Distant Object Manipulation with Adaptive Gains in Virtual Reality

Xiaolong Liu[1] Lili Wang[1;2;3 *] Shuai Luan[1] Xuehuai Shi[1] Xinda Liu[1]

[1]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, China
[2]Peng Cheng Laboratory, Shengzhen, China
[3]Beijing Advanced Innovation Center for Biomedical Engineering, Beihang University, Beijing, China.

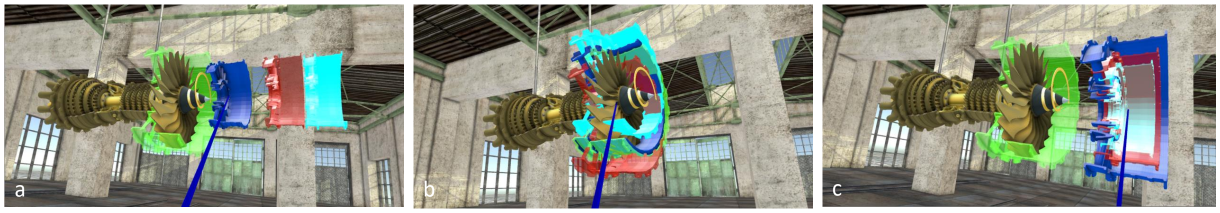

Figure 1: Views of component manipulation during aircraft engine assembly are displayed. Green indicates the target, cyan indicates the original position of the component to be manipulated, red shows the result of manipulation with $gain = 1$, and blue shows the result with our method. In (a) and (b), the user translates the component to the target. In (a), the original component is far from the target, and the gain computed with our method is greater than 1, so we translate the component faster than the manipulation with $gain = 1$. In (b), the original component is already very close to the target, and our adaptive gain is less than 1, so finer adjustments can be made to avoid the over manipulation problem caused by larger gains. In (c), the user scales the object to overlap the target. There is still a small distance between the original component and the target, and the gain calculated with our method is greater than 1, which can help the user quickly scale the component to a similar size to the target. When the two are closer, the gain of our method will become smaller, thereby helping users fine-tune scaling.

Abstract

Object Manipulation is a fundamental interaction in virtual reality (VR). The efficiency and accuracy of object manipulation are important to provide immersion to users. We propose a manipulation method with adaptive gains to improve the efficiency and accuracy of object manipulation in VR applications. First, we introduce manipulation gains. We then design an experiment to collect user behavior during manipulation to determine fitting functions for calculating manipulation gain. At last, we design a user study to evaluate the performance of our distant object manipulation method with adaptive gains. The results show that, compared with the state of the art methods, our method has a significant improvement in the completion time, and the manipulation accuracy of the tasks. Moreover, our method significantly increases usability and reduces task load.

Index Terms: Virtual reality—Object manipulation—Manipulation gains—Visual perception;

1 Introduction

Object manipulation is one of the most fundamental operations in 3D user interactions and can be used in many virtual reality (VR) applications, such as product design, 3D object modeling, and virtual object assembly. The efficiency and accuracy of the manipulation directly affect the effectiveness of the applications.

Existing methods mainly adopt two ideas to improve the accuracy and efficiency of object manipulation. The first idea is to manipulate objects through multiple manipulation points. These multiple points based manipulation methods [1, 10, 18, 19] require the user to manually input multiple points on the surface of the object or in 3D space, and then constrain the manipulation of the object based on these points or the axes formed by these points, which improves object manipulation accuracy. They are more suitable for direct manipulation of virtual objects by hand at close range, but when the manipulation object is far away from the user, it is difficult to manipulate. The second idea is to use the mapping between the physical motion of the user's hand and virtual objects to improve accuracy and efficiency. These methods [5, 13, 20, 21, 29] calculate the control/display ratio based on the speed of the user's hand motion, and manipulate objects in the virtual environment (VE) by using the ratio to adjust the user's hand motion. However, the efficiency and accuracy of manipulation are affected due to the imprecise perception of the user's hand motion speed.

To address these problems, we propose a distant object manipulation method with adaptive gains in VR to improve the efficiency and accuracy of manipulation. We introduce manipulation gains, and design experiments to collect user behavior data during manipulation and determine the fitting function used to adaptively calculate manipulation gains in the process of distant object manipulation. Our method works in the cases with the known targets, since this is true for many VR manipulation guidance applications. In order to evaluate the performance of our distant object manipulation method with adaptive gains, we design a user study with three tasks. Compared to the state of the art methods, our method achieves a significant improvement in the completion time, and the manipulation accuracy of the tasks. Our method significantly outperforms other methods in usability and significantly reduces task load. Fig. 1 illustrates our method in the scenario of an aircraft engine manipulation task.

In summary, our main contributions are as follows: 1) for the first time, we obtain the maximum and minimum manipulation gains that users can tolerate; 2) we propose a method to determine manipulation gains adaptively in the process of manipulation; 3) we design a user study to evaluate the efficiency and accuracy of our method.

*Corresponding Author: wanglily@buaa.edu.cn

2 Related Work

Many researchers have devoted themselves to object manipulation in the past two decades. For a more comprehensive understanding of object manipulation methods in VR, we recommend readers to read the survey paper [16].

When we manipulate objects in VE, the most intuitive idea is to manipulate virtual objects by hand [6, 10, 17, 23, 24]. Poupyrev et al. proposed a Go-Go method [22] to manipulate objects by growing the user's arm and nonlinear mapping to reach and manipulate distant objects. Bowman et al. proposed a ray casting based object manipulation method [2], in which users could grab and manipulate objects by intersecting rays with objects. In order to improve the accuracy and efficiency of object manipulation, Laurent et al. proposed the 3-hand manipulation method [1], which allows 2-3 users to collaboratively manipulate an object by manipulating three misaligned manipulation points. Nguyen et al. introduced the 3-Point++ tool technology [18], which contains the center of gravity (6DOF) and three manipulate points (3DOF) for the user to perform coarse and fine manipulations. They also proposed the 7-Handle manipulation technique [19] to generate a widget with multiple points, and manipulate the object according to the widget. Gloumeau et al. introduced PinNPivot manipulation technology [10], which uses pins to constrain the rotation of the object to improve the efficiency and accuracy of manipulation. These multiple point based manipulation methods improve the efficiency and accuracy for direct manipulation of virtual objects by hand at close range, but they are difficult to manipulate objects which are far away from the user.

Some researchers used the speed of user control to determine the factors of the manipulation. Frees et al. [5, 6] proposed the PRISM method, which divides the user's state into two modes according to the speed of the hand. In precision mode, objects are made to translate and rotate slower than the user's hands by increasing the control/display (CD) ratio; In normal mode, the user moves his hand, and the object moves an equal distance. Wilkes et al. [29] scaled the manipulation of the HOMER method [21] to make it more suitable for higher precision long- and short-range manipulation tasks without slowing down. They also combined the PRISM and Go-Go methods to improve the accuracy of the go-go method [30]. Noritaka [20] proposed two adjustment methods of position adjustment and viewpoint adjustment. When the user uses one hand, the factor in the two adjustment methods is determined based on the speed of the user's hand. When the user uses both hands, the distance between the hands determines the factor. Kim et al. [13] proposed a non-linear mapping method to improve the efficiency of the manipulation. It determines the combination coefficient through the speed and acceleration of the translation and rotation of the hand. In this method, the translation and rotation speed of the manipulated object is always greater than or equal to the hand speed, which may cause over-manipulation when the object is close to the target. A 3D user interface with anatomical 2D/3D shape matching was designed to allow the user to adjust the translation and rotation gains dynamically [14]. These methods use the speed of the user's input to determine the speed of the manipulation object. During the manipulations, users not only need to pay attention to the position and pose of objects and targets in the view they observe, but also need to pay attention to the speed of their hand motion, so the task load of the user is high, which reduces the efficiency and accuracy of the manipulation. Our method automatically adjusts the manipulation based on the object-target relation in the user's view, and the user only needs to focus on the view.

Recently, view analysis has been used to guide user interaction. Freitag et al. [7] adjusted the travel speed in the virtual scene by analyzing the viewpoint quality of the entire scene. They also designed an interactive assistance interface to guide users to the interesting regions through the analysis of view [8]. Cao [3] proposed a path redirection method based on the analysis of the visual features of the user views. Wang et al. [26] introduced an object selection method, which analyzes the occlusions in the user's view and generates an auxiliary view to improve the efficiency of VR selection. They also quantified the view quality of multiple users to determine the denominator of manipulation collaboration [27]. View analysis can be used in many areas [25]. In our method, we compute the manipulation gains according to the analysis of the target and the object to be manipulated in the user's views.

3 Manipulation Gains

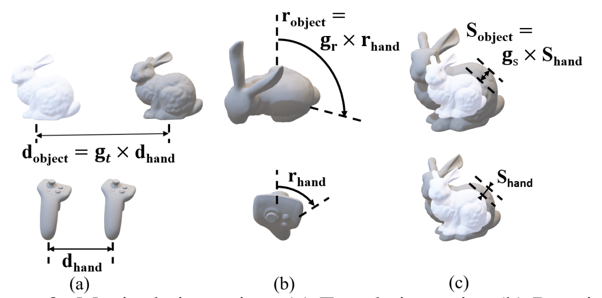

Figure 2: Manipulation gains. (a) Translation gain. (b) Rotation gain. (c) Scale gain.

In this section, we introduce manipulation gains in the context of manipulating virtual objects. In this paper, manipulation gains are defined as some adjustment factors that map the hand control motion to the motion of the manipulated virtual object. Inspired by the translation, rotation and curvature gains in redirected walking, we also divide the manipulation gains into three parts: translation gain, rotation gain and scale gain. These three gains do not affect each other in the process of manipulation. Compared to control/display ratio [5], the manipulation gains are more general. Scale gain is not included in the control/display ratio. Unlike the control/display ratio, manipulation gains are not necessarily related to the user's control. It is an independent mapping, and the user can set it manually or by using some functions. Manipulation gain is an extension of the control/display ratio concept.

3.1 Translation Gain

When the tracking system detects a change in the position of the user's hand in the real world, the change is defined by the vector $d_{hand} = d_{cur} - d_{pre}$, where d_{cur} is the position of the current frame and d_{pre} is the position of the previous frame, and the manipulated object is in the virtual world coordinate system move d_{hand} in the corresponding direction. The translation gain g_t is defined as the quotient of the motion vector of the object d_{object} in the VE to the motion vector of the user's hand d_{hand} in the real world: $g_t = d_{object}/d_{hand}$.

When the translation gain g_t is applied to the translation manipulation d_{hand}, the manipulated object is moved by the vector $g_t \times d_{hand}$ in the corresponding direction, as shown in Fig. 2 (a). If the manipulated object is far away from the target position, a translation gain greater than 1 helps the user quickly move the object to the vicinity of the target object, thus improving the manipulation efficiency. On the contrary, if the object and target are very close, the translation gain should be less than 1, for example, $g_t = 0.1$, which is particularly useful for improving the accuracy of manipulation.

3.2 Rotation Gain

The rotation of the hand in the real world is specified by a vector consisting of three angles. We recorded the rotation quaternion q_{cur} of the hand in the current frame and the rotation quaternion q_{pre} of the hand in the previous frame. The amount of change in the rotation of the real-world hand $q_{hand} = q_{cur} - q_{pre}$. The rotation angle r_{hand} is obtained by the quaternion q_{hand}. Then the object being manipulated in the VE rotates r_{hand}. The rotation gain gr is defined by the quotient of the rotation angle r_{object} of the manipulated object in the VE and the hand rotation angle r_{hand} in the real world: $g_r = r_{object}/r_{hand}$.

When the rotation gain g_r is applied to the rotation of the manipulated object in the VE, the angle of the object rotation is $r_{object} = g_r \times r_{hand}$, as shown in Fig. 2 (b). If $g_r < 1$, the rotation angle of the object is smaller than the rotation angle of the hand. In the case of $g_r > 1$, the rotation angle of the object is greater than the rotation angle of the hand. For example, if the user's hand is rotated 30 degrees, in the case of gain $g_r = 1$, the object is rotated 30 degrees in the VE; if the gain $g_r = 0.5$, the object is rotated 15 degrees; if the gain $g_r = 2$, the object is rotated 60 degrees.

3.3 Scale Gain

The user scales the manipulated object in the virtual world by specifying a scale vector. In our distant object manipulation, we increase or decrease the value of the scale vector by clicking two buttons on the handle. We set the initial value of the scale vector change of a click to s_{hand}. The scale gain g_s is defined by the quotient of the object scale factor s_{object} and the initial value s_{hand}, that is $g_s = s_{object}/s_{hand}$

When the scale gain g_s is applied to the scale of the manipulated object in the VE, the change in the scale of the manipulated object between two adjacent frames with a click is $s_{object} = g_s \times s_{hand}$, as shown in Fig. 2 (c). If $g_s < 1$, the change in the scale of the manipulated object between two adjacent frames is less than s_{hand}. In the case of $g_r > 1$, the change in the scale of the manipulated object between two adjacent frames is greater than s_{hand}.

4 Visual Perception based Adaptive Manipulation Gains Determination

In this section, we design two experiments to obtain the maximum/minimum manipulation gains and the fitting function to compute the gains adaptively in the manipulation process.

Participants. We have recruited 12 participants(7 males and 5 females), between 20 and 30 years old. Ten of our participants had used HMD VR applications before. Participants had a normal and corrected vision, and none reported vision or balance disorders. All 12 participants participated in Experiment 1 and Experiment 2, but did not participate in the evaluation part of Experiment 2.

Hardware and software setup. We used an HTC Cosmos VR HMD system with two handheld controllers that allow users to point a virtual laser at a virtual environment (VE) and manipulate virtual objects. The HMD was connected to its own workstation with a 3.6GHz Intel(R) Core(TM) i7-9900KF CPU, 16GB of RAM, and an NVIDIA GeForce GTX 1080 graphics card. The tracked physical space hosting the VR applications is 4m × 4m. We used Unity 2019.1 to implement our VR manipulation tasks. The virtual environments were rendered at 90fps for each eye.

Three scenes. *Cube* scene contains a cube to be manipulated and the target. The size of *Box* scene is 20m×20m, and the size of the target cube is $1.0m \times 1.0m \times 1.0m$. *Bunny* scene contains a bunny and the target. The size of *Bunny* scene is $20m \times 20m$, and the size of the target is about $0.7m \times 0.6m \times 0.6m$. $Aircraftengine$ scene contains a component and an aircraft engine with a target component. The size of $Factory$ scene is $60m \times 25m$ (Figure 1), and the size of target component of the aircraft engine is about $1.2m \times 2.6m \times 2.8m$.

Manipulation implementation. The object manipulation method we implement is from [28]. When the user keeps holding the 'on' button on the handheld controller, the translation and rotation of the handheld controller are mapped to the virtual space according to manipulation gains. The user moves the object with his/her right hand, and rotates the object with his/her left hand. The user clicks the 'Up' button on the right handheld controller to increase the scale transformation and clicks the 'Down' button on the left handheld controller to decrease the scale transformation of the object.

4.1 Experiment 1: Determine the Maximum and Minimum Gains

This section describes a visual perception experiment to obtain the maximum and minimum values of manipulation gains acceptable to the user. When gains are greater than the maximum value or less than the minimum value, the user feels that their hand motion is inconsistent with the visual feedback.

4.1.1 Experiment design.

This experiment includes 2 tasks: maximum gain task T_{MAX} and minimum gain task T_{MIN}. Each operation (translation, rotation and scaling) requires separate T_{MAX} and T_{MIN}. For translation operation, the target is fixed in the scene and each time the virtual object is initialized by translating randomly from the target, the distance to the target of T_{MAX} is within the range (1, 6), and of T_{MIN} is within the range (0.1, 1.5). For rotation operation, the target is also fixed in the scene, and the virtual object is initialized by rotating randomly from the target, the angle to the target of T_{MAX} is within the range (25,180), and of T_{MIN} is within the range (0.1,40). For scaling operation, the virtual object is initialized by scaling randomly from the target, the scale to the target of T_{MAX} is within the range (0.4,5), and of T_{MIN} is within the range (0.6,1.6). In T_{MAX}, we set the range of translate gain between 1 and 300, the range of rotate gain between 1 and 50, and the range of scale gain between 1 and 30. The initial translation gain, rotate gain and scale gain are set to 150.5, 25.5, 15.5 respectively. In T_{MIN}, we set the range of manipulation gains between 0 and 1. The initial gain is set to 0.5.

Tasks. The user is required to use one of the three operations (translation, rotation, scaling) to complete the task each time, for a total of 18 times (3 scenes × 2 tasks × 3 operations).

In T_{MAX}, the user is required to roughly manipulate the virtual object to the target position with a given gain. Through visual feedback, if the user feels that the motion of the virtual object is too large relative to the movement of his/her hand, he/she needs to press the "smaller" button on the handheld controller, and then the virtual object is set to the original position, and the system uses the dichotomy method to generate a small gain; on the contrary, the user has to press the "larger" button on the handheld controller, and then the virtual object is set to the original position, and the system uses the dichotomy method to generate a large gain. The user manipulates the object again using the newly generated gains, and repeats the above process until he/she feels that the motion of the virtual object is consistent with the magnitude of the motion of his/her hand. In T_{MIN}, the user is required to finely manipulate the virtual object to the target position with a given gain. The process is the same as that of T_{MAX}.

4.1.2 Results.

Fig. 3 shows the scatter plots of the maximum and minimum values of manipulation gains collected through the above experiment. Pearson correlation analysis provides evidence that the maximum manipulation gain users can tolerate is weakly correlated with distance (r=-0.168, p=0.327) and angle (r=0.162, p = 0.337) and scale ($r = 0.193, p = 0.259$) between the object and the target. Pearson correlation analysis also shows that the minimum manipulation gain that the user can tolerate is weakly related to the distance ($r = 0.074, p = 0.661$), angle ($r = 0.128, p = 0.456$) and scale ($r = 0.141, p = 0.412$) between the object and the target.

We average the maximum and minimum manipulation gains collected from the participants as the maximum and minimum manipulation gains that the user can tolerate. The maximum translation gain, rotation gain and scale gain are: 24.36, 5.34 and 5.96, the minimum gains are: 0.044, 0.053 and 0.057.

In T_{MAX} and T_{MIN}, we set the range of distance, angle, and scale to overlap. Because we think in the overlapping range, the maximum and minimum values are more likely related to distance, angle, and scale. But from Fig. 3, the maximum and minimum values in the overlapping range have nothing to do with distance, angle, and scale.

The maximum scale gain has larger variations of values. One reason maybe users use the handheld controller's translation and rotation to control the translation and rotation continuously, and use two buttons to control the scale. The discrete interactions may introduce the difference. Another reason is that the minimum scale gain is usually used to manipulate objects more precisely, and everyone feels basically the same, so the variance of the data is small. The maximum scale gain is used for fast scaling and rough interactions. Therefore, the user is not sensitive to this value, and therefore the variance is large.

4.2 Experiment 2: Find the View-dependent Fitting Functions for Gains

When users manipulate objects, they observe the virtual scene through the VR HMDs. The view seen by the user will change with the movement of the user's head, and the virtual objects in the scene will be transformed according to the user's hand interaction, so real-time visual feedback is very important for user interactions. For manipulation, the most important information in visual feedback includes object and target position, rotation, size, and relative relationship. So we design an experiment to explore the relationship between visual feedback and manipulation gains, and derive the fitting functions for visual metrics and manipulation gains. Because our method uses the information of the virtual object and the target in 3D scenes to determine the gains, if the handle devices can track the hand in real-time, our method does not need recalibration for different handles. However, if the handles have large latency, they need to be calibrated.

4.2.1 Experiment design

Object-target visual metrics. The object-target visual metrics are to measure the relationship of the distance, angle and projected area proportion between the manipulated object and the target under the current viewpoint and its corresponding auxiliary viewpoints. It is a triple (d, θ, p), where d, θ and p are the distance metric, the angle difference metric and the projection area ratio of the manipulated object and the target. The triple can be computed with algorithm 1.

Algorithm 1 Object-target visual metric

```
Input: object $o$, target $t$, viewpoint $V$, view $I$
Output: distance $d$, angle $\theta$, area proportion $p$
1: $A_o, A_t$ = Area($I, o, t$);
2: $p = A_o / A_t$;
3: $V_l, V_t$ = ConstrTopLeftFov($V$, $o$);
4: $b_o, b_t, b_{lo}, b_{lt}, b_{to}, b_{tt}$ = OBB($V$,$V_t$,$V_l$,$o$,$t$);
5: $c_o, c_t, c_{lo}, c_{lt}, c_{to}, c_{tt}$ = GetOBBCenter($b_o, b_t, b_{lo}, b_{lt}, b_{to}, b_{tt}$);
6: $l_o, l_t, l_{lo}, l_{lt}, l_{to}, l_{tt}$ = GetOBBAxis($b_o, b_t, b_{lo}, b_{lt}, b_{to}, b_{tt}$);
7: $d_1, d_2, d_3$ = Distance($c_o, c_t, c_{lo}, c_{lt}, c_{to}, c_{tt}$);
8: $\theta_1, \theta_2, \theta_3$ = GetAngle($l_o, l_t, l_{lo}, l_{lt}, l_{to}, l_{tt}$);
9: d=$\sqrt{d_1^2 + d_2^2 + d_3^2}$;
10: $\theta = \sqrt{\theta_1^2 + \theta_2^2 + \theta_3^2}$;
11: return $(d, \theta, p)$;
```

The inputs of this algorithm are the geometry of the manipulated object o and the target t, the current viewpoint V and the view I rendered from V. The output is the triple (d, θ, p). First, we calculate the projection area of the manipulated object and the target and the ratio p of the projection area (lines 1-2). Then we construct two auxiliary viewpoints based on the current viewpoint and the position of the operated object: V_l is on the left side of the object (V_{lo} is perpendicular to V_o), and V_t is directly above the object o. The distance from these two auxiliary viewpoints to the manipulated object is the same as the distance from the current viewpoint to the manipulated object (line 3). After this, we project the object and target from V, V_l and V_t separately and construct the oriented bounding box (OBBs) for them (line 4). The centers and the long axes of the OBBs are obtained (lines 5-6). The distance and angle difference between the object and the target of each viewpoint are calculated (lines 7-8). At the final distance d and the final angle difference θ between the object and the target are obtained (lines 9-10), and return with the projection area ratio p (line 11).

Task. This experiment includes 1 task. The target is fixed in the scene. The object to be manipulated is randomly placed in the scene, and the distance to the target is in the range of (0.1, 16), its size is 0.3-2 times the size of the target, and the rotation is completely random. The initial gains used are 12.4 for translation, 2.9 for rotation, and 2.9 for scaling, which is computed by averaging the maximum and minimum gains we obtained through the last experiment.

The task is to ask the user to adjust the manipulation gain by manipulating objects in the virtual scene to their corresponding targets multiple times until he/she feels the most appropriate gain. Each user is given 3 minutes of practice time before the formal experiment begin. In the practice, the user manipulates the object with different gains, so he/she has the most comfortable manipulation feeling, which is his/her motivation to keep repeating manipulation in the formal experiment to find the most comfortable gain. The user needs to complete the task in a random one of the 3 scenes. After the object is placed in the scene, the user manipulates the object with initial gains. If the user thinks that the current gains are not suitable, he/she presses the "reset" button to put the object back to the starting status(the starting position, rotation and scaling), increase or decrease the gains through the 'up' and "down" buttons on the handheld controller, and then manipulate the object to the target. The user can see the adjustment of their own manipulation gains through the slider in the virtual scene, as shown in Fig. 4. This process is repeated until the user thinks the current gains are appropriate. The user presses the "reset" button and then presses the "save" button to save the current gains and the object-target visual metric (d, θ, p) corresponding to those gains. After this, the system automatically resets the object to be manipulated randomly to reduce the distance between the object and the target. Then the user repeats the above process to select the appropriate gains, and the system records the gains and the corresponding (d, θ, p). The system resets the object to be manipulated four times. For each user, the system records five sets of data.

4.2.2 Results.

Translation Gain. We collected the translation gain obtained in the experiment and the corresponding d. A simple regression was verified the translation gain based on the corresponding d. We found that with quadratic ($F_{1,2} = 522.443, p < 0.001$), cubic ($F_{1,3} = 652.178, p < 0.001$), logarithmic ($F_{1,1} = 774.651, p < 0.001$) model regression equations are significant. The cubic model has R^2 of 0.970, which is greater than R^2 of the quadratic model (0.946) and logarithmic model (0.928). The cubic model has a regression standard error S of 1.396, which is smaller than that of the quadratic model (1.886) and the logarithmic model (2.170). So we chose the regression equation of the cubic model as our translation fitting function. We plot the translation gain and the corresponding d obtained in the experiment into a scatter plot, and perform fitting, as shown in Fig. 5. Thus the adaptive translation gain function is Equation 1. If the gain calculated using this equation is not within the range of the minimum gain and maximum gain in Section 4.1.2, [0.044, 24.36], then the gain is truncated, that is, if it is less than 0.044, use 0.044, and if it is greater than 24.36, use 24.36.

$$\begin{cases} g_t = 0.016d^3 - 0.488d^2 + 5.254d + 0.875 \\ g_t \in [0.044, 24.36] \end{cases} \tag{1}$$

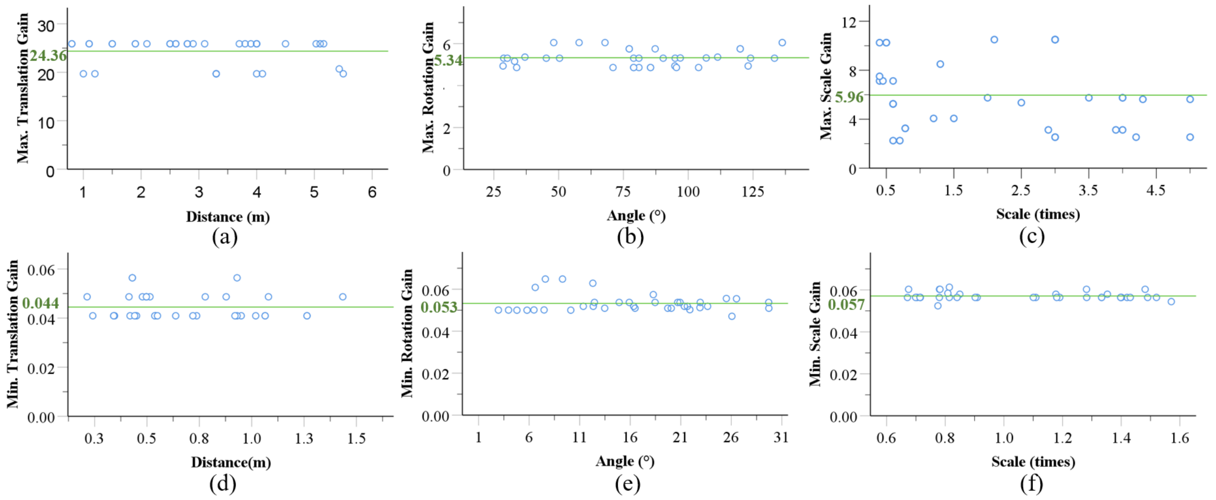

Figure 3: Minimum and maximum gains. (a) and (d) are the maximum and minimum translation gains.(b) and (e) are the maximum and minimum rotation gains. (c) and (f) are the maximum and minimum scale gains. The green line represents the average of the maximum or minimum manipulation gains.

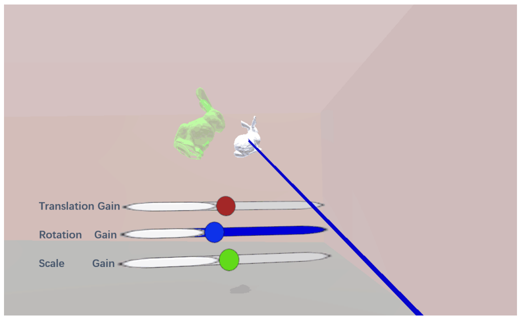

Figure 4: The user view of the experiment to find the fitting function of manipulation gains.

Rotation Gain. The same regression method was verified the rotation gain based on the corresponding θ. We found that with quadratic ($F_{1,2} = 440.075, p < 0.001$), cubic ($F_{1,3} = 443.969, p < 0.001$), logarithmic($F_{1,1} = 187.865, p < 0.001$) model regression equations are significant. The cubic model has an R^2 of 0.986, which is greater than the R^2 of the quadratic model (0.936) and logarithmic model (0.901). The cubic model has a regression standard error S of 0.218, which is smaller than S of the quadratic model (0.471) and the logarithmic model (0.587). So we chose the regression equation of the cubic model as our rotation fitting function. We plot the rotation gain and the corresponding θ into a scatter plot, and perform fitting, as shown in Fig. 5. We use the fitting function as the adaptive function of the rotation gain, the maximum value is 5.33, and the minimum value is 0.054, which are reported in Section 4.1.2. The adaptive rotation gain function is Equation 2.

$$\begin{cases} g_r = 1.645 \times 10^{-6}\theta^3 - 0.0006\theta^2 + 0.08\theta - 0.132 \\ \\ g_r \in [0.053, 5.34] \end{cases} \tag{2}$$

Scale Gain. Regression analysis was verified the scale gain based on the corresponding p. We found that with quadratic ($F_{1,2} = 111.890, p < 0.001$), cubic ($F_{1,2} = 252.211, p < 0.001$) model regression equations are significant. The cubic model has an R^2 of 0.876, which is greater than R^2 of the quadratic model (0.811). The cubic model has a regression standard error S of 0.565, which is smaller than S of the quadratic model (0.697). So we chose the regression equation of the cubic model as our final result. We plot the scale gain obtained in the experiment and the corresponding p into a scatter plot, and perform fitting, as shown in Fig. 5. We use the fitting function as the adaptive function of the scale gain, the maximum value is 5.96, and the minimum value is 0.057, which are reported in Section 4.1.2. The adaptive scale gain function is Equation 3.

$$\begin{cases} g_s = -4.889p^3 + 25.541p^2 - 37.278p + 17.380 \\ \\ g_s \in [0.057, 5.96] \end{cases} \tag{3}$$

Evaluation. We recruited 6 new participants(4 male, 2 female, 20-30 years old), repeated the experiment and collected the same data. Then we calculated the manipulation gains with three fitting functions using d, θ, and p in the collected data. We quantify the error e and error rate e_r of the fit functions by comparing the manipulation gains computed by the fitting functions $gains_{com}$ with the manipulation gains chosen by the participants we collected $gains_{choose}$. The error e is calculated using $|gains_{com} - gains_{choose}|$. The error rate e_r is calculated with Equation 4.

$$e_r = \frac{1}{N}\sum_{n=1}^{N}\frac{e}{gains_{choose}} \tag{4}$$

Fig. 6 shows the evaluation results for manipulation gains functions. The results show the errors are very small, and e_r of translation gain is 5.0%, rotation gain is 6.7%, and scale gain is 8.6%.

5 User Study

5.1 User Study Design

We designed a user study with a manipulation task in 3 scenes to evaluate the efficiency, accuracy, and task load of our method. The hardware settings and manipulation implementation used in the user study are the same as Sect. 4.

Participants. We have recruited 36 participants, 30 males and 6 females, between 20 and 30 years old. 24 of our participants had VR experience before. Participants had normal and corrected vision, and none reported vision or balance disorders. There are 4 control conditions and an experimental condition. Control condition CC_1 is with the traditional method, in which manipulation gains are 1. Control condition CC_2 is with the PRISM method [6]. Control condition CC_3 is the non-linear mapping method based on the velocity and acceleration of hand [13]. Control condition CC_4 is with the dynamic gains based on the velocity of hand [14]. None of these three methods discuss the scale gain, so we set scale gain to 1 for them. The experimental condition EC is with our method.

Hypotheses. Our method was designed to allow the user to manipulate an object to the target efficiently. Thus, we formulate the following hypotheses:

H1: Users can manipulate an object to the target faster with EC compared to CC_{1-4}.

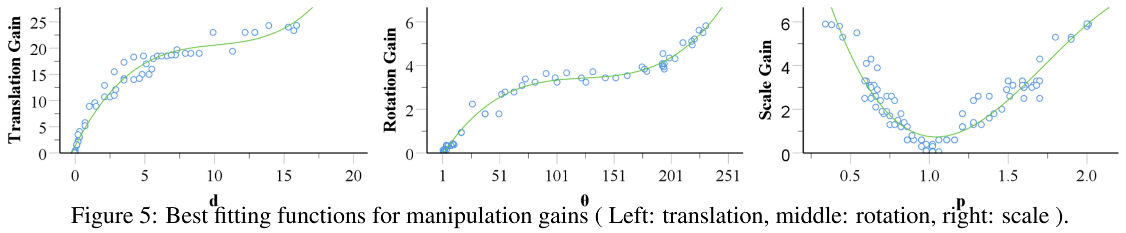

Figure 5: Best fitting functions for manipulation gains (Left: translation, middle: rotation, right: scale).

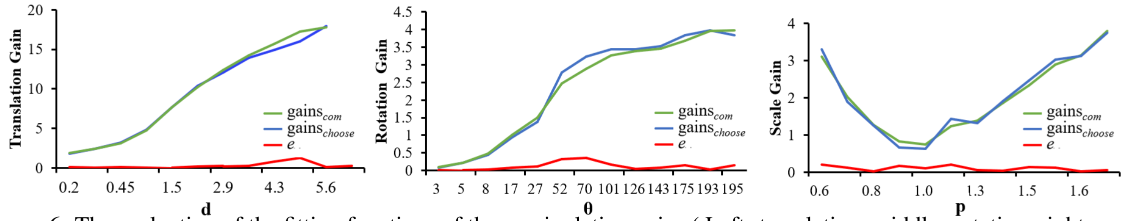

Figure 6: The evaluation of the fitting functions of the manipulation gains (Left: translation, middle: rotation, right: scale).

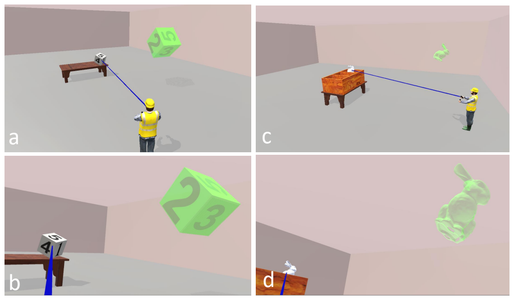

Figure 7: The first scene $S1$ (left) of our user study is shown in left column. The user is manipulating the cube to the green target position (a), (b) shows the view seen from the user; The second scene $S2$ is shown in right column. The user is manipulating bunny to the green target position (c), (d) shows the view seen from the user.

H2: Users can manipulate an object to the target more accurately with EC compared to CC_{1-4}.

H3:User task load of EC is lower than that of CC_{1-4}.

H4: EC is easier to use than CC_{1-4} .

Task. During the task, the user is required to manipulate the object as quickly and accurately as possible to a predefined target position. There are 3 scenes in the task. The target in each scene is fixed. The size of the object to be manipulated is randomly generated, which is about 0.4-5 times the target size. The object and the position of the user are placed at a random locations of the scene in the initialization. After the user manipulates the object to the target, he/she presses the “end” button to complete the task.

In *Table* scene (S1), the user is required to manipulate a cube on the table to the target (Figure 7 left). The size of *Table* scene is $20m \times 20m$ and the size of the target cube is $1.0m \times 1.0m \times 1.0m$. In *Box* scene (S2), the user is required to manipulate the bunny to the target.In the beginning, the bunny is placed in a wooden box, which provides more occlusions from the user’s viewpoint (Figure 7 right). The size of *Box* scene is $20m \times 20m$, and the size of the target is about $0.9m \times 0.8m \times 0.8m$. In *Factory* Scene (S3), the user is required to manipulate a piece of blue component to the target position in aircraft engine. There are many occlusions in the scene. The size of *Factory* scene is $60m \times 12m$ (Figure 1), and the size of target aircraft engine is about $1.2m \times 2.6m \times 2.8m$.

Procedure. The task with one scene is referred to as the session, so we have 3 sessions for the task. The order of the sessions is random. In each session, the user is required to perform the manipulation task 3 times with 5 conditions. The order of the task with different conditions $(5conditions \times 3times)$ is also random. The minimum interval between the session is one day and the maximum interval is three days. For each session, participants practice for 1 minute before the task starts. When the user points to the object need to be manipulated, our system starts recording time and other objective metrics. We tell the user that we will record and evaluate the task completion time, which indirectly encourages them to complete the task as soon as possible.

Metrics. Task performance was measured with the objective metrics: (1) task completion time, in seconds, indicates the time from the participant pointing to the object until he/she presses the ’end’ button to confirm the end of the manipulation; (2) position error, in millimeters, indicates the distance from the center of the manipulated object to the center of the target position when the participant presses the “end” button; (3) rotation error, in degrees, indicates the angle difference between the local coordinate system of the manipulated object and the target coordinate system when the participant presses the “end” button. If the angle difference of the three coordinate axes is α, β, γ, the rotation error is $\sqrt{\alpha^2+\beta^2+\gamma^2}$; (4) scale error, in times, indicates the ratio of the absolute value of the difference between the diagonal length of the bounding box of the manipulated object and the diagonal length of the target bounding box to the diagonal length of the target bounding box when the participant presses the “end” button. We also evaluated the perception with four subjective metrics: user task load, measured with the standard NASA TLX questionnaire [11, 12], and six aspects of usability measured with the usability questionnaire [13]. The six aspects of usability are ”Intuitiveness” (IN), ”Efficiency” (EF), ”Accuracy” (AC), ”Naturalness” (NA), ”Satisfaction” (SA) and ”Easiness” (EA). After each condition in the session, the data of the task-load questionnaire is collected. After all three sessions, the data of the usability is collected.

Statistical analysis. For each metric, the values of EC were compared to those of CC_1, CC_2, CC_3 and CC_4 respectively using a one-way repeated-measures ANOVA [9]. The sphericity assumption is evaluated using the Mauchly test [15]. When the assumption is violated, a Greenhouse-Geisser correction is applied. Then an overall ANOVA was conducted to investigate whether one can reject the null hypothesis that there is no statistically significant difference between the conditions. When the null hypothesis was rejected, the differences between the four pairs were analyzed with posthoc tests using the Bonferroni correction. For the time dependent variable we quantified the size of the effect using Cohen’s d [4].

5.2 Results

5.2.1 Task performance

Task completion time. Table 1 gives the task completion time. Statistical significance is indicated by an asterisk. The sphericity assumption is violated: $p < 0.001(S1, S2, S3)$. After applying the Greenhouse-Geisser correction, the overall ANOVA reveals significant differences between the five conditions: $(F_{2.100,50.402} =$

Table 1: The completion time, in seconds.

Task	Condi -tion	Avg ± std. dev.	$(CC_i\text{-}EC)$ / CC_i	p	Cohen's d	Effect size
S1	EC	53.30 ± 12.10				
	CC_1	81.27 ± 23.90	34.4%	$< 0.001^*$	1.48	Very Large
	CC_2	79.83 ± 24.91	33.2%	$< 0.001^*$	1.35	Very Large
	CC_3	74.90 ± 23.30	28.8%	$< 0.001^*$	1.16	Large
	CC_4	68.67 ± 10.96	18.3%	$< 0.001^*$	1.33	Very Large
S2	EC	44.00 ± 3.67				
	CC_1	86.75 ± 13.14	49.3%	$< 0.001^*$	4.43	Huge
	CC_2	81.50 ± 14.79	46.0%	$< 0.001^*$	3.48	Huge
	CC_3	66.75 ± 9.36	34.1%	$< 0.001^*$	3.20	Huge
	CC_4	65.25 ± 5.76	32.6%	$< 0.001^*$	4.40	Huge
S3	EC	51.75 ± 18.39				
	CC_1	86.61 ± 12.37	40.2%	$< 0.001^*$	2.22	Huge
	CC_2	85.13 ± 17.35	39.2%	$< 0.001^*$	1.87	Very Large
	CC_3	84.01 ± 32.66	38.4%	$< 0.001^*$	1.22	Very Large
	CC_4	83.37 ± 9.42	37.9%	$< 0.001^*$	2.16	Huge

$14.148, P < 0.001)$ for S1, $(F_{2.230,49.03} = 56.24, P < 0.001)$ for S2, and $(F_{1.658,8.138} = 24.672, P < 0.001)$ for S3. Post-hoc analysis reveals that EC was significantly shorter than for CC_1, CC_2, CC_3 and CC_4 for both scenes. Compared with all control conditions of all three scenes, our method significantly improves the task time performance, and the effect size ranges from "Large" to "Huge".

Position error. Table 2 shows the position errors all conditions for these three scenes. The sphericity assumption is violated: $p < 0.001(S1, S2, S3)$. After applying the Greenhouse-Geisser correction, the overall ANOVA reveals significant differences between the five conditions: $(F_{2.020,48.482} = 12.743, P < 0.001)$ for S1, $(F_{1.858,42.724} = 35.516, P < 0.001)$ for S2, and $(F_{2.612,60.070} = 11.340, P < 0.001)$ for S3. Post-hoc analysis reveals that EC was significantly smaller than for CC_1, CC_2, CC_3 and CC_4 for both scenes. Compared with all control conditions of all three scenes, our method reduced position error significantly, and the effect size ranges from "Medium" to "Very Large".

Rotation error. Table 3 gives the rotation error of all conditions for these three scenes. The sphericity assumption is violated: $p < 0.001(S1, S2, S3)$. After applying the Greenhouse-Geisser correction, the overall ANOVA reveals significant differences between the five conditions: $(F_{2.592,55.858} = 28.506, P < 0.001)$ for S1, $(F_{2.539,58.403} = 42.141, P < 0.001)$ for S2, and $(F_{2.391,54.990} = 26.746, P < 0.001)$ for S3. Post-hoc analysis reveals that EC was significantly smaller than for CC_1, CC_2, CC_3 and CC_4 for all scenes. Compared with all control conditions of all three scenes, our method reduced rotation error significantly, and the effect size ranges from "Large" to "Huge".

Scale error. Table 4 shows the scale errors of all conditions for these three scenes. The sphericity assumption is violated: $p < 0.001(S1, S2, S3)$. After applying the Greenhouse-Geisser correction, the overall ANOVA reveals significant differences between the five conditions: $(F_{2.100,50.402} = 14.148, P < 0.001)$ for S1, $(F_{2.230,49.03} = 56.24, P < 0.001)$ for S2, and $(F_{1.658,38.138} = 24.672, P < 0.001)$ for S3. Post-hoc analysis reveals that EC was significantly smaller than for CC_1, CC_2, CC_3 and CC_4 for all scenes. Compared with all control conditions of all three scenes, our method reduced scale error significantly, and the effect size ranges from "Large" to "Huge".

5.2.2 Perception

Figure 8 shows the results of the task load. The sphericity assumption is violated: $p = 0.032(S1)$ and $p < 0.001(S2, S3)$. After applying the Greenhouse-Geisser correction, the overall ANOVA reveals significant differences between the five conditions: $(F_{2.750,52.256} = 40.455, P < 0.001)$ for S1, $(F_{1.873,35.588} = 57.251, P < 0.001)$ for S2, and $(F_{2.085,39.619} = 59.698, P < 0.001)$ for S3. Post-hoc analysis reveals that the task load of EC was significantly smaller than that of CC_1, CC_2, CC_3 and CC_4 for all scenes.

Table 2: The position error, in millimeters.

Task	Condi -tion	Avg ± std. dev.	$(CC_i\text{-}EC)$ / CC_i	p	Cohen's d	Effect size
S1	EC	2.3 ± 1.2				
	CC_1	6.7 ± 5.1	66.0%	$< 0.001^*$	1.18	Large
	CC_2	5.8 ± 6.2	60.8%	0.0193^*	0.79	Medium
	CC_3	4.4 ± 2.3	48.6%	0.0011^*	1.15	Large
	CC_4	2.99 ± 4.3	64.1%	$< 0.001^*$	1.29	Very Large
S2	EC	2.0 ± 1.4				
	CC_1	3.9 ± 1.5	47.3%	$< 0.001^*$	1.26	Very Large
	CC_2	4.5 ± 3.6	54.4%	$< 0.001^*$	0.90	Large
	CC_3	4.0 ± 6.4	38.2%	$0.00433\ast$	0.98	Large
	CC_4	8.1 ± 4.3	44.9%	$0.00269\ast$	1.04	Large
S3	EC	3.2 ± 1.5				
	CC_1	6.3 ± 3.2	49.7%	$< 0.001^*$	1.25	Very Large
	CC_2	5.4 ± 3.0	41.2%	0.006^*	0.94	Large
	CC_3	7.7 ± 3.3	58.5%	$< 0.001^*$	1.75	Very Large
	CC_4	7.8 ± 3.2	59.3%	$< 0.001^*$	1.86	Very Large

Table 3: The rotation error, in degrees.

Task	Condi -tion	Avg ± std. dev.	$(CC_i\text{-}EC)$ / CC_i	p	Cohen's d	Effect size
S1	EC	0.88 ± 0.17				
	CC_1	2.62 ± 1.02	66.6%	$< 0.001^*$	2.40	Huge
	CC_2	1.60 ± 0.96	45.3%	0.002^*	1.05	Large
	CC_3	1.85 ± 0.48	52.5%	$< 0.001^*$	2.68	Huge
	CC_4	2.61 ± 1.00	66.5%	$< 0.001^*$	2.42	Huge
S2	EC	1.38 ± 0.79				
	CC_1	5.25 ± 1.38	73.7%	$< 0.001^*$	3.43	Huge
	CC_2	2.79 ± 1.57	50.5%	0.0012^*	1.13	Large
	CC_3	4.68 ± 1.18	70.5%	$< 0.001^*$	3.28	Huge
	CC_4	5.22 ± 1.78	73.5%	$< 0.001^*$	2.79	Huge
S3	EC	1.07 ± 0.34				
	CC_1	2.67 ± 0.84	59.9%	$< 0.001^*$	2.51	Huge
	CC_2	1.68 ± 0.69	36.4%	0.0012^*	1.12	Large
	CC_3	1.88 ± 0.30	43.0%	$< 0.001^*$	2.51	Huge
	CC_4	2.20 ± 0.69	51.3%	$< 0.001^*$	2.08	Large

The usability results are shown in Fig. 9. Over all six questions, the sphericity assumption is violated $(p = 0.001)$. After applying the Greenhouse-Geisser correction, the overall ANOVA reveals significant differences between the five conditions $(F_{3.029,157.516} = 44.768, p < 0.001)$, and post-hoc analysis reveals that EC is significantly easier to use than CC_1, CC_2, CC_3 and CC_4. The scores for the six questions violate the sphericity assumption $(p < 0.001, p = 0.014, p < 0.001, p < 0.001, p = 0.001, and p < 0.001)$, after applying the Greenhouse-Geisser correction, the overall ANOVA reveals significant differences between the five conditions $(F_{2.712,70.516} = 13.162, p < 0.001; F_{1.885,18.851} = 17.434, p < 0.001; F_{1.934,32.880} = 12.446, p < 0.001; F_{1.934,32.880} = 22.815, p < 0.001; F_{2.521,32.768} = 40.276, p < 0.001; F_{1.803,36.058} = 11.577, p < 0.001)$.

5.3 Discussion

The results in Table 1 support ***H1:*** The reasons for the efficiency of our method may be: (1) When the object is far away from the target in the user's view, the manipulation gains obtained by our method in each frame are greater than 1, which allows the user to manipulate the object to the nearby region of the target quickly. When the object is near the target in the user's view, our method generates the gains smaller than 1, which reduces the chances of over manipulation. (2) The manipulation gain is automatically calculated according to the view, the user does not need to consider the speed of the hand

Table 4: The scale error, in times.

Task	Condi -tion	Avg ± std. dev.	(CC_i-EC) / CC_i	p	Cohen's d	Effect size
S1	EC	0.003 ± 0.001				
	CC_1	0.013 ± 0.005	73.3%	$< 0.001^*$	2.50	Huge
	CC_2	0.011 ± 0.007	69.7%	$< 0.001^*$	1.49	Very Large
	CC_3	0.012 ± 0.007	70.9%	$< 0.001^*$	1.62	Very Large
	CC_4	0.011 ± 0.007	70.4%	$< 0.001^*$	1.61	Very Large
S2	EC	0.03 ± 0.012				
	CC_1	0.058 ± 0.023	47.7%	$< 0.001^*$	1.49	Very Large
	CC_2	0.053 ± 0.021	43.2%	$< 0.001^*$	1.33	Very Large
	CC_3	0.047 ± 0.018	35.5%	$< 0.001^*$	1.07	Large
	CC_4	0.059 ± 0.028	49.3%	$< 0.001^*$	1.34	Very Large
S3	EC	0.007 ± 0.004				
	CC_1	0.012 ± 0.005	45.1%	$< 0.001^*$	1.19	Large
	CC_2	0.0014 ± 0.010	54.0%	0.0024^*	1.05	Large
	CC_3	0.019 ± 0.010	65.4%	$< 0.001^*$	1.58	Very Large
	CC_4	0.015 ± 0.008	57.1%	$< 0.001^*$	1.44	Very Large

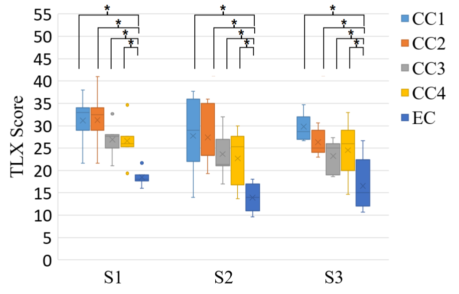

Figure 8: Box plots for task load scores of the five conditions and the three scenes. Asterisks denote statistical significance.

motion, and the speed of the hand motion is not easy to control.

Generally, CC_3 and CC_4 are more efficient than CC_1 and CC_2, because the manipulation gains of CC_3 and CC_4 can be greater than 1, and the manipulation gains of CC_2 and CC_1 are always equal to or smaller than 1. When the object is far away from the target, CC_4 is faster than CC_3. The motion of the hand from a static state to high speed to a static state usually requires a start-up phase and a cool-down phase. During the start-up phase and cool-down phase, the hand speed is not large, but the acceleration is large. CC_3 only considers the speed of hand motion, CC_4 considers both the speed and acceleration of hand motion, so CC_4 can also achieve a larger gain during the start and cooling phases of hand movement. CC_1 is slower than CC_2 because the gain of CC_2 is less than 1 and reduces the chance of over manipulation when the object is near to the target.

The results in Table 2, Table 3, and Table 4 support ***H2:*** The position, rotation and scale errors of EC were significantly smaller than those of CC_1, CC_2, CC_3 and CC_4 for all scenes. This is because during the manipulation, when the differences between the position, rotation, and scaling of the object and the target in the current user view and the auxiliary views are getting smaller and smaller, our manipulation gain will continue to decrease automatically, allowing the user to more finely adjust the object to be manipulated to the target, thereby improving the accuracy of the manipulation.

The results in Fig. 8 support ***H3:***. Compared with all control conditions, the task load of our method is reduced significantly. This is because the gains of CC_1 and CC_2 are always equal to or smaller than 1, and it takes a long time for the user to manipulate the object to the target, which makes the user fatigued. For CC_2, CC_3 and CC_4, the gains are related to the speed of hand motion, the user had to pay more attention to both his/her hand motion and his/her view, then manipulate the object. With our method, the gains are computed adaptively according to the user's view automatically, so the user only needs to focus on his/her view, then manipulate the object.

The results in Fig. 9 support ***H4:*** With our method, the user

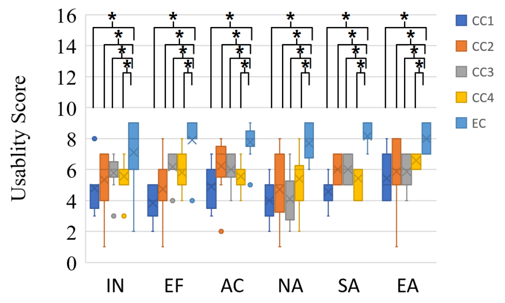

Figure 9: The usability score of per condition. Significant differences are denoted with asterisks.

observes his/her current view, obtains the relationship between the object and the target, and manipulates the object, which is the same experience as in reality. Our method provides a more intuitive and natural way to manipulate objects, and it is easy for users to use. In control conditions (C2, C3, and C4), the user had to focus on the speed of the hand motion and the content of his/her view simultaneously to manipulate the object. Moreover, the better performance of our method makes the user feel more effective and accuracy of our method. The experience makes the user more satisfied.

6 Conclusions, Limitations, and Future Work

We have proposed a distant object manipulation method based on adaptive gains in VR to improve the efficiency and accuracy of manipulation. We introduced manipulation gains, and constructed the fitting functions to model the relations between the manipulation gains and the object-target visual metrics. We designed a user study to evaluate the performance and perception of our method. The results showed compared to the state of the art methods, our method achieved a significant improvement in the efficiency and the accuracy of the manipulation. Our method significantly reduces task load, and outperforms other methods in usability.

One limitation of our method is that we compute the object-target visual metrics with the current user view and two auxiliary views. The gain may also vary significantly when the visual metrics of the user view change little or no change, while the visual metrics of the secondary view vary greatly. The user may feel that the hand motion is inconsistent with the visual experience. Another limitation is that currently we only consider the gain calculation method when the user is standing in a relatively fixed position to manipulate the object. If the user needs to move around during the manipulation, and the position changes too much, both the user view and the auxiliary views may change greatly, and the calculated gain may also change greatly, resulting in the discontinuity of the gain. Our method now doesn't work for the cases with unknown targets since we use the information of the target to compute the gains.

One future work can be done is to remove auxiliary views and add some manipulation points to improve the intuitiveness and naturalness of the manipulation. We will extend our work to more general cases in VR, such as the users moving and manipulating the object in the scene. In order to alleviate the requirement of prior knowledge of the target pose and scale, one future work is to have a prediction step that tries to identify the most likely target in the scene or captures the user's gaze at the target area, and guides the calculation of gains when the target is missing in freely manipulating objects for novel creations. In our user study, the users were required to manipulate the predefined number of objects to the targets. In future, an interesting user study to investigate the distant object manipulation accuracy when users complete manipulation tasks with different durations can be designed.

Acknowledgments

This work was supported by National Key RD plan 2019YFC1521102, by the National Natural Science Foundation of China through Projects 61932003 and 61772051.

References

[1] L. Aguerreche, T. Duval, and A. Lécuyer. 3-hand manipulation of virtual objects. In *Eurographics*, 2009.

[2] D. A. Bowman and L. F. Hodges. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. *ACM*, 1999.

[3] A. Cao, L. Wang, Y. Liu, and V. Popescu. Feature guided path redirection for vr navigation. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 137–145, 2020. doi: 10.1109/VR46266.2020.00032

[4] J. Cohen. Statistical power analysis for the behavioral sciences. Academic press, 2013.

[5] S. Frees and G. Kessler. Precise and rapid interaction through scaled manipulation in immersive virtual environments. In *IEEE Proceedings. VR 2005. Virtual Reality, 2005.*, pp. 99–106, 2005. doi: 10.1109/VR.2005.1492759

[6] S. Frees, G. D. Kessler, and E. Kay. Prism interaction for enhancing control in immersive virtual environments. *ACM Trans. Comput.-Hum. Interact.*, 14(1):2–es, May 2007. doi: 10.1145/1229855.1229857

[7] S. Freitag, B. Weyers, and T. W. Kuhlen. Automatic speed adjustment for travel through immersive virtual environments based on viewpoint quality. In *3d User Interfaces*, 2016.

[8] S. Freitag, B. Weyers, and T. W. Kuhlen. Interactive exploration assistance for immersive virtual environments based on object visibility and viewpoint quality. In *IEEE Virtual Reality 2018*, 2018.

[9] A. Gelman. Analysis of variance . *Quality Control Applied Statistics*, 20(1):págs. 295–300, 2005.

[10] P. C. Gloumeau, W. Stuerzlinger, and J. H. Han. Pinnpivot: Object manipulation using pins in immersive virtual environments. *IEEE Transactions on Visualization and Computer Graphics*, PP(99):1–1, 2020.

[11] S. Hart. Nasa-task load index (nasa-tlx); 20 years later. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 50:904 – 908, 2006.

[12] S. Hart and L. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. *Advances in psychology*, 52:139–183, 1988.

[13] H. Kim, G. A. Lee, and M. Billinghurst. A non-linear mapping technique for bare-hand interaction in large virtual environments. In *Australasian Computer-Human Interaction Conference*, 2015.

[14] K. Kim, R. L. Lawrence, N. Kyllonen, P. M. Ludewig, A. M. Ellingson, and D. F. Keefe. Anatomical 2d/3d shape-matching in virtual reality: A user interface for quantifying joint kinematics with radiographic imaging. In *Symposium on 3D User Interfaces*, 2017.

[15] J. W. Mauchly. Significance Test for Sphericity of a Normal n-Variate Distribution. *The Annals of Mathematical Statistics*, 11(2):204 – 209, 1940. doi: 10.1214/aoms/1177731915

[16] D. Mendes, F. M. Caputo, A. Giachetti, A. Ferreira, and J. Jorge. A survey on 3d virtual object manipulation: From the desktop to immersive virtual environments. In *Computer Graphics Forum*, 2019.

[17] D. Mendes, M. Sousa, R. Lorena, A. Ferreira, and J. Jorge. Using custom transformation axes for mid-air manipulation of 3d virtual objects. In *Acm Symposium*, pp. 1–8, 2017.

[18] T. T. H. Nguyen and T. Duval. 3-point++: a new technique for 3d manipulation of virtual objects. 2013.

[19] T.-T. H. Nguyen, T. Duval, and C. Pontonnier. A new direct manipulation technique for immersive 3d virtual environments. In *International Conference on Artificial Reality and Telexistence*, 2014.

[20] N. Osawa. Two-handed and one-handed techniques for precise and efficient manipulation in immersive virtual environments. In G. Bebis, R. Boyle, B. Parvin, D. Koracin, P. Remagnino, F. Porikli, J. Peters, J. Klosowski, L. Arns, Y. K. Chun, T.-M. Rhyne, and L. Monroe, eds., *Advances in Visual Computing*, pp. 987–997. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[21] J. S. Pierce and R. Pausch. Comparing voodoo dolls and homer: exploring the importance of feedback in virtual environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 105–112, 2002.

[22] I. Poupyrev, M. Billinghurst, S. Weghorst, and T. Ichikawa. The go-go interaction technique: Non-linear mapping for direct manipulation in vr. 1998.

[23] P. Song, W. B. Goh, W. Hutama, C.-W. Fu, and X. Liu. A handle bar metaphor for virtual object manipulation with mid-air interaction.

[24] R. Stoakley, M. J. Conway, and R. Pausch. Virtual reality on a wim: Interactive worlds in miniature. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, p. 265–272. ACM Press/Addison-Wesley Publishing Co., USA, 1995. doi: 10.1145/223904.223938

[25] H.-W. Wang, z.-Z. Hu, and J.-R. Lin. Bibliometric review of visual computing in the construction industry. In *Visual Computing for Industry, Biomedicine, and Art*, vol. 3, 2020. doi: 10.1186/s42492-020-00050-0

[26] L. Wang, J. Chen, Q. Ma, and V. Popescu. Disocclusion headlight for selection assistance in vr. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*, pp. 216–225, 2021. doi: 10.1109/VR50410.2021.00043

[27] L. Wang, X. Liu, and X. Li. Vr collaborative object manipulation based on viewpoint quality. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 60–68, 2021. doi: 10.1109/ISMAR52148.2021.00020

[28] L. Wang, X. Liu, and X. Li. Vr collaborative object manipulation based on viewpoint quality. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 60–68, 2021. doi: 10.1109/ISMAR52148.2021.00020

[29] C. Wilkes and D. A. Bowman. Advantages of velocity-based scaling for distant 3d manipulation. In *Proceedings of the 2008 ACM Symposium on Virtual Reality Software and Technology*, VRST '08, p. 23–29. Association for Computing Machinery, New York, NY, USA, 2008. doi: 10.1145/1450579.1450585

[30] C. Wilkes and D. A. Bowman. Advantages of velocity-based scaling for distant 3d manipulation. In *Virtual Reality Software and Technology*, 2008.