Pliant Motion
: Integration of Virtual Trajectory Control into LCP Based Physics Engines

Takashi Tokizaki * Yuuichi Tazaki † Hironori Mitake ‡ Shoichi Hasegawa *
University of Electro-Communications * Dept. of Math. and Comp. Sci, Tokyo Institute of Technology †
P & I lab. Tokyo Institute of Technology ‡

Keywords: Iterative LCP Solver, Physics simulation, virtual creature

1 Introduction

Interactive applications such as Video Games require characters, which generate motions corresponding to user's interaction. Motion capture is an effective technique to reproduce realistic motion. However, to produce a motion which is appropriate to the operation of the user, a lot of motions must be prepared and one of the motions which is suitable for the user's operation must be selected and played. Because the user's operation changes the motion trajectory, unexpected contact to objects may happen. The amount of change on a trajectory depends on not only the trajectory of motion but also internal tensions of skeletal muscles - co-contraction level, when a person put one's hand down on a table or collides with an object. [Hogan 1984] proposed that reaching motion of human is supposed to be generated by spring damper characteristics of muscles dragging to the virtual trajectory. Human controls not only trajectories of motions but also spring-damper characteristics of muscles by changing co-contraction levels. Realistic character motions contacting to objects can be generated easily with virtual trajectory tracking control which is integrated to physics engines for character motions.

However, it is not easy to make a virtual trajectory by hand. Though a real trajectory tracks the virtual trajectory, the real trajectory parts from virtual trajectory by effects of inertia and Coriolis force. Thus making a virtual trajectory which generates specified real trajectory is not easy. Thus, we propose a realtime method to generate a virtual trajectory tracking any real trajectory for any articulated bodies with dynamics simulator based on iterative LCP solvers.

2 Related works

The method of implementing motion of reaction into a Motion Capture data or a key frame animation is being studied. [Zordan et al. 2005] proposed the method of using physics engine to interpolate Motion Captured data.

3 Computation of virtual trajectory

Proposed method computes joint torques to track desired (real) trajectory using physics engines (1st step), then executes one step of simulation to update posture and velcity of the articulated body with PD control targeting to the desired trajectory (2nd step). Because the computed torque from 1st step is almost enough to track desierd trajectory, the feedback coefficients for the PD control are not matter. Therefore, the generated trajectories have no difference between small and large feedback coefficients. Though proposed method do not compute virtual trajectory explicitly, virtual trajectory can be found from the torques computed in 1st step by supposing that the torques are generated from the PD control targeting to the virtual trajectory.

3.1 torque computation (1st step)

Torque computation for trajectory tracking is realized by constraints of the angular velocities on joints. Physics engines compute constraint forces, which satisfy constraint conditions on joints and things like that. By introducing velocity constraints into movable degrees of freedom on joints, physics engines compute constraint forces. Then, joints move by specified velocities.

This algorithm can be implemented into LCP solver of physics engine, by slightly changing the formulations of constraint conditions. For example, ball joints are implemented by constraints of three translation degrees of freedom. Velocity control for the ball joint results in adding a constraint of $\omega = \omega_{desierd}$ into three degrees of freedom on the angle.

Generally, physics engines update velocities of rigid bodies by integrating forces acting on them after the computation of constraint forces. However, the purpose of this computation is calculation of torques and we do not update the velocities of rigid bodies here.

3.2 motion generation (2nd step)

Proposed method removes the constraints on movable degrees of freedom of joints which were introduced in 1st step, then converts the torques computed in 1st step into constraints of PD control with offset torque for angular freedom of joints to give the joint torque to track the desired trajectories.

4 Physics simulator with LCP solver

This section introduces a simulation method for rigid body dynamics constrained by joints and things like that. Section 4.1 briefly describes how to resolve computation of constraint forces into a LCP. Section 4.2 and section 4.3 describes the method to implement the velocity control and the PD control with offset torque into the LCP solver respectively.

4.1 Resolving to LCP

Let $p(t)$ and $v(t)$ be a position and a velocity of a rigid body in time t. Constraints on a rigid body can be categolized into two - equal constraints and linear complementary constraints. For example, a connection of two rigid bodies by a joint resolved into equal constraints and a contact between two rigid bodies resolved into linear complementary constraints. Here, let w_e and w_c be variables for velocities which constraint by equal and linear complementary constraints respectively. Let λ_e and λ_c be corresponding constraint forces. These constraints can be present as

$$w_e(t) = J_e(t)\, v(t) \tag{1}$$
$$w_c(t) = J_c(t)\, v(t) \tag{2}$$

*e-mail:{tokizaki, hase}@hi.mce.uec.ac.jp
†e-mail:tazaki@cyb.mei.titech.ac.jp
‡e-mail:mitake@hi.pi.titech.ac.jp

with Jacobian matrix J_e and J_c. Here, we simply and represent it as

$$w = \begin{bmatrix} w_e \\ w_c \end{bmatrix},\ J = \begin{bmatrix} J_e \\ J_c \end{bmatrix},\ \lambda = \begin{bmatrix} \lambda_e \\ \lambda_c \end{bmatrix} \tag{3}$$

Newton-Euler's equation of motion gives the following formulation:

$$M\dot{v}(t) = f(t) + J(t)^T \lambda(t) \tag{4}$$

where M, $f(t)$ are a mass matrix and a vector of external force and Coriolis term. While, the constraint condition can be formulated as

$$w_e(t) = \mathbf{0} \tag{5}$$

$$w_c(t) \geq \mathbf{0},\ \lambda_c(t) \geq \mathbf{0},\ w_c(t)^T \lambda_c(t) = 0 \tag{6}$$

where sign of inequalities for vectors mean each elements of vector satisfy the inequalities. Now then, we introduce update rules based on ordinary differential equations to simulate dynamics.

In following equations, let t be descritized time and let $[t]$ be value of a variable at time t. For example, $p[t]$ represent position p at time t. Proposed method update following rules at time t.

1. Compute constraint force $\lambda[t]$
. 2. Update velocities of rigid bodies .

$$v[t+1] = v[t] + M^{-1}(f[t] + J[t]^T\lambda[t])h \tag{7}$$

3. Update positions of rigid bodies.

$$p[t+1] = p[t] + v[t+1]h \tag{8}$$

where h is integration time step. The constraint force computed in step1 ($\lambda[t]$) should choose to satisfy constraints on $v[t+1]$ at the timing of just after the step2 and just before the step 3. Therefore the constraint conditions become

$$w_e[t+1] = J_e[t]v[t+1] \tag{9}$$

$$w_c[t+1] = J_c[t]v[t+1] \tag{10}$$

From equation (7) equation (8), the problem to solve constraint force resolved into the following LCP.

$$\begin{bmatrix} w_e[t+1] \\ w_c[t+1] \end{bmatrix} = A \begin{bmatrix} \lambda_e[t] \\ \lambda_c[t] \end{bmatrix} + b \tag{11}$$

$$A = J[t]M^{-1}J[t]^T h \tag{12}$$

$$b = J[t]\{v[t] + M^{-1}f[t]h\} \tag{13}$$

$$w_e[t+1] = \mathbf{0} \tag{14}$$

$$w_c[t+1] \geq \mathbf{0},\ \lambda_c[t] \geq \mathbf{0},\ w_c[t+1]^T \lambda_c[t] = 0 \tag{15}$$

4.2 Integrating velocity control into LCP solver

Controller for angular velocity of joints can be implemented as equal constraints in previous section. Let $\boldsymbol{w}_0$ be a desired angular velocitiy of a joint. The velocity control is implemented as

$$w_e[t+1] = \boldsymbol{w_0} \tag{16}$$

4.3 Integrating spring, damper and offset torque into LCP solver

PD-controller with offset torque of joints also can be implemeted as equal constraints. In addition to w_e and w_c in section 4.1, let w_s be velocity of degrees of freedom where PD controllers working on. w_s is also represented as

$$w_s(t) = J_s(t)\, v(t) \tag{17}$$

Here, let λ_s to corresponding constraint force (sum of the PD control force and offset torque f_0). The relation between constraint force and propotional and differential error for PD controller is

$$\begin{aligned} \lambda_s[t] &= -Kq_s[t+1] - D(w_s[t+1] - w_{tar}[t+1]) + f_0 \\ &\approx -K(q_s[t] + w_s[t+1]h) - D(w_s[t+1] - w_{tar}[t+1]) + f_0 \end{aligned} \tag{18}$$

where $q_s[t]$ is proprtional error for PD control and w_{tar} is target velocity of trajectory tracking. Then, we get

$$\begin{aligned} w_s[t+1] = &-(D+Kh)^{-1}\lambda_s[t] \\ &-(D+Kh)^{-1}(Kq_s[t] - Dw_{tar}[t+1] - f_0) \end{aligned} \tag{19}$$

Here, we redefine w and J as

$$w = \begin{bmatrix} w_e \\ w_c \\ w_s \end{bmatrix},\ J = \begin{bmatrix} J_e \\ J_c \\ J_s \end{bmatrix},\ \lambda = \begin{bmatrix} \lambda_e \\ \lambda_c \\ \lambda_s \end{bmatrix} \tag{20}$$

and get

$$\begin{bmatrix} w_e \\ w_c \\ w_s \end{bmatrix} = A \begin{bmatrix} \lambda_e \\ \lambda_c \\ \lambda_s \end{bmatrix} + b \tag{21}$$

$$A = J[t]M^{-1}J[t]^T h \tag{22}$$

$$b = J[t]\{v[t] + M^{-1}f[t]h\} \tag{23}$$

Finally, we substitue equation (21) with equation (19) and get

$$\begin{bmatrix} w_e[t+1] \\ w_c[t+1] \\ \mathbf{0} \end{bmatrix} = \tilde{A} \begin{bmatrix} \lambda_e[t] \\ \lambda_c[t] \\ \lambda_s[t] \end{bmatrix} + \tilde{b} \tag{24}$$

$$\tilde{A} = J[t]M^{-1}J[t]^T h + \begin{bmatrix} O & O & O \\ O & O & O \\ O & O & (D+Kh)^{-1} \end{bmatrix} \tag{25}$$

$$\tilde{b} = J[t]\{v[t] + M^{-1}f[t]h\} + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ (D+Kh)^{-1}(Kq_s \\ -Dw_{tar}[t+1] - f_0) \end{bmatrix}$$

As shown in the above equations, PD-controller with offset torque resolved into equal constraints of LCP with modified coefficient matrix and vector of $\tilde{A}, \tilde{b}$.

5 Result

We create examples of constrainted motion with proposed and previous method. We generate motion of pushing three boxes on a desk by tracking pre-defined trajectory with (a) simple PD control, (b) proposed method with stiff (large) feedback coeffiecent and (c) proposed method with soft (small) feedback coefficent.

Figure1 shows examples of the generated motion.

In (a), the hand of the character penetrates into the box. Though, large contact force acts on the hand, the tracking force breaks down the contact force. In (b) and (c), the contact force prevent the penetration through the box. In (b), The hand gives enough force to move the box and both boxes move, while (c), the hand gives not enough force and the hand stops when the small box contact to large one.

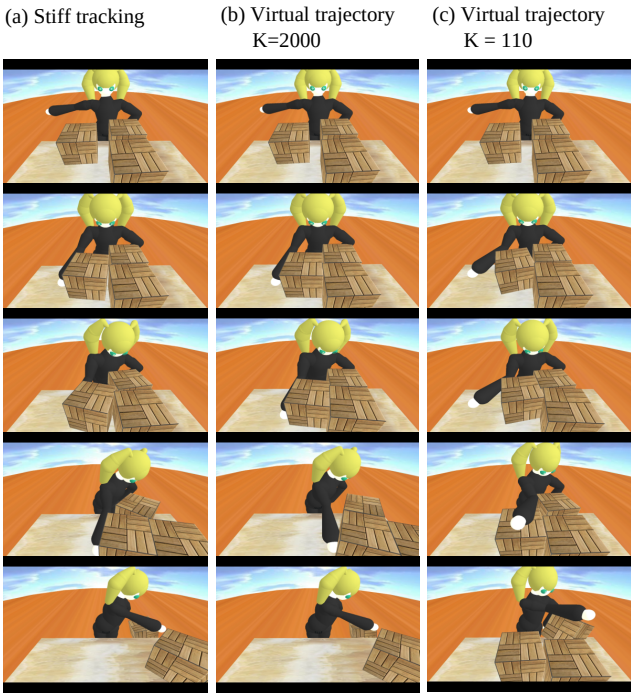

Figure 1: *Comparison of generated motion with (a) simple PD control, (b) proposed method with stiff feedback coefficient and (c) proposed method with soft feedback coefficient*

6 Conclusion

It is difficult to generate a trajectory which trace surfaces of desks or walls. We think human beings generate such motions by complying with environmental constraints using virtual trajectory tracking. Proposed method mimics this motion generation method of human and easily generates realistic motion trajectories. Proposed method generates motion trajectory in realtime with low computatioal amount using LCP solvers in physics engines.

References

HOGAN, N. 1984. An organizing principle for a class of voluntary movements. *Journal of Neuroscience 4*, 2745–2754.

ZORDAN, V. B., MAJKOWSKA, A., CHIU, B., AND FAST, M. 2005. Dynamic response for motion capture animation. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, ACM, New York, NY, USA, 697–701.