Natural User Interface for Physics-Based Character Animation

C. Karen Liu[1] and Victor B. Zordan[2]

[1] Georgia Institute of Technology
[2] University of California, Riverside
karenliu@cc.gatech.edu,
vbz@cs.ucr.edu

Abstract. Using natural user interface to interact with digital worlds is becoming commonplace in our daily life, as evidenced by the high demand for Microsoft Kinect since its launch in 2010. However, comparatively little research effort has focused in the past on harnessing these capabilities to create great applications. In this paper, we introduce unified framework for combining natural user interface and physics-based character animation. Our framework takes the form of a continuous controllable space for the combination of the two techniques. We also propose a human-in-the-loop control paradigm which allows a player or performer to sense and act for the character. With the information encapsulated in the human performance, the proposed framework accomplishes its goal in two steps: first, by recognizing and potentially modifying the performance such that it is appropriate for the given scenario; and second, by balancing interactivity and control in order to maintain both physical responsivity to the virtual world and faithfulness to the human performance.

1 Introduction

Character animation is a prevalent mechanism in media for entertainment, training, and socializing. The quality of the experience felt by the media participants often depends on the believability of the avatar's motion in the context of the virtual environment. Non-trivial interaction with the environment is a necessity to make a character appear capable and realistic as well as to allow the character to navigate an interesting world and take action to advance her own dilemma. One method for creating rich, intuitive forms of interaction is through the use of a physically simulation. In particular, a worthy goal is the generation of a physically consistent virtual world, where every aspect of the world is able to be manipulated in an interactive, physically-based manner.

To date, much research work has focused on creating virtual avatars capable of moving and manipulating in a physically simulated world. Most existing control algorithms operate at the mechanical level of the movement (e.g. compute the required joint torques) and assume that cognitive decisions (e.g. to walk or to run) are made by an external decision maker. For applications that involve real-time participants, an equally important but much less explored area is the user

J.M. Allbeck and P. Faloutsos (Eds.): MIG 2011, LNCS 7060, pp. 1–14, 2011.

interface for controlling a physically simulated character. A character typically consists of more than 40 degrees of freedom (DOFs), constrained by the dynamics equations of motion and the kinematic limitations. How we design a user interface to control such a complex, high-dimensional dynamic system remains a daunting task.

Part of the problem seems to be answered by the recent development in motion input hardware, such as Microsoft Kinect. These so called *Natural User Interfaces*, in theory, are more suitable to control physically simulated characters, because they can capture higher dimensional and more expressive user's movements. However, gathering the data from the users motion only addresses the first hardware problem. The second problem, how to use the full-body motion is still very much unexplored.

Integrating real-time user performance with a physically simulated character immediately introduces two major challenges. First, due to discrepancies between the virtual and real world, the real-time performance might be unrealistic (or invalid) when directly applied to the virtual character. For example, if the performer is standing on the floor attempting to control a suspended virtual character hanging by her arms (say, to traverse a monkeybar set), the lower body motion of the performance is clearly a poor fit. Second, if the animation system is physically simulated, the character might not be able to produce the performers motion due to a lack of sophistication in control. As such, physics should only be used as necessary to capitalize on the richness of the human performance. These two challenges suggest that the control scheme must dynamically determine both the appropriate level of sensitivity to the performers motion and the level of physical responsiveness the character has to the virtual environment.

We propose a unified framework for combining natural user interface and physics-based character animation. Our framework takes the form of a continuous controllable space for the combination of the two techniques. The first challenge in this work is to devise a control method capable of combining the potentially conflicting inputs from the user's movements and physics engine. We propose to divide our control along two dimensions. On one dimension, the system can interpret the input from a human user in one of two manners, as a literal performance which is to be played through the character with little or no modification or as a symbolic motion, which is recognized and adjusted to meet the goal-based constraints of the identified behavior. Along the second dimension, the system adjusts the constraints applied to the characters motion from purely kinematic to purely dynamic. In this way, at one extreme, the character can act out the motion perfectly, ignoring internal and external influences such as balance or impacts. At the other extreme, the character will obey physical demands while attempting to accomplish the motion in a physically valid manner. To uphold flexibility, we propose to breakdown the control of the character both spatially and temporally, taking advantage of the best technique for controlling each section of the body, limb, or individual segment (body unit). That is, at any given moment, any body unit of the character is able to respond to the given

situation in an appropriate manner, by following the movement symbolically or literally, dynamically or kinematically as appropriate.

2 Human-in-the-Loop Control

In Figure 1 we show our control space spanned by two axes that precisely address the two challenges mentioned in Section 1. The vertical axis reflects the first challenge on the sensitivity to the real-time user motion. At the bottom of the vertical dimension, the virtual character is extremely sensitive to the performance and literally mimics the entire pose. The control of the virtual character becomes increasingly symbolic and dissociated with the real-time performance as we move upward along the vertical axis. The second challenge defines the horizontal axis along which the responsiveness to the simulated environment varies. At the left extremity, the virtual character is completely kinematically controlled, oblivious to any physical perturbations applied upon her. At the other end of the axis, the motion of the virtual character is fully simulated under Newtonian physics. Here, the performance may be interpreted as desired input but the physics will be the overriding driver of the overall animation of the body. We denote these two dimensions as the literal/symbolic (LS) axis and the kinematic/dynamic (KD) axis.

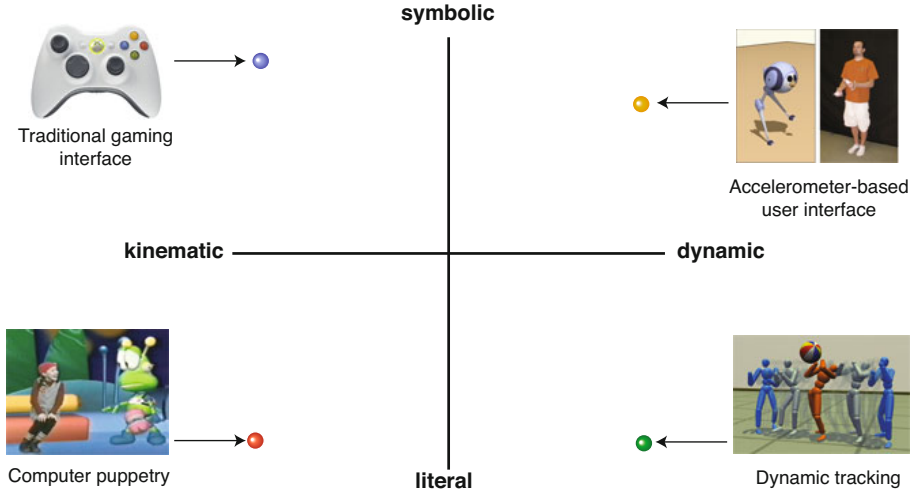

Fig. 1. Unified framework of the proposed KD/LS control space. Select applications highlight aspects of the different subspaces.

This two-dimensional space provides a systematic way to organize existing control methodologies and techniques for online character animation (examples are denoted in figure.) A typical video game controller can be represented by

a point in the upper left corner of this control space, because the simple control interface maps a fully prescribed motion to a symbolic command, such as a button press. A computer puppetry system kinematically controls the virtual character through direct mapping of degrees of freedom between the performer and the virtual character. Shin and colleagues [3] provided an illustrative example of literal and kinematic control located in the bottom left corner of the control space. Researchers have also explored different control schemes on the dynamic side of the space. A dynamically tracking system can be represented as a point in the lower right corner if the input motion is an online performance [7,8]. Researchers have also developed symbolic and dynamic control on simple characters [5] as shown in the upper right corner. Our hypothesis is that all portions of this control space are indeed valuable, including the interior even though the bulk of current applications exist on the periphery.

To fully explore all possible control schemes in the domain, we propose a generic architecture which incorporates real-time user performance with physical simulation, augmented with an example motion database (Figure 2). At each time instance, a real-time pose is processed via pose composition and physical simulation steps. The architecture employs three intelligent modules to select the optimal example pose and control parameters based on the real-time performance and the state of the virtual environment. For pose composition, the system derives an intermediate pose from the input real-time posture and the optimal example pose, modulated by the control parameters in the LS dimension. The example pose is selected from a data set of pre-recorded motions or procedurally generated by inverse kinematics process. In the physics step, based on the control parameters in the KD dimension, the simulator updates the input pose, taking into account external forces and internal torques computed to track the intermediate pose. In this architecture, both the LS and the KD control parameters are determined by intelligence modules which take into account the real-time poses and the virtual environment. In particular, a recognition system matches the real-time performance with known behaviors and supervised learner aids in the selection of the control parameters based on observed training examples. The result is a final pose influenced by a combination of real-time motion and example motion, in conjunction with a modulated physical simulation.

One of the challenges of this architecture is the selection of control parameters. An important observation is that these control parameters should vary not only over time, but also across the body. In a scenario where the performer attempts to reach a virtual object with her hand, the position and orientation of the hand should be generated symbolically (e.g. using IK) in order to precisely match the objects virtual position. However, the rest of the motion can follow the performers movement literally as it provides valid joint configurations and does not interfere with the performers intention in this particular action. Likewise, if the actors arm was incidentally hit by an object, the kinematic and dynamic control could see the same divide in the body. This spatial separation is used for negotiating to produce a balanced motion which is effective but also responsive to external stimuli.

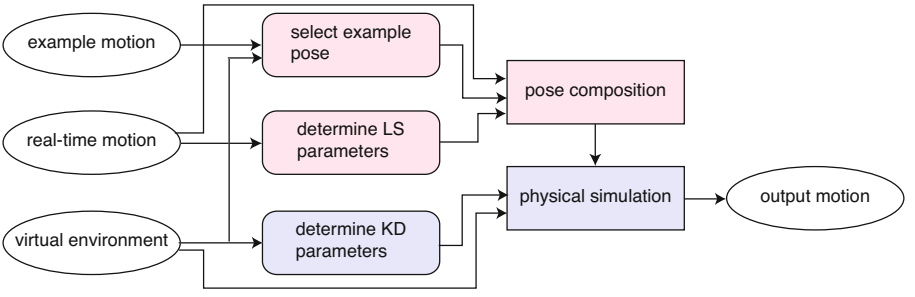

Fig. 2. Control architecture

3 Literal vs. Symbolic Mapping

To establish a mathematically meaningful domain for literal/symbolic control, we need to define a continuous function that maps the control space to the pose space. We first define a point in the control space as $\mathbf{x} \in R^n, \mathbf{0} \leq \mathbf{x} \leq \mathbf{1}$. Each component of $\mathbf{x}$ indicates the LS parameter for a particular body unit, e.g. the left hand. Any monotonic function that satisfies $f(\mathbf{0}) = \mathbf{q}_{rt}$ and $f(\mathbf{1}) = \mathbf{q}_{exp}$ can be a valid mapping between the control space and the pose space, where $\mathbf{q}_{rt}$ and $\mathbf{q}_{exp}$ denote the real-time pose and the example pose respectively. As a first step, we assume $f(\mathbf{x})$ is a simple linear function that interpolates between a real-time pose and an example, *symbolic* pose: $f(\mathbf{x}) = \text{diag}(\mathbf{q}_{rt})(\mathbf{1} - \mathbf{x}) + \text{diag}(\mathbf{q}_{exp})\mathbf{x}$, where $\text{diag}(\mathbf{v})$ is a diagonal matrix which diagonal elements are $v_1, \cdots v_n$.

Evaluating the function f requires solving the following two problems. First, we need to determine the blending factors (LS parameters) of the interpolation for each individual body unit at each time instance. Second, we need to generate a proper example pose $\mathbf{q}_{exp}$ from a dataset instantly based on the performer's action and the current state of the environment. We solve the first problem based on the constraint state of the end-effectors. By comparing the constraint state of the end-effectors on the user against those on the virtual character, we can partition the end-effectors into constrained and unconstrained sets for the user, $\mathcal{S}_c^u$ and $\mathcal{S}_f^u$, as well as for the virtual character $\mathcal{S}_c^{vc}$ and $\mathcal{S}_f^{vc}$. Assuming a body unit is defined as a joint angle, the LS parameters are defined as follows: $x_i = 1$, if joint i is on the branch of an end-effector in $\mathcal{S}_c^u \cap \mathcal{S}_f^{vc}$. Otherwise, $x_i = 0$. For the body units where branches with different LS parameters meet, we assign x_i to an interpolated value. Intuitively, this algorithm follows the user's real-time motion unless the end-effectors of the user are constrained while those of the virtual character are free to move.

The second problem, generating an appropriate pose $\mathbf{q}_{exp}$ from the example motion, requires a recognition algorithm which selects/interpolates a proper example from the motion data base. Further, to be useful in an online setting, this recognition must be conducted in an efficient and timely manner. In particular, to be valuable, a motion example must be retrieved from the database early in a given behavior.

One simple way to recognize the performer's intention is to compare the current pose of the performer against a set of example motion clips, each of which is annotated as a certain action. We propose a simple algorithm that analyzes the correlation of the end effectors of each limb to identify which action the performer intends to imitate. We have implemented a real-time version of this for the game interface system described in Section 5. The proposed algorithm determines whether the performer is imitating a given example motion clip $\mathbf{Q}_k$ based on the performer's current unconstrained end-effectors. For example, if the performer is simply standing on the floor, the end effectors on the feet are constrained while those on the hands are unconstrained. The algorithm concludes that the performer is attempting to perform action k, if the unconstrained end effectors are mimicking the sample motion.

Our system can recognize the mimicking behavior via a simple but robust analysis in low dimensional space, invariant to users' different styles in movements or different skeletal dimensions. We use Principal Component Analysis (PCA) to project the position and the velocity of end effectors to a low dimensional space. Based on the user's current pose $\hat{\mathbf{q}}$, we first identify a set of unconstrained end effectors D. For each pose $\mathbf{q}_i^k$ in the example motion $\mathbf{Q}_k$, we compose a feature vector $\mathbf{w}_i = [\mathbf{x}_i, \mathbf{v}_i]$ which contains positions and velocities of the end effectors in D. We then apply PCA to the example motion represented by its feature matrix, $\mathbf{W} = [\mathbf{w}_1, \cdots, \mathbf{w}_n]$, where n denotes the number of frames in $\mathbf{Q}_k$. The first m principal components constitute a matrix A_k, which is defined to map a feature vector between its original and low-dimensional spaces:

$$\mathbf{u} = A_k^T \mathbf{w} \tag{1}$$

$$\mathbf{w} = A_k \mathbf{u} \tag{2}$$

To compare the similarity of the current user's pose $\hat{\mathbf{q}}$ to the example motion, we project the feature vector $\hat{\mathbf{w}}$ composed from $\hat{\mathbf{q}}$ to the low dimensional space via A_k^T, and back project into the high-dimensional space using A_k:

$$\tilde{\mathbf{w}} = A_k(A_k^T \hat{\mathbf{w}}) \tag{3}$$

Our assumption is that if action k is close to the current performance, the above operation should return a close approximation of the original features. Once the reconstruction error $e_t = ||\hat{\mathbf{w}} - \tilde{\mathbf{w}}||^2$ is obtained, we smooth e_t using a factor α as:

$$E_t = \alpha e_t + (1 - \alpha) E_{t-1} \tag{4}$$

and if E_t is smaller than the threshold value, the intention recognizer will return a positive flag indicating that the behavior has been recognized.

Once we obtain the real-time pose $\mathbf{q}_{rt}$, and determine the example pose $\mathbf{q}_{exp}$ and LS parameters $\mathbf{x}$, an interpolated pose $\hat{\mathbf{q}}$ can be trivially computed via $f(\mathbf{x})$. However, $\hat{\mathbf{q}}$ does not take into account the constraints imposed by the environment. For example, after interpolating a pose in a monkeybar motion, the hand from which the character is swinging might no longer be in contact with the bar.

To solve this issue, each example motion, in addition to joint configurations, can also store a set of kinematic constraints, such as maintaining certain contact points or facing direction while executing the example motion sequence. To compose a pose that actively interacts with the environment, we can then formulate an optimization to solve for the optimal pose that satisfies the environment constraints associated with the example motion and the internal joint limits of the character, while minimizing the difference from the interpolated pose, $\hat{\mathbf{q}}$.

4 Kinematic and Dynamic Continuum

An important concept in the proposed technique is to formalize a continuum between kinematics and dynamics approaches. Our take on this idea is that dynamics, given the proper external forces, can be forced to act like a kinematic system. We exploit this observation to create a Kinematic-Dynamic (KD) continuum between the use of physics for response and remaining faithful to the human performance. We introduce a domain for KD control defined as the continuous space that combines two distinct interpretations of the motion performance, $\hat{\mathbf{q}}$, first as a pure kinematic pose and second as a physical model following this pose as a desired setpoint. We treat the latter as a forward simulation tracking $\hat{\mathbf{q}}$ via a control routine as in [7,6,6]. By employing our KD framework, we sidestep coordinated control in lieu of two naive controllers and instead incorporate intelligence and high-level control through the interactive performance.

Formally, starting with the two interpretations of the performance, one for kinematic playback of $\hat{\mathbf{q}}$ and the other the result of a physical system tracking $\hat{\mathbf{q}}$, we must define a mathematical operation that transforms the input signals into a single KD pose. To unify the two representations, we propose to implement kinematics as a special case of the dynamics where we add additional forces, f_K, to the dynamics which override other dynamics influence of gravity and contact. The result is effectively to drive the dynamics to act kinematically by applying proper values for f_K. Next, we define a point on the KD continuum as $\mathbf{y} \in R^n, \mathbf{0} \leq \mathbf{y} \leq \mathbf{1}$. One component of $\mathbf{y}$ indicates the KD parameter for a single body unit with n bodies. In the KD continuum, we define $\mathbf{y} = \mathbf{1}$ as pure kinematics and $\mathbf{y} = \mathbf{0}$ as pure dynamics. Within our unified framework, the influence of $\mathbf{y}$ on a single body can be interpreted simply by scaling f_K which creates a smooth continuous span between the two extremes. Finally, our choice of the weighting vector, $\mathbf{y}$, will determine the influence felt by each body for each instance in time.

4.1 Kinematic Control

Without loss of generality, we can interpret motion capture playback as a set of accelerations for each body across each timestep. Assigning dynamics parameters for the mass and inertia of each body there is an equivalence between these accelerations and a set of generalized forces which will lead to the analogous accelerations of these bodies. Effects due to gravity and external forces can also

be accounted for within this framework. Indeed, this transformation is equivalent to inverse dynamics. However, rather than performing exact inverse dynamics, we pose a simpler problem which is to derive a set of forces, f_K, which will lead the character to follow the performance despite gravity and contact forces.

To compute f_K, we can propose a simple controller, for example, a Cartesian-based PD-servo as

$$f_K = k_p F(||\hat{p} - p||)(\hat{p} - p) - d_p(\dot{p}) \tag{5}$$

where $\hat{p}$ is the body position derived from forward kinematics using $\hat{q}$ and gains k_p and d_p are manually tuned constants.

In this case, we would likely employ a modified logistic function, or another similar function, to saturate the kinematic forces so they do not go unbounded.

$$F(x) = \frac{1}{1 + e^{-sx}} - \frac{1}{2} \tag{6}$$

where the value of the term s would control how quickly the controller saturates to the maximum. $F(x)$ would limit the bounds of the forces to avoid undesirably large influences and would help to keep the system stable. The values of k_p and d_p must be chosen such that they are large enough to become the dominant influences (overcoming gravity and contact forces) without leading to instability.

4.2 Dynamic Control

Our dynamic controller follows the performance by tracking the joint angles of the motion capture data using internal joint torques. Without the presence of disturbances, these torques would lead the character to match the joint trajectories of the performance. However, the joint tracker does not control the root body and the lack of coordination for balance would lead the character to fall over using a tracking controller alone, simply due to the influences of gravity. Rather than focusing on joint coordination using torques, we propose to employ this tracker in conjunction with f_K described and transform the character from purely dynamic to a hybrid system which combines both balance and responsivity in a simple, controllable fashion.

Specifically, the joint tracker is to compute joint torques based on the tracking error, $\hat{q} - q$, and joint velocities, $\dot{q}$, as follows

$$\tau = k_q F(||\hat{q} - q||)(\hat{q} - q) - d_q(\dot{q}) \tag{7}$$

which follows the pose $\hat{q}$ based on the maximum value, k_q and damping d_q. As in Equation 5, $F(x)$ controls how quickly the controller saturates to k_q following Equation 6. This function allows the joint tracker to resist against deviations from the performance with a substantial torque even for small errors, but will give up stiff control when some maximum value is reached.

4.3 Integrated KD Control

The two KD controllers both guide the character to follow the human performance. However, each has its own characteristics. The forces computed in Equation 5 will maintain global Cartesian positions and will resist external forces in an unrealistic manner, in the limit ignoring them completely to follow the motion kinematically. Instead, the joint torques from Equation 7 make corrections in local joint coordinates and, therefore, will respond in a more believable manner. In contrast, the Cartesian forces provide more precise control from the perspective of the performance and can aid in balancing the full-body character, while the joint torques only provide a limited amount of control and no balance. Our proposed technique is to combine these two complementary inputs in order to get the best of both.

While there are many ways in which we can combine these two signals, we introduces a simple algorithm suitable for many free-standing behaviors. In this case, we propose to join the KD signals within the body based on the proximity of each body part to the ground plane, drawing from the assumption that the ground is the source of the reaction forces which lead to the analogous corrections that are embedded in the forces from Equation 5. In this way, we are using the f_K to make up for the under-actuation of the root seen by the joint tracker. As such, we could propose a simple, temporally changing allocation rule to combine the KD signals and incorporate the following value for $\mathbf{y}$

$$\mathbf{y} = \alpha^{k(t)} \tag{8}$$

where α is the discount applied for each subsequent body in the shortest chain between the body and the ground. k bodies is the count of body parts from the current body and the nearest support, which can change at any instant in time. The support bodies which are in contact with the floor at a given time are assumed to have $\mathbf{y} = \mathbf{1}$ to be able to resist external content and follow the data kinematically. All other bodies are discounted based on α and their body count from the floor. For example if $\alpha = 1/2$ when the character is standing, then the pelvis which is the third body up from the feet feels $1/8$ of the kinematic forces computed. The result would make the character resilient to impacts which hit body parts close to support bodies while the system is also able to track well across the entire body.

5 Applications

To date, the concept of online performance animation is narrowly perceived as a real-time motion reconstruction technique that can be used in commercial applications. However, this limited definition completely ignores the most unique feature of the performance-based approach, that is, the information provided by human intelligence in real-time. In our broader view of performance animation, the performer is not only providing motion trajectories, but also revealing how humans make intelligent decisions and strategic actions based on the external

situations and internal factors, such as biomechanical constraints, physiological states, or personal preferences. To ground the investigation and further discussion, we focus on two generic applications: 1) Performance capture with physical interactions, and 2) Natural user-interface for controlling avatars.

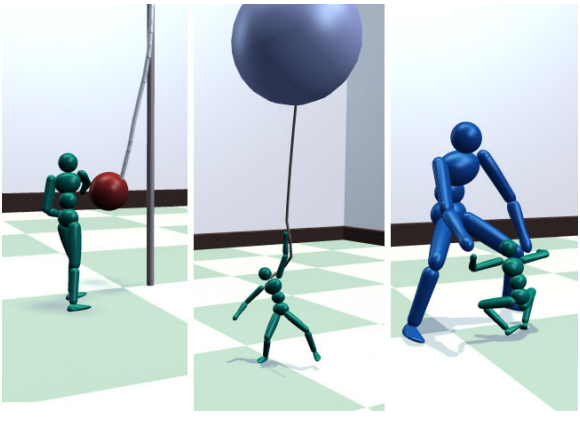

Fig. 3. Interactions from various scenarios created easily by combining motion performance and a physics-based representation of the character(s)

5.1 Performance Capture with Physical Interactions

Creating scenes with complex interaction between actors and the environment is crucial to many applications in movies and games. Although computer-generated effects enable actors to appear visually realistic in virtual worlds, producing realistic interaction between real actors and virtual objects, features, or characters remains challenging. Because actors often perform outside the context of the virtual scene, seamless integration relies heavily on manual post-processing efforts to synchronize an actor's performance with CG effects.

We introduce a technique that previsualizes final, integrated scenes at the time of performance acquisition. Our technique seeks to enforce realistic dynamic interaction in the virtual world while faithfully preserving the nuances of the actor's performance. The system combines motion capture in the form of a real-time performance with physical simulation to generate visible response to the virtual interaction. We present a hybrid approach for combining pure (kinematic) motion capture and a dynamic simulation of the character in order to create the appearance of a more immersive performance, in real time. The seamless integration of these two animation signals (kinematic and dynamic) is accomplished by transforming the kinematic signal into a dynamics representation and then balancing the influence of the original data and the physical response across the body and across time.

The description of our system architecture follows the layout diagram appearing in Figure 4. The raw performance starts with the actor and is input into a

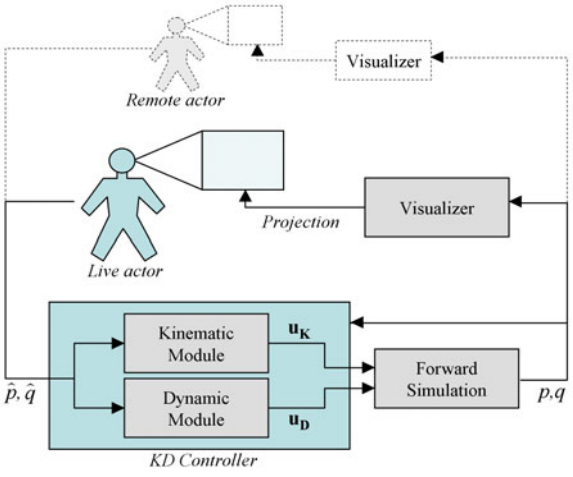

Fig. 4. The layout of the system shows the basic flow of information from the actor(s) through the motion capture system to the KD controller and the simulation. The simulation state is fed back to the control and input to the visualizer(s) which project(s) the scene for the actor(s).

combined kinematicdynamic (KD) controller. The forward simulation of the virtual world along with the physical model of the character is advanced in time based on the output of the KD control inputs. In this step, collisions are resolved by modifying the character motion and the objects in the scene. The actor gets visual feedback from the projection output and modifies the online performance appropriately.

At the core of our technique is the KD controller which seamlessly combines the (kinematic) motion performance with a physics model. The domain for our KD controller is defined as the continuous space that combines two distinct interpretations of the motion performance, first as a pure kinematic playback and second as a physical model following the performance as a desired setpoint trajectory. In our system, we implemented the latter as a forward simulation following the performance via a tracking control routine. We sidestep coordinated control in lieu of a naive tracking controller and incorporate intelligence and coordination through online performance capture.

Our system creates a single simulation within which a host of interactions are possible. For example, by adding additional simulations we can created coupled simulations like those seen in previous work [2,1,4] Also shown, with dashed lines, in Figure 4 is a networked remote actor. We extend our basic system to allow a remote performance capture and interaction between two actors in long-distant real-world locations. The networked clients remotely report the motion performance of the actors while a single local server manages the simulation that resolves interactions and generates a consistent state. This state is fed back to each remote client which uses its own visualizer that can uniquely control camera view, etc. for each actor.

5.2 Natural User-Interface for Controlling Avatars

We propose a new control interface for navigating and manipulating virtual worlds based on motion reconstructed in real-time from a human performance. Our control interface intelligently maps online motion capture data to a virtual characters action, such that the user can directly control the virtual character using her own body movement. The proposed system leverages a physical simulation, a small set of offline action examples, and an optimal integration process to synthesize the characters motion. When interacting with an object in the virtual world, the simulation ensures that the global movement of the character obeys the laws of physics, rather than directly tracking the performance. In addition, we propose a novel method for deriving active control from the performance to give intuitive control over the simulation. For local joint configurations, we integrate the online performance with offline example motions that demonstrate specific desired interactions with virtual artifacts. Recognizing when and knowing how to integrate the online and offline motion capture poses are particularly challenging because of the online nature of the input motion and real-time computation requirements. Furthermore, the identification needs to be robust against different styles of motion and different body types of users. We propose efficient algorithms to recognize the appropriate moments for integration and which body parts to include.

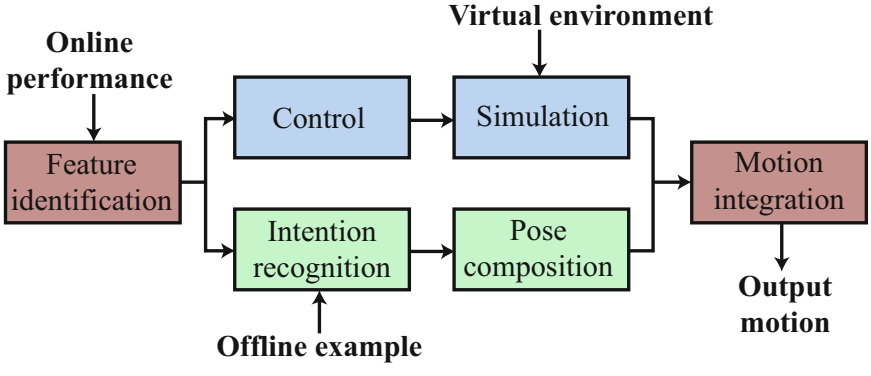

Fig. 5. Overview of the system

Our system takes as input a pose stream recorded by a motion capture device, and produces, in real time, a character animation that maps the users movements to the virtual world. To use the input motion for control of the virtual character, we need to modify the real-time input motion so that the virtual characters actions reflect the performers intention, preserve her motion style, and satisfy physical constraints in the virtual world. The designer of the virtual world plays a key role in this process by predefining the features of the world that the user can interact with and associating these features with motion sequences of possible interactions.

The overall system, shown in Figure 5, comprises both a dynamic process and a kinematic process to produce the virtual characters motion. First, we cycle through the features of the environment to check if the input motion matches their conditions for engagement. Once the interacting feature is identified, the dynamic process simulates the global motion of the virtual character by combining active control inferred from the users performance with dynamic influences from the virtual environment. In tandem, the kinematic process produces realistic context-appropriate poses by matching the users motion with the example interaction. Finally we formulate an optimization process to combine the global motion from the dynamic process and the poses from the kinematic process to produce an output motion that conforms to kinematic constraints in the virtual environment.

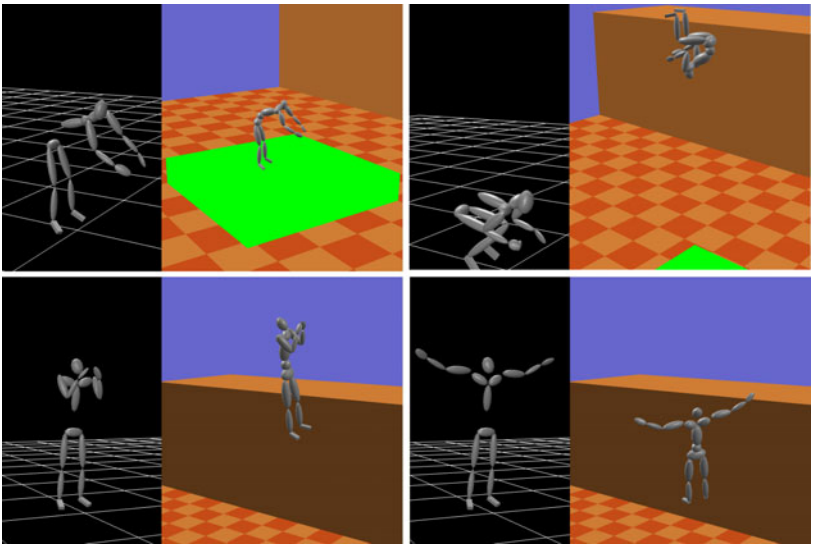

Fig. 6. The trampoline demonstrates the user's control over the dynamics of the character. Top Left: The user leans forward while jumping to generate forward angular momentum. Top Right: Tucking into a ball increases angular velocity, allowing the virtual character to flip. Bottom Row: Controlling the angular velocity by pulling in (left) and extending the arms (right).

6 Discussion

The proposed research is positioned to address the issues of generating more realistic character motion by taking advantage of online human performance, specifically recorded from full-body human motion capture, to add humanlike detail and intelligence to the animation generation process along with physical models employed for flexible, believable virtual interactions. In this research,

a novel solution is proposed: a unified framework for physics and performance-based character animation. In the described method, the proposed human-in-the-loop control paradigm allows a player or performer to sense and act for the character. Specifically, a performer will: assess the situation and make a decision about the strategy that is appropriate for a given scenario; perform high-level and intermediate path planning for the character; and provide motion examples that are suitable for the current conditions. With this information encapsulated in the human performance and feedback about interaction given as the human is acting, the proposed framework accomplishes its goal in two steps: first, by recognizing and potentially modifying the performance such that it is appropriate for the given scenario; and second, by balancing interactivity and control in order to maintain both physical responsivity to the virtual world and faithfulness to the human performance.

References

1. Karen Liu, C., Hertzmann, A., Popović, Z.: Composition of complex optimal multi-character motions. In: 2006 ACM SIGGRAPH / Eurographics Symposium on Computer Animation (September 2006)
2. O'Brien, J.F., Zordan, V.B., Hodgins, J.K.: Combining active and passive simulations for secondary motion. IEEE Computer Graphics & Applications 20(4) (2000)
3. Shin, H.J., Lee, J., Gleicher, M., Shin, S.Y.: Computer puppetry: An importance-based approach. ACM Trans. on Graphics 20(2), 67–94 (2001)
4. Shinar, T., Schroeder, C., Fedkiw, R.: Two-way coupling of rigid and deformable bodies. In: 2008 ACM SIGGRAPH / Eurographics Symposium on Computer Animation (July 2008)
5. Shiratori, T., Hodgins, J.K.: Accelerometer-based user interfaces for the control of a physically simulated character. ACM Trans. on Graphics 27(5), 123:1–123:9 (2008)
6. Yin, K., Cline, M.B., Pai, D.K.: Motion perturbation based on simple neuromotor control models. In: Pacific Graphics, p. 445 (2003)
7. Zordan, V.B., Hodgins, J.K.: Motion capture-driven simulations that hit and react. In: Eurographics/SIGGRAPH Symposium on Computer Animation, pp. 89–96 (2002)
8. Zordan, V.B., Majkowska, A., Chiu, B., Fast, M.: Dynamic response for motion capture animation. ACM Trans. on Graphics (SIGGRAPH) 24(3), 697–701 (2005)