An Intuitive Interface for Human Performance Tracking with Simulated Characters

Gökçen Çimen
ETH Zurich

Martin Guay
Disney Research

Stelian Coros
Carnegie Mellon University

Robert W. Sumner
Disney Research

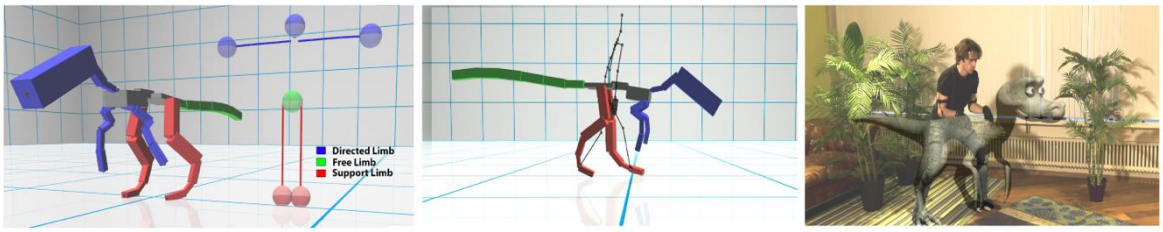

Figure1: We present an interface (Abstract humanoid on the left) for tracking a human actor with a simulated character. Our interface simplifies to simply clicking and dragging, the process of creating an articulated rigid body system for a biped digital character, together with the control parameters used to compute torques for tracking the human actor in real-time. A side effect of our simulated character is that free limbs—such as tails—are automatically activated during tracking.

ABSTRACT

One of the main challenges with tracking a live actor with a simulated character is modeling or specifying the character's mechanical system, together with the control parameters required to compute the torques. This process is so intricate, that it is mainly done by researchers and engineers—leaving this technology out of reach for casual users and digital artists. In this paper, we introduce a user-friendly interface that allows casual users to quickly model the character's mechanical system, together with the control parameters for tracking a live actor. A side effect of tracking with a simulated character is that free limbs such as tails are automatically activated into the full body control, instead of remaining static or moving in a predetermined fashion. We show various motion examples with two biped characters: a raptor dinosaur, as well as an alien.

KEYWORDS

Performance tracking, simulated character, intuitive animation interface.

1. INTRODUCTION

Motion capture is heavily used by the visual effects industry to allow a professional actor to craft the performance of a virtual creature. However, because the creature's shape and morphology often differs significantly from the actor's, a motion retargeting step is inevitable. A leading approach to retarget a human

actor's motion onto a virtual character is to track shared features such as the feet and the hips using inverse kinematics. The problem with this approach is that it leaves free limbs, such as tails, static and without motion.

To address this issue, we embed the character into a physics-based simulation framework and track the actor's motion using optimal control. As a result, free limbs, such as a virtual creature's tail, are automatically animated to move in unique ways that are consistent with the character's overall performance. While this simulation control concept is a promising approach to synthesize unique movements, its biggest drawback is its complex nature. Physics-based simulation unfortunately requires intricate knowledge of mechanics and control mechanisms, leaving it out of reach for most visual effects artists that lack scientific training.

This complexity comes from two main sources. First, the character must be provided with an articulated mechanical system including corresponding rigid body and joint properties. And second, a controller general enough to track various human motions in real-time must be set up and tuned so that its parameters are appropriate for the new character. In our work we use model-based inverse dynamics ([Abe et al. (2007)])

In our research, we make physics-based retargeting more accessible with an intuitive user interface, designed around a general control framework, that allows both quickly designing the character's mechanical system as well as setting the controller parameters for optimal tracking. Our work relies on the core observation that, while *limbs* may be different, they often share a common function. For example, legs interact with the ground, and free limbs, such as tails, are used for balancing and controlling the character's overall orientation. Hence, we devised a bipedal limb-based abstraction where the user simply drags-and-drops from the abstract limbs to the simulated character's limbs to automatically fill the controller's parameters for tracking.

Our method comes with additional benefits. First of all, we can automatically clean-up contacts from the often noisy captured motion. This is possible thanks to the notion that certain limbs are used for support and interact with the ground. Secondly, our controller penalizes deviations from a natural pose, which can be used to control the style of the motion, simply by specifying a new default pose. And finally, the tracking can be performed in real-time. We show results of a raptor and alien tracking different motions.

2. RELATED WORK

In practice, tracking a human actor with a digital character is done as a feature-based optimization problem (inverse kinematics, or IK), where the distance between a set of features —such as the feet and hands positions—are minimized at each frame of the animation [Choi & Ko (1999)], [Shin et al. (2001)], [Tak & Ko (2005)]. Several works address foot contacts and foot skating issues in the context of tracking by altering the actor's captured trajectories [Kovar et al. (2002)], [Ikemoto et al. (2006)]. When the characters have different topologies (i.e. different joints hierarchies) and/or do not move in the same way, the tracked motion is likely to be unrealistic. For example, the legs of a dinosaur do not bend the same way as the legs of human, and the free limbs humans do not possess—such as a tail—remain static.

Controlling a simulated character to perform different motions is a problem that has been researched for several decades now. Yet it remains a challenge to go from a kinematic—strictly positional—signal, to a controlled motion using a simulated articulated character.

One of the motion skills that has been well studied is locomotion, and there are now robust controllers specifically designed for this. Typically, a set of locomotion *poses* are tracked with Proportional Derivative (PD) control of target joint angles, and a balancing strategy based on the inverted pendulum (IP) model is used to adjust the target poses online [Yin et al. (2007)], [Tsai et al. (2010)], [Lee et al. (2010)]. While these controllers are very robust, they can only track locomotion, and do not extend easily to other types of motions.

As a general approach to track human motion with a simulated character, Liu and colleagues developed a sampling-based approach that can successfully learn a feedback function to track contact-rich motion capture sequences [Liu et al. (2010)], [Liu et al. (2015)]. They learn a constant feedback function over the motion clip by using stochastic optimization along a black box rigid body simulator. Their framework has only been demonstrated with human character with well-studied proportions, masses and PD stiffnesses. In other words, this cannot be used directly to track human motion with non-humanoid characters such as a dinosaur.

Another line of works—so-called online optimization controllers—consider the control at a single time step and recompute the torques each time via inverse dynamics; assuming knowledge of the equations of motion [Abe et al. (2007)], [Da Silva et al. (2008)], [Macchietto et al. (2009)], [Mordatch et al. (2012)], [de Lasa et al. (2010)], [Rabbani et al. (2014)], [Levine & Popovic (2012)]. When the equations of motion of an articulated

rigid body system are expressed in generalized coordinates, a linear relation between joint torques and joint accelerations can be established for a single time step. As a consequence, it becomes possible to minimize a collection of quadratic objectives under the hard linear constraint that the equations of motion hold.

Abe et al. (2007) focused on motions that remained in balance (static contacts) and subsequent works focused on planning and engineering features for various motion skills [Da Silva et al. (2008)], [Macchietto et al. (2009)], [Mordatch et al. (2012)], [de Lasa et al. (2010)], [Rabbani et al. (2014)], or relaxing physical realism for robustness [Levine & Popovic (2012)]. While this type of controller has been well studied, it requires a scientist or engineer for each new character or motion. In contrast, our automates the process for bipedal characters and allows casual users to set up the character's rigid body system together with the parameters of the controller, simply by clicking on our abstract limb-based interface.

Closest to our are methods that combine dynamics to alter a kinematic motion model as to better cope with interactions in a simulated environment [Ishigaki et al. (2009)], [Nguyen et al. (2010)]. These methods rely on simplified models of dynamics (not the full body) and the dynamics are used to alter a motion model that relies on pre-exiting motion clips; in both cases human captured motion. In contrast, our method is based on adapting motion capture data to virtual characters which have different body structures than human. The only prior information required is the user's character design choice.

To our knowledge, only a few prior works placed the control of simulated characters into the hands of casual users. Coros et al. (2010) allowed the user to change the body proportions of the character, resulting in different motions. Unfortunately, their method is specific to human locomotion. Closer to our effort is the work of Levine et al. (2012), which also uses multi-objective control to track animations. They use the bone lengths to initialize the rigid bodies, but relax the physical accuracy by allowing root activation. With their method, the character can only track the same character, and there is no retargeting or free limbs activation. In contrast, we allow the user to track human actors with different characters as well as different body parts associations.

3. OVERVIEW

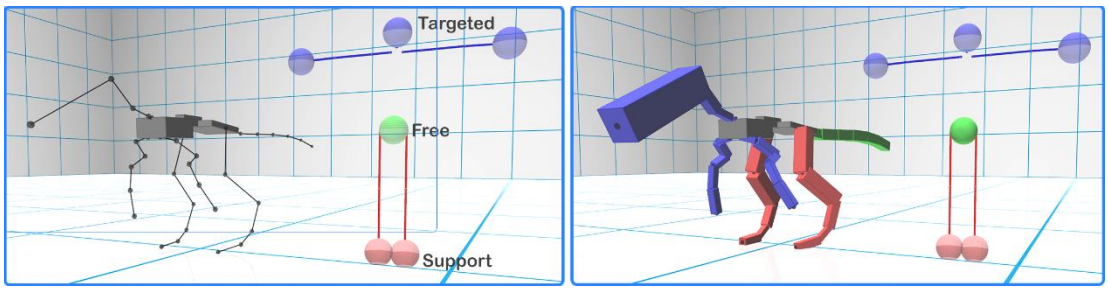

Figure 2: The user interface we use to create an articulated rigid body system and to quickly set control objectives based on a limb abstraction of the bipedal character. The user simply drags and drops a limb type from the abstract humanoid onto the limbs of the character to get it ready for simulated tracking.

Our goal is to have a bipedal creature follow the motion of a human actor in real-time. While characters and the actor may be different morphologically, we observe that they often share a set of common features. In particular, most characters share a global position, global orientation, as well as a set of limbs whose end effector (EE) trajectories match, but at a different scale (see Fig. 3).

While tracking only the re-targeted positions conveys the essence of the actor's motion, it leaves the free limbs of the character, such as the tail, without motion. To active the free limbs and provide the character with additional realism, we model the full body dynamics and regulate angular momentum with an additional control objective.

In order to mix different objectives while satisfying the equations of motion, we formulate our tracking as model-based multi-objective control [Abe et al. (2007)]:

$$\begin{aligned} \min_{\ddot{q},\tau,f} \quad & \sum w_i \, E_i \\ \text{s.t.} \quad & M(q)\,\ddot{q} - C(q,\dot{q}) = J_x^T f + [0 \;\; \tau]^T, \end{aligned} \tag{1}$$

which exploits the linear relation between joint accelerations $\ddot{q}$ and torques τ when the equations of motion are expressed in generalized coordinates, where the vector q is the root position, the root orientation and the set of joint relative orientations. The matrix $M(q)$ is the generalized mass matrix, $C(q, \dot{q})$ is the vector that combines gravitational forces, coriolis and centrifugal terms. The jacobian transpose J_x^T measures the change in position w.r.t. the generalized degrees of freedom and maps the cartesian ground reaction forces f into generalized forces, with the construction $[0\ \tau]^T$ avoiding root activation. Setting the weights w_i of the objectives is a cumbersome and time consuming task. To avoid setting all the weights manually, we introduce a limb abstraction—summarized in Table 1—that automatically sets the weights for each objective. Here we define a limb as a connected linear chain of bones attached to the body frame which is comprised of the pelvis and upper body (shown in grey in Fig. 2).

Table 1. Our limb abstraction is comprised of three types. Support limbs, which drive the body to a desired location through ground reaction forces, and alternate between swing and stance states. Free limbs, which do not track a human body part, but help balance and control the character by regulating angular momentum. And finally, Targeted limbs which track a part of the human body such as the hands or head.

Type	**Role**
Support	Carry the body through ground reaction forces and ensure contact constraints
Free	Does not track a human part, such as a tail, but participate in angular momentum control.
Targeted	Tracks a human part, but without contacts (e.g. head or hands).

The first step to realize our simulated tracker concept is to model the character's articulated rigid body system and to specify the type of limb, as well as target location on the human actor's skeleton. Hence, in the next section we describe our intuitive user interface to get the character simulation- and tracking-ready.

4. SIMULATED TRACKING INTERFACE

The input to our modeler is a bipedal character skeleton, and we provide the necessary widgets to track a human skeleton through our limb-based optimal control. We begin by modelling the character's mechanical system, i.e. the linked rigid bodies that approximate the mechanical properties of the character. The second step consists in attributing each of the character's limbs a type from one of our limbs (support, targeted or free). The user creates rigid bodies by clicking on the bones of the skeleton. We first initialize the rigid body shape with the orientation and size of the skeleton bone, and set the mass and friction to default values. In most cases, the user can use a facilitating function that fills a linear chain of bones with a linear chain of connected rigid bodies, starting from the root, as shown in our accompanying video.

Our limb-based abstraction implicitly encodes which human body part to track. Hence when setting the type of limb by drag-and-dropping a limb type from the limb-based abstraction (shown in Fig. 2), the controller is automatically set ready for tracking the human actor. The ordering of the process does not matter, the user can set the type of limb during or after having modeled the articulated rigid body system.

5. ONLINE FEATURE RETARGETTING

Generally, the features we track are human positions $x_r(t)$ and orientations $\theta_r(t)$, where r denotes reference motion. In particular, we track each of the human's end effector positions, together with the root position and orientation.

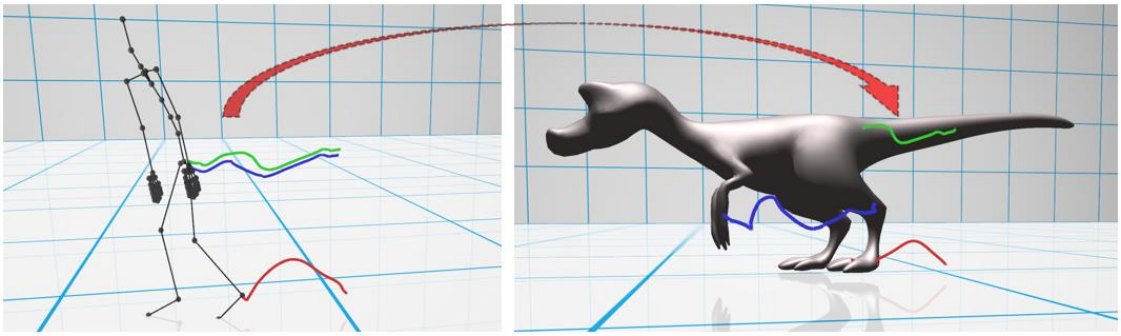

Figure 3: Some of the features in both the character and the human actor's match but at a different scale (e.g. the hand, feet and head, and pelvis positions). We retarget these trajectories to the position and scale of the simulated character for tracking.

While the orientations can be tracked directly (i.e. $\theta_{des} = \theta_r$ where θ_{des} is the desired orientation), the positions need to be translated and scaled as to be reachable by a character with differently sized limbs. Secondly, because the captured motion can have noisy contact trajectories, we perform online cleaning-up of the end effectors at the extremity of support limbs.

Retargeting Human Position Trajectories. To ensure feasibility of tracked end effector positions, we first compute the differential coordinates of the actor's positions $\Delta x_r(t) = x_r(t + \Delta t) - x_r(t)$, and scale it down based on the proportions $\alpha = l_c \backslash l_r$, where l_* is the distance between a limb's end effector and root position for the rest pose, with c denoting character. This results in:

$$x_{des}(t) = \alpha \ \Delta x_r(t) + x(t), \tag{2}$$

where $x(t)$ is the end effector's position. This rescaling is illustrated in Fig. 4.

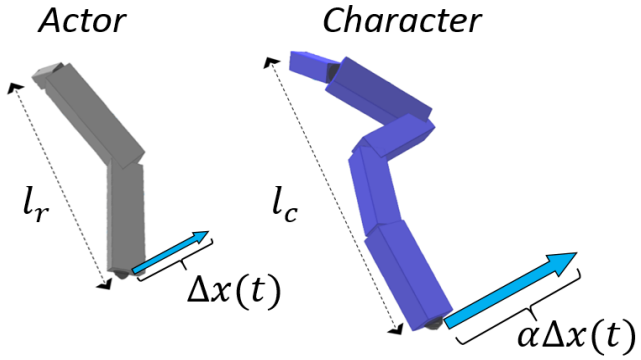

Figure 4: When retargeting the end effector positions to the character, we scale the relative displacements proportionally to each limb length, that we define as the distance between the end effector and the root of the limb.

Online Contacts Clean-up. We process the end effectors (EEs) at the extremity of support limbs, as to clean the contacts in real-time. This is particularly challenging when the capture is being streamed in real-time, and we do not have the full trajectory to determine whether a position is supposed to be in contact and remain fixed, or it should be moving.

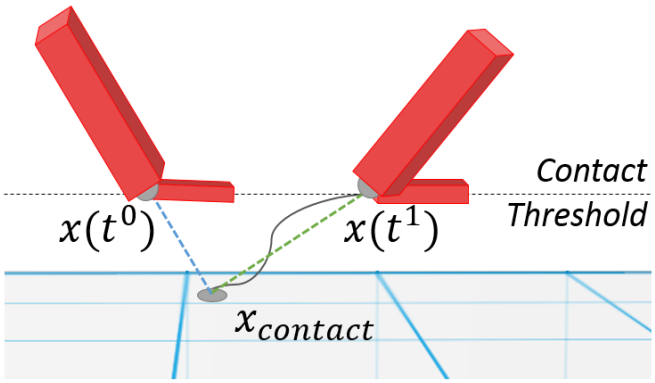

Figure 5: To clean contacts in real-time, we perform smooth-in and smooth-out transitions between the fixed (below the contact threshold) and the moving (above the contact threshold) end effector positions.

Our solution to this problem is to keep the actor's end effector fixed when close to the ground (below a contact threshold), and to perform a smooth-in and a smooth-out to transition between the fixed contact position, and the moving position beyond the contact threshold. When the EE position gets below a threshold at time t^0, we project the position onto the ground using its velocity, and define this position as the contact position $x_{contact}$. We then perform a smooth transition between the EE position at the threshold position $x(t^0)$ and the contact position $x_{contact}$ using linear interpolation over a small time window. When the actor's EE position leaves beyond the contact threshold at time t^1, we perform a smooth out transition between the contact position $x_{contact}$ and $x(t^1)$ (see Fig. 5).

6. CONTROL OBJECTIVES

We describe our objectives that include both tracking re-targeted positions to be reached by the character, as well as objectives for regulating angular momentum, and controlling style. Each type of limbs contributes to the overall sum of weighted objectives, and we describe at the end of the section how we automatically prioritize the weights based on the type of limb.

Our limb-based controller tracks the global root position, root orientation, the collection of end-effectors at the extremity of the limbs, as well as influences the tracking with additional full body angular momentum and pose regularization. We assemble this sum of weighted objectives (detailed below), and solve problem (1) for the optimal torques.

Target Position and Orientation. These objectives are used to track the retargeted positions x_{des} and orientations θ_{des}, by the character. We compute the desired acceleration for the concerned rigid body based on proportional-derivative (PD) control:

$$\begin{aligned} \ddot{x}_{des} &= k_p(x_{des} - x) + k_d(\dot{x}_r - J_x \dot{q}), \\ \ddot{\theta}_{des} &= k_p(\theta_{des} - \theta) + k_d(\dot{\theta}_r - J_\theta \dot{q}), \end{aligned} \tag{3}$$

where x and θ are the rigid position and orientation, k_p the proportional stiffness and k_d, the derivative value, which both remain constant for all motions. Here J_θ denotes the change in orientation for the rigid body, w.r.t. the all the joint orientations. From these desired accelerations, we measure the error to the character's current accelerations:

$$E_p = \| \ddot{x}_{des} - (J_x \ddot{q} + \dot{J}_x \dot{q}) \|^2, \tag{4}$$

$$E_o = \| \ddot{\theta}_{des} - (J_\theta \ddot{q} + \dot{J}_\theta \dot{q}) \|^2. \tag{5}$$

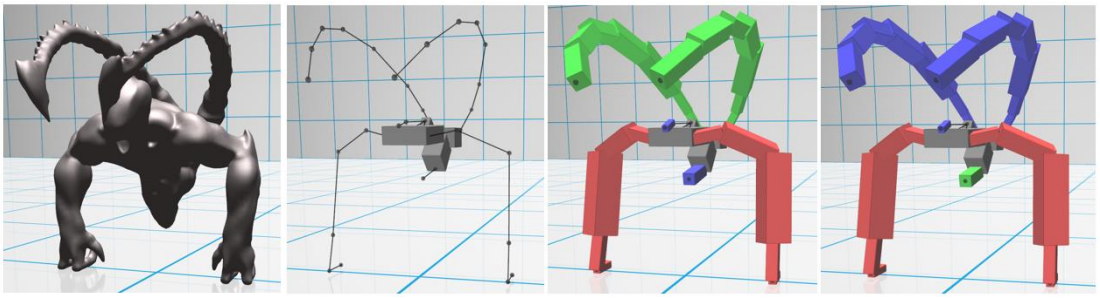

Figure 6: In this figure we see the result of setting two different types of limbs for the alien character. The limb association on the left sets the tails to free limbs, while the association on the right sets the tails to the targeted limbs, which allows the stingers at the tip to perform attacking motions.

Angular Momentum (AM). The total angular momentum about a point (often the character's center-of-mass) gives a measure of the system's internal rotations. For example, when the actor's upper body bends forward, the total angular momentum changes, and we can minimize this change in AM w.r.t. to all the links (including the tail) to activate the tail's motion.

Hence we minimize the change in total angular momentum w.r.t. to the center of mass:

$$E_{AM} = \left|\dot{L}_{AM}\right|^2, \tag{6}$$

Where

$$\begin{aligned} \dot{L}_{AM} &= R(q)\, M(q)\, (J_x^T \ddot{q} + \dot{q}_x^T \dot{q}), \\ R(q) &= [\, [\, r_1(q)\,]_x \ldots [\, r_m(q)\,]_x\,], \end{aligned}$$

where $r_i(q)$ are the position vectors of the i^{th} body links relative to the center-of-mass, $[\, r_i(q)\,]_x$ are the skew symmetric coefficients from the cross product, $J_x^T = [\, J_{x_1}^T \ldots J_{x_m}^T\,]$ is the jacobian transpose mapping generalized coordinates to cartesian positions, and $M(q)$ is the mass matrix of the entire articulated rigid body system.

Pose Regularization. To control the style of the animation, as well as to prevent the character from entering unrealistic configurations, we penalize deviations from a rest pose. This regularization objective is the sum of errors between joint angle accelerations computed through PD control (similarly to equation (5)) with the desired pose defined as the rest pose in equation (3) resulting: E_j^{reg} for each joint j.

Total Sum of Objectives. Using equation (4) and (5), we define a root position and orientation objective E^{root}, as well as the set of end effector objectives $E_i^{EE}, i = 1, \ldots, m$ with m limbs. With the angular momentum regulation and the pose regularization, the total sum of objectives we minimize is:

$$w^{root} E^{root} + \sum_i w_i^{EE} E_i^{EE} + w^{AM} E^{AM} + \sum_i w_j^{reg} E_j^{reg}, \tag{7}$$

which we set as the objectives in problem (1). But before solving the problem, we first need to set the weights of each objective.

Limb-based Prioritization. Solving the torques that minimize this sum of weighted objectives (7) is challenging in practice as the objectives may be conflicting and need to be prioritized. We observed that certain limbs play a more important role when it comes to performance tracking. For example, the support limbs need to provide clean contact positions and should be weighted higher. Hence, to accommodate the user in setting the weights, we use our limb-based abstraction to automatically set values.

To express the relative priorities of the objectives, we first define a maximum weight value w_{max}, and define each weight based on this maximum value, and on the type of limb. Hence, the global orientation and position being visually important are weighted with the maximum value $w_{root} = w_{max}$, the angular momentum playing a lesser role $w_{AM} = 0.5 w_{max}$, and the objectives that depend on the type of limb (the end effectors, as well as the pose regularization) are summarized below:

Limb	w_i^{EE}	w_j^{reg}
Support	w_{max}	0.01 w_{max}
Targeted	0.5 w_{max}	0.02 w_{max}
Free	-	0.03 w_{max}

where $w_{max} = 4000$, and depends on the mass of the system.

7. RESULTS

Character setup. We created two characters using our interface described in Section 4. The first is a raptor dinosaur and the second an alien that walks on his hands and has two tails. In both cases, an artist created a mesh and a skeletal rig in Maya ([Autodesk (n.d.)]) without constraints regarding its skeleton. Our interface for the character set up allows quickly creating rigid bodies based off the skeleton, as well as setting different types of limbs for the tracking (see Fig. 6 for different types of limb associations). All of the character set ups including the full articulated rigid body systems were created by the authors of this paper under three minutes each.

Capturing human motion. We captured all the actor's motions using an Axis Neuron ([Noitom (n.d.)]) full body motion capturing system. The system as 13 captors and samples the motion at 120 frames per second. The character's motion is not always accurate and may include body interpenetrations, shakiness in the feet or hands, as well as unstable foot contacts.

Solver. We solve problem (1) at each time step $\Delta t = 0.01$, with linearly-constrained quadratic programming. We then integrate the generalized accelerations using the generalized equations of motion. Note that we do not send the torques to a cartesian rigid body simulator, but always perform the simulation in generalized coordinates. Our single threaded implementation runs in real-time on a 4.00 GHz Intel Core i7 machine. We used the same objectives and weights provided by our limb-based abstraction, to track human locomotion (forward and backward), as well as various expressions gestured with hands and upper body such as roaring, being scary and biting (shown in our accompanying video).

Using our limb-based controller across different characters. It is often the case that for each new character, the control objective weights must be adjusted to this new character's proportions. We tested using a similar limb attribution on both the raptor—which has a long and heavy tail with a long upper body—and an alien character—which has two long tails and no legs, but stands on his hands. We found our limb-based control framework to be quite robust in that regard, adapting quite well to changes in character morphology and producing motions that are characteristic to the character's intrinsic shape (shown in our accompanying video).

Controlling style through different rest poses. We experimented with our pose regularization to provide the motion with a different feel. For example, we changed the raptor's pose to have its head slightly tilting forward, which resulted in a sadder look for the motion (shown in our accompanying video).

8. LIMITATIONS AND FUTURE WORK

One of the main intricacies associated with multi-objective inverse dynamics [Abe et al. (2007)] are the conflicts between objectives and constraints, which may become unfeasible—causing the simulator to diverge and blow up. One of our remedies was to relax the hard position constraints and replace them with objectives (with a large weight value provided by the support-type of limb).

To alter the character's motion style, we simply changed a single pose regularization, to avoid becoming depending on large collections of data. However, the single pose regularization could easily be replaced with a data-driven pose function learned from a collection of example poses (using for example [Grochow et al. (2004)])

While our limb-based abstraction supports walking on hands and having multiple tails, it is designed specifically for bipedal characters. In the future, we could imagine extending our abstraction to the case of

quadrupeds by introducing a coordinated stance-limb planner which transfers end-effector trajectories to the appropriate support limbs.

9. CONCLUSION

Setting the control parameters for a new simulated character is traditionally complex and requires engineering skills. In this paper, we introduced an intuitive interface for casual users to track human actors with a simulated character in real-time. Our limb-based abstraction simplifies the initial set up to clicking and dragging on a few nodes. A side effect of controlling the simulated character is the activation of free limbs such as the tails. While we greatly simplified the process of tracking human actors with bipedal characters, we left out tracking with quadrupeds and multi-legged creatures, which could addressed in the future by coordinating the stance limbs of the character.

REFERENCES

Abe, Y. et al, 2007. Multiobjective control with frictional contacts. *In 'Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '07, Eurographics Association*. Aire-la-Ville, Switzerland, pp. 249–258.

Autodesk (n.d.), Maya. *Computer Animation and Modeling Software*.

Choi, K.-J. & Ko, H.-S, 1999. On-line motion retargeting. *In Proceedings of the 7th Pacific Conference on Computer Graphics and Applications, PG '99, IEEE Computer Society*.

Coros, S. et al, 2010. Generalized biped walking control. *ACM Trans. Graph.* Vol. 29, No. 4, pp. 130:1–130:9.

Da Silva, M. et al, 2008. Simulation of human motion data using short-horizon model-predictive control. *Computer Graphics Forum*. Vol. 27, No. 2, pp. 371–380.

de Lasa, M. et al, 2010. Feature-based locomotion controllers. *ACM Trans. Graph.* Vol. 29, No. 4, pp. 131:1–131:10.

Grochow, K. et al, 2004. Style-based inverse kinematics, *ACM Trans. Graph.* Vol. 23, No. 3, pp. 522–531.

Ikemoto, L. et al, 2006. Knowing when to put your foot down. *In Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games, I3D '06, ACM*, pp. 49–53.

Ishigaki, S. et al, 2009. Performance-based control interface for character animation. *In ACM SIGGRAPH 2009 Papers, SIGGRAPH '09, ACM*. New York, NY, USA, pp. 61:1–61:8.

Kovar, L., Schreiner, J. & Gleicher, M, 2002. Footskate cleanup for motion capture editing. *In Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '02, ACM*. pp. 97–104.

Lee, Y., et al, 2010, Data-driven biped control, *ACM Trans. Graph.* Vol. 29, No. 4, pp. 129:1–129:8.

Levine, S. & Popovic, J, 2012. Physically plausible simulation for character animation. *In* Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '12. pp. 221–230.

Liu, L. et al, 2015. Improving sampling-based motion control. *Comput. Graph. Forum*. Vol. 34, No. 2, pp. 415–423.

Liu, L. et al, 2010. Sampling-based contact-rich motion control. *ACM Trans. Graph.* Vol. 29, No. 4, pp. 128:1–128:10.

Macchietto, A. et al, 2009. Momentum control for balance. *ACM Trans. Graph.* Vol. 28, No. 3, pp. 80:1–80:8.

Mordatch, I. et al, 2012. Discovery of complex behaviors through contact-invariant optimization. *ÁCM Trans. Graph.* Vol. 31, No. 4, pp. 43:1–43:8.

Nguyen, N. et al, 2010. Performance capture with physical interaction. *In Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '10, Eurographics Association*. Aire-la-Ville, Switzerland, Switzerland, pp. 189–195.

Noitom (n.d.), Perception Neuron. *https://neuronmocap.com/content/axis-neuron-software*

Rabbani, A. H. et al, 2014. Anticipatory balance control. *In Proceedings of the Seventh International Conference on Motion in Games', MIG '14.* pp. 71–76.

Shin, H. J. et al, 2001. Computer puppetry: An importance-based approach. *ACM Trans. Graph.* Vol. 20, No. 2, pp. 67–94.

Tak, S. & Ko, H.-S, 2005. A physically-based motion retargeting filter. *ACM Trans. Graph.* Vol. 24, No. 1, pp. 98–117.

Tsai, Y. Y. et al, 2010, Real-time physics-based 3d biped character animation using an inverted pendulum model. *IEEE Transactions on Visualization and Computer Graphics*. Vol. 16, No. 2, pp. 325–337.

Yin, K., et al, 2007, Simbicon: Simple biped locomotion control. *ACM Trans. Graph.* Vol. 26, No. 3.