Report of Mini-Project #3

Peihong Guo

Department of Computer Science and Engineering

November 28, 2011

1 Brief Introduction

In this miniproject, I implemented both forward kinematics and inverse kinematics, and tested the performance of both methods using the motion capture data from CMU Graphics Lab Motion Capture Database[1]. The implementation for forward kinematics is straightfoward with the help of given start-up code, and the inverse kinematics is implemented following the papers by Zhao[3] and Buss [1]. The levmar[2] nonlinear optimization library is utilized for solving the optimization problem in IK process.

The human model used has 30 bone segments and 62 degree of freedom. Root position and rotation is also provided for each frame in the motion capture data. Figure 1 shows the model in initial state and one frame from a motion sequence.

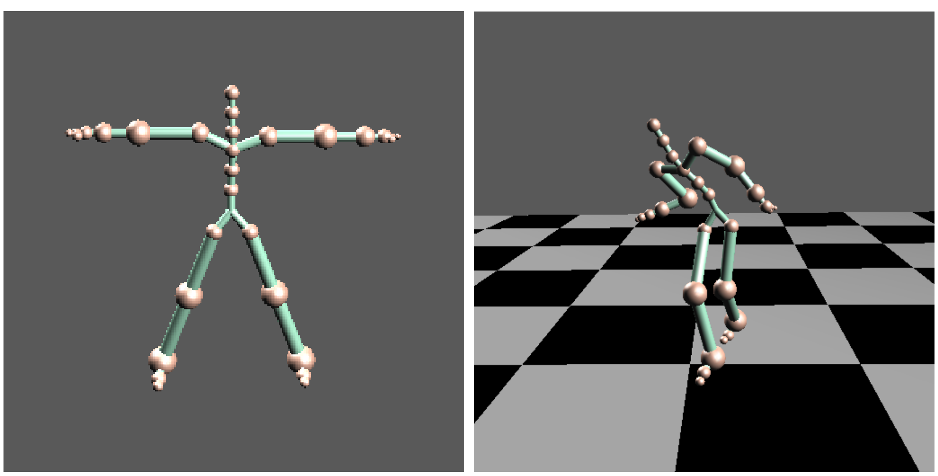


Figure 1: Left: Human model; Right: a frame from one motion capture data.

[1] http://mocap.cs.cmu.edu/

2 Results

2.1 Forward Kinematics

The tareget for forward kinematics is to calculate the global coordinates of any position on a model from given combination of joint angles. One major challenge for this process is deriving correct transformation between the local coordinates frames of adjacent bone segments. Given the rotation of local coordiantes frames against global coordiantes frame, the transformation can be determined as a combination of several rotational transformations and translational transformation.

Figure 2 is a result for forward kinematics: a position on the human model is picked and its trace in the motion sequence is visualized as a golden curve in the 3D space. Please refer to the video clip *trace.avi* for more details about the movement and trace of a specified position on the model.

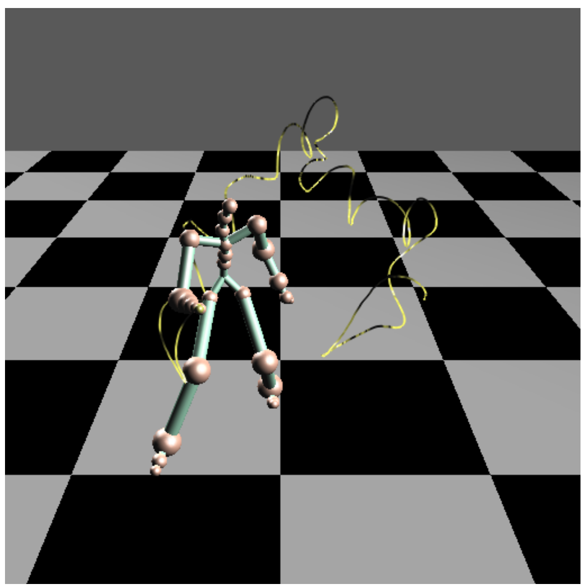


Figure 2: Trace of a picked position on the model. Picked position is labeled with a golden sphere on the model.

2.2 Inverse Kinematics

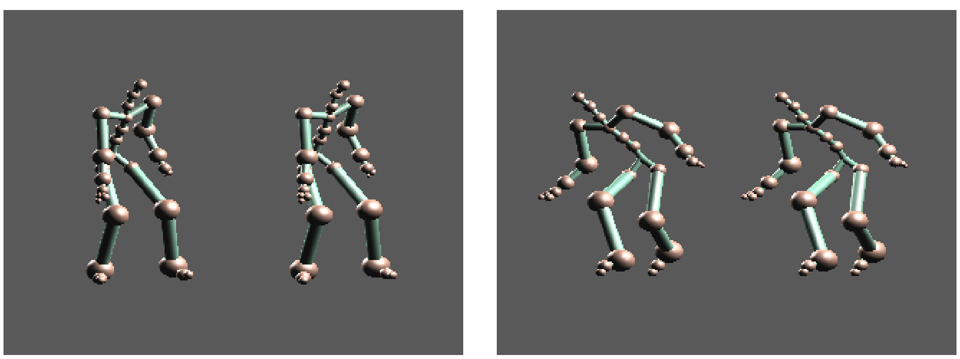
The inverse problem of forward kinematics is to estimate a combination of joint angles which corresponds to a model configuration that satisfies certain constraints, which is known as inverse kinematics. In this miniproject, 2 scenarios are used to test the performance of the implemented inverse kinematics algorithm.

2.2.1 Per-frame inverse kinematics

The first scenario is a per-frame inverse kinematics, in which the constraints are all joints positions in a specified frame. Figure 3 shows some results for this scenario. In each image, the left one is the reference configuration and the right one is the result by inverse kinematics. Initially all joint angles are assigned random values, and the result given by IK algorithm converge nicely to the reference configuration.

To achieve nice results with the IK algorithm, a good starting point is crucial. One possible choice for good starting point is the joint angles from a neighboring frame, which is usually very similar to the target state. However, for the cases of random start point, a iterative scheme for progressily improving results should be applied. The basic idea is to use privously achieved configuration, which is usually close to the global optimal state, as a starting point for next iteration. In my implementation, on average 5 iterations are enough for highly precise convergence.

The video clips $ik_{r}andom.avi$ (IK with random starting points) and $ik_{n}eighbor.avi$ (IK with neighboring frame configurations as starting points) shows the performance of the IK algorithm over a long sequence of animation.

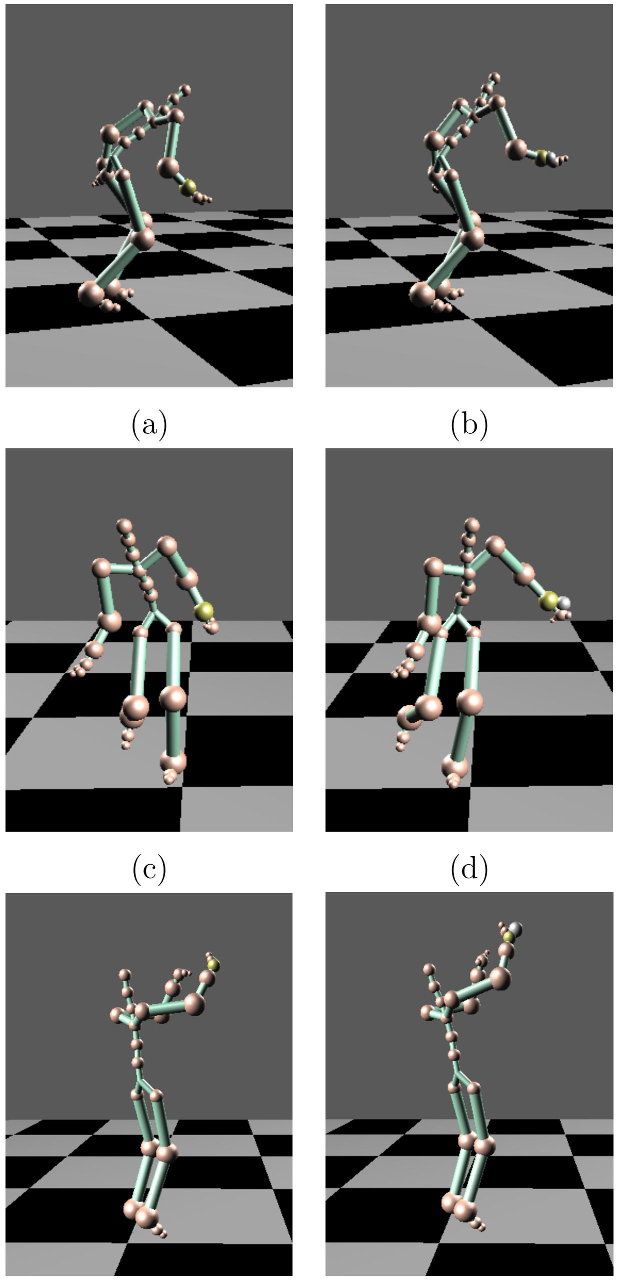


(a) (b)

Figure 3: Result of inverse kinematics of 2 frames. In each image, the left one is reference state, and the right one is the result of inverse kinematics. Note that orthogonal projection is used to eliminate the visual difference introduced by projection transformation.

2.2.2 Interactive Manipulation

Interactive manipulation is a little different from per-frame inverse kinematics. For interactive model manipulation, we only need to consider one end effector, while we also need to maintain smooth transition from previous configuration to target configuration. Therefore the optimization aims to minimize the spatial difference between the position of end effector and the specified position, as well as the total amount of difference between newly generated configuration and previous configuration. Joint angle limits are utilized for this scenario to avoid impractical configurations. Figure 4 shows several results.



(a) (b)

(c) (d)

(e) (f)

Figure 4: Results for direct manipulation of model. The golden sphere represents the position of end effector, and the silver sphere is corresponding target position.

References

[1] Samuel R. Buss. Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. Technical report, IEEE Journal of Robotics and Automation, 2004.

[2] M.I.A. Lourakis. levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++. [web page] `http://www.ics.forth.gr/~lourakis/levmar/`, Jul. 2004. [Accessed on 31 Jan. 2005.].

[3] Jianmin Zhao and Norman Badler. Inverse kinematics positioning using nonlinear programming for highly articulated figures. *ACM Transactions on Graphics*, 13:313–336, 1994.

Manual

Qt library is required to build the system. To compile the source code, please make sure Qt 4 is correctly installed and configurated. The source code is tested ONLY in 64 bit Linux system.

Compile Procedures:

1. tar xvf *package*.tar.gz
2. cd *package*/
3. qmake (or qmake-qt4)
4. make

Usage: The program provides a GUI for convenience.

- Load model: Use the "Load Model" entry in "File" menu.
- Load animation sequence: Use the "Load Sequence" entry in "File" menu.
- Replay animation sequence: Press the play button (Green triangle icon) on the GUI. Note that you must first load a model then load a sequence before doing replay.
- Interactions:
  1. Show trace: Select the bone segment and set corresponding normalized bone position, the press the show trace button (Curve icon) on the GUI. The trace will appear when replay the motion.
  2. Show all joint positions: Press J when finish loading an animation sequence data.
  3. Interactive manipulation: Press the cube icon to enter interaction mode, then press Ctrl key to enter direct manipulation mode. Now you can pick any joint and drag it to a wanted position. New configuration of the model will be generated upon release of mouse button.
  4. View per-frame IK result: Press I. Note that the IK process may take a few seconds, so it is better not perform replay in the same time.
  5. Jump between frames: PageUp/PageDown to move to next/previous frames. Or you can simply specify a frame number in the control panel.