FingerOscillation: Clutch-free Techniques for 3D Object Translation, Rotation and Scale

Siju Wu, Amine Chellali, Samir Otmane

▶ To cite this version:

FingerOscillation: Clutch-free Techniques for 3D Object Translation, Rotation and Scale

Siju Wu,[*]   Amine Chellali,[†]   Samir Otmane[‡]

IBISC Lab, Evry University, France

Abstract

In this paper, we present three freehand interaction techniques for 3D content manipulation, the *FingerShake*, for object translation, the *FingerRotate*, for object rotation and the *FingerSwing* for object scale. These three techniques refer to a more generic concept which we call *FingerOscillation*. The main contribution is to interact with the machine by using finger oscillation movements. We introduce the design and the implementation of these techniques.

Keywords: freehand gesture, 3D object manipulation, remote control, oscillatory motion.

1   Introduction

With the help of tracking devices such as the Kinect, it is possible to make remote interaction by using freehand movements in air. However, using freehand movements to manipulate objects displayed on the display intuitively and efficiently is still a challenge task. In this paper we present three techniques, the *FingerShake*, the *FingerRotate* and the *FingerSwing*, to manipulate objects in the 3D environment by using only finger movements. These techniques are inspired by the *CycloStar* which is presented in [Malacria et al. 2010] and can be generalized by a more generic concept, which we name the *FingerOscillation*.

2   FingerOscillation

The *FingerOscillation* is an approach to generalize techniques based on straight and circular periodical 3D finger movements by an elliptic oscillatory model. This model is an extension of the 2D model proposed in the work of [Malacria et al. 2010].

The *FingerOscillation* has several important properties:

- Humans are apt to make reasonably accurate cyclic gestures after training. The hand structure makes it easy to oscillate the hand in a quasi-harmonic fashion.

- Oscillatory gestures allows more continuous control of variables to be manipulated than other gestures such as dragging and pinching. Clutching can be avoided when repeating the same gesture.

- When performing oscillatory finger movements, hand and arm movements of large range are avoided and fewer muscles are involved in. The user can keep the hand in a relatively

small space in front of the chest. Interaction process becomes less tiring.

- The user does not need to change the hand posture to indicate the onset and the end of the manipulation explicitly.

3   FingerShake

The *FingerShake* is designed to translate virtual objects in the 3D space by using to-and-fro oscillatory finger gesture. This technique is inspired by the *CycloPan* proposed in [Malacria et al. 2010] and it is an extension of the *CycloPan* in the 3D space. To translate an object, the index finger should be dragged in the wanted direction (Figure 1). After the first stroke finishes, a second stroke can be done by dragging the index finger in the opposite direction. The object continues moving in the same direction although the stroke direction reverses. By dragging the finger forward and backward alternatively, the object keep translating in the initial direction until the finger stops moving. The translation distance and speed can be controlled by the amplitude and frequency of finger movements.
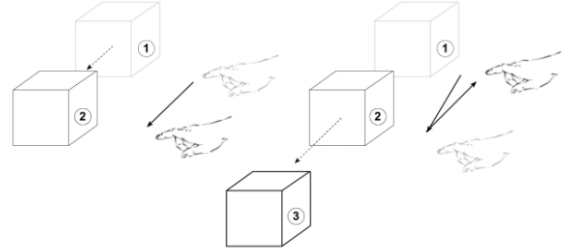

Figure 1: *Perform the FingerShake to translate an object.*

3.1   Implementation

To translate an object by using linear finger oscillation, first it is necessary to segment the finger trajectory to different strokes. For each frame, we check the curvature of the finger trajectory around the new generated finger position. If the curvature is small, the new finger position belongs to the active stroke. Otherwise we segment the trajectory and a new stroke is generated.

If the *FingerShake* is inactive, we check whether the new generated stroke satisfies a set of conditions to start the translation. Because the *FingerShake* requires the user to drag the finger in a linear way, so first we check whether the stroke can be fitted to a line. To avoid triggering the *FingerShake* involuntarily, the average speed of the starting stroke should exceeds 150 mm/s. In addition, it is also necessary to limit the length of the stroke, otherwise the *FingerShake* can be launched by a long drag gesture. In our implementation, we set the maximum length of the first stroke to 120 mm.

Once the *FingerShake* starts, the direction of the first stroke is chosen as the translation direction. The translation amplitude is calculated by using the following functions:

$$f_i = \frac{1}{2} * (T_{end\_i} - T_{start\_i}) \tag{1}$$

---

[*]e-mail:siju.wu@ibisc.univ-evry.fr

[†]amine.chellali@ibisc.univ-evry.fr

[‡]e-mail:samir.otmane@ibisc.univ-evry.fr

$$g_i = \max(1, k_1 * \frac{1}{3} * \sum_{i-2 \leq j \leq i} f_j) \qquad (2)$$

$$d_n = \sum_{1 \leq i \leq n} g_i * a_i * k_2 \qquad (3)$$

where f_i is the frequency of the ith stroke. T_{start_i} and T_{end_i} is the starting and end time of the ith stroke. g_i is the gain value which is used to control the translation amplitude. d_n is the total translation distance of the object and a_i is the amplitude of the ith stroke.k_1 and k_2 are two ratio parameters. The use of g_i makes it possible switch the translation velocity. To avoid jerky speed variations, the calculation of g_i takes use of the frequency of the last three strokes. The translation stops when the length of finger movement is shorter than 10 mm in 5 continuous frames.

4 FingerRotate

Similar to the *FingerShake*, the *FingerRotate* is also inspired by the work of [Malacria et al. 2010]. We have extended the concept of *CycloZoom+* to the 3D space. To rotate an object, the user has to oscillate the index finger in a circular way (Figure 2). Once a quasi-circle is drawn by the first stroke, the *FingerShake* is activated. The first stroke is fitted to a circle and the normal vector of the circle is used to determine the rotation axis. After that the object can be rotated around the selected axis by continuing the same finger movement. The rotation speed can be controlled by the circle radius.

Figure 2: *Perform the FingerRotate to rotate an object.*

4.1 Implementation

When dragging the finger in a circular way, the direction of the index movements changes fluently, so there is no jerky change of the trajectory curvature. We have proposed another algorithm to make stroke segmentation. For each frame, we approximate the active stroke to an arc segment of a circle and check the angle corresponding to the arc segment. When a complete circle is drawn, the angle reaches $360°$. At this time the *FingerRotate* is activated and the finger trajectory is segmented to generate a new stroke. When the user continue drawing a circle, we do the same thing to the new stroke to check the angle variation. Every time a complete circle is drawn, the center and the radius of the circle are saved. The normal vector of the circle is chosen as the rotation axis. For each frame, the rotation angle of the object is calculated by using the following function:

$$a_i = \frac{\|Q_{i-1}Q_i\|}{R} * k_3 \qquad (4)$$

where a_i is the angle variation.Q_{i-1} and Q_i are the projection points of the $(i-1)^{th}$ and i^{th} point of the stroke. R is the radius of the fitting circle and k_3 is a ratio parameter.

5 FingerSwing

We have proposed a technique, which is named *FingerSwing*, to make uniform scale of an object. The object can be magnified (or shrunk) by repeating drawing the upper (or the lower) part of an circle in the plane parallel to the display (Figure 3).

Figure 3: *Perform the FingerSwing to scale an object.*

5.1 Implementation

The stroke segmentation of the *FingerSwing* can be done by the same algorithm of the *FingerShake*. When the first stroke of the *FingerSwing* finishes, the system tries to fit the first stroke to an ellipse. If both of the long axis and the short axis of the fitting ellipse are longer than 4 mm, the system waits for the second stroke. When a drag direction reverses again, we compare the directions of two strokes. If they are performed in opposite directions, for example one in clockwise and the other in anticlockwise, the *FingerSwing* is recognized. After that, every time the stroke direction is reversed, the system uses the last two strokes to fit a circle. The scale of the object is refreshed by using the following function in each frame:

$$b_i = \frac{\|Q_{(i-1)}Q_i\|}{R} \qquad (5)$$

$$s_i = s_{i-1} * (1 + k_4 * b_i) \qquad (6)$$

where s_i is the scale of the object in the ith frame. b_i is the radian distance between the $(i-1)^{th}$ and i^{th} stroke point projections on the fitting plane. R is the radius of the fitting circle and k_4 is a ratio parameter.

6 Setup

We have implemented all the three techniques in a Thinkpad T510i laptop with a screen of 15 inches. A Leap Motion controller locates in front of the laptop and is used to capture hand and finger movements. The application development is done in the environment of the Unity3D by using C#.

7 Conclusion and perspectives

In this paper we present three techniques, *FingerShake*, *FingerRotate* and *FingerSwing* for object manipulation. All the three techniques refer to a more generic concept which we call *FingerOscillaiton*. In the future work, we want to compare the performance of the *FingerOscillation* with some other object manipulation techniques. We are going to make several experiments to examine the efficiency and the precision of our techniques.

References

MALACRIA, S., LECOLINET, E., AND GUIARD, Y. 2010. Clutch-free panning and integrated pan-zoom control on touch-sensitive surfaces: the cyclostar approach. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 2615–2624.