Boxer: A Multimodal Collision Technique for Virtual Objects

Byungjoo Lee
Aalto University
Graduate School of Cuture Technology, KAIST
byungjoo.lee@kaist.ac.kr

Qiao Deng
Aalto University
qiao.deng@aalto.fi

Eve Hoggan
Aalto University
Aarhus University
eve.hoggan@cs.au.dk

Antti Oulasvirta
Aalto University
antti.oulasvirta@aalto.fi

ABSTRACT

Virtual collision techniques are interaction techniques for invoking discrete events in a virtual scene, e.g. throwing, pushing, or pulling an object with a pointer. The conventional approach involves detecting collisions as soon as the pointer makes contact with the object. Furthermore, in general, motor patterns can only be adjusted based on visual feedback. The paper presents a multimodal technique based on the principle that collisions should be aligned with the most salient sensory feedback. Boxer (1) triggers a collision at the moment where the pointer's speed reaches a minimum after first contact and (2) is synchronized with vibrotactile stimuli presented to the hand controlling the pointer. Boxer was compared with the conventional technique in two user studies (with temporal pointing and virtual batting). Boxer improved spatial precision in collisions by 26.7% while accuracy was compromised under some task conditions. No difference was found in temporal precision. Possibilities for improving virtual collision techniques are discussed.

CCS CONCEPTS

• **Human-centered computing** → **Interaction techniques**;

KEYWORDS

Virtual Reality; Collision Detection; Temporal Pointing; Batting.

ACM Reference Format:
Byungjoo Lee, Qiao Deng, Eve Hoggan, and Antti Oulasvirta. 2017. Boxer: A Multimodal Collision Technique for Virtual Objects. In *Proceedings of 19th ACM International Conference on Multimodal Interaction (ICMI'17).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3136755.3136761

1 INTRODUCTION

Virtual collision techniques are a class of interaction techniques in VR wherein the collision of a pointer is used to invoke an event on an object in a scene. As with direct manipulation of physical objects, virtual objects are selected and manipulated through collisions with the (virtual) body. However, such events can be triggered only when a collision between the user's virtual representation (the pointer) and the virtual object is detected. The various ways in which this collision can be implemented have rarely been investigated. The paper shows that significant improvements may be possible over the standard collision detection paradigm.

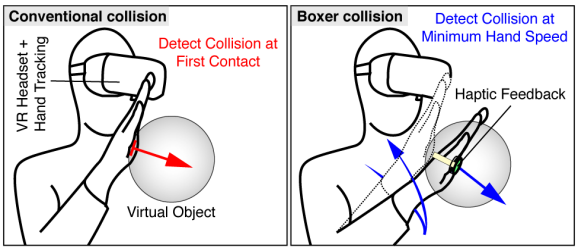

Figure 1: The study compared collision detection methods in VR {Conventional, Boxer}. The Conventional method detects the collision on the basis of geometric conditions between two objects (first contact) without haptic feedback from the system. The Boxer method postpones collision identification until the point at which the speed of the virtual pointer reaches a minimum after first contact, and it is synchronized with vibrotactile stimuli presented to the hand controlling the pointer.

There is a large number of possible techniques for detecting and triggering a collision event [4]. Let us consider the case of touching a virtual button with a virtual finger. Should the collision deemed to take place when the finger first contacts the button, when the finger finally exits from the button, or at some time between those two points? Outside VR, there is evidence that poor input performance is related to the timing of detecting the input event [3, 11, 21, 22]. Improvements in performance of up to 10% were reported in a touchpad-based study using a timing task (non-VR) [11]. While that study looked at timing performance in a non-VR task, the study presented here evaluated collision detection techniques in a VR task for both timing and spatial precision (direction of collision).

We hypothesized that a critical variable for defining collision detection techniques is how well users are able to perceive and predict the moment of collision. In conventional methods, the moment of collision can be perceived only *visually*, because the collision is detected when certain geometric conditions are met — for example, initial contact between surfaces of objects. For instance, the commercial physics engine in `Unity` [17] provides two basic callbacks

ICMI'17, November 13–17, 2017, Glasgow, UK

ACM ISBN 978-1-4503-5543-8/17/11…$15.00
https://doi.org/10.1145/3136755.3136761

for detecting collisions: `onCollisionEnter` and `onCollisionExit`. In these methods, the collision is triggered at the moment of the first contact or the moment of the last contact of two objects. However, since the speed of collision is usually high, the moment of collision can be difficult for the user to perceive visually.

Following up on the hypothesis that co-alignment of sensory events improves user performance, this paper examines a novel technique called Boxer that triggers collisions when they are co-aligned with the user's proprioceptive perception (see Figure 1). This is best understood via a shadowboxing analogue: although the boxer does not have anything visible in the air to hit, (s)he can practice, because there is still an impact transferred at the end of the punch. Collisions in AR and VR are somewhat similar to this situation. Users have nothing tangible to hit in the air, and, because the duration of the collision is usually too short, the user cannot perceive the moment of the collision through visual feedback alone. Since haptic feedback is more immediate and faster than visual feedback in certain situations [20], we would expect better user perception of collision timing with Boxer for instantaneous tasks.

Our research compared the performance of Boxer with a conventional baseline. In particular, we were interested in how participants utilize sensory information from different modalities. In the first user study (on *temporal pointing* in VR), we investigated how various collision techniques affect the user's timing performance. For this case, we assumed that triggering the collision at the exact moment predicted by the user will reduce the user's timing error. In the second user study (on *virtual batting*), we compared the spatial performance of the Boxer technique (i.e., accuracy and precision of virtual shots) with the conventional baseline. From the second experiment, we observe that user performance varies, depending on the given collision technique and collision situation. Especially in use of the Boxer technique, the spatial precision of the collision intended by the user increased by up to 26.7% in certain situations.

The main contribution of this study is in revealing that the performance of users in virtual space can vary significantly, depending on the difference between collision techniques. This result indicates that the sensory and cognitive processes of a user should be carefully considered when one is designing input-sensing methods in virtual space. We hope that this study will trigger a wide range of future research on more common tasks such as pushing and pulling in VR.

2 BACKGROUND AND MOTIVATION

This section describes the characteristics of collision-based tasks, with comparison to some other typical tasks in human–computer interaction. We provide background for our suggestion that different collision techniques can yield different user performance. We also summarize research on multimodal sensations, pointing out problems that may arise in applying conventional collision techniques in modern VR situations.

2.1 Existing Collision Techniques

Algorithms for detecting collisions between virtual objects have been studied extensively in computer science. They are aimed primarily at enabling realistic physics simulations in virtual scenes. However, with the advent of applications of VR, users are no longer a passive audience watching physical simulations but can directly interact with virtual objects through their avatars in virtual scenes. However, in the field of computer science, collision detection algorithms do not reflect the user's motor or cognitive characteristics; they consider only computational efficiency, visual appeal, ease of implementation, and robustness [4]. With our study, we tried to explore the design space of collision detection from the perspective of user performance.

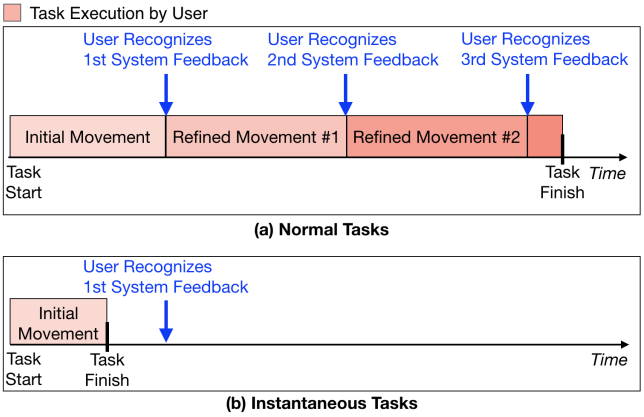

Figure 2: In (a) non-instantaneous tasks, feedback given by the system can aid in correction of the user's present state during (closed-loop) interaction. However, when a user tries to carry out (b) an instantaneous task that occurs in a very short time, such as a collision, the feedback provided by the system after detection of the collision is too late. Movement is controlled in open-loop fashion.

2.2 User Performance in Instantaneous Tasks

Given the importance of collision techniques for VR, the issue of how to time detection has been addressed surprisingly rarely. Previous studies have mostly addressed what happens after a collision is detected, considering elements such as issues of system feedback [1, 2, 9, 10, 13, 16] and system latency [12, 15]. However, collision is an *instantaneous task* that occurs in a very short time, at the moment of collision, and any feedback that comes some time after detection may not be directly useful for correcting the motor pattern. In instantaneous tasks, the time it takes the user to complete the task is shorter than the time it takes to perceive the feedback from the system and refine the behavior (see Figure 2).

Instead, in the case of a collision, the user is forced to perform with a preprogrammed and anticipated motion and the problem of when the system detects the user's intention to apply input is critical. For example, in the case of a task in which a user shoots a virtual ball through a virtual pointer, the direction in which the ball moves may vary greatly, depending on when the system detects the collision (see Figure 3). At this time, if the direction of the collision interpreted by the system differs from the user's intended direction, an error results. But can the computer know exactly when the user intended the collision to be? To answer this question, we need to understand how humans utilize sensory information from various modalities.

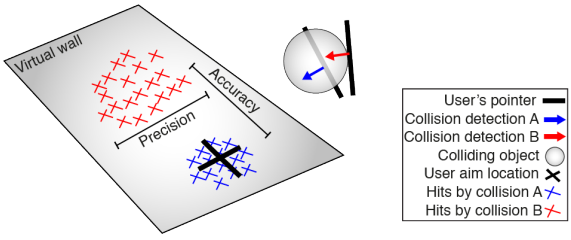

Figure 3: In this picture, the user intended to collide with the ball at the moment of B. However, if the system detects a collision at A, the movement of the ball is different from what the user originally intended.

2.3 Integration of Sensory Feedback

We hypothesize that users can benefit from integrating information from multiple sensory modalities. This might help the user program a better motor command for collision. Here, we consider the integration of two modalities: (1) vision and (2) haptics. It is well known that sensory processing within the visual sense is modulated by information from the haptic sense [6]. Although our visual sense tends to dominate, haptic dominance can occur when the variance associated with haptic estimation is lower than that associated with visual estimation [5]. Furthermore, multisensory integration (in terms of visual and haptic feedback) is sensitive to temporal inconsistencies [7].

As pointed out in the introduction, the collision techniques provided by conventional physics engines limit the user to visual feedback. However, humans sometimes use haptic information rather than visual information in situations where the haptic modality is dominant, as is often the case in instantaneous tasks. Therefore, we hypothesize that users would benefit from haptic feedback. A recent study published in HCI supports this hypothesis.

2.4 TouchMax: Detecting Touch at Maximum Proprioceptive Saliency

A tapping input on a touchscreen can be considered as an instantaneous task similar to a virtual collision. Also this event can be detected in a variety of ways. A recent study report [11] points out that the existing touch input APIs detect input only when the finger first touches the screen (touch down) or when it is removed (touch up). These approaches, we believe, suffer from the same issue pointed out for conventional collision techniques in VR. The authors presented a novel tapping method called *TouchMax*, which follows the principle of temporal co-alignment: the moment the input is triggered is synchronized with the peak of the tactile feedback from the fingertip hitting the touchpad.

The TouchMax algorithm was designed to detect the input when the area of contact between the finger and the touchscreen is at its maximum (see Figure 4). At this moment, the speed of the finger becomes zero, so the user can recognize the largest impact. Through a task of acquiring a blinking target, they found that TouchMax has an error rate as low as 10%, far below those of other, conventional input methods. Although their experiments used a limited dependent variable (error rate) to show the performance of different collision techniques (or tapping in this case), their work showed

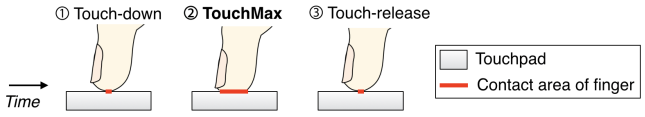

Figure 4: TouchMax [11] was designed to detect the input when the area of contact between the finger and the touchpad is at its maximum. At this moment, the speed of the finger becomes zero, so the user can recognize the largest proprioceptive impact.

that the problem of when to detect collisions can have a significant impact on usability.

In summary, there is evidence that the usability of instantaneous input varies greatly with when the input is detected, but there has been very little work studying the relevant techniques. In the following sections, we address this issue with a novel collision technique named Boxer. First, we explain the operation principle and implementation process of Boxer, and then we discuss two user studies investigating how various collision techniques can affect usability.

3 IMPLEMENTATION OF BOXER

Boxer collision can be thought of as a virtual-collision version of TouchMax (see Figure 4). Boxer postpones the detection of the collision event until the point where the speed of the virtual pointer reaches a minimum after the initial contact. We hypothesized that this moment is co-aligned with the point of maximum proprioceptive saliency — in this case, abrupt change in the speed of the hand or finger controlling the pointer. From users' perspective, the moment of collision detected with Boxer is perceived predominantly through their proprioception. Finally, at the time of collision detection, a vibrotactile actuator is used to simulate the impact of actual physical collision by vibrating the palm of the hand at the same moment (see Figure 1).

For purposes of an empirical study, we implemented Boxer on top of a conventional physics engine provided via `Unity`. Boxer uses three steps to detect a collision and finally provides a unit direction vector $\vec{u}_B$ and corresponding force magnitude F_B at the moment when user pointer speed become zero (see Algorithm 1). Providing the direction ($\vec{u}$) and the magnitude (F) as a result of a collision detection is the same as with conventional collision detection algorithms (see Figure 5). In the physics engine of `Unity`, two callbacks for collision detection, `onCollisionEnter` and `onCollisionExit`, provide the same information, although the rules for detecting a collision are different.

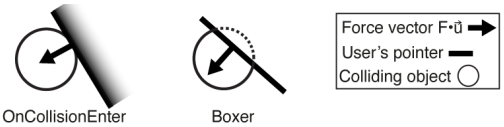

Figure 5: Unity's collision detection algorithms (`onCollisionEnter` and `onCollisionExit`) and the Boxer algorithm both return the magnitude (F) and direction ($\vec{u}$) of the collision.

Algorithm 1 Boxer technique implemented in `Unity`

define $v_{pointer}$, F_{sum}, N, v_p, and v_{pp}
N, F_{sum}, v_p, $v_{pp} \leftarrow 0$
procedure Step 1 `OnCollisionEnter`:
    $F_{sum} \leftarrow F_{sum} + \left|\vec{F}_{current}\right|$, $N \leftarrow N + 1$
    $v_p,\ v_{pp} \leftarrow \vec{v}_{pointer}$
procedure Step 2 `OnCollisionStay`:
    if $v_p < v_{pp}$ *and* $v_p < \left|\vec{v}_{pointer}\right|$ **then** Step 3
    else
        $v_{pp} \leftarrow v_p$
        $v_p \leftarrow \left|\vec{v}_{pointer}\right|$
        $F_{sum} = F_{sum} + F_{current}$, $N \leftarrow N + 1$
procedure Step 3 Detection of a collision:
    return $F_{Boxer} = F_{sum}/N$ *and* $\vec{u}_{Boxer} = \vec{u}_{pointer}$
    add force to the object ($F_{Boxer} \cdot \vec{u}_{Boxer}$) and vibrate palm

3.1 Step 1: Detection of Initial Contact

The overall process of Boxer begins with the detection of geometric contact between a user's pointer and the colliding object, using a conventional collision detection method: `onCollisionEnter`. The callback returns a force vector $\vec{F}_1$, and the physics engine of `Unity` automatically tries to apply the force to the colliding object. However, at this time, Boxer prevents the system from moving the ball, by setting its location constant. Then Boxer adds the initial magnitude of force output $|\vec{F}_1|$ to the value of our force summation variable:

$$F = |\vec{F}_1|, \qquad N = 1 \tag{1}$$

N is the total number of magnitudes currently summed into the variable and later used to obtain the average collision magnitude.

3.2 Step 2: Accumulating Intermediate Forces

In the second step, we use the `onCollisionStay` callback from `Unity`, which repeatedly detects a collision whenever the pointer and the ball stay in contact with each other. For every ith force output from callback F_i, Boxer adds its magnitude to the force summation variable until the speed of the pointer is at a minimum:

$$F = F + |\vec{F}_i|, \qquad N = N + 1 \tag{2}$$

3.3 Step 3: Detection of a Collision

Just after the speed of the pointer is minimized in Step 2, Boxer enters Step 3. This is the final step to determine both magnitude and direction of the collision. In this step, the direction pointed by the user's pointer at the moment of the speed becoming minimum is determined to be the direction of the collision ($\vec{u}_B$). Furthermore, we divide the magnitude value (F) that we have accumulated in Step 1 and Step 2 by the total number of collision detections (N) to determine the magnitude of the Boxer collision (F_B) as follows:

$$F_B = F/N, \qquad \vec{u}_B = \vec{u}_{Pointer} \tag{3}$$

Using the force vector obtained above, we apply the `AddForce` function provided by `Unity` to move the colliding object in magnitude F_B and direction $\vec{u}_B$. After adding force to the object, Boxer utilizes the `onCollisionExit` event for not detecting duplicate collisions until the user's pointer is completely separated from the object.

In the empirical studies reported upon here, the user's pointer was assumed to be a plane attached parallel to the palm of the hand (see Figure 6). Therefore, when the speed of the pointer is minimized, the direction pointed by the pointer ($\vec{u}_{Pointer}$) can be represented as a vector normal to that plane. For more complex shapes of pointers, however, further research is needed on how to determine the direction of the pointer at the moment of Boxer collision.

4 STUDY 1: A TIMING TASK IN VR

In two studies, the performance of various collision techniques was compared in terms of performance of two basic input tasks: *temporal pointing* [11] and *virtual batting*. This allows us to assess two important characteristics of collision: (1) whether the collision was triggered at the user-intended time and (2) whether the collision was triggered in the user's intended direction.

In Study 1, we first focused on measuring the timing performance of different collision techniques. Participants collided a virtual pointer into a virtual target that was blinking with a regular interval (T_D) and duration (T_W) at a certain index of difficulty (ID). In this task, called *temporal pointing*, if a collision is detected while the target is "on," success in the trial is recorded, and if a collision is detected while the target is "off," a failure is recorded (see Figure 7). In this setup, the experiment is repeated until statistically meaningful error rates ($= \frac{Number\ of\ failed\ trials}{Number\ of\ total\ trials}$) are obtained for the different blinking patterns. In this case, the error rates (E) are fitted to the equation of the temporal pointing model as shown below, and consequently the amount of timing noise (c_σ) contained in the user input can be accurately measured.

$$E(ID) = 1 - \frac{1}{2}\left[erf\left(\frac{(1-c_\mu)}{c_\sigma 2^{(ID+0.5)}}\right) + erf\left(\frac{c_\mu}{c_\sigma 2^{(ID+0.5)}}\right)\right] \tag{4}$$
$$ID = \log_2(D_t/W_t)$$

The ID term refers to the index of difficulty (in bits) and is a unitless variable like that of Fitts' law [8], and $erf(x)$ is the (known) error function originating from integration of the normal distribution:

$$erf(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt \tag{5}$$

c_σ is the standard deviation of the participant's timing response and is considered to be the sum of the noise generated from the participant's internal clock and motion implementation. The larger the c_σ value, the more inaccurate the timing performance of the participant.

4.1 Method

4.1.1 Participants: Sixteen paid participants (2 female) were recruited. The mean age of participants was 27.2 yrs (σ=5.83). All participants were right-handed, and three of them wore eyeglasses during the experiment.

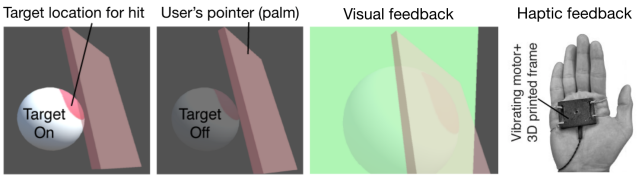

Figure 6: At left, task screens from Study 1; at right, two modalities of system feedback (Visual and Visual·Haptics).

4.1.2 Experiment Design: The experiment followed a $2 \times 2 \times 3 \times 2$ within-subject design with four independent variables: *duration of blinking (T_W)* {80, 160, and 240 ms}, *interval of blinking (T_D)* {900 and 1500 ms}, *modality of system feedback* {Visual and Visual·Haptics}, and *timing of collision detection* {First Contact and Minimum Speed}:

- Visual: Just after detection of a collision, only the visual change of the colliding object is presented to the participant to indicate that the collision was successful.
- Visual·Haptics: In addition to the visual change of the colliding object, haptic vibration is applied to the palm of the participant just after the detection of the collision, indicating a successful hit.
- First Contact: Collision is registered from a callback provided in the physics engine of `Unity` (`onTriggerEnter`). The algorithm detects a collision when the user pointer first touches the target.
- Minimum Speed: Detection of a collision is postponed until the point where the speed of the virtual pointer reaches a minimum.

Six indices of difficulty (IDs) were computed from each T_D and T_W pair: 1.90, 2.49, 2.64, 3.22, 3.49, and 4.22 (bits). Fifty trials per condition resulted in 1,200 trials per participant.

4.1.3 The Task: Participants used the palm of the dominant hand in the virtual space to tap on the target (see Figure 6). They were instructed to anticipate the target blinking and do the selection as quickly and as accurately as possible from the onset because the target duration is shorter than typical human reaction time (~250 ms). Only one tap was allowed per blink, and participants were not allowed to skip any. By setting the target's `isTrigger` option to `True` and omitting the execution of the `addForce` function in the Boxer collision, the ball was made to remain stationary after the collision during the experiment.

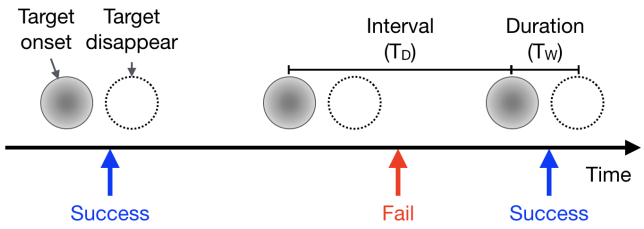

Figure 7: Study 1, with a timing task in VR. Timing error was assessed in a temporal pointing task in which the user tries to select a regularly blinking target.

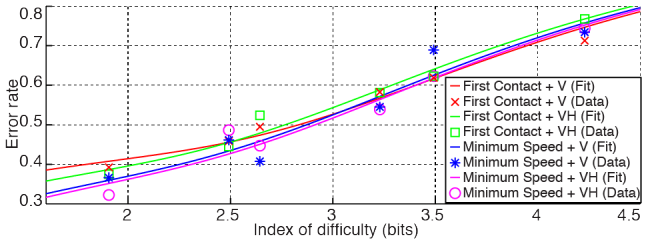

Figure 8: Study 1, with regression of empirical error rates to the model of temporal pointing. When the collision detection rule and the feedback signal are working in the same modality, the noise (c_σ) from participants' internal clock is minimized: First Contact + Visual (0.079), First Contact + Visual·Haptics (0.088), Minimum Speed + Visual (0.084), Minimum Speed + Visual·Haptics (0.082).

4.1.4 Materials: The target was represented as a white ball with 15 cm diameter (see Figure 6). To enable constantly controlling the participants' hitting strategy, we instructed them to knock the red part of the ball. The transparency of the ball suddenly becomes 0% from 100% upon the change from "on" to "off" state. The participant's hand was represented by a red plane 2 cm thick placed parallel to the center of the palm (15 cm × 20 cm). Detecting collisions between a plate and a sphere is mathematically one of the simplest cases of collision detection problem. We believe this simple implementation could minimize uncontrolled effects of the `Unity` engine on the experiment. The plane representing the user's hand is designed transparently (50% alpha) in order not to prefer either visual or haptic modality.

In the Visual feedback condition, a green flash in the background was shown in addition to the immediate switch to target "off" state, to indicate a successful collision (see Figure 6). In the Visual·Haptics condition, a 250 Hz sine wave with 100 ms duration was also used to vibrate the palm of each participant (see Figure 6). To minimize the effect of fatigue, we provided an elbow supporter to all participants (see Figure 9).

4.1.5 Procedure: Participants sat on a regular office chair, with the location of the virtual target as depicted in Figure 9. The light was adjusted to be the best for recognition with the `Leap Motion` controller used, and we calibrated all the devices each time before the experiment. Before starting 50 trials for a blinking *ID*, 10 blinks were demonstrated so that participants could learn the blinking pattern of the upcoming condition. After each *algorithm for collision detection*, participants could take a break. We counterbalanced the ordering of *timing of collision detection* × *modality of the system feedback* provided to the participants. Within a *timing-feedback* condition, the order of *ID*s was randomized. The experiment took an hour to finish.

4.1.6 Apparatus: The task scene was built in `Unity` (Version 5.3.5), run on a desktop computer (64-bit Windows 8.1, Intel Core i7-5930K CPU @ 3.50 GHz). We used an `Oculus Rift DK2 Orion Beta` unit as the VR display (Version 3.1.3), and a `Leap Motion` controller attached in front of it was used to track each participant's hand. We used a standard library provided by Leap Motion

Table 1: From Study 1, model fitting in the temporal pointing task. Empirical error rates are reported for six IDs. The difference between the conditions was not statistically significant.

Condition of Collision	c_σ	c_μ	R^2
First Contact+ Visual	0.079	0.070	0.97
First Contact+ Visual·Haptics	0.087	0.097	0.97
Minimum Speed+ Visual	0.084	0.11	0.92
Minimum Speed+ Visual·Haptics	0.082	0.12	0.95
Overall	0.083	0.10	0.99

to integrate the controller with the `Oculus` unit. The refresh rate of the application was 75 fps. We used an `ATAC C2-Tactor` to provide vibrotactile feedback. The tactor is a moving magnet linear actuator, resonant at 250 Hz. The tactor was enclosed in a 3D-printed case and attached to the user's palm with a rubber band (see Figure 6, right). All experiment conditions were coded in C#. The default setting for the physics engine of `Unity` was maintained throughout the experiment.

4.2 Results

4.2.1 Error Rates: The overall error rate of collisions was 53% (σ=23.3%) for the Minimum Speed condition and 54.8% (σ=23.6%) for the First Contact condition. The main effect of *timing of collision detection* on error rate was not significant (F(1,15)=0.345, p=0.57). Also, the main effect of *modality of system feedback* on error rate was not significant (F(1,15)=0.003, p=0.96). The interaction effect between *timing of collision detection* and *modality of system feedback* was also non-significant (F(1,15)=0.13, p=0.72). So was the interaction effect between *ID* and *timing of collision detection* (F(1,15)=1.575, p=0.23).

4.2.2 Model Fitting: The previously used model of temporal pointing fits very well to the empirical error rates (see Table 1), robustly replicating the model in more complex settings. In the previous temporal pointing study [11], the TouchMax algorithm showed significantly better temporal pointing performance than did conventional collision detection algorithms, but our study did not find a significant difference in timing performance across varying *timing-feedback* conditions.

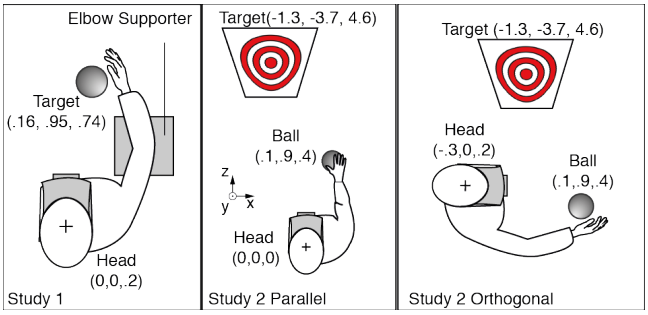

Figure 9: Geometric design of tasks in Study 1 (left) and Study 2 (right). The numbers are the absolute coordinate values of each element in space.

4.3 Discussion

Overall, the effect of the *modality of system feedback* and *timing of collision detection* factors on timing error was not statistically significant. This may be due to the tracking accuracy or the delay created by the sensing pipeline adding far more noise to the human process than taps on a touchpad. However, the fit between the temporal pointing model and all data was very high, and the amount of temporal noise (c_σ) measured for each situation followed an interesting pattern (see Table 1).

When the collision detection rule and the feedback signal were imposed on the same modality (e.g., Visual+ First Contact and Visual·Haptics+ Minimum Speed), timing contained less noise (see Figure 8 and Table 1). In comparison to the Visual condition ($c_\sigma = 0.079$), the noise in First Contact collision increased 10.6% when Visual·Haptics feedback was provided from the system ($c_\sigma = 0.087$). Also, in comparison to the Visual·Haptics condition ($c_\sigma = 0.082$), the noise in Minimum Speed collision increased 2.3% when there was only Visual feedback from the system ($c_\sigma = 0.084$).

In Figure 8, Boxer seems to show a larger difference from conventional collision for lower ID values. Research on moving target interception suggests that the more difficult the targets a person has to select, the faster the movement of the hand [18, 19]. The faster the motion, the greater the dispersion of the motion [14], so Boxer's precision can be lower for difficult targets.

Even though no significant performance differences were found between collision techniques in timing performance, this does not mean that there is no difference in spatial performance between them. Very small timing differences can be amplified significantly in tasks such as shooting and batting (see Figure 10). In the experiment on virtual batting described below, we compared the spatial accuracy and precision of the Boxer technique with the conventional baseline.

5 STUDY 2: VIRTUAL BATTING

In Study 2, we measured spatial accuracy and precision of batting, comparing two collision techniques: (1) Conventional collision and (2) Boxer collision (see Figure 1). Participants again collided their virtual pointer with a virtual object, but this time the virtual object was moving with respect to the direction of collision (as in batting). The goal given to the participants was to hit the ball at the center of the target wall, without any time limit. (See Figure 11.)

The visibility of the target wall and the ball to be hit was varied as an independent variable. The idea was that if a participant can see the ball and the wall at the same time, the visual sense can be used to perceive the direction in which the ball must be hit before the arm actually starts to move. If the performance of a collision

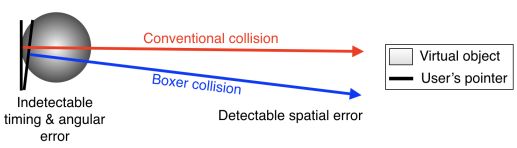

Figure 10: Study 2 showed that the angular error induced from the very small timing error can be amplified in proportion to distance in virtual batting situations.

technique shows a significant difference based on the visibility of the ball and the wall, it can be shown that said collision technique should adaptively change in accordance with the given VR situation for the user rather than operate as a fixed algorithm.

5.1 Method

5.1.1 Participants: The 12 participants were the same as in Study 1.

5.1.2 Experiment Design: The experiment followed a 2×2 within-subject design with two independent variables: *batting direction* (Parallel and Orthogonal) and *collision technique* (Conventional and Boxer).

- Conventional technique: A collision is detected from the `onCollisionEnter` callback in the physics engine of `Unity`. The function responds to the initial contact with the object but also moves the object immediately after the collision, with a force vector estimated via the physics engine.
- Boxer technique: A collision is detected through the Boxer algorithm (see Algorithm 1). The detection is postponed until the point where the speed of the virtual pointer reaches a minimum after the first `onCollisionEnter` event. At the moment of collision, the object starts to move in line with an estimated force vector ($F_{Boxer} \cdot \vec{u}_{Boxer}$) and a haptic vibration is applied to the participant's palm. The direction of force was estimated from the instantaneous normal vector of the virtual pointer at the moment of collision.
- Parallel direction: Participants can see the target and the ball simultaneously within the viewing angle when hitting the ball (see Figure 9).
- Orthogonal direction: Participants cannot see the ball and the target within a single view at the same time when hitting the ball (see Figure 9).

We had two dependent variables: *batting accuracy* (mean error) with respect to the center of the target wall and *batting precision* (or standard deviation) in the shot distribution. Participants completed 50 trials per condition, for 2,400 shots in total.

5.1.3 The Task: Participants were instructed to tap on a virtual object, which was initially fixed in the virtual scene. In response to the direction of the collision event, participants could hit the object to the distant target (see Figure 11). Without any time pressure, they were encouraged to aim at the center of the target with maximum accuracy and precision. They were also instructed to tap harder on the object to produce high enough speed. In addition, they were told to make very short impacts without slowly pushing or penetrating the ball. A shot was recorded as successful when the object arrived inside the target within a certain time (3 sec). The application regenerated the target at the initial location with zero speed whenever the object was successfully shot to the target or exceeded the maximum flight time. After each regeneration, the color of the object temporarily became red, for 0.5 seconds, and no collision was detected for that period, to exclude unintentional shots by participants.

5.1.4 Material: The object was a white sphere presented in a dark visual space in `Unity` (see Figure 11). We retained the shape of the virtual pointer from the previous study, which was determined

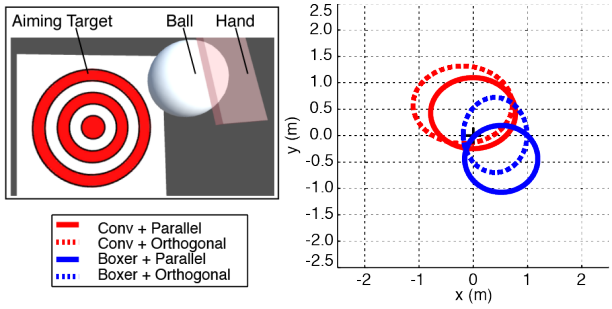

Figure 11: Accuracy and precision of batting for each experimental condition in Study 2. Using Boxer collision, batting precision was 1.171, which was significantly higher than conventional collision by 26.7%.

by the dominant hand of the participants. However, we did not mark the location toward which the the participants had to hit the ball, because, unlike in Study 1, users were able to hit targets in a uniform way. This was possible because the target wall (see 11) that the ball must fly into was clearly shown: there was a target wall (5 m × 5 m) at a fixed distance from the participant (exact coordinates are described in Figure 9). We used the same materials as in the previous study for producing haptic feedback in the Boxer condition.

5.1.5 Procedure: At the beginning, a brief introduction to the task was given, explaining the key factors of successful shots. We allowed the participants to practice batting until they got used to it. They were fully aware that there was no time constraint in the experiment and that they were able to rest whenever they felt tired. We counterbalanced the ordering of *collision technique* × *batting direction* provided to the participants. The experiment took around half an hour for each subject to complete.

5.1.6 Apparatus: To achieve the most accurate result for collision in the Conventional condition, we turned on the *continuous collision* option in `Unity`'s physics engine. Otherwise, we used the same settings as in Study 1.

5.2 Results

5.2.1 Batting Precision: There was a significant main effect of *collision technique* on *batting precision* ($F(1,11)=12.576$, $p=0.005$). The mean precision of Conventional was 1.597 m ($\sigma=0.124$), and that of Boxer was 1.171 m ($\sigma=0.131$). There was no significant interaction effect between *collision technique* and *batting direction* on *batting precision*, with $F(1,11)=4.782$, $p=0.051$: In Parallel direction, precision in the Conventional condition was 1.386 m ($\sigma=0.151$), while the precision with the Boxer condition was 1.171 ($\sigma=0.168$). In Orthogonal direction, the precision with the Conventional condition was 1.807 m ($\sigma=0.143$) and that in the Boxer condition was 1.171 m ($\sigma=0.117$).

5.2.2 Batting Accuracy: There was no significant main effect of *collision technique* on *batting accuracy* ($F(1,11)=1.56$, $p=0.24$): Mean accuracy with the Conventional condition was 0.667 m ($\sigma=0.051$), and it was 0.73 m ($\sigma=0.066$) for the Boxer condition. However, we did find a significant interaction effect between *batting direction* and *collision technique* on *batting accuracy* ($F(1,11)=10.969$,

Table 2: Precision and accuracy observed with the collision conditions in Study 2. The numbers marked with an asterisk (*) indicate statistically significant differences. Overall, the Boxer algorithm showed 26.7% higher precision (1.171) than the Conventional method did (1.597). However, in one situation (Parallel batting), the accuracy of Boxer was significantly reduced compared to conventional collision.

Technique	Precision		Accuracy	
	Orthogonal	Parallel	Orthogonal	Parallel
Conventional	1.597*		0.667	
	1.807*	1.386	0.819	0.591*
Boxer	1.171*		0.730	
	1.171*	1.171	0.67	0.852*

p=0.007). Pairwise comparison showed that the difference between Conventional and Boxer is significant for the Parallel direction (p=0.006): in the Parallel direction, the accuracy in the Conventional condition was 0.591 m (σ=0.061) while 0.852 m (σ=0.081) was seen for the Boxer condition. Pairwise comparison showed the difference between Conventional and Boxer not to be significant for the Orthogonal direction (p=0.076): In the Orthogonal direction, the accuracy in the Conventional condition was 0.819 m (σ=0.081), while it was 0.670 m (σ=0.089) in the Boxer condition.

5.3 Discussion

Since instantaneous tasks end before the first feedback from the system reaches the user, the user must prepare a complete movement from the given information before starting the task. The results of Study 2 show that this can vary greatly, depending on the given situation, so it is necessary to design a collision detection algorithm adaptively co-aligning to the user's preparation.

Firstly, participants' preparation for the moment of collision seems to depend mainly on their proprioceptive sensation. This conclusion is supported by the fact that the Boxer condition, in which collisions are triggered at the moment when the participant's proprioceptive sensation is maximized, showed uniform and high precision for all situations in the experiment. Participants, however, showed different strategies for the direction of collision, depending on the given task condition. For example, if the ball and target are visible simultaneously (in Parallel), the user appears to prepare the direction of the collision on the basis of the given visual information. This is evidenced by the fact that Conventional collisions showed particularly increased accuracy in Parallel battings. However, in Orthogonal conditions, where the relative position of the ball and the target cannot be determined through visual information, this relationship has changed and the accuracy of the Boxer algorithm has been measured to be higher. In summary, the direction of the collision is prepared mainly from the given visual information, while the precision of repeating batting in a certain direction seems to be informed by proprioception.

Perhaps the most surprising finding of the two studies is that the collision technique can have such a large impact on user performance, particularly with regard to spatial precision (26% to 45%). It is interesting to note that the two methods, which did not show a significant difference in temporal performance (Study 1), showed large differences when consideration was confined to spatial performance (Study 2).

With the fast development of hardware and VR applications, the results strongly suggest that it is not enough just to create visually pleasing effects. Rather, successful techniques should consider the motor, perceptual, and cognitive aspects of the task. The first take-away from the research is that the conventional collision technique is overly grounded in the user's visual sensation. A new design possibility, for applications where accuracy and precision are important, is to allow choice among collision detection methods according to preference.

To improve collision detection even further, our results point toward an "ideal pipeline" for collision detection in VR/AR in the future: (1) a classification method to distinguish normal tasks from instantaneous ones, (2) a method for inference of the user-intended moment of collision, and (3) a method for optimal co-alignment of sensory feedback with the moment. More research on human sensory integration and motor planning is needed.

6 CONCLUSION

This paper has presented a comparison of some collision detection algorithms widely used in VR with a novel idea called Boxer. The results of the study support an earlier theory according to which user performance of precision-requiring tasks is improved if collisions are triggered at a moment that is highly salient for users [11]. The results of the second study confirm an earlier finding [5] that users use haptic information and visual information differently, in line with the given situation. A take-away from the two studies, therefore, is that different collision detection algorithms are required, with the best choice depending on the task at hand.

Future research needs to address some limitations of this work. Firstly, the implementation of Boxer builds on a commercial physics engine, and it is impossible to fully exclude the possibility that some undocumented factors in its low-level implementation affected the experiment. In addition, the assumption that the user's pointer is flat leaves room for the implementation of another form of Boxer collision, for more realistic geometry of pointers.

Secondly, comparing the conventional collision technique with Boxer showed that user performance varies greatly with the design of the collision technique. However, overall, Boxer was not always the optimal collision technique. More sophisticated collision techniques should be examined in the future, if we are to achieve optimal overall performance in a wider range of situations. For example, one might eliminate the directional bias of Boxer by rotating the direction of the collision by a certain amount toward the direction of the collision detected in the first contact. These limitations notwithstanding, the two studies have broadened our understanding of collision tasks in VR.

ACKNOWLEDGMENTS

The research received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement 637991). It was supported also by the Basic Science Research Program through the National Research Foundation of Korea (NRF), funded by the Ministry of Science, ICT & Future Planning (NRF-2017R1C1B2002101).

REFERENCES

[1] Bruno Araujo, Ricardo Jota, Varun Perumal, Jia Xian Yao, Karan Singh, and Daniel Wigdor. 2016. Snake Charmer: Physically Enabling Virtual Objects. In *Proceedings of the TEI'16: Tenth International Conference on Tangible, Embedded, and Embodied Interaction*. ACM, 218–226.

[2] Mahdi Azmandian, Mark Hancock, Hrvoje Benko, Eyal Ofek, and Andrew D Wilson. 2016. Haptic Retargeting: Dynamic Repurposing of Passive Haptics for Enhanced Virtual Reality Experiences. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 1968–1979.

[3] Kimberly Chu and Chui Yin Wong. 2011. Mobile input devices for gaming experience. In *User Science and Engineering (i-USEr), 2011 International Conference on*. IEEE, 83–88.

[4] Christer Ericson. 2004. *Real-time collision detection*. CRC Press.

[5] Marc O Ernst and Martin S Banks. 2002. Humans integrate visual and haptic information in a statistically optimal fashion. *Nature* 415, 6870 (2002), 429–433.

[6] Marc O Ernst, Martin S Banks, and Heinrich H Bülthoff. 2000. Touch can change visual slant perception. *Nature neuroscience* 3, 1 (2000), 69–73.

[7] Marc O Ernst and Heinrich H Bülthoff. 2004. Merging the senses into a robust percept. *Trends in cognitive sciences* 8, 4 (2004), 162–169.

[8] Paul M Fitts. 1954. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of experimental psychology* 47, 6 (1954), 381.

[9] Sidhant Gupta, Dan Morris, Shwetak N Patel, and Desney Tan. 2013. AirWave: non-contact haptic feedback using air vortex rings. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*. ACM, 419–428.

[10] Anuruddha Hettiarachchi and Daniel Wigdor. 2016. Annexing Reality: Enabling Opportunistic Use of Everyday Objects as Tangible Proxies in Augmented Reality. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 1957–1967.

[11] Byungjoo Lee and Antti Oulasvirta. 2016. Modelling Error Rates in Temporal Pointing. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 1857–1868.

[12] Ming Li, Katrin Arning, Luisa Vervier, Martina Ziefle, and Leif Kobbelt. 2015. Influence of temporal delay and display update rate in an augmented reality application scenario. In *Proceedings of the 14th International Conference on Mobile and Ubiquitous Multimedia*. ACM, 278–286.

[13] Pedro Lopes, Alexandra Ion, and Patrick Baudisch. 2015. Impacto: Simulating Physical Impact by Combining Tactile Stimulation with Electrical Muscle Stimulation. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. ACM, 11–19.

[14] Richard A Schmidt, Howard Zelaznik, Brian Hawkins, James S Frank, and John T Quinn Jr. 1979. Motor-output variability: A theory for the accuracy of rapid motor acts. *Psychological review* 86, 5 (1979), 415.

[15] Richard HY So and German KM Chung. 2005. Sensory motor responses in virtual environments: Studying the effects of image latencies for target-directed hand movement. In *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*. IEEE, 5006–5008.

[16] Rajinder Sodhi, Ivan Poupyrev, Matthew Glisson, and Ali Israr. 2013. AIREAL: interactive tactile experiences in free air. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 134.

[17] Unity Technologies. 2016. Unity. Game Engine. (2016). Retrieved June 2, 2016 from https://unity3d.com.

[18] J Tresilian, J Oliver, and T Carroll. 2003. Temporal precision of interceptive action: differential effects of target size and speed. *Experimental brain research* 148, 4 (2003), 425–438.

[19] James R Tresilian and Andrew Lonergan. 2002. Intercepting a moving target: effects of temporal precision constraints and movement amplitude. *Experimental Brain Research* 142, 2 (2002), 193–207.

[20] Robert J van Beers, Daniel M Wolpert, and Patrick Haggard. 2002. When feeling is more important than seeing in sensorimotor adaptation. *Current biology* 12, 10 (2002), 834–837.

[21] Loutfouz Zaman and I Scott MacKenzie. 2013. Evaluation of nano-stick, foam buttons, and other input methods for gameplay on touchscreen phones. In *International Conference on Multimedia and Human-Computer Interaction-MHCI 2013*. 69–1.

[22] Loutfouz Zaman, Daniel Natapov, and Robert J Teather. 2010. Touchscreens vs. traditional controllers in handheld gaming. In *Proceedings of the International Academic Conference on the Future of Game Design and Technology*. ACM, 183–190.