MotionBender: A Gesture-based Interaction Technique For Editing Motion Paths

Frederic Raber[l], Quan Nguyen[l], Michael Kipp[2]

[l]DFKI GmbH, 66123 Saarbrücken
[2]Augsburg University of Applied Sciences, An der Hochschule 1, 86161 Augsburg

`{Frederic.raber,quan.nguyen}@dfki.de,michael.kipp@hs-augsburg.de`

Abstract. Precision tasks in 3D like object manipulation or character animation call for new gestural interfaces that utilize many input degrees of freedom. We present MotionBender, a sensor-based interaction technique for post-editing the motion of e.g. the hands in character animation data. For the visualization of motion we use *motion paths*, often used for showing e.g. the movement of the hand through space over time, and allow the user to directly "bend" the 3D motion path with his/her hands and twist it into the right shape. In a comparative evaluation with a mouse-based interface we found that subjects using our technique were significantly faster. Moreover, with our technique, subject movement was more coordinated, i.e. movement was done in all three dimensions in parallel, and the participants preferred our technique in a post-experiment questionnaire. We also found a gender effect: male users both like the gesture interaction better and achieve better performance.

Keywords: coordination; character animation; motion trajectory; Kinect; 3D user interfaces

1 Motivation

Character animation is required in many fields, from movie production and games to anthropomorphic interfaces. However, animation is a complex craft that requires high expertise. Current animation approaches can be divided into two categories. In *interactive methods* animation is created on-the-fly. The animator either moves a virtual model using his/her limbs or a controller. The model's motion is recorded during this animation process. In *keyframe methods* the animator defines several decisive "key" frames, using a 3D modeling tool. In-between frames are automatically computed by interpolation. This is the standard method in the industry. Neff et al. [5] presented an interactive system where this animation is done via the mouse. The movement of the mouse is mapped to respective joint movements by correlation maps. Kipp/Nguyen [3] explored the possibilities of a multitouch-surface to interactively animate the arm and hand of a puppet. There are several articles [4][2] which describe keyframe methods, using a motion trajectory for visualizing and editing the animation of each joint.

In these solutions the trajectory is modified with conventional input devices like a mouse which have few degrees of freedom.

We propose a novel interaction technique called MotionBender (Figure 1) for editing motion paths by combining the possibilities of direct manipulation using a vision-based motion sensor (Kinect) and a simple button for switching between edit modes. The modes differentiate between selecting where to "grab" the path and moving these parts to "bend" the path into a new shape.
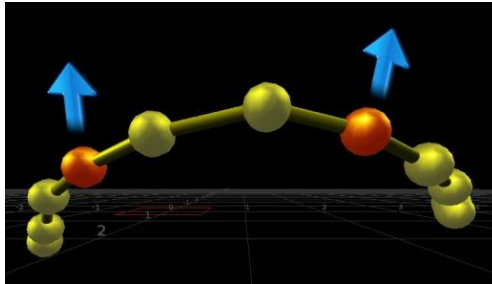
Fig. 1. In character animation, the movement of a joint, like the hand/wrist is visualized as a motion path. With our technique the user can modify this path by grabbing it (two red spheres) and bending it. The goals of our editing technique are:

- **Effective**: Suitable and useful for post-processing character animation data.
- **Efficient**: At least as fast as the current industry-standard techniques
- **Intuitive**: Can be directly used without too much instruction.
- **Joy of use**: Users should enjoy working with this technique because they feel "in control" in the sense of a *Natural User Interface* (NUI)

For such a gesture-based interface, it is important that the user makes use of all degrees of freedom that such an interface offers. If s/he uses the system in a sequential way, one dimension at a time, the benefit against conventional input systems is decreased. We used a coordination measure in our evaluation to measure how coordinated the user's movement is.

2 Interaction technique

2.1 Overview

Our system is called *MotionBender*. It was created with a system like Min et al.'s [4] in mind, providing the possibility for a sensor-based interactive editing of the motion paths. Like in Min et al.'s work, every joint (e.g. the wrist of one hand) has a motion path, describing the movement of this joint during the animation, as shown in Figure 1. The spheres represent the positions of a single joint at successive time points. The user selects two of the spheres on the curve (marked red) and moves them to another location in the virtual world. The surrounding parts of the motion path are automatically adjusted. After the user has finished, MotionBender recomputes the

animation of the joint according to the new motion path. Figure 2 shows a user using MotionBender in a typical working situation. The corresponding video can be found online[1].
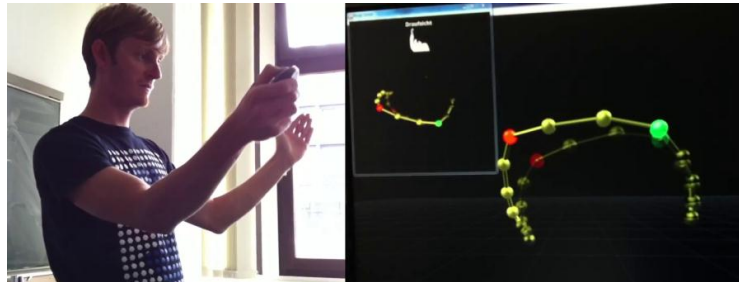
Fig. 2. User working with the MotionBender interface.

2.2 Workflow

We focus on the editing of a single motion path. The manipulation is done with a Kinect sensor and a two-button device in one hand. There are two modes: *grab* and *bend*. The user stands in front of the computer. In grab mode, the user selects two points on the motion path (marked red in Figure 1) where the path is "grabbed". This is done by clicking and holding the first button and moving the hands apart and together on a horizontal axis in front of the user. In bend mode, activated by clicking and holding the second button, the two selected spheres and their neighboring spheres follow the user's hand movements in 3D space (indicated by the blue arrows in Figure 1). Figure 3 shows how neighboring points (yellow) are moved in relation to the grabbed points (red).
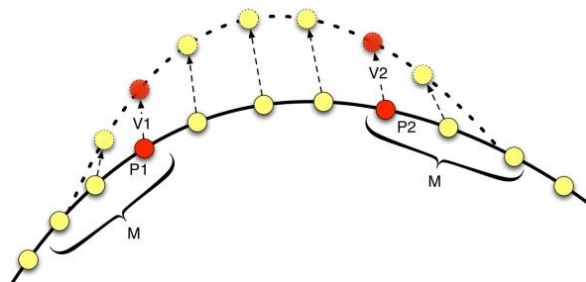
Fig. 3. Effect of the neighbor function

To compute the change for the neighboring points, we differentiate between three types of points. Let us assume that points P1 and P2 have been moved by vectors V1 and V2.

---

[1] http://tiny.cc/MotionBender

- Points between P1 and P2 are moved by a vector which is a weighted interpolation between V1 and V2, where the weight corresponds on the distance to the points.
- All other points, if they are less than M points away, are moved into the same direction as the nearest point moved by the user, but with a decreased distance. Formally, for every point P which is nearer to P1, its movement vector V will be calculated as $V = k * V1$, where k decreases linearly with increasing distance between P and P1.
- All other points are not moved.

3 Experiment

3.1 Methods

Participants

One of the main goals of our work was to create a tool which can also be used by novices in character animation or similar fields. Therefore, we decided not to take experts for our evaluation, but lay people. We tested 21 participants. Two were taken out due to aborted runs. We thus analyzed 19 people (12 female, 7 male) of age 18-56 years (average 24). Two of the subjects are left-handed but use the mouse with their right hand. Participants were not paid.

Control Condition (Mouse Interface)

When trying to design the control condition, we started out with a conventional one-mouse interface, using a graphical widget to allow control over the various degrees of freedom. However, as Owen's work on bimanual curve manipulation [6] suggests, this would lead to an unfair comparison, as bimanual input is more efficient than unimanual. Since the Kinect also has more DOF, we had to extend this to make a fairer comparison. To allow the simultaneous usage of two hands, we designed an interface with two mice, one for each "grab point". As for the control mapping of each mouse, we made the x/y mouse motion control the x/y motion on the screen (frontal plane), whereas the scroll wheel would move along the depth axis (z axis). One could argue that the scroll wheel is usually used for up/down motion (y axis). However we deemed the mapping from x/y mouse motion to x/y motion in 3D space even more common, and since the mouse is usually pointing toward the screen the scroll wheel acts like physical metaphor of rolling with a wheel into the screen. In user studies, participants did not negatively comment on the scroll wheel mapping.

There are 12 trials, divided into two subsets A and B of 6 items each. Each subset was randomized in a controlled manner (e.g. reversed) to obtain new sets A' and B'. Each participant worked on either A or A' for one condition, and B or B' for the other condition. To avoid training effects on one condition, the first half of the subjects started using the condition "Kinect" and continued with the "Mouse" condition, and vice versa for the second half of the subjects.

Apparatus

For the experiment the user had to stand in front of a 22 inch screen in a distance of about two meters. For the two-mouse interface, the subject was sitting directly in front of the screen. The complete surface of the office desk was available for him to act upon with the mice (Figure 4). We are aware of the fact that the difference in the user-screen distance could give a slight advantage for the mouse condition, based on the assumption that for precision tasks it is better to be closer.
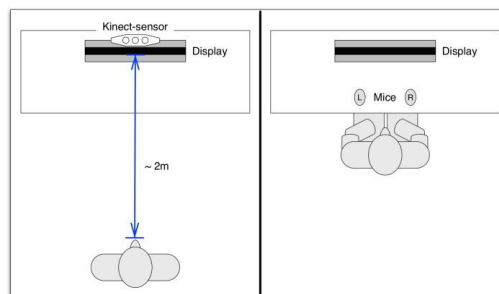
Fig. 4. The Kinect setup (left) and the two-mouse interface (right).

Task

The task was to match a given trajectory to a target trajectory (Figure 2). Each subject uses both interface versions and has to match six different trajectories for each interface. A trial is done if either the distance between given and target trajectory is below a certain threshold (successful match) or the maximum edit time (120 sec) has expired. In either case, the user directly continues with the next trial.

Procedure

The test procedure for all test candidates was as follows:

- At the start of the experiment the subject reads the written instructions.
- After the subject finishes reading, the experimenter gives a demo of the interfaces and gives hints on solution strategies.
- After a short question and answer session, the subject can practice on two sample trials, which will not be part of the main evaluation phase later. The sample trials and their order are always the same.
- The main evaluation begins, where the subject is presented 6 trials to be matched in the first interface condition (Kinect or Mouse).
- Phases 2-4 are repeated for the second condition (Mouse or Kinect)
- The experiment ends with a short two-page questionnaire, where the subject expresses his/her experiences with both interfaces in a multiple-choice form on the first page, and gives free feedback about problems/issues on the second page
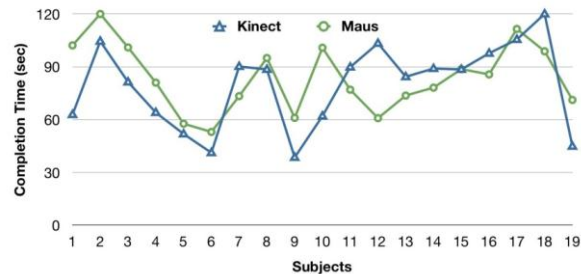
Measures

For our analysis, we computed three measures.

1. **Completion time:** Average time needed to complete a trial.
2. **Mean squared error** of all points which have not been matched yet.
3. **Coordination**: For comparing the movements regarding coordination, we use the measure defined by Kipp/Nguyen [3]. Coordination measures whether movement is performed in all three dimensions in parallel (high value) or sequentialized, i.e. performed along one dimension at a time (low value). Alternative coordination measures [1][7] are based on a fixed optimal solution path. For our problem, the optimal solution may be counterintuitive. The various selection possibilities for selecting grab points further complicate the matter. Therefore, such optimal-path metrics may yield misleading results. Kipp/Nguyen's measure is independent of optimal paths, taking only the raw movement vectors into account.

3.2 Results

Completion time

Figure 5 shows the average completion times over all participants and motion path samples. A paired t-test proved that subjects are significantly faster with the Kinect (M=79.36; SD=23.9) compared with the mouse (M=83.71; SD=19.02): t(18)=3.89, p<0.01. To test for gender, we analyzed males and females separately. While the performance for the mouse was statistically equal, we found that, for the Kinect, male participants were much faster (M=65.24) than female ones (M=87.59) which we found to be highly significant (t(17), p-value<0.001).

Fig. 5. Average completion times over all subjects.

Coordination

Using the above mentioned coordination measure [3] we compared the performance for the left and right hand separately over all participants and devices (see Figure 6). In both cases the Kinect interface encourages significantly higher coordination: For the mouse device the coordination values are almost zero (left hand 0.009, right hand 0.007), whereas the Kinect interface reaches a value of 0.29 for the left hand and 0.26 for the right hand.
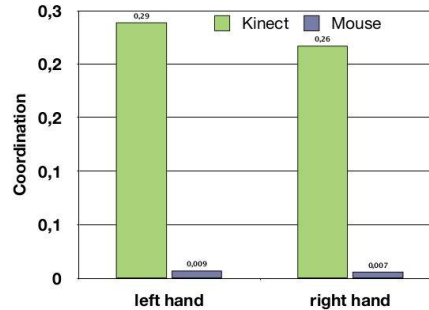
Fig. 6. In terms of coordination Kinect (green) significantly outperforms mouse (blue).

Questionnaire

In a post-experiment questionnaire we asked the participants questions which discriminated between Kinect and mouse on a 5-point differential scale where the middle position was labeled "both or equal". This questionnaire contained 17 questions regarding usability and user experience. For analysis, we transformed this to an interval between -2 for mouse and +2 for Kinect. We summed up ratings for each question over all subjects and computed a chi square statistic, comparing the ratings against the expected neutral value of zero. Only three questions reached significance, all three in favor of the Kinect interface.

- Which interface was **easier** to use? (M=.21; chi²(2, n=19) = 11.47; p<0.005)
- Which device was more **fun** to use? (M=.84; chi² (2, n=19) = 8.58; p<0.05)
- Which device do you **prefer**? (M=.47; chi² (2, n=19) = 11.47; p<0.005)

4 Discussion

The results show that our MotionBender interface is objectively more efficient and subjectively better liked. Performance was significantly better, movement was clearly more coordinated and participants preferred it over the mouse interface and enjoyed it more. We also found statistical evidence that male users seem to profit more from our gesture-based technique than female users. Due to the low number of participants this may be an artifact and must be validated in future studies, which might also reveal a possible cause of that effect. However, time improvements were relatively small, compared to the overall task time. Our experiment design might have favored the mouse interface due to the closer user-screen distance. However, even if this was the case it only strengthens the results.

The Kinect was more coordinated than mouse. For mouse, users mostly acted separately, so they stopped moving the mouse when adjusting the depth position by rotating the scroll wheel. Some found this a limiting factor, others liked the fact that no unintentional movement along the z-axis was possible. The left hand was more coordinated than the right one in the mouse condition. We often observed that people are acting serially in matching the two selected points. They concentrated on the first

(left), neglecting the second one initially. On the sensor-based interface, there is less of a difference. This might also be a result of the reduced cognitive difficulty with the direct-manipulation concept. Some problems emerged because subjects had problems to visually understand the 3D scene. Some suggested that a 3D stereoscopic view could reduce problems. We had two views on the scene, a top and a front view. Some users were confused by these two different views, and temporarily lost orientation.

In one interesting case, a subject had general problems with precise hand movements. She found the Kinect interface much easier to use because due to the whole-body motion some focus was taken away from the precision of her hand movements.

Conclusion

We presented an interaction technique called MotionBender, suitable for Character Animation. We used the concept of motion paths for the visualization of the animation and provided an editing method for post processing the animation by grabbing and bending the motion path. For this, we used a combination of a motion sensor (Kinect) and a simple button controller (wireless mouse) as input devices. The technique was intended to be efficient, intuitive and enjoyable. We validated our technique against a mouse-based interface and found MotionBender to be faster, easier to use and more enjoyable. Although that a comparison of our technique against a single-mouse interface subjectively appears to be unfair, that question should be checked in a future work. We also found that our technique encouraged highly coordinated movement. A gender effect indicates that this technique particularly appeals to male users but this finding needs further study. Future work must examine how this technique could fit into the larger character animation workflow and whether experts would prefer this technique over the accepted industry standards.

5 References

1. Balakrishnan, R., and Hinckley, K. Symmetric bimanual interaction. In: Proceedings of CHI, 2000, 33–40.
2. Kim, M., Hyun, K., Kim, J., and Lee, J. Synchronized multi-character motion editing. In: ACM Trans. Graph. 28, 3, July 2009, 79:1–79:9.
3. Kipp, M., and Nguyen, Q. Multitouch puppetry: creating coordinated 3D motion for an articulated arm. In: ACM International Conference on Interactive Tabletops and Surfaces, ITS '10, ACM (2010), 147–156.
4. Min, J., Chen, Y.-L., and Chai, J. Interactive generation of human animation with deformable motion models. In: ACM Trans. Graph. 29, 1, Dec. 2009, 9:1–9:12.9
5. Neff, M., and Kim, Y. Interactive editing of motion style using drives and correlations. In: Proceedings of SCA, ACM, 2009, 103–112.
6. Owen, R., Kurtenbach, G., Fitzmaurice, G., Baudel, T., and Buxton, B. When it gets more difficult, use both hands: exploring bimanual curve manipulation. In: Proceedings of Graphics Interface 2005, GI '05, 2005, 17–24.
7. Zhai, S., and Milgram, P. Quantifying coordination in multiple dof movement and its application to evaluating 6 dof input devices. In: Proceedings of CHI, ACM Press/Addison-Wesley Publishing Co., 1998, 320–327.