Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2016)
Jan Bender (Editor)

Data-driven Finger Motion Synthesis with Interactions

M. Bitan[1] and S. Jörg[2] and S. Kraus[1]

[1]Bar-Ilan University, Israel
[2]Clemson University, South Carolina

1. Introduction

The film and video-game industries' rapidly increasing demand for realistic virtual characters is pushing for the development of fast and efficient character animation techniques. The use of motion capture has become an industry standard. While motion capture systems allow the production of high fidelity full-body motions, in most cases, the complex finger movements are later animated manually. Several methods have been suggested to automatically synthesize finger motions. An overview can be found in Wheatland et al. [WWS*15]. Most of these methods either do not consider interactions and collisions or are computationally intensive.

In this poster, we present a data-driven approach that takes interactions into account but is less computationally expensive than some previous approaches. We expand Jörg et al.'s data-driven approach for synthesizing finger motion for conversational characters [JHS12] and provide support for generating finger motion in cases where the hands interact with objects or with the virtual character itself. Our system takes into consideration the proximity of the fingers from colliders as well as the adjustments needed to correctly animate the fingers while avoiding intersections with objects in the scene. Using this technique, animators can quickly and easily generate finger movements for previously captured or manually animated body motion clips.

2. Proposed Method

We created a database of 135 clips captured using a Vicon motion capture system. An actor performed different actions interacting with a variety of objects and body parts, for example, scratching his head or picking up a cup. In addition to the standard body marker set, 24 markers were placed on each hand.The data was thoroughly post-processed and retargeted to a virtual character.

Our system receives as input a full body motion clip and the interacting geometry, and creates finger motions as output. The synthesis process consists of the following steps:

1. Segmentation The input motion and database are segmented based on the wrist proximity to the colliding geometry.

2. Collision Correction Selected promising finger motion segments from the database are applied to the input body motion, which can result in slight intersections. We use collision detection and inverse kinematics [AL11] to alter the finger motions to avoid intersections with the geometry and create a corrected database.

3. Segment Cost The algorithm searches for the k-most similar segments in the corrected database. The similarity is calculated using the following segment cost function c_s:

$$\frac{1}{n}\sum_{t=1}^{n}\left(w_p\|P_t^I - P_t^D\| + w_r\|R_t^I - R_t^D\| + w_c\sum_{j=1}^{m}\|R_{t,j}^{D^*} - R_{t,j}^D\|\right)$$

where n is the number of frames, m is the number of finger joints, I, D and D^* are the input, database and corrected database segments, respectively, P_t and R_t are the wrist position and orientation, and w_p, w_r and w_c are weights for the position, rotation and collision correction, respectively.

4-6. Transition Cost; Motion Graph; Segment Combination Following Jörg et al. work [JHS12] we compute transition costs for consecutive segments, create a motion graph that allows to combine any of the segments from step 3, find the shortest path through the graph considering segment and transition costs, and correct small differences when combining the final finger motion clips.

7. Collision Correction In order to ensure that the output finger motion does not penetrate the geometry, the collision correction step is executed a second time. Lastly, Kalman filter was used to reduce any jittering that may occur due to the collision correction.

3. Discussion

While first results are promising, our method has drawbacks. If motions are being retargeted to a new character, the whole hand motion might have to be corrected using the IK solver, which can lead to body motions that are different from the original input motion. Furthermore, the collision correction produces jittering fingers and the Kalman filter may re-introduce subtle collisions. Finally, we need to thoroughly evaluate and optimize our segment selection method.

References

[AL11] ARISTIDOU A., LASENBY J.: FABRIK: A fast, iterative solver for the inverse kinematics problem. *Graphical Models 73*, 5 (2011), 243–260.

[JHS12] JÖRG S., HODGINS J., SAFONOVA A.: Data-driven finger motion synthesis for gesturing characters. *ACM Transactions on Graphics 31*, 6 (2012), 189:1–189:7.

[WWS*15] WHEATLAND N., WANG Y., SONG H., NEFF M., ZORDAN V., JÖRG S.: State of the art in hand and finger modeling and animation. *Computer Graphics Forum 34*, 2 (2015), 735–760.

DOI: 10.2312/sca.20161244