alone and two implementations of DTW (using Type I and Type III local constraints [Rabiner and Juang 1993]). The results of these algorithms for styles with large pose variations (such as *crouch*) were different, but equally poor. This was expected, as stylistically similar frames exhibited large numerical differences. In contrast, IMW gave consistently better results. A comparison is shown in Figure 2, and a similar result is shown in the supplemental material.

When applying IMW, we found that it took remarkably few iterations to approach a near-optimal solution (less than ten), and few more for full convergence. To show the quality of correspondences independently of translation, we extracted single gait cycles from the aligned motions, transformed them to the representation described in §4.1, and applied bilinear analysis to separate style and speed [Vasilescu 2002; Tenenbaum and Freeman 2000]. The coordinates of the motions were then interpolated and extrapolated to allow for continuous control of style and speed parameters. We looped the gait cycle to create infinite locomotion. These results are shown in the supplemental material.

5.2 Translation

We applied our system to two style translation tasks. In the first, we translated normal locomotion into a variety of styles. In the second, we translated weak fighting moves into more aggressive ones.

Locomotion. Our locomotion data set included several styles: *normal*, *catwalk*, *crouch*, *limp*, and *sideways*. The examples consisted of gaits that exhibited variations of speed: *slow*, *medium*, *fast*, and turning angles: *hard left*, *soft left*, *straight*, *soft right*, *hard right*. One to two minutes of motion were captured for each style.

Translation models were built from *normal* to other styles by pairing motion clips based on their speed and turning angle. These clips were aligned using iterative motion warping, placed into the proper representation, passed to N4SID for system identification, and so on (Figure 3). This entire process was fully automated, but slow: about an hour, using interpreted MATLAB code. Faster implementations are possible, but our algorithmic choices for model estimation focused on quality, not speed. It is the translation performance that is critical.

This objective was certainly achieved. The preprocessing and postprocessing operations were certainly executed at real-time speeds. Since these operations derive from proven techniques, their high performance should not be surprising. The more important figure is the speed of the LTI model simulation. To evaluate this, we preprocessed a locomotion in batch mode and then fed it through a translation model with a 98-dimensional state space. Averaged over multiple runs, we observed an effective frame rate of 10000 per second on a 2.4 GHz Pentium 4 PC. This clearly indicates that style translation models are fast enough for usage in interactive applications. But more importantly, it implies that they have a negligible performance impact when used in conjunction with existing techniques for constraint satisfaction, motion blending, and so on.

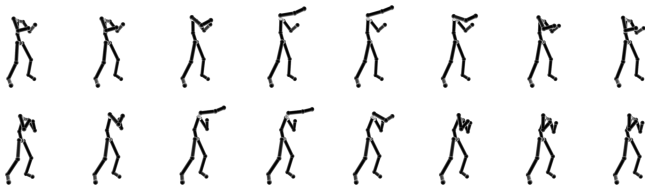


Figure 5: A weak punch (TOP) is translated into a more aggressive one (BOTTOM). Notice the differences in posture and speed.

In our quality evaluations, shown in the accompanying video and in Figure 1, we applied various styles to new normal locomotions. Heuristic generalization was enabled for all tests using identical parameters. Our first evaluation used a walking motion with random changes in speed and direction. This motion was intuitively in the space of the examples. Style translation yielded consistently good results that exhibited transfer of large-scale content and fine detail.

To emphasize the transfer of details, we edited a locomotion to exaggerate the arm swings. Our results show that this detail was transferred in a stylistically consistent manner. Finally, we evaluated heuristic generalization by translating significant deviations from the examples: stepping over an obstacle, ducking, and limboing. Without heuristic generalization, the results of translation were unpredictable. Note, however, that when the input motion returned from the deviation to normal locomotion, translation recovered very rapidly. This was due to the stability of the dynamical system matrix, as discussed in §4.3. With heuristic generalization enabled, translation seamlessly blended in the deviation from the input.

Fighting. Our fighting evaluation demonstrates the versatility of style translation. Here, the goal was to translate punches and kicks of a weak style into an aggressive one. Our examples consisted of 9 kicks (three low, middle, and high kicks for the right leg) and 18 punches (three low, middle, and high punches for each arm) in each style. Using IMW, matching motions were placed into correspondence using identical parameters to those in the locomotion evaluation. These motions were used to estimate a style translation model with a 100-dimensional state space.

We captured a long fighting sequence that contained randomly-ordered punches and kicks in the weak style. The translation model successfully transformed the sequence into the aggressive style: the posture was more defensive and the blows were more forceful. This is shown in Figure 5 and in the accompanying video. The translation speed was approximately equal to that for locomotion. This was to be expected, as the sizes of the state spaces in both evaluations were nearly the same.

6 Conclusion

Our translation system combines proven and novel techniques to produce realistic translations of style in human motion. The new techniques draw on the observation that stylistic variation is well-approximated by the combination of time warp and linear transformations of individual degrees of freedom. Our IMW algorithm uses this combination of time and space warps to compute dense correspondences between stylistically different motions. The alignment process is fully automatic, making our algorithm ideal for computing correspondences between many motions—a task required by many methods in statistical analysis. Similarly, our translation model combines the time-warp representation with the linear operations of the LTI model to translate style independently for each joint. When assembled, these design choices form a fast, effective translation system that can be used for a variety of applications.

Discussion. Style translation can operate on complex inputs; however, the translation itself must be relatively simple. The mapping from normal locomotion to limping locomotion, although difficult for humans to describe mathematically, can be compactly parameterized by several matrices. The same can not be said for all situations in which an input motion maps to an output motion. LTI models would fail, for instance, to parameterize the complex interaction between dance partners, as the mapping itself is quite complex [Hsu et al. 2004]. Similarly, one should not expect LTI models to add or delete choreographic elements [Brand and Hertzmann 2000]. Doing so would contradict our goal of retaining details in the input motion. For such applications, techniques with explicit content models are more appropriate. For style translation, however, LTI models balance modeling capability and simplicity to produce quality results efficiently.

Our system is intended for applications in which motions are specified by performance. This implies that our present system should not be used to generate motions from constraints, such as paths or goals. However, we believe that our system can be extended to support such constraints by incorporating well-researched techniques for the control of LTI models [Stengel 1994].

In the present system, the decision to model joints independently has led to the assumption that the examples of input and output styles share a common skeleton. This restriction prevents its use in computer-puppetry applications, where an actor controls a different digital character [Sturman 1998]. This could be resolved in the future by combining our system with published techniques such as the retargeting approach of Shin et al. [2001].

It was not our objective to supplant existing motion synthesis techniques, but rather to present concepts that could be integrated into past and future work. Iterative motion warping, for instance, could be used to enhance the quality of motion blending techniques, just as we have adapted the insights from previous work in this area. In fact, such techniques can be applied directly to remove another temporary limitation of our system, which, as presented, only translates motions into particular styles. Existing blending techniques can interpolate the output of multiple translation models to generate arbitrary linear combinations of output styles.

Future Work. Our translation model does not generate physically correct motions. As this is a common weakness of most blending and statistical methods, it should not be surprising that physically-based approaches could be used in postprocessing. However, we believe that the simplicity of our translation model provides additional opportunities for integration with spacetime techniques [Popović and Witkin 1999]. This assertion is supported by the extensive history of LTI systems in the modeling of dynamic processes [Luenberger 1979]. Furthermore, the wealth of available control techniques indicates that our translation model could also be used in conjunction with robot controllers [Faloutsos et al. 2001].

This latter connection motivates another fascinating direction that we hope to pursue in future work. Experimental results in biomechanics continue to provide a variety of insights about human motion that physics alone cannot explain. By applying analysis techniques from control theory and other related areas, we hope to find new insights into the nature of style in human motion.

Acknowledgements

We would like to thank the following people for their valuable advice and assistance: Yeuhi Abe, Barbara Cutler, Bill Freeman, Sommer Gentry, Paul Green, Jan Kautz, Zoran Popović, Daniel Vlasic, our anonymous reviewers. Funding for this work was provided by the MIT Oxygen Project. Eugene Hsu was supported by an NVIDIA Graduate Fellowship.

References

- AMAYA, K., BRUDERLIN, A., AND CALVERT, T. 1996. Emotion from motion. In *Graphics Interface '96*, 222–229.
- ARIKAN, O., AND FORSYTH, D. A. 2002. Synthesizing constrained motions from examples. *ACM Transactions on Graphics* 21, 3 (July), 483–490.
- ARIKAN, O., FORSYTH, D. A., AND O'BRIEN, J. F. 2003. Motion synthesis from annotations. *ACM Transactions on Graphics* 22, 3 (July), 402–408.
- BRAND, M., AND HERTZMANN, A. 2000. Style machines. In *Computer Graphics (Proceedings of ACM SIGGRAPH 2000)*, ACM SIGGRAPH, Annual Conference Series, 183–192.
- BROCKWELL, P. J., AND DAVIS, R. A. 2002. *Introduction to Time Series and Forecasting*, 2nd ed. Springer-Verlag.
- BRUDERLIN, A., AND WILLIAMS, L. 1995. Motion signal processing. In *Computer Graphics (Proceedings of SIGGRAPH 95)*, ACM SIGGRAPH, Annual Conference Series, 97–104.
- DE LA TORRE, F., AND BLACK, M. 2001. Dynamic coupled component analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 643–650.
- DONTCHEVA, M., YNGVE, G., AND POPOVIĆ, Z. 2003. Layered acting for character animation. *ACM Transactions on Graphics* 22, 3 (July), 409–416.
- DUDA, R. O., HART, P. E., AND STORK, D. G. 2000. *Pattern Classification*, 2nd ed. John Wiley & Sons, Inc., New York.
- FALOUTSOS, P., VAN DE PANNE, M., AND TERZOPOULOS, D. 2001. Composable controllers for physics-based character animation. In *Computer Graphics (Proceedings of ACM SIGGRAPH 2001)*, ACM SIGGRAPH, Annual Conference Series, 251–260.
- FREEMAN, W. T., TENENBAUM, J. B., AND PASZTOR, E. 2003. Learning style translation for the lines of a drawing. *ACM Transactions on Graphics* 22, 1 (Jan.), 33–46.
- GIESE, M. A., AND POGGIO, T. 2000. Morphable models for the analysis and synthesis of complex motion patterns. *International Journal of Computer Vision* 38, 1 (June), 59–73.
- GLEICHER, M. 1998. Retargeting motion to new characters. In *Computer Graphics (Proceedings of SIGGRAPH 98)*, ACM SIGGRAPH, Annual Conference Series, 33–42.
- GRASSIA, F. S. 1998. Practical parameterization of rotation using the exponential map. *Journal of Graphics Tools* 3, 3, 29–48.
- GROCHOW, K., MARTIN, S. L., HERTZMANN, A., AND POPOVIĆ, Z. 2004. Style-based inverse kinematics. *ACM Transactions on Graphics* 23, 3 (Aug.), 522–531.
- HERTZMANN, A., JACOBS, C. E., OLIVER, N., CURLESS, B., AND SALESIN, D. H. 2001. Image analogies. In *Computer Graphics (Proceedings of SIGGRAPH 2001)*, ACM SIGGRAPH, Annual Conference Series, 327–340.
- HSU, E., GENTRY, S., AND POPOVIĆ, J. 2004. Example-based control of human motion. In *2004 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 69–77.
- ILG, W., AND GIESE, M. A. 2002. Modeling of movement sequences based on hierarchical spatial-temporal correspondence of movement primitives. In *Biologically Motivated Computer Vision*, 528–537.
- JEBARA, T., AND PENTLAND, A. 1999. Action reaction learning: Automatic visual analysis and synthesis of interactive behaviour. In *International Conference on Computer Vision Systems (ICVS)*, vol. 1542, 273–292.
- KOVAR, L., AND GLEICHER, M. 2003. Flexible automatic motion blending with registration curves. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 214–224.
- KOVAR, L., AND GLEICHER, M. 2004. Automated extraction and parameterization of motions in large data sets. *ACM Transactions on Graphics* 23, 3 (Aug.), 559–568.
- KOVAR, L., GLEICHER, M., AND PIGHIN, F. 2002. Motion graphs. *ACM Transactions on Graphics* 21, 3 (July), 473–482.
- KOVAR, L., SCHREINER, J., AND GLEICHER, M. 2002. Footskate cleanup for motion capture editing. In *ACM SIGGRAPH Symposium on Computer Animation*, 97–104.
- LEE, J., AND SHIN, S. Y. 1999. A hierarchical approach to interactive motion editing for human-like figures. In *Computer Graphics (Proceedings of SIGGRAPH 99)*, ACM SIGGRAPH, Annual Conference Series, 39–48.
- LEE, J., CHAI, J., REITSMA, P. S. A., HODGINS, J. K., AND POLLARD, N. S. 2002. Interactive control of avatars animated with human motion data. *ACM Transactions on Graphics* 21, 3 (July), 491–500.
- LI, Y., WANG, T., AND SHUM, H.-Y. 2002. Motion texture: A two-level statistical model for character motion synthesis. *ACM Transactions on Graphics* 21, 3 (July), 465–472.
- LJUNG, L. 1999. *System Identification: Theory for the User*, 2nd ed. Prentice Hall PTR.
- LUNENBERGER, D. G. 1979. *Introduction to Dynamic Systems: Theory, Models, and Applications*, 1st ed. Wiley.
- PARK, S. I., SHIN, H. J., AND SHIN, S. Y. 2002. On-line locomotion generation based on motion blending. In *ACM SIGGRAPH Symposium on Computer Animation*, 105–112.
- PARK, S. I., SHIN, H. J., KIM, T. H., AND SHIN, S. Y. 2004. On-line motion blending for real-time locomotion generation. *Computer Animation and Virtual Worlds* 15, 3–4, 125–138.
- PERLIN, K. 1995. Real time responsive animation with personality. *IEEE Transactions on Visualization and Computer Graphics* 1, 1 (Mar.), 5–15.
- POPOVIĆ, Z., AND WITKIN, A. P. 1999. Physically based motion transformation. In *Computer Graphics (Proceedings of SIGGRAPH 99)*, ACM SIGGRAPH, Annual Conference Series, 11–20.
- PULLEN, K., AND BREGLER, C. 2002. Motion capture assisted animation: Texturing and synthesis. *ACM Transactions on Graphics* 21, 3 (July), 501–508.
- RABINER, L., AND JUANG, B.-H. 1993. *Fundamentals of Speech Recognition*. Prentice Hall, New Jersey.
- ROSE, C., COHEN, M. F., AND BODENHEIMER, B. 1998. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics and Applications* 18, 5, 32–40.
- SHIN, H. J., LEE, J., GLEICHER, M., AND SHIN, S. Y. 2001. Computer puppetry: An importance-based approach. *ACM Transactions on Graphics* 20, 2 (Apr.), 67–94.
- SOATTO, S., DORETTO, G., AND WU, Y. N. 2001. Dynamic textures. In *International Conference on Computer Vision (ICCV)*, 439–446.
- STENGEL, R. F. 1994. *Optimal Control and Estimation*. Dover Books on Advanced Mathematics, New York, NY.
- STURMAN, D. J. 1998. Computer puppetry. *IEEE Computer Graphics and Applications* 18, 1, 38–45.
- TENENBAUM, J. B., AND FREEMAN, W. T. 2000. Separating style and content with bilinear models. *Neural Computation* 12, 1247–1283.
- UNUMA, M., ANJYO, K., AND TEKEUCHI, R. 1995. Fourier principles for emotion-based human figure animation. In *Computer Graphics (Proceedings of SIGGRAPH 95)*, ACM SIGGRAPH, Annual Conference Series, 91–96.
- VAN OVERSCHEE, P., AND DE MOOR, B. 1996. *Subspace Identification for Linear Systems: Theory, Implementation, Applications*. Kluwer Academic Publishers, Dordrecht, Netherlands.
- VASILESCU, M. A. O. 2002. Human motion signatures: analysis, synthesis, recognition. In *International Conference on Pattern Recognition (ICPR)*, vol. 3, 456–460.
- WITKIN, A., AND POPOVIĆ, Z. 1995. Motion warping. In *Computer Graphics (Proceedings of SIGGRAPH 95)*, ACM SIGGRAPH, Annual Conference Series, 105–108.