Multi-touch Interface for Character Motion Control

Masaki Oshita
Kyushu Institute of Technology
oshita@ces.kyutech.ac.jp

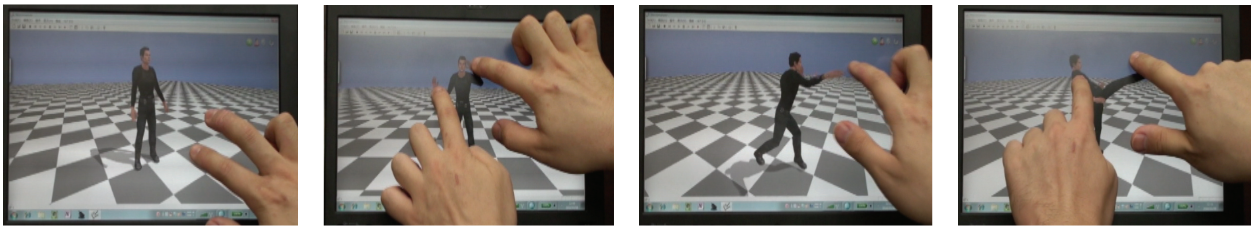

Figure 1: Example of a multi-touch interface for character motion control. By simply dragging body parts using the multi-touch input device, a user can control a character's pose freely and make the character perform various actions.

1 Introduction

Currently the flexibility of character motion control in interactive applications is very limited. Since common input devices such as a gamepad, mouse or keyboard. have only a small number of buttons, the user can do nothing but simply select an action from a few pre-created actions by pressing an associated button and make a character perform any fixed action such as walking, punching or kicking. Sometimes a user wants to pose the character freely or make the character perform various actions in the user's own style. This is particularly important in some applications such as fighting games, a dancing animation or online communication using avatars. Using inverse kinematics (IK), a user could control a character's pose in detail. However, with conventional input devices, the user can control only one end-effect at a time.

We propose a multi-touch interface for character motion control with which a user can control a character's pose freely and make the character perform various actions by simply dragging the character's body parts using a multi-touch input device (Figure 1).

Our method employs a style-based IK [Crochow et al. 204][Shin and Lee 2006]. In general, a style-based IK generates a natural-looking pose by mapping a large number of sample postures onto a low-dimensional latent space in advance and finding a pose that satisfies given constraints in the latent space. However, the style IK has several problems. First, it cannot handle many kinds of actions in a single style IK model because appropriate poses depend on the kind of action to be expressed. Second, it is difficult to generate a continuous motion from given touch strokes of body parts because there is no guarantee that a continuous trajectory in Cartesian space is mapped to a continuous trajectory in latent space. Previous studies [SMHP04][BL06] only generate motions from a given trajectory in latent space or modify existing motions based on a modified trajectory in Cartesian space.

2 Our Approach

We solve the first problem by preparing different style IK models for each action and choosing an appropriate model based on the user's input. More specifically, we use a single style IK model for posing and many styles of IK models for actions. Although [Shin and Lee 2006] used several styles of connected IK models, unlike our system, the user of their system had to switch between the models manually.

For posing control, we construct a single style IK model from various poses and the initial poses of the actions. By finding a pose based on touch inputs, i.e., screen positions which are converted to half lines in the Cartesian space, the user can make the character pose and perform some gestures (e.g., moving one's hands or shaking one's head). Since there is no guarantee that the given constraints are always satisfied, we also apply a conventional IK after a style IK.

For action control, when a touch stroke for a body part matches the initial trajectory of the primary body part of a prepared action model, the system switches to action control by using the style of IK model that is trained from the motion data corresponding to the action. We compare a touch stroke with the projected trajectory of each action. When the angle between them is small, the action is eliminated from the candidate. For example, to make the character punch, the user has to control the camera to see the character from the side. We think that this is a proper constraint. During action control, the system finds a point that matches the touch inputs in the latent space in the same way as posing control. However, when the found point is away from the current point, the system progresses the point toward the found point to ensure the continuity of the generated action. When the user lifts the finger, the system automatically progresses or rewinds the action time until the action finishes and then goes back to posture control.

3 Implementation and Future Work

We implemented the proposed method using the Windows 7 Touch API on a multi-touch-enabled PC. We are also developing an iPad/iPhone version. We tested the proposed system with a small number of action data. As can be observed in the accompanying video, the generated pose sometimes looks unnatural because we apply a conventional IK to satisfy touch constraints. In addition to conducting a user study, finding a balance between satisfying the constraints and achieving a natural motion will be part of the future study.

References

Grochow, K., Martin, S. L., Hertzmann, A., Popovic, Z. 2004. Style-based Inverse Kinematics. *ACM Transactions on Graphics (SIGGRAPH 2004)*, 23, 3, 522-531.

Shin, H. J., Lee, J. 2006. Motion Synthesis and Editing in Low-Dimensional Spaces. *Computer Animation and Virtual Worlds (CASA 2006)*, 17, 3-4, 219-227.

SIGGRAPH 2010, Los Angeles, California, July 25 – 29, 2010.
ISBN 978-1-4503-0210-4/10/0007