Guided Control : A System for Directable Characters

Daniel Taranovsky and Michiel van de Panne
Department of Computer Science, University of Toronto
{dannyt | van} @ dgp.utoronto.ca

Introduction

Controlling the motion of virtual characters with many degrees of freedom can be difficult and time-consuming. For some applications, complete control of all joints at every time step is not necessary and actually hinders the creative process. On the other hand, endowing the character with autonomous behaviour and decision-making capabilities does not allow the user to clearly specify his/her intentions. In many circumstances the ideal level of control is task-level specification accompanied by timing and stylistic parameters. We implemented a prototype animation system that addresses this relatively unexplored level of motion control termed *guided control*.

Guided control endows the user with less ambiguous motion specification than behavioural techniques without cumbersome low-level details typical of keyframing methods. The contribution of this work is a guided control prototype that realistically responds to a rich set of motion directives specified interactively by the user.

There are several main issues to be resolved in implementing guided control. First, the user must have a mechanism for specifying objects, space, and tasks to generate relatively complex motions. In particular, object manipulation, cooperative use of both hands, and concurrent execution of tasks is supported by our prototype system. Second, the puppet should respond to the user's directives with human realism. Some synthetic intelligence must be endowed to realistically execute tasks in arbitrary environments.

Task Specification

The prototype implements a virtual puppet seated at a table as illustrated in Figure 1. The user inputs a script indicating the position and dimensions of the table and objects in the scene. The user associates objects and space with tasks, and the puppet adapts its motion according to the state of the environment. Figure 1 illustrates the command "*reach object A with left hand*" executed in four different scenarios.

Ideally one would like to offer a set of tasks and motion primitives rich enough to not limit the user's creativity. There are three categories of tasks implemented in our system, each associated with a different set of motion parameters. Reaching motions place one of the puppet's hands at an object or pre-defined point in space. When reaching for an object the user specifies whether the top, side, or bottom of the

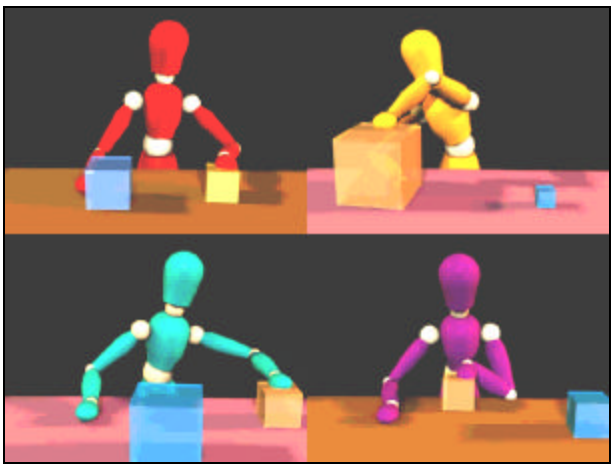

Figure 1: Performing a task in variable environments.

object is to be grasped. Sliding motions move one of the puppet's hands relative to its current position. The user specifies a vector and matrix corresponding to the hand's differential position and goal orientation respectively. Finally, general tasks are pre-defined motions that cannot easily be produced as reaching or sliding tasks. An example of each type of task is presented in Table 1.

Reaching	"*reach object A with left hand*"
Sliding	"*move left hand (x, y, z, q, y , d)*"
General	"*wave with left hand*"

Table 1: User-specified commands.

Every task is associated with a set of *critical body segments*, which refer to the subset of the puppet's anatomy that is preoccupied with the current task. What to do with non-critical parts (referred to as *secondary body segments*) is an issue to be resolved. Our system assigns default motion to the secondary segments that can be overridden interactively by the user. For example, the torso and left arm are critical to the task "*reach object A with left hand*", while the head and right arm are secondary. The system default will position the head to observe the reaching hand while the right arm rests on the table, as illustrated in Figure 1.

Default secondary motion can be overridden by the user with *locks* or *concurrent execution*. Locking a task to a hand will result in overriding the system default whenever the particular hand is among the set of secondary body segments. For example, if one locks the task "*reach object A with left hand*", then the left hand

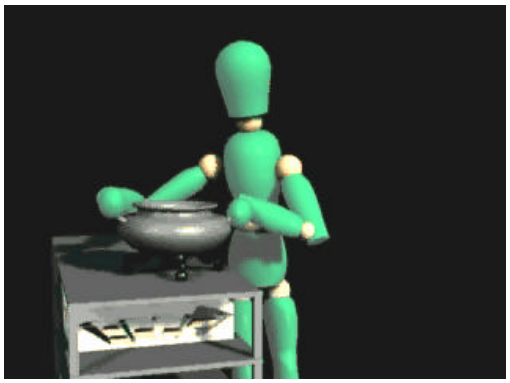

Figure 2: Moving cauldron with both hands.

will rest on object A while the right hand is free to perform other tasks as directed by the user.

The user can also specify two tasks to be executed concurrently. If two tasks' sets of critical segments are mutually exclusive, or the intersection of the two sets can be shared among multiple tasks, then motion concurrency is possible. The user can direct the puppet to perform two independent tasks, or use both hands cooperatively to perform one task. Figure 2 illustrates the puppet grasping and lifting a cauldron with both hands.

Motion Synthesis

Ideally, the system should map elements in task-space to a motion signal that satisfies the intentions of the user with realism indistinguishable from human motion. In our prototype, motion is synthesized by interpolating between humanly natural postures.

Modeling unconstrained multi-joint movements is an on-going research area, although some scientists have observed a relationship between a joint's velocity and its potential for moving the end effector towards the target. The velocity function of constrained single joint movements is observed to be a smooth bell-shaped curve. This curve is used as an interpolation function for the entire body after scaling the function to each joint's amplitude of motion.

The system computes postures by translating the user's command into a hand position and orientation that accomplishes the task. Once end effector goals are determined, an inverse kinematics engine is invoked. The algorithm proceeds as follows:

1. Estimate initial natural posture.
2. Invoke IK algorithm to satisfy hand position and orientation constraints.
3. Score the solution(s) found in step 2 to determine the best posture.

The first step uses neurophysiological data to position the arm naturally with respect to the hand's positional goal. The IK algorithm is invoked three times with varying stiffness applied to the joints. From the three postures computed, the best is determined according to a score function. The IK algorithm attempts to satisfy the geometric constraints imposed by the task while preserving the naturalness of the original estimate.

Our score function quantifies the naturalness of a posture based on principles from ergonomics research. Certain posture characteristics are responsible for uncomfortable working conditions and serious injury if maintained for long periods. To achieve a comfortable seated position, one should :

- Avoid the limits of joints' range of motion.
- Minimize torsion of the torso and waist.
- Keep elbows low.

The score function rewards postures for satisfying position and orientation constraints while respecting the above characteristics of a natural seated posture.

Conclusions and Future Work

The prototype implemented has produced animations such as stacking blocks, pouring coffee, and playing cards. The experience gained from this project suggests that guided control is most practical for producing motion where cost is a priority over quality. Characters found in the background of scenes, a large crowd of figures each performing unique tasks, or preliminary motion for foreground characters to be later refined are all cases that would benefit from a guided control system.

The postures in Figure 1 each took approximately thirty seconds to compute on a Pentium II processor. More complicated postures, such as the one demonstrated in Figure 2, can take several minutes to compute. Much more can be done to improve the performance of the IK engine. However, if a real-time algorithm is developed, interaction issues will arise when trying to quickly specify sophisticated tasks.

The current motion model of interpolating between natural postures needs to be upgraded in two respects. First, applying a scaled interpolation function to all joints does not correspond to observed biomechanical data. It is clear that not all joints achieve their peak velocity at the same time even though a general relationship between a joint's acceleration and the hand's target location has not been discovered. Furthermore, planning motions to navigate arbitrary environments is an on-going research area. Our current implementation ensures collisions with the table is avoided, but does not consider objects placed in the scene.

References

[1] Norman Badler, Cary Phillips, Bonnie Webber. *Simulating Humans.* Oxford University Press, Oxford, NY, 1993.

[2] Mark Latash. *Control of Human Movement*. Human Kinetics Publishers, Champaign, IL, 1993.