Techniques d'interaction pour la sélection d'objets et le contrôle d'expressions faciales en réalité virtuelle

Marc Baloup

▶ **To cite this version:**

Marc Baloup. Techniques d'interaction pour la sélection d'objets et le contrôle d'expressions faciales en réalité virtuelle. Interface homme-machine [cs.HC]. Université de Lille, 2021. Français. ⟨NNT : 2021LILUB024⟩. ⟨tel-03508652⟩

HAL Id: tel-03508652

https://hal.science/tel-03508652v1

Submitted on 18 Mar 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

HAL Authorization

Thèse

de

Doctorat en Informatique

École doctorale MADIS-631

Techniques d'interaction pour la sélection d'objets et le contrôle d'expressions faciales en réalité virtuelle

Présentée et soutenue publiquement le

15 décembre 2021

par

Marc Baloup

pour obtenir le grade de docteur délivré par

l'Université de Lille

devant un jury composé de

Présidente	**Indira Thouvenin**	Professeure, Université de Technologie de Compiègne
Rapporteurs	**Thierry Duval**	Professeur, IMT Atlantique
	Laurence Nigay	Professeure, Université de Grenoble
Examinateurs	**Olivier Chapuis**	Chargé de Recherche, Université Paris-Saclay
	Laurent Grisoni	Professeur, Université de Lille
Directeur de thèse	**Géry Casiez**	Professeur, Université de Lille
Encadrants	**Martin Hachet**	Directeur de recherche, Inria Bordeaux
	Thomas Pietrzak	Maître de conférences, Université de Lille

Remerciements

Tout d'abord, j'aimerais remercier mes encadrants Géry Casiez, Thomas Pietrzak et Martin Hachet, pour leurs conseils et leur soutien tout au long de la thèse.

Merci aux rapporteurs Laurence Nigay et Thierry Duval pour leurs remarques et critiques intéressantes sur ce manuscrit. Merci aussi à eux et aux autres membres du jury, Olivier Chapuis et Laurent Grisoni, d'avoir été présent et pour leurs questions et commentaires constructifs.

Je veux aussi remercier les membres de l'équipe Loki, qui m'ont accueilli pendant mon stage de master et ma thèse, et avec lesquels j'ai passé de bons moments (en particulier dans le Starter 🙂).

Merci aussi à mon père, qui m'a toujours encouragé à aller le plus loin possible dans mes études. Il a surtout su prendre soin de moi, mes frères et ma sœur, même dans les moments qui ont été difficiles, ce qui m'a permis d'arriver jusqu'ici aujourd'hui.

Enfin, un grand merci à Lucie, qui m'apporte beaucoup de soutien depuis ces deux dernières années, notamment pendant ma thèse et les moments où j'en avais le plus besoin.

Ah oui au fait : c'est bon, j'ai trouvé! 🥳 (Certains comprendront.)

Résumé

Les technologies de réalité virtuelle connaissent un essor sans précédent par le développement de casques qui possèdent des résolutions d'affichage et des capacités de suivi du mouvement inédits. Ces casques sont associés à des contrôleurs dotés de nombreux degrés de liberté qui offrent la possibilité d'améliorer et de développer de nouvelles techniques d'interaction. L'interaction dans ces environnements immersifs reste en effet bien souvent plus laborieuse que dans la réalité. Nous nous sommes concentrés dans cette thèse sur l'amélioration de la tâche de sélection et le développement de techniques dédiées au contrôle d'expressions faciales.

Pour la tâche de sélection d'objets 3D, la main virtuelle et le *Raycasting* sont les techniques les plus fréquemment utilisées. Bien que *Raycasting* permette de sélectionner des objets hors de portée, la sélection est d'autant plus difficile que les objets sont éloignés, petits ou partiellement occultés. Nous avons mis en évidence comment réduire les erreurs de sélection de *Raycasting* par l'utilisation d'un filtrage adapté. Nous avons également développé *RayCursor*, une amélioration de *Raycasting* qui repose sur l'ajout d'un curseur que l'utilisateur peut déplacer le long du rayon. Ce curseur permet de combiner *Raycasting* avec des techniques de facilitation du pointage par proximité. Nous avons étudié un ensemble de retours visuels adaptés à la visualisation et la sélection de cibles à proximité. Une série d'expériences contrôlées a permis de mettre en évidence les bénéfices de *RayCursor* par rapport à un ensemble de techniques de référence de la littérature.

Pour le contrôle d'expressions faciales, nous avons proposé un ensemble de techniques non isomophiques pour dépasser les limitations des techniques isomorphiques proposées dans la littérature, qui reposent sur la détection ou l'estimation de l'expression réelle de l'utilisateur pour la transposer à son avatar. Le contrôle précis d'expressions est difficile et il est impossible à l'utilisateur de contrôler une expression autre que celle de son visage. Les techniques que nous proposons reposent sur une décomposition du contrôle d'expressions faciales en sous-tâches que nous formalisons dans un espace de conception : la sélection de l'expression faciale, le contrôle de son intensité et de sa durée. Des expériences contrôlées ont permis d'isoler les techniques les plus pertinentes pour réaliser chacune des sous-tâches, afin de concevoir une technique que nous appelons *EmoRayE*. Celle-ci a été validée dans une expérience écologique.

Abstract

Virtual reality technologies are evolving fast with the development of headsets that have increased display resolutions and motion tracking capabilities. These headsets are associated with controllers with many degrees of freedom that offer the possibility to improve and develop new interaction techniques. Indeed, interaction in these immersive environments is often more laborious than in reality. In this thesis, we focused on the improvement of the selection task and the development of techniques dedicated to the control of facial expressions.

For the 3D object selection task, the virtual hand and *Raycasting* are the most frequently used techniques. Although *Raycasting* allows selecting objects out of reach, the selection is more difficult when the objects are far away, small or partially occluded. We have highlighted how to reduce the selection errors of *Raycasting* by using an adapted filtering. We also developed *RayCursor*, an improvement of *Raycasting* that relies on adding a cursor that the user can move along the ray. This cursor allows *Raycasting* to be combined with proximity-based pointing facilitation techniques. We studied a set of visual feedbacks adapted to the visualization and selection of nearby targets. A series of controlled experiments highlighted the benefits of *RayCursor* over a set of reference techniques from the literature.

For the control of facial expressions, we have proposed a set of non-isomorphic techniques to overcome the limitations of the isomorphic techniques proposed in the literature, which rely on the detection or estimation of the users' real expressions to transpose them to their avatar. It is difficult to control precisely expressions and it is impossible for users to control an expression different from the expression on their face. Our techniques are based on the decomposition of facial expression control into subtasks that we formalize in a design space : facial expression selection, intensity and duration control. Controlled experiments allowed us to isolate the most relevant techniques to perform each of the subtasks, in order to design a technique that we call *EmoRayE*, that was validated in an ecological experiment.

Table des matières

Introduction

Après avoir connu l'essor des interfaces graphiques pour ordinateurs de bureau, celui des interfaces tactiles pour périphériques mobiles, nous assistons aujourd'hui à l'essor des interfaces humain-machine pour la réalité virtuelle immersive. Ces dernières sont non seulement utilisées dans le domaine du divertissement mais aussi dans des contextes professionnels, par exemple pour l'entraînement dans des situations dangereuses comme le saut en parachute. Les casques de réalité virtuelle immersive grand public ont commencé à apparaître dans les années 1990, par exemple avec le Virtual-Boy de Nintendo ou le PUD-J5A de Sony pour la PlayStation 2. Cependant, ces dispositifs étaient très limités par le manque de capacités à suivre les mouvements de l'utilisateur et par la faible résolution des écrans. Les nouveaux casques de réalité virtuelle grand public de cette dernière décennie règlent ces soucis en proposant un suivi à 6 degrés de liberté des mouvements de l'utilisateur (le premier étant Oculus Rift DK2 [66]). Les caractéristiques évoluant au fil des années et les prix attractifs pour le grand public ont permis, entre autres, le développement d'applications et de jeux vidéo bien plus immersifs que précédemment. Ces casques sont associés à des contrôleurs dotés de nombreux degrés de liberté qui offrent la possibilité d'améliorer et de développer de nouvelles techniques d'interaction.

Si l'interaction avec les objets de la vie de tous les jours demande le plus souvent un effort cognitif faible, proposer des interactions équivalentes dans le cadre d'environnements virtuels n'est pas évident. Les interfaces de bureau ou mobiles actuelles ont du mal à proposer des interfaces qui sont transparentes pour l'utilisateur, qui doit donc apprendre de nouvelles techniques d'interaction. Par exemple pour la prise en main d'un objet réel, un humain utilise généralement ses doigts pour le saisir et le manipuler, et est capable de le ressentir grâce au sens du toucher. Sur un ordinateur, l'utilisateur doit cependant utiliser un périphérique comme une souris pour effectuer cette tâche. La réalité virtuelle immersive réduit la frontière avec les interactions réelles, grâce à la possibilité de transposer plus facilement les mouvements de l'utilisateur dans l'environnement virtuel, et à la capacité de fournir des retours sensoriels (visuels, auditifs et tactiles principalement). Les techniques développées pour les environnements immersifs cherchent alors à rendre les interactions au moins aussi faciles que dans la réalité. Pour prendre un objet virtuel en main, il est par exemple possible d'utiliser des périphériques d'entrée pouvant d'une part capter la position des doigts et, d'autre part, faire ressentir à l'utilisateur la force exercée par l'objet sur les doigts. En plus d'essayer de reproduire le mieux

que possible les interactions de la réalité, les environnements virtuels permettent aussi des interactions difficiles ou impossibles à réaliser dans le monde physique. En effet, que ce soit en immersion dans un casque de réalité virtuelle, ou dans une application sur ordinateur, l'environnement virtuel peut se passer des limitations de la physique. Par exemple, il est possible de copier un objet virtuel, de voler et de se téléporter dans l'espace virtuel, ou encore de sélectionner et manipuler des objets à distance.

Utiliser notre corps comme moyen d'interaction avec l'environnement virtuel, comme c'est le cas dans le monde physique, nécessite que celui-ci incarne un *avatar*. Il s'agit d'une représentation graphique de soi qui est visible dans le monde virtuel. En réalité virtuelle, ces avatars sont des modèles 3D, souvent de forme humaine ou humanoïde. En plus des interactions avec l'environnement, la réalité virtuelle permet de rencontrer des personnes dans des mondes immersifs en ligne, tels que Facebook, VR Chat ou Rec Room [27, 98, 82, 63]. Les capacités de suivi de mouvement des casques et des manettes permettent un contrôle *isomorphique* de son avatar, c'est-à-dire que le système transpose les mouvements de l'utilisateur à ceux de son avatar de la manière la plus transparente possible. Accompagnés par la communication vocale, ceux-ci permettent des interactions plus engageantes [31] entre les utilisateurs par rapport à la communication textuelle, audio ou vidéo.

Contrairement aux avatars hors de la RV immersive, comme le personnage que l'on contrôle dans un jeu vidéo ou une image de profil dans une communauté en ligne, les avatars en réalité virtuelle immersive présentent des défis supplémentaires. La recherche actuelle tente donc d'étudier la sensation d'immersion et d'incarnation vis-à-vis des avatars [28, 29], et améliorer leur contrôle [23] et leur rendu visuel [6]. Dans ce contexte, ma thèse fait partie d'un projet de recherche appelé Avatar [1], un groupement de plusieurs équipes de recherche Inria ayant pour but de concevoir des avatars offrant une meilleure incarnation et étant plus interactifs et plus sociaux. Pour cela, le projet cherche à améliorer tous les aspects du contrôle [23] et du rendu d'un avatar [6], c'est-à-dire l'acquisition [68] ou la simulation d'un avatar, les paradigmes d'interaction et les retours multisensoriels [80] liés aux avatars.

Dans cette thèse, nous avons choisi deux sujets que nous trouvons fondamentaux dans un contexte immersif, et plus particulièrement dans un environnement social virtuel. Ces sujets nous ont inspiré, en nous basant sur différentes contributions et limitations de la littérature. D'une part, nous avons cherché à faciliter la tâche de sélection en réalité virtuelle, en nous basant sur *Raycasting*, une technique de sélection d'objets 3D à distance. D'autre part, nous avons développé des techniques dédiées au contrôle d'expressions faciales d'un avatar, en cherchant à compenser les limitations des techniques isomorphiques.

Pour la tâche de sélection d'objets 3D, la main virtuelle et le *Raycasting* sont les techniques les plus fréquemment utilisées, car elles sont généralement incluses dans la plupart des environnements de développement de réalité virtuelle. La main virtuelle permet

1. https://avatar.inria.fr/

la sélection d'objets que l'utilisateur peut atteindre avec sa main, ce qui se rapproche le plus de l'interaction entre une personne et un objet physique. Cependant, cette technique ne permet pas la sélection d'objets hors de portée. *Raycasting* règle ce problème, grâce à l'utilisation d'un rayon, qui s'apparente à un pointeur laser. Néanmoins, la sélection est d'autant plus difficile que les objets sont éloignés et petits. En effet, les tremblements de la main de l'utilisateur et les erreurs de suivi sont d'autant plus marqués que la cible est éloignée et petite. Dans nos travaux, nous avons mis en évidence comment réduire les erreurs de sélection de *Raycasting* par l'utilisation d'un filtrage adapté. Aussi, *Raycasting* ne permet pas la sélection d'objets occultés, car seul le premier qui est intersecté par le rayon est pris en compte. Nous avons alors développé *RayCursor*, une amélioration de *Raycasting* qui repose sur l'ajout d'un curseur que l'utilisateur peut déplacer le long du rayon. Ce curseur permet de combiner *Raycasting* avec des techniques de facilitation du pointage par proximité. Nous avons comparé différentes fonctions de transfert pour le contrôle du curseur le long du rayon, en utilisant la surface tactile de la manette HTC Vive. Nous avons aussi étudié un ensemble de retours visuels adaptés à la visualisation et la sélection de cibles à proximité. Une série d'expériences contrôlées a permis de mettre en évidence que *RayCursor* est plus rapide et moins sujette à des erreurs de sélection, par rapport à un ensemble de techniques de référence de la littérature.

Pour le contrôle d'expressions faciales, les techniques existantes dans la littérature essayent de transposer l'expression faciale de l'utilisateur à celui de son avatar [52, 90]. Elles utilisent pour cela des caméras externes ou des capteurs intégrés au casque de réalité virtuelle. Ces techniques isomorphiques peuvent cependant non seulement manquer de précision [90] mais aussi nécessiter une installation complexe. Il est par ailleurs impossible à l'utilisateur de contrôler une expression autre que celle de son visage. Nous proposons donc des techniques non isomorphiques, c'est-à-dire que l'utilisateur choisit manuellement l'expression du visage de son avatar en utilisant des techniques d'interaction. Ces techniques reposent sur une décomposition du contrôle d'expressions faciales en sous-tâches que nous formalisons dans un espace de conception : la sélection de l'expression faciale, le contrôle de son intensité et de sa durée. Pour la sous-tâche de sélection d'une expression, nous avons réalisé quatre techniques. *RayMoji* et *EmoTouch* proposent des émojis correspondant à des expressions faciales, que l'utilisateur peut sélectionner. *RayMoji* dispose les émojis en grille, comme les menus d'émojis dans les messageries instantanées, et l'utilisateur peut faire sa sélection grâce à un *Raycasting*. Dans *EmoTouch*, les émojis sont arrangés en fonction des émotions qu'ils représentent, dans un menu circulaire navigable avec la surface tactile de la manette. *EmoGest* reconnaît des gestes symboliques sur la surface tactile, qui activent les expressions correspondantes. Enfin, *EmoVoice* active des expressions sur le visage de l'avatar, en fonction de mots-clés prononcés par l'utilisateur pendant l'appui sur un bouton. Une expérience contrôlée nous a permis de concevoir une nouvelle technique combinant les avantages des techniques précédentes, que nous appelons *EmoRay*. Pour les sous-tâches de contrôle d'intensité et de la durée de l'expression, nous avons conçu cinq techniques, qui exploitent soit la pression sur la gâchette, l'inclinaison ou la secousse de la manette, la longueur d'un élastique virtuel ou la position visée sur un menu circu-

laire. Une deuxième expérience contrôlée a permis de déterminer que l'élastique virtuel était la technique préférée des participants. Nous la combinons donc avec *EmoRay*, afin de concevoir une technique que nous appelons *EmoRayE*. Enfin, une expérience écologique a montré qu'*EmoRayE* était utilisable dans des réseaux sociaux virtuels.

Ce manuscrit de thèse est découpé en trois chapitres. Dans le premier chapitre, nous proposons un état de l'art sur les techniques d'interaction utilisées en réalité virtuelle. Nous nous concentrons sur la tâche de pointage ainsi que les techniques de contrôle de l'expression faciale de son avatar. Nous présentons aussi un espace de conception permettant d'organiser les techniques existantes sur le contrôle d'expression faciale, et de concevoir de nouvelles techniques. Dans le chapitre II, nous décrivons tout d'abord *RayCursor*, avant de présenter les expériences ayant guidé la conception de la technique. Nous terminons ce chapitre par une dernière expérimentation montrant les avantages de *RayCursor* par rapport à des techniques de la littérature. Le dernier chapitre présente l'ensemble des techniques que nous avons développées permettant le contrôle non isomorphique de l'expression faciale de son avatar. Nous décrivons leur implémentation ainsi que les expérimentations permettant de les comparer. Ce chapitre se conclut par la description d'*EmoRayE*, et sa validation lors d'une expérience écologique. Enfin, nous concluons ce manuscrit par une présentation des travaux futurs envisagés.

Chapitre I

État de l'art

Dans ce premier chapitre, nous explorons différents types d'interactions qui permettent à l'utilisateur d'interagir dans l'environnement virtuel immersif. Dans la première section, nous discuterons des techniques permettant à l'utilisateur d'interagir avec son environnement. Nous évoquerons les techniques de pointage puis de manipulation d'objets 3D avant de parler des déplacements de l'utilisateur en réalité virtuelle. Enfin, dans la dernière section, nous décrivons les techniques d'interaction permettant de prendre le contrôle de l'avatar que l'on incarne. Nous discutons des solutions existantes pour contrôler l'avatar de manière globale, avant de nous concentrer sur le contrôle de l'expression faciale.

I.1 Interagir avec l'environnement virtuel

L'interaction avec l'environnement comprend la sélection et la manipulation d'objets virtuels ainsi que les déplacements de l'utilisateur. La tâche de sélection est divisée en deux sous-tâches : le pointage et la validation de la sélection. La classification de Bowman *et al.* [10] regroupe les tâches de sélection et de manipulation car elles sont étroitement liées : par exemple, manipuler un objet nécessite tout d'abord de le sélectionner. Aussi, ils formalisent les déplacements comme étant le changement de point de vue de l'utilisateur dans la scène virtuelle.

Dans cette section, nous commençons par décrire des techniques de pointage, puis nous discutons des techniques de manipulation existantes avant d'explorer les techniques de déplacement.

I.1.1 Techniques de pointage

Dans les jeux et applications de réalité virtuelle qui utilisent des manettes à 6 degrés de liberté, les techniques de sélection utilisées la plupart du temps sont la main virtuelle et le *Raycasting* [10, 61]. La main virtuelle permet de sélectionner directement un objet en approchant la manette de celui-ci. Cependant, elle ne permet pas de sélectionner directement des cibles un peu plus éloignées, à moins de se déplacer dans l'espace virtuel.

Le *Raycasting* règle en partie ce problème : cette technique s'apparente à un pointeur laser, il suffit de viser une cible avec le rayon qui part de la manette pour sélectionner l'objet voulu. Le problème de cette technique est qu'il est difficile de viser des petites cibles qui sont éloignées, à cause des tremblements de la main ou de la précision des capteurs de la manette. De plus, si une cible est cachée derrière un obstacle, celui-ci ne pourra pas être sélectionné, puisque *Raycasting* ne considère que le premier objet intersecté par le rayon.

Dans cette section, nous commençons par présenter les différents mécanismes de désambiguïsation qui ont été proposés pour ces techniques. Nous présentons ensuite des techniques qui ajoutent des degrés de liberté supplémentaires au rayon, avant de couvrir les techniques de facilitation du pointage qui reposent sur la sélection par proximité.

I.1.1.1 Techniques de désambiguïsation

Argelaguet et Andujar fournissent une taxonomie des différentes techniques conçues pour améliorer le *Raycasting* [5]. La plupart de ces techniques sont basées sur l'utilisation d'un volume au lieu d'un rayon, ce qui nécessite l'utilisation de techniques de désambiguïsation pour la sélection. Ils distinguent trois groupes de mécanismes de désambiguïsation : manuel, heuristique et comportemental.

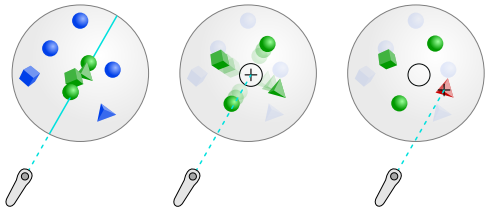


FIGURE I.1 – Flower Ray [33] : si le rayon intersecte plusieurs cibles, celles-ci sont réparties sur l'écran pour faciliter la sélection d'une cible particulière.

L'approche manuelle nécessite des étapes supplémentaires pour sélectionner manuellement une cible parmi celles qui sont mises en évidence. Par exemple, dans Flower Ray, Grossman *et al.* affichent dans un menu circulaire les objets traversés par le rayon [33] (figure I.1). La technique Menu Cone affiche également les cibles à désambiguïser dans un menu et l'utilisateur fait un geste pour sélectionner la cible désirée [79]. De la même manière, la technique SQUAD, proposée par Kopper *et al.*, adapte le *Raycasting* pour projeter une sphère sur la surface en intersection la plus proche du rayon, afin de déterminer quels objets sont susceptibles d'être sélectionnés [45]. Les objets sélectionnables sont ensuite répartis sur quatre quadrants et l'utilisateur affine la sélection jusqu'à ce que l'objet souhaité puisse être sélectionné. SQUAD a montré des performances nettement meilleures que *Raycasting* pour les cibles de petites tailles et les faibles densités de cibles, mais elle présente également une dégradation significative des performances pour les cibles de grandes tailles et les densités importantes, en raison du nombre accru d'étapes pour sélectionner une cible. Cashion *et al.* proposent une variante de SQUAD, appelée Expand, qui ajoute la possibilité de zoomer [14]. Ils montrent qu'Expand est plus rapide que SQUAD pour des densités d'objets élevées. Sans recourir

aux menus, le Depth Ray [33] (figure I.2 a)) utilise un curseur fixe au milieu d'un rayon dont la longueur est également fixe. Lorsque le rayon intersecte plusieurs objets, celui qui est le plus proche du curseur peut être sélectionné. L'utilisateur peut alors ajuster la position du Depth Ray en changeant la position et l'orientation de sa main. Les auteurs l'ont utilisé avec un écran volumétrique, autour duquel l'utilisateur peut se déplacer. En réalité virtuelle (RV), l'espace est grand, potentiellement infini, et ce curseur fixe n'est ni plus ni moins qu'une main virtuelle distante. Grossman et Balakrishnan ont également proposé la technique du Lock Ray qui consiste à verrouiller tous les objets en intersection, avant d'en sélectionner un à l'aide du curseur [33], en bouclant entre les objets sélectionnés [40]. Cependant, cela n'améliore pas le temps de mouvement. En utilisant un smartphone pour sélectionner des objets dans le monde physique, Delamare *et al.* ont proposé deux techniques pour désambiguïser les cibles sélectionnées dans un cône [22]. Avec le P2Roll, l'utilisateur effectue un geste de roulement pour sélectionner la cible souhaitée. Avec le P2Slide, il réalise un geste de glissement sur la surface tactile. Leurs techniques n'ont été évaluées qu'avec un maximum de 16 cibles. Plasson *et al.* [73] proposent une amélioration de *Raycasting* avec l'ajout d'une loupe. Cet outil, manipulable dans l'espace virtuel par l'utilisateur, facilite la sélection d'une cible dans un environnement très dense. L'utilisateur déplace d'abord la loupe de manière à mieux distinguer la cible parmi les distracteurs, puis sélectionne celle-ci directement sur la loupe. La manipulation de la loupe étant une étape supplémentaire, cela affecte aussi le temps de sélection.

L'approche heuristique permet de déterminer la cible que l'utilisateur souhaite sélectionner selon des choix prédéterminés. La technique Flashlight, par exemple, met en évidence l'objet qui est le plus proche de l'axe central du cône de sélection [49]. Le Skicky-Ray, basé sur *Raycasting* [89], laisse quant à lui sélectionnable le dernier objet intersecté, jusqu'à ce qu'un autre soit touché. Le rayon virtuel est dévié vers les objets qui peuvent être sélectionnés, n'ayant plus ainsi de retour visuel pour sélectionner un autre objet. Cette technique n'a pas été évaluée. Schmidt *et al.* ont proposé différents algorithmes probabilistes basés sur le pointage pour déduire la cible que l'utilisateur veut sélectionner, mais cela nécessite un réglage complexe de schémas de pondérations selon l'application [86].

Enfin, les approches comportementales prennent en compte les actions de l'utilisateur avant la confirmation de la sélection pour déterminer l'objet à sélectionner. Par exemple, IntenSelect utilise une fonction de notation basée sur le temps pour déterminer le score des objets qui se situent dans un volume de sélection conique. L'objet avec le score le plus élevé peut être sélectionné [21]. De la même manière, le Smart Ray pondère continuellement les cibles en fonction de leur proximité au curseur du rayon [33]. Cependant, cette dernière technique est moins performante que les techniques telles que Flower Ray ou Depth Ray [33].

En résumé, les techniques d'interaction actuelles qui améliorent le *Raycasting* nécessitent un mécanisme de désambiguïsation qui ajoute des étapes supplémentaires pour faire la sélection, pour pouvoir sélectionner des cibles occultées. Lorsqu'elles sont évaluées, ces techniques sont plus performantes pour la sélection de petits objets dans des

environnements denses, mais elles sont également moins performantes que le *Raycasting* pour la sélection de cibles de tailles importantes. Grossman et Balakrishnan ont montré que le Depth Ray est plus performant que le Lock Ray, Flower Ray ou Smart Ray en raison du temps plus court nécessaire pour la phase de désambiguïsation [33]. Au lieu d'avoir à désambiguïser entre différentes cibles en plusieurs étapes, une autre approche consiste à ajouter des degrés de liberté supplémentaires au *Raycasting* pour aider à ajuster la sélection de la cible tout en manipulant le rayon.

I.1.1.2 Ajout de degrés de liberté supplémentaires

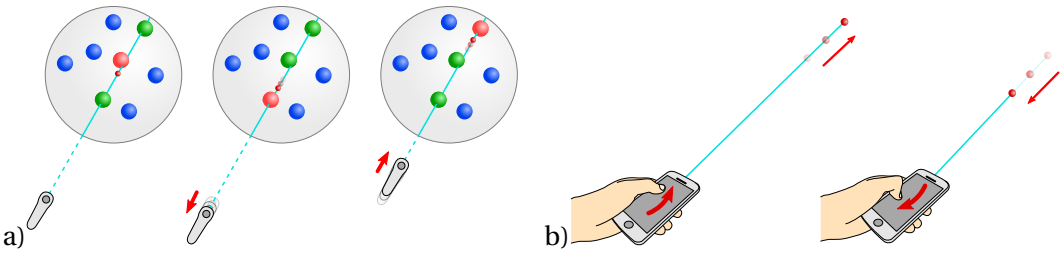


FIGURE I.2 – a) Depth Ray [33] : le curseur étant fixé sur le rayon, l'utilisateur doit déplacer la manette en avant et en arrière pour déplacer le curseur sur la profondeur; b) Technique de Ro *et al.* [81] : l'utilisateur déplace le curseur le long du rayon en glissant son doigt de haut en bas sur l'écran tactile d'un smartphone.

Grossman et Balakrishnan ont simplement ajouté un degré de liberté supplémentaire en ajoutant un curseur fixe au milieu d'un rayon [33], ce qui s'est avéré être la technique la plus efficace pour sélectionner des cibles sur un écran volumétrique. Cependant, dans le contexte d'environnements immersifs tels que les casques RV, l'utilisation de cette technique nécessiterait des déplacements importants de l'utilisateur pour désambiguïser les cibles. Au lieu d'avoir un curseur fixé sur un rayon, Ro *et al.* ont introduit la possibilité d'ajuster la profondeur du rayon en utilisant les déplacements relatifs d'un doigt sur l'écran tactile d'un smartphone [81]. Cependant, leur technique n'a pas été comparée à d'autres techniques et la fonction de transfert utilisée pour contrôler la longueur du rayon n'est pas détaillée. Des études récentes combinent les mouvements des mains avec le suivi de la tête et des yeux pour le pointage en réalité augmentée, comme le Pinpointing [47]. Cette technique est beaucoup plus précise que les techniques basées uniquement sur le regard. Cependant, elle présente des inconvénients communs avec le *Raycasting* : sensibilité à l'occultation, tremblement des mains et précision des périphériques d'entrée.

I.1.1.3 Techniques de facilitation du pointage

Diverses stratégies ont été étudiées pour faciliter le pointage. Par exemple, le pointage sémantique élargit les cibles dans l'espace moteur [8] Cette technique a été conçue pour le pointage 2D, mais une autre étude l'a étendue à la 3D en utilisant une souris d'ordinateur sur écran standard [26]. Cette technique améliore les performances de pointage dans des environnements qui présentent une faible densité de cibles mais est affectée par des cibles qui se trouvent sur le chemin du curseur (cibles distractives) [16]. Une autre stratégie consiste à remplacer le pointage par des gestes symboliques [37]. Elle per-

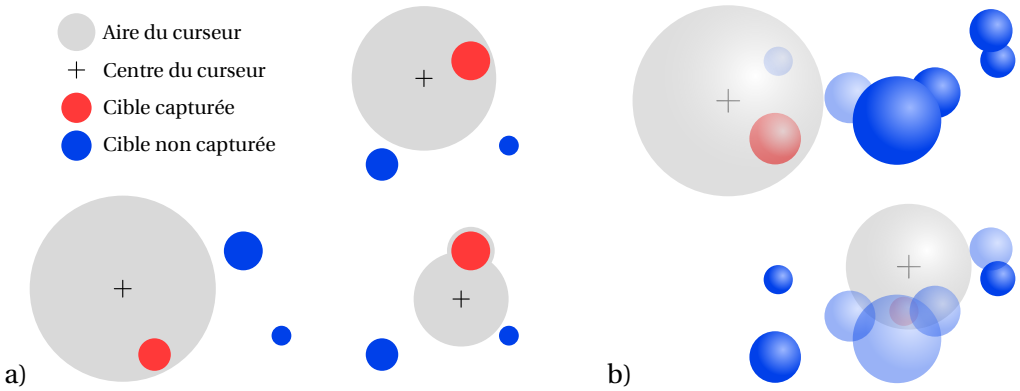


FIGURE I.3 – a) Bubble Cursor [32] : une bulle, centrée sur le curseur, adapte sa taille en temps réel, pour englober la cible la plus proche. Si jamais elle risque d'englober plusieurs cibles, la bulle forme une excroissance pour englober une seule cible ; b) 3D Bubble [94] : même principe que Bubble Cursor, mais en 3D. Les cibles à proximité de la bulle sont rendues de manière translucide afin faciliter l'identification de celles qui sont cachées.

met d'atténuer les problèmes dus aux gestes de pointage, mais elle n'est pas adaptée à des cibles arbitraires. À l'inverse, des études proposent des gestes 3D pour pointer des cibles sur des écrans 2D mais ces résultats sont difficilement applicables dans le cadre de la sélection de cibles 3D [97, 64].

L'une des techniques de facilitation du pointage les plus efficaces en 2D consiste à sélectionner la cible la plus proche du curseur. Par exemple, le Bubble Cursor affiche un disque (une bulle) centré sur un curseur de souris dont le rayon est ajusté en fonction de la distance à la cible la plus proche [32]. Cette technique est particulièrement efficace lorsque la densité des cibles est faible, quelle que soit leur taille. Le principal inconvénient reste le retour visuel introduit par la bulle qui change constamment de rayon. Guillon *et al.* ont évalué l'impact de plusieurs retours visuels sur la performance du Bubble Cursor et ont constaté qu'une simple mise en surbrillance de la cible la plus proche est efficace [35].

Vanacken *et al.* ont développé une version 3D du Bubble Cursor, appelée 3D Bubble, en utilisant une technique de main virtuelle pour contrôler un pointeur 3D [94] (figure I.3). Leur technique utilise une sphère semi-transparente en 3D englobant la cible la plus proche. Ils montrent que le Depth Ray est plus efficace que la 3D Bubble, elle-même plus efficace que *Raycasting*. De même, Vickers a défini un cube sensible autour d'un curseur 3D manipulé par une manette [95]. Lorsqu'un objet se trouve dans le cube sensible, le curseur saute sur l'objet.

En résumé, les techniques de désambiguïsation semblent efficaces pour sélectionner de petites cibles dans des environnements denses, mais elles augmentent globalement le temps de sélection en raison des étapes supplémentaires qu'elles introduisent. Les techniques de sélection utilisant la proximité des cibles semblent efficaces mais elles nécessitent l'utilisation d'un retour d'information approprié, notamment en 3D. Inspirés par la 3D Bubble, le Depth Ray et le rayon de longueur ajustable introduit par Ro *et al.*, nous proposons dans le chapitre II *RayCursor*, une technique combinant plusieurs des

avantages que ces techniques offrent, sans introduire d'étapes de désambiguïsation supplémentaires que de nombreuses techniques requièrent.

La thèse de Plasson ([72], section 4.1) propose un autre état de l'art sur l'amélioration de *Raycasting*. Celui-ci offre un point de vue différent de notre état de l'art, en classifiant les techniques selon ces critères : les techniques qui dépendent des cibles ou non et la caractéristique du rayon exploitée (longueur, forme ou courbure).

I.1.2 Techniques de manipulation

En 3D, la sélection précède souvent la manipulation d'un objet 3D. Bowman et Hodges [9] définissent la manipulation d'un objet virtuel comme étant une tâche consistant à en modifier les caractéristiques. Mendes *et al.*, dans leur état de l'art [58], énumèrent certaines de ces caractéristiques, par exemple la transformation spatiale (position, rotation et taille), le changement de propriétés visuelles (couleur, texture) ou la déformation libre. Cependant, Bowman et Hodges [9] ne considèrent que la transformation spatiale comme étant formellement de la manipulation. En effet, le changement de propriétés implique principalement l'utilisation de menus (donc de la sélection et de la manipulation) et la déformation libre d'un objet peut être formalisée comme étant la transformation spatiale de sous-parties de cet objet. Dans cette sous-section, nous considérons donc uniquement les manipulations visant à déplacer, orienter ou redimensionner un élément de l'environnement virtuel.

Le déplacement et la rotation sont des interactions courantes avec les objets réels [58]. Elles sont faciles à reproduire en réalité virtuelle, grâce à des manettes à 6 degrés de liberté (en position et rotation), par exemple avec la technique de main virtuelle [10]. Cette technique fait correspondre directement la position et l'orientation de la manette à celles de l'objet manipulé. Cependant, comme pour la sélection, cette technique ne permet pas la sélection d'objets hors de portée. D'autres techniques comme Go-Go [77] ou Depth Ray [33] permettent en partie de résoudre ces problèmes, en permettant à l'utilisateur de déplacer des objets hors de portée. Une autre approche est d'utiliser des gants de réalité virtuelle [38] ou un dispositif de capture de mouvement des mains [91]. Ces périphériques d'entrée proposent une correspondance directe entre la posture des mains de l'utilisateur et celles de son avatar. Les interactions qu'elles permettent sont donc encore plus proches des interactions physiques, comparées aux techniques utilisant une manette à 6 degrés de liberté [2]. D'autres techniques d'interaction permettent un contrôle de la position et de l'orientation d'un objet 3D avec le clavier et la souris, comme celles utilisées dans les logiciels de modélisation 3D tels que Blender.

Pour le redimensionnement, une correspondance exacte avec une transformation physique équivalente n'est pas évidente [10, 58], puisqu'elle nécessite des périphériques d'entrée spécialisés, comme une manette télescopique ou un élastique dont on peut mesurer la longueur. Formellement, c'est une distance entre deux points contrôlés par l'utilisateur, que le système fait correspondre à la taille de l'objet [102]. Cette technique a été popularisée par l'interaction sur écrans tactiles, et est aussi utilisée en réalité virtuelle : par exemple, la métaphore de la barre de traction de Song *et al.* [88] permet, en plus de

manipuler la position et la rotation de l'objet sélectionné, de contrôler sa taille, en se basant sur la distance entre les deux manettes.

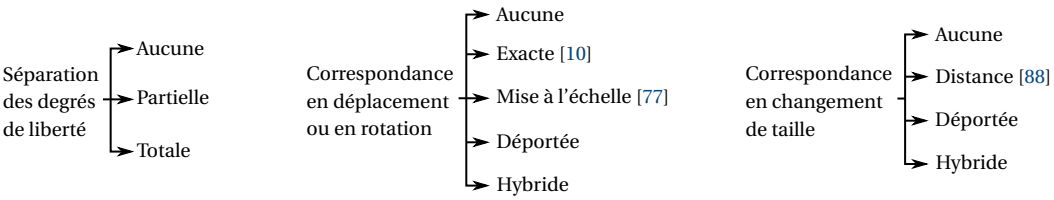


FIGURE I.4 – Taxonomie adaptée de Mendes *et al.* [58] des différentes approches pour la correspondance entre les entrées utilisateur et la transformation spatiale d'un objet 3D.

Mendes *et al.* [58] ont proposé une taxonomie, illustrée sur la figure I.4 qui décrit différentes approches pour faire correspondre les degrés de liberté d'entrée aux propriétés de transformation spatiale d'un objet 3D. Tout d'abord, ils distinguent les différents niveaux de séparation des degrés de liberté. Par exemple, la main virtuelle [10] contrôle les 3 axes de position et de rotation de manière non-séparée, c'est-à-dire de manière simultanée. Au contraire, un contrôle de ces valeurs, avec des potentiomètres par exemple, représente une séparation complète. La séparation peut être partielle, dans le cas où seulement un sous-ensemble des axes peut être manipulé en même temps. Ensuite, ils décrivent différentes catégories d'approches pour la correspondance entre une entrée utilisateur et une des composantes de déplacement, rotation et changement de taille. Pour le déplacement et la rotation, Mendes *et al.* proposent la correspondance exacte ou mise à l'échelle. Les techniques utilisant ces types de correspondance prennent directement une mesure réelle (par exemple la position d'une manette) pour la transposer à l'objet virtuel. Contrairement à la correspondance exacte, celle mise à l'échelle applique un coefficient ou une fonction de transfert à la valeur mesurée (par exemple la technique Go-Go [77]). Pour redimensionner un objet, une correspondance exacte ou mise à l'échelle est moins évidente. À la place, leur taxonomie considère la mesure d'une distance comme entrée pour contrôler la taille d'un objet. La taxonomie considère aussi la correspondance déportée, lorsque le contrôle passe par un degré de liberté sans lien avec la propriété contrôlée. Par exemple, la position de la manette pourrait être reliée à la taille à la taille de l'objet virtuel sur les trois axes. Cette correspondance est généralement utilisée sur les interfaces de bureau. Dans Blender par exemple, des widgets permettent d'utiliser la souris (position 2D) pour contrôler la position, rotation ou taille d'un élément 3D. Enfin, la correspondance hybride combine différents types de correspondances, comme par exemple une correspondance exacte sur certains degrés de liberté et une correspondance déportée pour les autres.

Bowman *et al.* [10] ont proposé d'autres méthodes de classification des techniques de manipulation. Les techniques peuvent être classifiées selon leur isomorphisme, c'est-à-dire quand les manipulations virtuelles correspondent aux manipulations que l'utilisateur effectue réellement, ou non. Par exemple, la main virtuelle est une technique isomorphique, alors que l'utilisation d'un rayon virtuel ne l'est pas. Il est aussi possible de classer les techniques en utilisant la décomposition en sous-tâches. Ceci permet de décrire une technique en fonction des tâches atomiques qui la compose. Par exemple

pour une technique de sélection, Bowman *et al.* proposent les sous-tâches « Indication de l'objet », « Confirmation de la sélection » et « Retour d'information ». Enfin, nous pouvons classer les techniques selon la métaphore utilisée. Bowman *et al.* utilisent cette méthode pour leur classification [10]. Ils évoquent notamment la métaphore de la prise en main, comme avec la main virtuelle [10] ou la Go-Go [77] ; la métaphore du pointage avec, entre autres, la technique *Raycasting* ; ou la métaphore bi-manuelle avec, par exemple, la barre de traction de Song *et al.* [88].

Toutes ces manières différentes de classifier les techniques offrent différents points de vue sur les techniques, selon différents critères. De ce fait, elles ont chacun leurs avantages et offrent ensemble une vision globale de l'espace de conception des techniques.

Nous venons de voir les techniques permettant d'interagir avec l'environnement virtuel, c'est-à-dire la sélection et la manipulation d'objets. Dans la section suivante, nous explorons les interactions permettant à l'utilisateur de se déplacer dans l'espace virtuel.

I.1.3 Se déplacer pour explorer l'espace virtuel

Se déplacer dans un espace virtuel est une tâche fondamentale, tout comme se déplacer dans la réalité. Nous décrivons le déplacement en réalité virtuelle comme le fait de modifier le point de vue de l'utilisateur dans la scène virtuelle. Bowman *et al.* [10] proposent différents critères de classification des techniques de déplacement dans les environnements 3D. Tout d'abord, une technique peut être active ou passive. La technique est active si c'est l'utilisateur qui contrôle de déplacement du point de vue, et passive si c'est le système qui le contrôle. Aussi, une technique de déplacement peut être physique ou virtuelle. La technique est physique si les déplacements correspondent directement ou indirectement aux déplacements de la tête de l'utilisateur. Autrement, elle est virtuelle.

La technique utilisant une correspondance exacte entre les déplacements physiques et virtuels de l'utilisateur semble être le plus écologique, puisque l'utilisateur se déplace comme il le ferait dans la réalité. Avec un suivi précis et à faible latence de la position et de la rotation du casque de RV, cette technique permet de réduire au maximum l'effet de cinétose, ou mal des transports. En effet, cette technique maintient la cohérence entre la perception visuelle de l'utilisateur et la perception vestibulaire (l'oreille interne). Cependant, cette technique est limitée par espace disponible dans le monde physique et d'éventuelles contraintes techniques. Plusieurs solutions consistent à introduire un décalage entre les déplacements physiques et virtuels. C'est le cas par exemple de la marche redirigée [78] qui introduit un décalage entre l'orientation réelle et virtuelle de l'utilisateur, de manière imperceptible. Ceci lui permet de se déplacer plus loin dans la scène virtuelle pour un même déplacement physique. Liu *et al.* [51] proposent une technique équivalente, la rotation se faisant cette fois lorsque l'utilisateur effectue une téléportation dans la scène virtuelle. Cependant, introduire un décalage entre les déplacements physiques et virtuels peut poser problème si plusieurs personnes utilisent le même espace de suivi. Dans ce cas, la position d'une personne par rapport à l'autre dans l'espace virtuel est alors décalée par rapport à la réalité.

Une autre solution, très utilisée dans les jeux en réalité virtuelle, utilise un stick analogique ou des boutons directionnels pour déplacer le point de vue de l'utilisateur. Par exemple, orienter le stick vers le haut le fera se déplacer vers l'avant. Un tapis de course spécialisé, comme le KAT Walk C [44], permet de garder la marche comme moyen de contrôle du déplacement, mais sans être limité par l'espace réel. L'utilisateur marche dans la direction qu'il souhaite, ce qui provoque un déplacement dans l'espace virtuel, alors que le corps de l'utilisateur est maintenu au centre d'un tapis roulant. Les déplacements passifs, c'est-à-dire contrôlés par le système, libèrent l'utilisateur du contrôle de son déplacement. Ceci lui permet de se concentrer sur une tâche plus importante de l'application virtuelle, comme un jeu de tir immersif. L'avantage de ces techniques est que l'utilisateur n'a plus besoin de se déplacer réellement, ce qui évite le problème d'espace réel limité. Cependant, ceci introduit un décalage entre la perception visuelle liée aux déplacements dans la scène virtuelle, et l'oreille interne qui ne perçoit aucun mouvement ou des mouvements incohérents.

D'autres techniques permettent de réduire l'effet de la cinétose tout en gardant la possibilité de se déplacer sans limites physiques. Il est possible d'incorporer dans le champ de vision de l'utilisateur des éléments visuels dont la position reste fixe par rapport à l'espace réel, indépendamment de la scène virtuelle [18]. Par exemple, des études ont essayé d'ajouter des nuages [50], ou un motif de grille dans la scène VR [24], ce qui réduit significativement la sensation de cinétose. Une autre approche consiste à se téléporter à une destination dans l'espace virtuel, que l'utilisateur choisit avec une technique de sélection. Le déplacement entre le point de départ et le point d'arrivée étant instantané, les symptômes de cinétose sont limités.

Nous remarquons que les techniques doivent souvent trouver un compromis entre, d'une part le réalisme des déplacements, c'est-à-dire la correspondance exacte entre les mouvements de la tête de l'utilisateur et les mouvements de la caméra virtuelle, et d'autre part les limites physiques de l'espace de suivi. En effet, il n'est pas toujours possible d'utiliser un espace de suivi assez grand pour l'environnement virtuel immersif, ni de réduire la taille de l'environnement virtuel pour l'adapter à l'espace réel. Dans ce cas, il est nécessaire d'utiliser des techniques alternatives qui, comme nous l'avons vu ci-dessus, augmentent souvent l'effet de cinétose et réduisent l'effet de réalisme de l'expérience immersive.

Dans la section suivante, nous explorons les interactions permettant de communiquer avec les autres utilisateurs de l'espace virtuel.

I.2 Interagir avec les autres, au travers de son avatar

Les applications de réalité virtuelle multi-utilisateurs permettent à plusieurs personnes de se rencontrer dans des environnements immersifs. Dans ce contexte, le principal moyen de communication est d'utiliser sa voix, avec un microphone.

Dans les échanges en face-à-face, la voix est souvent accompagnée par la communication non-verbale, qui joue un rôle essentiel [96]. Par exemple, différentes expressions

faciales ou gestuelles apportent une signification plus précise de ce qu'on dit à l'oral. En réalité virtuelle, la communication non-verbale est plus engageante et permet une meilleure présence sociale [31]. Elle nécessite que l'utilisateur incarne un avatar qu'il pourra contrôler, et qui est capable de reproduire de telles expressions.

Nous divisons le contrôle d'avatar en trois catégories, représentant différents niveaux de précisions ou parties du corps à contrôler : le contrôle de la posture générale de l'avatar ; le contrôle précis de la posture des doigts ; et le contrôle de l'expression faciale. Ces catégories de contrôle permettent chacune d'exprimer différentes choses de manière non-verbale. Pour le corps, la posture permet par exemple de saluer à distance 🙋, hausser les épaules 🤷 ou danser 💃. Les doigts des mains permettent entre autres, de valider 👍 ou invalider 👎 une proposition, pointer une direction 👉 ou compter avec ses doigts. Enfin, le visage permet par exemple d'exprimer une grande variété d'émotions, comme la joie 😂, la tristesse 😢 ou la colère 😠.

Cette section détaille les techniques permettant le contrôle de son avatar, principalement en environnement immersif. Nous verrons que les techniques pour le contrôle de la posture de l'avatar ou le contrôle des mains ont été beaucoup étudiées. Elles couvrent des techniques avec une correspondance directe entre la posture de l'utilisateur et celui de son avatar, jusqu'à un contrôle indirect via une entrée utilisateur, en passant par une correspondance partielle telle que la cinématique inverse. Cependant pour le contrôle d'expressions faciales, la littérature se concentre plus sur le contrôle isomorphique. Nous élargirons alors notre exploration hors de la réalité virtuelle immersive, pour nous inspirer de techniques d'interaction dans des environnements immersifs.

I.2.1 La posture de l'avatar

La capture de mouvement permet une correspondance exacte entre le corps de l'utilisateur et celui de l'avatar. Elle fournit un contrôle complet de la posture, la maîtrise du contrôle n'étant limitée que par les capacités de mouvement de l'utilisateur. Une première approche utilise des marqueurs positionnés sur l'ensemble du corps ainsi que la présence de caméras calibrées et des logiciels dédiés. Cette technique est couramment utilisée pour l'enregistrement de jeu d'acteur pour les films d'animation ou les jeux vidéo. Une autre solution consiste à utiliser un exosquelette [55]. Bien qu'il ne nécessite pas la présence d'appareil de suivi externe, l'exosquelette reste une solution intrusive pour l'utilisateur.

Pour une installation moins encombrante, il est possible d'utiliser des caméras de profondeur, comme par exemple la Kinect [60] qui est un appareil grand public. Ce type de caméra est capable de reconnaître les formes du corps de l'utilisateur et d'en extraire la posture [52]. La Kinect reste malheureusement un appareil supplémentaire à installer dans la pièce. Ahuja *et al.* [1] proposent une technique pouvant fonctionner en réalité virtuelle sur smartphone [30]. Elle exploite la caméra arrière du téléphone, en face duquel se trouvent deux miroirs sphériques permettant d'obtenir une vue stéréoscopique du corps de l'utilisateur. Le logiciel intégré au smartphone déduit de ces reflets la posture de l'utilisateur, qui peut être transposée à son avatar.

Il est aussi possible d'utiliser un nombre plus limité de marqueurs sur le corps de l'avatar. Les casques VR grand public étant généralement fournis avec deux manettes, le système a donc accès à la position et la rotation de la tête et des deux mains. Dans ces conditions, le système n'a pas connaissance de la posture du corps entier de l'utilisateur. Deux solutions sont alors envisagées par les concepteurs d'applications. Une première solution est d'enlever de l'avatar les éléments qui ne sont pas suivis. Par exemple, l'avatar incarné dans Rec Room [82] n'est composé que de la tête, du buste et des deux mains. La tête et les mains de l'avatar suivent la position de la tête et des mains de l'utilisateur, et la position et la rotation du buste sont déduites de celles de la tête. Une autre solution est d'utiliser la cinématique inverse [70, 83] : le système déduit la posture d'une partie ou de l'intégralité de l'avatar, en se basant uniquement sur les entrées fournies par l'équipement de réalité virtuelle. L'ajout de marqueurs supplémentaires, comme le Vive Tracker [42], réduit le risque d'erreurs dues à la cinématique inverse.

Dans le cas où il ne serait pas possible de suivre les mouvements effectués par l'utilisateur, celui-ci peut contrôler son avatar en utilisant des périphériques d'entrée tels que des manettes, un clavier ou une souris. En dehors de la réalité virtuelle, les entrées utilisateurs vont souvent permettre les déplacements de l'avatar (voir section I.1.3) ou de le faire interagir avec l'environnement. L'avatar est alors animé automatiquement en fonction de mouvements pré-enregistrés qui sont joués selon les commandes de l'utilisateur.

I.2.2 Le contrôle des doigts

Plusieurs techniques existent pour transposer exactement la posture des mains de l'utilisateur à celles de son avatar. La capture de mouvement est possible en utilisant des techniques similaires à la capture pour le corps. Par exemple, il est possible d'utiliser des marqueurs sur les doigts et des caméras externes pour les localiser [53]. Une autre solution est d'utiliser des capteurs de courbure pour déterminer le pliage des doigts [20, 54]. D'autres gants utilisent des exosquelettes [38, 34] permettant, en plus de la captation de la posture, de fournir un retour d'effort pour augmenter l'immersion. Une solution moins intrusive consiste à utiliser des caméras de profondeur, comme le Leap Motion [91], qui est comparable à une Kinect, mais conçu spécialement pour les mains. L'Oculus Quest intègre nativement le suivi des mouvements des doigts grâce aux caméras intégrées au casque [59]. Cependant, ces deux solutions utilisant uniquement des caméras, elles ne fournissent pas de retour haptique ni de retour d'effort.

Les manettes de réalité virtuelle grand public incluent des capteurs permettant d'obtenir partiellement la posture des doigts. La manette HTC Vive (voir annexe A.1) ne possédant aucun capteur pour les doigts, la position des doigts ne peut être déduite que des boutons que l'utilisateur actionne. Toucher la surface tactile implique que le pouce est abaissé, appuyer plus ou moins sur la gâchette implique que l'index est plus ou moins fermé, et le serrage de la manette permet de déduire l'état des trois derniers doigts (ils sont soit tous ouverts, soit tous fermés). Les manettes du Valve Index (voir annexe A.2)) utilisent le même principe pour le pouce et l'index. Cependant, ils sont équi-

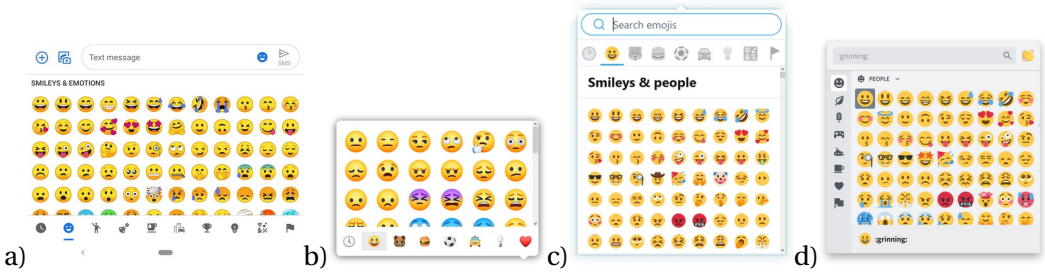
pés de capteurs capacitifs permettant de déterminer l'amplitude d'ouverture des trois derniers doigts.

I.2.3 Le contrôle de l'expression faciale

Nous commençons par un aperçu des techniques d'interaction liées à la sélection des expressions faciales dans des contextes non RV, avant de détailler les travaux antérieurs sur les techniques de contrôle isomorphiques et non isomorphiques dans la réalité virtuelle.

I.2.3.1 Le cas de la messagerie instantanée et des réseaux sociaux

En dehors de la réalité virtuelle, sur les réseaux sociaux et les messageries instantanées, les échanges entre les utilisateurs reposent principalement sur du texte. Malheureusement, ceci ne permet pas de communiquer les subtilités qui sont permises par la communication non-verbale. Pour outrepasser ce problème, le standard Unicode intègre des émojis, c'est-à-dire des petites images, intégrées comme des caractères, dans les échanges textuels. Le standard actuel (Émoji v13) comporte 3 292 émojis répartis en neuf catégories : *Smileys et émotions*, *Personnes, gestes et corps*, *Animaux et plantes*, *Nourriture et boissons*, *Voyage et lieux*, *Activités*, *Objets*, *Symboles* et *Drapeaux* [92]. La catégorie qui nous intéresse, *Smileys et émotions*, contient des expressions faciales ainsi que d'autres émojis en rapport avec le visage.



a) b) c) d)

FIGURE I.5 – Menu de sélection d'émoji de diverses applications : a) Google Messages (Application Android) ; b) Facebook Messenger (sur navigateur) ; c) Twitter ; d) Discord.

Les utilisateurs sélectionnent généralement les émojis dans des grilles (figure I.5), mais des alternatives existent dans la littérature. Alvina *et al.* [3] ont enrichi un clavier de smartphone de gestes symboliques permettant de sélectionner des émojis sans avoir à utiliser un menu spécifique. Avec cette technique, l'utilisateur est également en mesure de contrôler la taille de l'émoji ou l'intensité de l'expression. Pohl *et al.* [76] ont proposé un nouveau menu d'émoji pour les smartphones. L'idée est que l'utilisateur voit un panorama de tous les émojis disponibles, et doit taper sur une zone spécifique pour zoomer dessus. L'utilisateur peut ensuite taper sur l'émoji souhaité. Par rapport aux menus d'émoji courants sur les smartphones, il n'est pas nécessaire de faire défiler une liste d'émojis.

Compte tenu de la large utilisation des émojis dans la messagerie mobile et de bureau, ils peuvent être exploités pour concevoir des techniques d'interaction permettant

de contrôler les expressions faciales dans la RV, afin de favoriser un transfert d'apprentissage d'un contexte à l'autre. En outre, les gens sont de plus en plus habitués à utiliser les émojis à des fins autres que les émotions, ce qui est à prévoir également dans le cadre de la RV et doit être pris en compte lors de la conception des techniques d'interaction.

I.2.3.2 Contrôle isomorphique de l'expression faciale

Nous parlons de contrôle du visage *isomorphique* lorsque le système détecte le comportement des utilisateurs pour faire correspondre l'expression de leur visage à celui de leur avatar.

Le contrôle isomorphique des expressions faciales repose principalement sur l'utilisation de caméras. Le système capte les expressions sur le visage de l'utilisateur pour les transposer sur l'avatar qu'il incarne. La capture de mouvement est un exemple de technologie permettant, entre autres, la capture d'expressions faciales. Facebo [52] est une technologie combinant la Kinect pour la capture de la posture du corps et un Primesense pour la capture du visage. Leur prototype fournit un module Unity avec un avatar générique, qui combine les données de suivi du visage et du corps de l'utilisateur. Weise *et al.* [99] proposent d'utiliser directement un Kinect pour la capture de l'expression faciale.

Ces techniques ne sont cependant pas utilisables en réalité virtuelle, le casque occultant la visibilité du visage pour les capteurs. Pour contourner ce problème, d'autres solutions consistent à installer des capteurs à l'intérieur du casque de RV. Suzuki *et al.* [90] utilisent des capteurs optiques, et Li *et al.* [48] proposent d'attacher une caméra RGB-D (qui capte aussi la profondeur) au HMD et filmant la partie basse du visage, ainsi que des jauges de déformation pour détecter les mouvements du visage au niveau de la mousse du casque. Le problème est que ces techniques nécessitent une phase de calibration avant utilisation, étant donné le placement du casque de RV qui peut varier d'une utilisation à l'autre, ainsi que la morphologie du visage qui est différente pour chaque personne.

Une autre solution, utilisée dans Facebook Spaces, consiste à analyser la voix de l'utilisateur à l'aide d'un microphone et à en déduire l'expression du visage [27]. Cette méthode pose des problèmes de confidentialité, car les utilisateurs n'ont aucune garantie sur ce qui est enregistré et sur la façon dont c'est interprété. De plus, cette technique est limitée aux expressions audibles (rire fort, par exemple). Enfin, en raison des performances des algorithmes de reconnaissance, les utilisateurs peuvent être amenés à exagérer les expressions.

Des applications mobiles utilisent la même approche. Les Animoji [4] exploitent les capteurs Face ID pour suivre le visage de l'utilisateur en temps réel et le transposer sur l'avatar affiché à l'écran. D'autres applications comme Snapchat ou Instagram proposent un suivi du visage basé sur des filtres pour augmenter l'expression du visage de l'utilisateur avec des décorations. Ces techniques permettent un contrôle manuel de l'appa-

rence de l'avatar, mais visent à fournir une correspondance isomorphique entre le visage des utilisateurs et l'orientation ainsi que l'expression du visage de leur avatar.

En premier lieu, le problème du contrôle isomorphique est qu'il suppose que les utilisateurs souhaitent transposer leur expression faciale réelle à leur avatar dans l'environnement virtuel. Or, les utilisateurs souhaitent parfois afficher une expression différente, ou avec une intensité différente. Les utilisateurs peuvent également vouloir exprimer des expressions irréalistes telles que des yeux en forme de cœur ou de dollar, comme le montre la figure I.6. Ceci est partiellement possible grâce aux applications mobiles décrites ci-dessus, mais le choix du filtre ou de l'expression doit tout de même être fait manuellement par l'utilisateur. Enfin, il faut aussi prendre en compte les utilisateurs qui ne veulent pas, voire ne peuvent pas réaliser l'expression souhaitée avec leur propre visage.

Pris ensemble, ces inconvénients minimisent les avantages de la détection automatique dans le cas général. Le contrôle non isomorphique des expressions du visage est une alternative pour surmonter ces problèmes.

FIGURE I.6 – Exemples d'expressions faciales qui existent sous forme d'émojis, mais qui ne peuvent pas être exprimées par un contrôle isomorphique[1]. De gauche à droite : *baver sur de l'argent, yeux en cœur, tête qui explose, sourire avec lunettes de soleil, visage avec masque médical, visage qui a froid* et *visage qui a chaud.*

I.2.4 Contrôle non isomorphique de l'expression faciale

Nous parlons de contrôle *non isomorphique* du visage lorsque les utilisateurs choisissent manuellement l'expression du visage de leur avatar en utilisant des techniques d'interaction. Cela nécessite des techniques d'interaction adéquates, car il s'agit généralement d'une tâche secondaire.

Les interactions proposées peuvent permettre différents niveaux de précision pour le contrôle du visage, qui sont décrits dans cette sous-section : le contrôle individuel des degrés de liberté du visage d'un avatar, jusqu'à la sélection d'une émotion.

I.2.4.1 Le contrôle précis des degrés de liberté du visage

Les expressions faciales impliquent des mouvements de nombreux muscles. Pour réduire le nombre de degrés de liberté, le Facial Action Coding System (FACS) [25] décrit 24 unités d'action (AU), représentant des mouvements atomiques du visage, et la norme MPEG-4 définit 68 paramètres d'animation faciale (FAP) [69]. Même avec cette simplification des degrés de liberté, contrôler manuellement 24 ou 68 degrés de liberté intégrés en temps réel ne semble pas possible et une simplification supplémentaire est nécessaire.

1. Apparence des émojis utilisés dans cette thèse : *twemoji* par Twitter, CC-BY. https://twemoji.twitter.com/

I.2.4.2 Le contrôle par la sélection d'une émotion

Au lieu d'essayer de contrôler individuellement chaque degré de liberté d'un visage, une approche consiste à laisser l'utilisateur choisir l'expression du visage. Les expressions faciales servent, entre autres, à exprimer des émotions [71]. De ce fait, il est possible d'utiliser une représentation des émotions comme intermédiaire pour déterminer l'expression faciale souhaitée. Ceci permettrait à l'utilisateur de ne pas avoir à contrôler le grand nombre de degrés de liberté d'un visage. Plutchik [74] a déterminé un espace de conception pour les émotions, basé sur 8 émotions de base (joie, confiance, peur, surprise, tristesse, dégoût, colère, appréhension). Plusieurs de ces émotions peuvent être combinées, avec des intensités différentes, pour former des émotions supplémentaires. Par exemple, l'intensité de la joie va de la sérénité à l'extase. Mehrabian et Russel [57] placent les émotions dans l'espace PAD (*pleasure-arousal-dominance*).

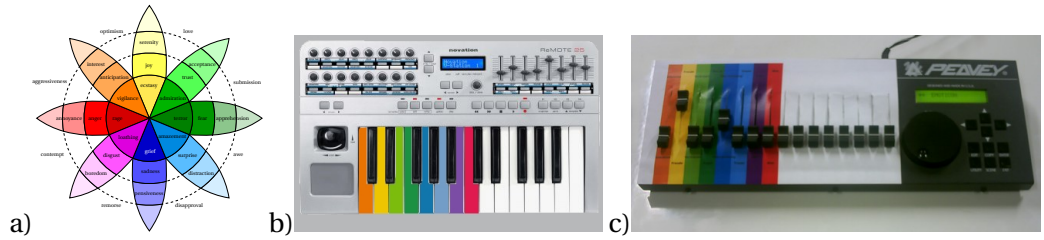


FIGURE I.7 – Les 3 interfaces testées par Bittorf *et al.* [7] : a) une interface contrôlée à la souris, représentant la roue de Plutchik ; b) un clavier MIDI et c) un mixeur qui permettent de contrôler les 8 émotions de base de Plutchik [75]. Illustrations b) et c) tirées de [7].

Plusieurs interfaces ont été proposées pour des applications de bureau afin de permettre aux utilisateurs de sélectionner une émotion et de laisser le système traduire cette émotion en une expression faciale. Par exemple, *EmoCoW* [87] est une interface graphique qui présente une roue chromatique émotionnelle (configurable en amont) pour ajuster l'expression faciale en temps réel. Cette interface permet aux animateurs de films d'animer un visage en temps réel à l'aide d'un équipement peu coûteux. Cependant, le contrôle en temps réel est limité à un petit ensemble prédéfini d'émotions. Ainsi, le contrôle de l'expression du visage en utilisant toutes les émotions possibles n'est pas possible en temps réel. Les travaux de Bittorf *et al.* [7] (figure I.7) proposent 3 interfaces permettant à l'utilisateur de choisir une émotion dans la représentation des émotions de Plutchik : une interface 2D, utilisant une représentation graphique du modèle de Plutchik [75], ainsi que deux interfaces tangibles tels qu'un mixeur audio et un clavier MIDI. Sur ces deux dernières interfaces, chaque émotion de base de Plutchik est mise en correspondance avec une entrée continue (un potentiomètre du mixeur, ou une touche du clavier). Les utilisateurs peuvent également combiner les émotions de base en activant plusieurs entrées en même temps. Alors qu'ils ont un contrôle fin de l'émotion qu'ils expriment, les techniques basées sur le clavier leur demandent d'apprendre la correspondance des entrées et des émotions. En conséquence, les auteurs signalent une charge cognitive élevée. Cela rend leurs techniques moins adaptées à une tâche secondaire et inappropriées à la RV, car les dispositifs d'entrée ne sont pas adaptés.

Les concepteurs d'applications peuvent également souhaiter compléter les expressions faciales liées aux émotions par des expressions abstraites, telles qu'illustrées figure I.6. Cependant, cela n'est pas possible avec seulement un modèle d'émotion. Pour surmonter cette limitation, une alternative consiste à sélectionner une expression faciale parmi un ensemble prédéfini d'expressions faciales. Ces expressions faciales peuvent représenter des émotions ou des expressions abstraites.

I.2.4.3 Interfaces pour la sélection d'expression faciale

Rec Room, une application sociale de RV, utilise cette approche pour permettre aux utilisateurs d'afficher une expression faciale sur leur avatar [82] (figure I.8). L'utilisateur ouvre un *marking menu* [46] avec le bouton *menu* de la manette. Le menu principal contient 2 sous-menus proposant chacun 3 expressions faciales. Le premier sous-menu contient des expressions positives (*rire* 🤣, *yeux en cœur* 😍 et *tirer la langue* 😛), et le second des expressions négatives (*pleurer* 😢, *en colère* 😠 et *visage neutre* 😐). Une fois activée, l'expression reste sur le visage de l'avatar entre 2 et 3 secondes. Par rapport à un menu d'émoji, le choix est limité, et l'utilisateur ne peut pas contrôler d'autres paramètres tels que la durée ou l'intensité de l'expression. Un système de détection de gestes (comme dans Mojiboard [3]) permettrait à l'utilisateur de préciser l'intensité de l'émotion, et donc de l'expression faciale.

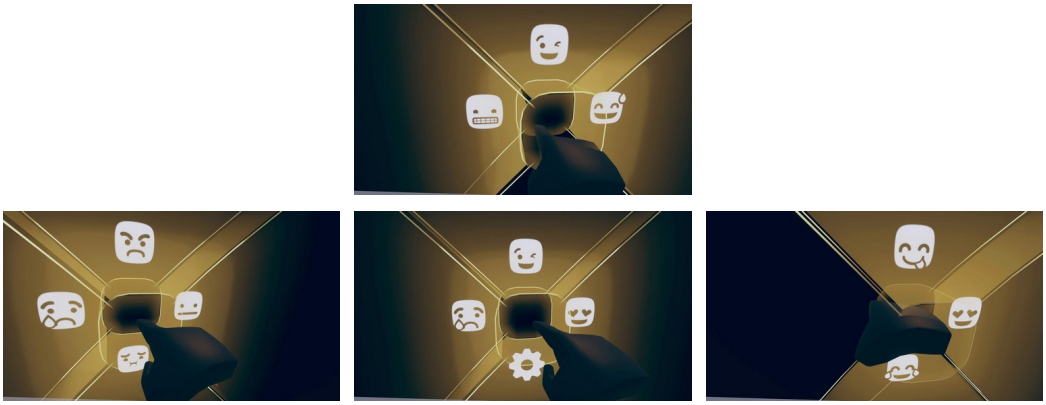


FIGURE I.8 – Marking menu utilisé dans l'application de RV *Rec Room*, pour le contrôle de l'expression faciale. Au centre, le menu principal, avec 3 boutons menant à des sous-menus pour les expressions faciales. En haut, à gauche et à droite, l'illustration de ces sous-menus. [2]

I.2.5 Espace de conception des techniques de contrôle d'expression faciale

Une technique de sélection d'expression faciale permet à l'utilisateur de contrôler le visage de son avatar à différents niveaux de contrôle. Nous proposons un espace de conception pour les techniques d'interaction permettant de contrôler les expressions faciales. Cet espace de conception vise deux finalités distinctes. D'une part, il permet de décrire et classifier les techniques existantes dans l'état de l'art. Aussi, il aide à concevoir de nouvelles techniques grâce à sa capacité générative, que nous exploitons dans

2. Images CC-BY-SA. https://rec-room.fandom.com/wiki/Basic_Controls

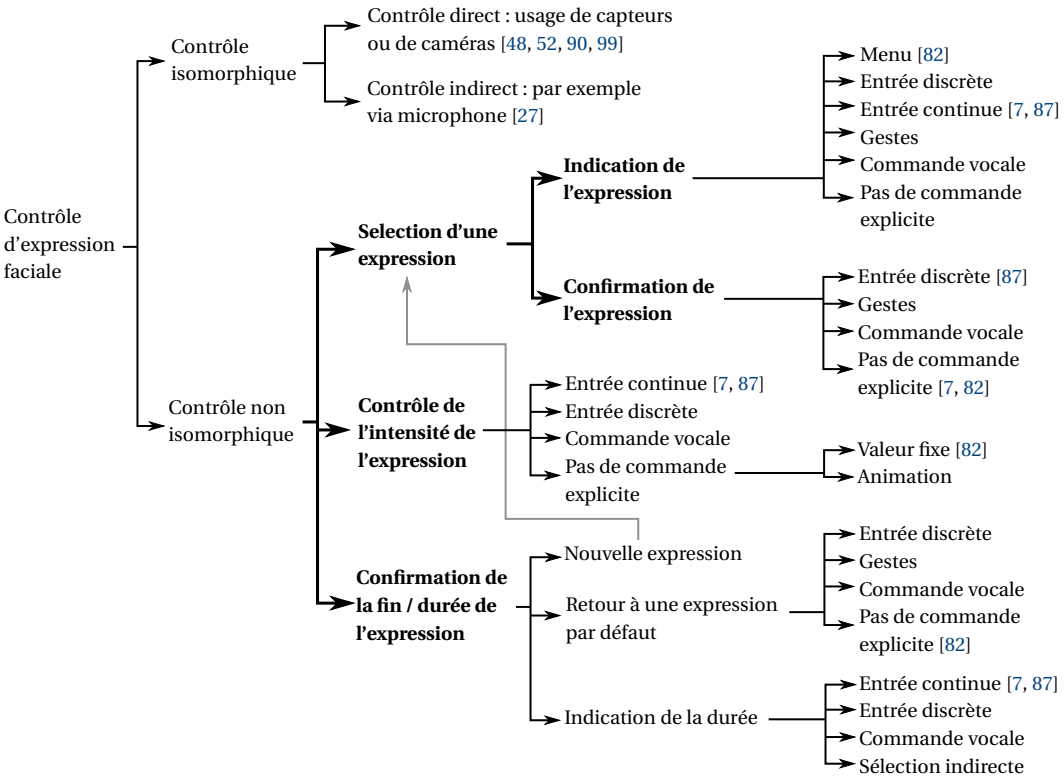


FIGURE I.9 – Classification des techniques de contrôle d'expression faciale d'un avatar. Les éléments en **gras** représentent une décomposition en sous-tâches.

le chapitre III. Nous divisons les techniques en fonction de la nature isomorphique du contrôle.

I.2.5.1 Contrôle isomorphique

Dans la littérature, nous avons distingué deux types de contrôle isomorphique de l'expression faciale. D'une part, le contrôle direct, qui correspond au fait de capter en temps réel la forme du visage de l'utilisateur pour la transposer directement sur l'avatar. Ces techniques utilisent généralement des capteurs ou des caméras, couplés à des algorithmes qui transforment les données des capteurs en un ensemble de valeurs décrivant le niveau de déformation des différentes parties du visage [48, 52, 90, 99]. D'autre part, le contrôle indirect essaye de déduire l'expression faciale de l'utilisateur via des entrées qui ne permettent pas directement de reconnaître l'expression faciale [27].

I.2.5.2 Contrôle non isomorphique

L'espace de conception pour le contrôle non isomorphique est basé sur la décomposition de tâche, similaire à la taxonomie des techniques de sélection et de déplacement de Bowman *et al.* [10]. Nous décomposons la tâche de contrôle de l'expression faciale en sous-tâches qui peuvent être considérées comme indépendantes. Ces sous-tâches suivent un ordre chronologique qui consiste d'abord à sélectionner une expression, puis à contrôler son intensité et enfin à contrôler sa durée ou sa fin (figure I.9). Cette approche présente l'avantage de décrire des techniques existantes et de créer de nouvelles techniques, en utilisant des blocs de construction qui peuvent être combinés.

Sélection d'une expression La sélection d'une expression est subdivisée de nouveau en 2 sous-tâches : l'indication de l'expression et sa confirmation. La sélection indirecte peut être utilisée pour l'indication d'une expression, avec l'aide d'un marking menu comme dans Rec Room [82], mais d'autres moyens peuvent être conçus comme de la sélection vocale ou des gestes. Il est aussi possible que le système définisse automatiquement une expression en fonction de certains événements (par exemple, un visage heureux lorsqu'on reçoit un cadeau), ce qui peut retirer la nécessité d'une commande manuelle. Une expression peut également être indiquée en utilisant une sélection directe avec une entrée discrète, par exemple en appuyant sur un bouton physique comme dans EmoCoW [87].

Une fois l'expression indiquée, elle peut être confirmée par une entrée discrète, par exemple en appuyant sur un bouton physique. D'autres moyens peuvent être conçus, comme l'utilisation de gestes ou d'une commande vocale. Aucune commande explicite n'est nécessaire si la confirmation peut être omise ou si un temps d'attente est utilisé, par exemple.

Contrôle de l'intensité de l'expression L'intensité d'une expression faciale peut être contrôlée à l'aide d'une entrée continue. Par exemple, avec EmotiCon [7], l'utilisateur contrôle l'intensité de l'expression en ajustant la pression sur les touches d'un clavier MIDI ou en ajustant les faders d'une table de mixage. D'autres types de dispositifs isotoniques, élastiques ou isométriques pourraient être utilisés ou d'autres entrées continues telles que la hauteur de la voix. On peut aussi utiliser une entrée discrète, comme appuyer plusieurs fois sur un bouton ou utiliser un temps d'arrêt. Il est aussi possible de ne pas laisser le contrôle à l'utilisateur, si celle-ci est gardée constante ou automatiquement ajustée dans le temps par une animation. Certaines expressions, comme celles illustrées figure I.6, peuvent ne pas être adaptées à un contrôle d'intensité. Dans ce cas, le contrôle de l'intensité peut être désactivé pour l'expression particulière ou être limité à un certain aspect de l'expression.

Détermination de la durée ou de la fin de l'expression Cette sous-tâche consiste à déterminer la durée ou la fin de l'expression du visage. Une première approche est de terminer l'expression courante en sélectionnant une nouvelle expression. Une alternative est de revenir à une expression par défaut, comme un visage neutre, en utilisant une entrée discrète comme un bouton ou un geste. Par ailleurs, aucune commande explicite n'est nécessaire si la fin est déterminée par une certaine durée par défaut (Entre 1 et 2 secondes dans Rec Room [82]), lorsqu'elle est déclenchée par l'environnement, ou si le fait de définir une intensité nulle correspond à la fin d'une expression. Au lieu de déterminer la fin d'une expression, l'utilisateur peut également déterminer sa durée en utilisant une entrée continue ou discrète, voire une commande vocale ou une sélection indirecte.

I.3 Conclusion

Nous avons exploré les différentes catégories d'interaction en réalité virtuelle : les interactions avec l'environnement telles que le pointage et la manipulation d'objets 3D ; les déplacements dans l'espace virtuel ; et les interactions avec les autres utilisateurs via son avatar. Nous avons élaboré un espace de conception pour le contrôle de l'expression faciale de son avatar. Celui-ci détaille les types d'interaction pour le contrôle isomorphique, et les sous-tâches d'interaction pour le contrôle non isomorphique. Aussi, nous avons identifié deux pistes qui seront détaillées dans les chapitres suivants.

Les techniques de sélection basées sur la main virtuelle et sur *Raycasting* présentent des limitations. La main virtuelle ne permet pas de sélectionner des objets hors de portée. *Raycasting* règle ce problème, mais la précision de sélection est limitée à mesure que l'objet à sélectionner est éloigné et petit. De plus, *Raycasting* ne permet de sélectionner que l'objet le plus proche de l'utilisateur, intercepté par le rayon, ce qui pose un problème d'occultation. Dans le chapitre II, nous présentons une amélioration de *Raycasting*, que nous appelons *RayCursor*, et qui permet de surmonter les limitations des techniques évoquées.

Les techniques de la littérature pour contrôler l'expression faciale de son avatar présentent aussi des limites. La plupart de ces techniques sont isomorphiques. Elles sont donc limitées par la précision des capteurs et par le fait que l'utilisateur ne puisse pas réaliser certaines expressions, ou ne veuille pas contrôler l'expression faciale de son avatar avec son propre visage. Dans le chapitre III, nous présentons la conception de techniques non isomorphiques, qui couvrent les différentes sous-tâches de l'espace de conception, dont *EmoRayE* résultant de plusieurs expériences contrôlées.

Chapitre II

RayCursor, une technique de sélection d'objets 3D

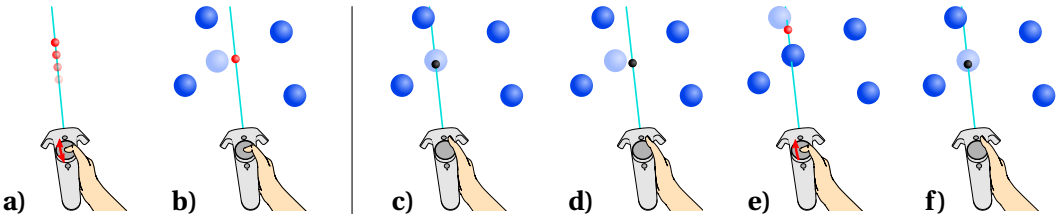


FIGURE II.1 – Illustration du *RayCursor* manuel : a) l'utilisateur contrôle un curseur le long du rayon grâce au déplacement relatif de son pouce sur le pavé tactile ; b) la cible la plus proche du curseur est mise en surbrillance. Illustration du *RayCursor* semi-auto : c) par défaut, il s'utilise comme *Raycasting*. Le curseur (en noir) se positionne automatiquement à l'intersection du rayon avec une cible ; d) la cible reste mise en surbrillance même lorsque le curseur s'éloigne, jusqu'à ce qu'il soit plus proche d'une autre cible ; e) l'utilisateur peut reprendre le contrôle du curseur en utilisant le pavé tactile, pour sélectionner une cible occultée (le curseur devient rouge pour indiquer le mode manuel) ; f) si l'utilisateur n'utilise plus le pavé tactile pendant 1s, le curseur retrouve le comportement décrit en c).

Publications

Marc Baloup, Veïs Oudjail, Thomas Pietrzak, and Géry Casiez.
Techniques de Pointage pour Cibles Distantes en Réalité Virtuelle.
Dans *Proceedings of the AFIHM Conférence Francophone sur l'interaction Humain-Machine*, IHM 2018, octobre 2018.

Marc Baloup, Thomas Pietrzak, and Géry Casiez.
RayCursor: A 3D Pointing Facilitation Technique Based on Raycasting.
Dans *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, mai 2019.

Marc Baloup, Thomas Pietrzak, and Géry Casiez.
Amélioration du Raycasting par utilisation de la sélection par proximité et du filtrage.
Dans *Journal d'Interaction Personne-Système*, décembre 2019. Special issue : the best of IHM'2018.

Nous l'avons montré dans le chapitre état de l'art, le pointage est une tâche fondamentale dans tout système interactif comportant un espace d'interaction 2D ou 3D. Malgré de nombreuses techniques destinées à améliorer *Raycasting* [45, 33, 89, 81, 14, 79, 49, 86, 21] (voir [5] pour un aperçu plus exhaustif), le *Raycasting* standard et la main virtuelle restent les deux techniques par défaut disponibles avec des systèmes tels que l'environnement Steam VR pour Unity.

Les techniques de mains virtuelles offrent un isomorphisme entre la main réelle et la main virtuelle [5]. De ce fait, elles ne permettent pas la sélection d'objets hors de portée de la main sans déplacement de l'utilisateur. La technique *Raycasting* permet de contourner ce problème, car son rayon permet de pointer des objets éloignés. Lorsque le rayon intersecte plusieurs objets, celui qui est le plus proche de l'utilisateur peut être sélectionné. Cette technique permet de sélectionner des cibles avec une difficulté croissante lorsque les cibles sont plus éloignées ou plus petites, en raison des limites du suivi du mouvement et des capacités motrices humaines. Cette technique est également affectée par l'occultation et les cibles distractives, car seule la cible la plus proche peut être sélectionnée. Ainsi, pour sélectionner des cibles cachées, il faut changer la position et l'orientation du rayon.

De nombreuses techniques ont déjà été conçues et améliorées pour surmonter ces limites. De nouveaux périphériques d'entrée et de nouveaux contextes d'interaction offrent de nouvelles possibilités d'améliorer les techniques de pointage. Les nouveaux contrôleurs, disponibles sur le HTC Vive, offrent des degrés de liberté supplémentaires tels que des pavés tactiles qui n'étaient pas disponibles auparavant.

Nous proposons une technique de *Raycasting* améliorée, appelée *RayCursor*[1], qui utilise un curseur sur le rayon. Cette technique a été conçue principalement pour les environnements immersifs utilisant le HTC Vive, car elle utilise un touchpad. Elle pourrait également être utilisée dans d'autres contextes 3D comportant un contrôleur 6 degrés de liberté avec un pavé tactile et des boutons. L'utilisateur peut contrôler le curseur le long du rayon en utilisant les déplacements relatifs de son pouce sur le pavé tactile. Comme pour le *Bubble Cursor* [32, 94], la cible la plus proche du curseur peut être sélectionnée lorsque l'utilisateur appuie sur un bouton.

Après avoir décrit la conception de *RayCursor* et ses caractéristiques (retour visuel, fonction de transfert du curseur et filtrage du rayon), nous expliquons la mise en place générale de nos études expérimentales, et une série d'expériences examinant chacune des caractéristiques de la technique. Enfin, nous détaillons une expérience utilisateur, comparant notre technique au *Raycasting* standard et la technique de la littérature la plus proche [81].

1. Ressources additionnelles disponibles à l'adresse ns.inria.fr/loki/raycursor

II.1 RayCursor

Nous décrivons la conception de *RayCursor*, une amélioration de *Raycasting*. Tout d'abord, nous ajoutons un curseur sur le rayon, que l'utilisateur peut manipuler. Ensuite, nous ajoutons une stratégie consistant à sélectionner la cible la plus proche du curseur, de la même manière que Bubble Cursor [32, 94]. Nous évoquons différentes variantes de retours visuels, inspirés des travaux de Guillon *et al.* pour la 2D [35]. Ensuite, nous décrivons différentes fonctions de transfert possibles pour le contrôle du curseur. Enfin, nous détaillons les techniques de filtrage que nous avons utilisées pour réduire les effets des tremblements du rayon, en utilisant le *1€ Filter* [17].

II.1.1 Ajout d'un curseur sur le rayon

L'idée d'un curseur contrôlable a été introduite par Ro *et al.* pour sélectionner des cibles dans des environnements de réalité augmentée en utilisant un téléphone mobile [81]. Cependant, cette technique n'a pas été conçue pour faciliter le choix de petites cibles. Au lieu d'utiliser un téléphone portable, l'utilisateur de *RayCursor* effectue des déplacements vers l'avant ou vers l'arrière sur le pavé tactile d'un contrôleur Vive situé sous son pouce (voir l'annexe A pour une description de la manette HTC Vive). Il serait possible d'implémenter cette technique sur n'importe quel autre contrôleur à 6 ddl ayant au moins un degré de liberté supplémentaire tel qu'une molette.

II.1.2 Fonction de transfert pour le contrôle du curseur

La fonction de transfert, qui calcule les déplacements du curseur, est un aspect essentiel de la technique d'interaction. En effet, des recherches antérieures ont montré que la fonction de transfert influence les performances de pointage [15]. Nous considérons deux variables pour la conception de la fonction de transfert :

— v_{pad} la vitesse du point de contact sur le pavé tactile en m/s. Il s'agit d'une variable habituelle pour des fonctions de transfert non linéaires.

— d_{cur} la distance entre la manette et le curseur en m. Il s'agit de la distance dans l'espace virtuel, c'est-à-dire la distance entre la représentation virtuelle de la manette et le curseur. Nous utilisons cette variable car la vitesse du curseur projetée à l'écran parait d'autant plus importante que celui-ci est proche de l'utilisateur. Nous émettons l'hypothèse que des fonctions de transfert exploitant d_{cur} influencent le temps de sélection.

La vitesse du curseur est donc $v_{cur} = g(v_{pad}, d_{cur}) \times v_{pad}$. Nous proposons plusieurs fonctions de transfert, que nous évaluons dans le cadre d'une expérience dédiée.

II.1.2.1 Fonction de transfert dépendant de la vitesse du doigt

Les fonctions de transfert non linéaires usuelles adaptent la vitesse du curseur à celle du périphérique d'entrée [15], de manière à tirer parti des phases balistique et corrective d'une tâche de pointage. Pendant la phase balistique, l'utilisateur cherche à se déplacer

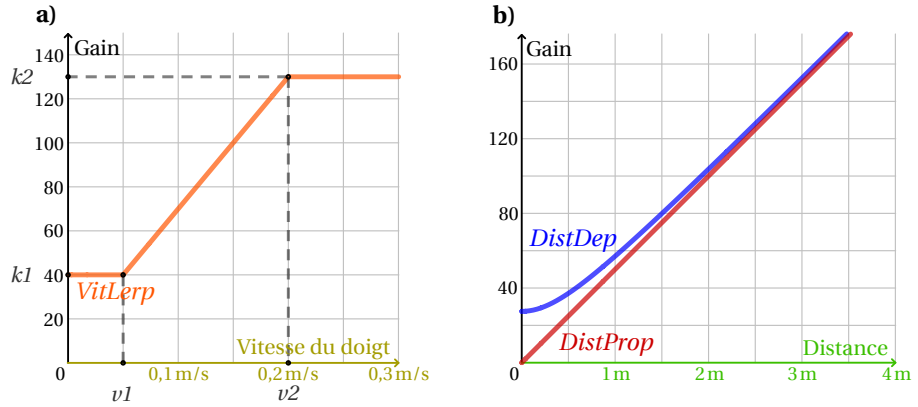


FIGURE II.2 – Courbes des fonctions de gain en fonction de **a)** la vitesse du doigt sur le pad et de **b)** la distance manette-curseur.

aussi vite que possible pour se rapprocher de la cible alors que dans la phase corrective, l'utilisateur a généralement besoin d'être précis pour sélectionner la cible. Pour ce faire, les fonctions de transfert non linéaires utilisent un gain plus faible à basse vitesse et un gain plus élevé à haute vitesse. Nous avons conçu cette fonction de transfert comme une interpolation linéaire bornée (voir figure II.2 **b)**) :

$$VitLerp(v_{pad}) = \begin{cases} k_1 & \text{si } x \leq v_1 \\ k_2 & \text{si } x \geq v_2 \\ k_1 + \frac{k_2 - k_1}{v_2 - v_1}(v_{pad} - v_1) & \text{sinon} \end{cases}$$

II.1.2.2 Fonction de transfert dépendant de la position du curseur

Avec une fonction de transfert qui utilise un gain constant, la vitesse du curseur affiché à l'écran semble dépendre de sa distance à l'utilisateur dans l'espace virtuel : plus rapide quand il est proche, et plus lente quand il est éloigné. Nous avons d'abord conçu une fonction de transfert basée sur un gain qui est proportionnel à la distance du curseur par rapport au contrôleur dans l'espace virtuel, pour atténuer ce problème (voir *DistProp* dans la figure II.2 (a)). Bien que cette fonction permette de rendre visuellement cohérente la vitesse de déplacement du curseur à des positions moyennes et éloignées, elle provoque des déplacements trop lents lorsque le curseur est très proche du contrôleur. Nous avons donc développé une deuxième version de cette fonction appelée *DistDep* (voir figure II.2 (a)), qui garantit un gain minimum pour des distances plus proches. La formule de cette fonction est la suivante, avec k un facteur de proportionnalité, d_{cur} la distance entre le contrôleur et le curseur, et d une constante correspondant à une distance estimée entre l'œil de l'utilisateur et la manette :

$$DistDep(d_{cur}) = k \times \sqrt{d_{cur}^2 + d^2}$$

II.1.3 Retours visuels

Les travaux de Guillon *et al.* ont montré une influence du retour visuel rétroactif sur l'efficacité du pointage avec la technique *Bubble Cursor* pour le pointage 2D [35]. L'étude de Vanacken *et al.* sur le 3D Bubble ne prenait en compte que le retour visuel de la bulle [94]. Nous adaptons à la 3D les retours visuels conçus par Guillon *et al.* pour l'interaction 2D. Nous décrivons ci-dessous ces retours visuels selon deux dimensions : la mise en évidence de la cible et la représentation de la distance entre le curseur et la cible la plus proche.

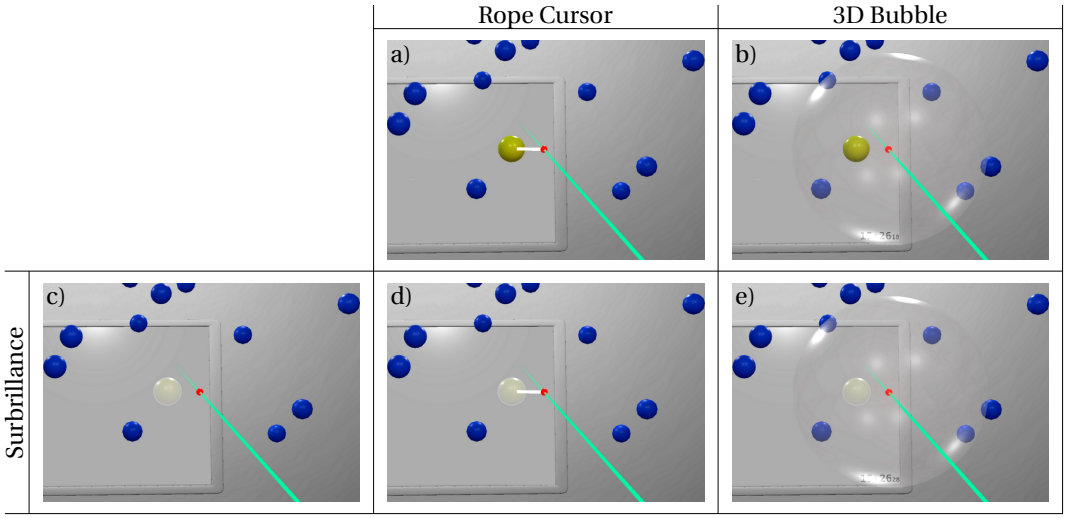


FIGURE II.3 – *RayCursor* avec différents retours visuels. Le curseur est rouge et le rayon est cyan. **(c,d,e)** *Surbrillance* : la cible la plus proche est mise en surbrillance. **(a,d)** *Rope Cursor* [35] : une ligne blanche relie le curseur à la cible la plus proche. **(b,e)** *3D Bubble* [94] : une bulle 3D centrée sur le curseur, adapte son rayon pour englober la cible la plus proche.

II.1.3.1 Surbrillance de la cible

Dans notre implémentation, la mise en évidence de la cible la plus proche consiste à appliquer une couleur plus claire que les autres cibles (figure II.3 (c,d,e)). L'avantage de ce retour visuel par rapport aux suivants est que l'encombrement visuel est minimal. Nous pouvons également le combiner avec d'autres retours visuels.

II.1.3.2 Représentation de la distance *cible-curseur*

Représenter la distance entre le curseur et la cible la plus proche est susceptible d'aider à déterminer la cible qui peut être sélectionnée à l'aide du curseur. Il existe plusieurs manières de représenter cette distance. Dessiner une sphère semi-transparente, centrée sur le curseur et dont le rayon est la distance *cible-curseur*, est une adaptation 3D du *Bubble Cursor* (figure II.3 (b,e)). Guillon *et al.* ont également proposé le *Rope Cursor*, que nous adaptons à la 3D en affichant un segment blanc entre le curseur et la cible la plus proche (figure II.3 (a,d)). Ce retour visuel provoque moins d'encombrement visuel que la bulle. Une autre proposition de Guillon *et al.* est la région de Voronoï de chaque cible. Ces régions représentent les zones d'influence de chaque cible, en prenant en compte les distances entre elles. Bien que nous ayons implémenté cette rétroaction visuelle sous forme de volumes semi-transparents, nous l'avons rejetée parce que l'encombrement visuel

créé la rend difficile à utiliser (figure II.4) et les performances de l'application sont fortement impactées (baisse significative du nombre d'images par seconde) lorsque toutes les régions sont affichées.

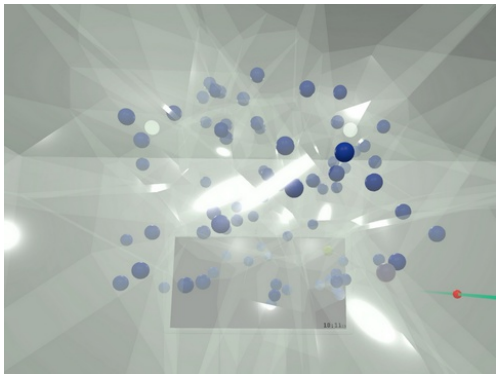
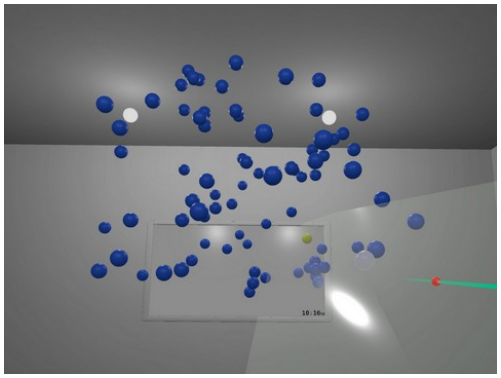
a)  b) 

FIGURE II.4 – *RayCursor* avec le retour visuel représentant les régions de Voronoï : **a)** affichage simultané de toutes les régions de Voronoï ; **b)** affichage de la région de Voronoï de la cible la plus proche.

Dans la section II.3.1, nous présentons une étude comparative entre les retours visuels suivants ainsi que des combinaisons entre eux (figure II.3) : *3D Bubble*, *Rope Cursor* et *Surbrillance*.

II.1.4 Filtrage du rayon

La précision de la sélection des petites cibles est un problème courant avec *Raycasting*. Elle est due à la fois à des tremblements de la main et à des entrées bruitées. Bowman *et al.* [11] décrivent aussi un autre problème, qu'ils ont nommé « l'effet Heisenberg de l'interaction spatiale », qui correspond au décalage accidentel du rayon lors de l'action de sélection (appui sur un bouton). Nous proposons de réduire ces problèmes en filtrant le rayon. Nous appliquons le filtrage, à la fois pour *Raycasting* et notre technique, en utilisant le *1€ Filter* car il est rapide, simple à régler, et offre un bon compromis entre réduction du bruit et introduction de latence. Nous avons conçu deux modes de filtrage. Dans le premier mode, nous filtrons l'orientation du rayon mais uniquement pour calculer l'intersection avec des objets virtuels. Le rayon affiché n'est pas filtré. Nous appelons ce mode $1€_{M}$, car le rayon est seulement filtré dans l'espace moteur. Dans le second mode, nous filtrons l'orientation du rayon dans les espaces moteur et visuel ($1€_{VM}$). L'avantage de $1€_{M}$ par rapport à $1€_{VM}$ est que l'utilisateur ne perçoit visuellement aucun temps de latence supplémentaire mais bénéficie quand même du filtrage pour le point d'intersection du rayon avec des objets. La latence due au filtrage est tout de même présente dans l'espace moteur. En revanche, avec $1€_{VM}$, l'utilisateur dispose d'un retour visuel plus cohérent sur les résultats de ses actions.

Nous décrivons une étude comparative de ces deux modes de filtrage et un *Raycasting* non filtré dans la section II.3.3.

II.2 Configuration expérimentale générale

Toutes les expériences de ce chapitre se basent sur la même configuration telle que décrite dans cette section.

II.2.1 Matériel

Pour nos expériences, nous avons utilisé un ordinateur équipé d'un casque de réalité virtuelle *HTC Vive* [41]. Les participants manipulaient le contrôleur HTC Vive avec leur main dominante, et n'avaient pas le droit d'utiliser l'autre. Le rayon était contrôlé par les 6 degrés de liberté de mouvement du contrôleur, et le curseur était manipulé par le pouce sur le pavé tactile (figure II.1). La sélection était effectuée au moment de l'appui sur la gâchette du contrôleur, en utilisant l'index. L'application de l'expérience a été développée en *C#* avec *Unity 3D* et la librairie *SteamVR*.

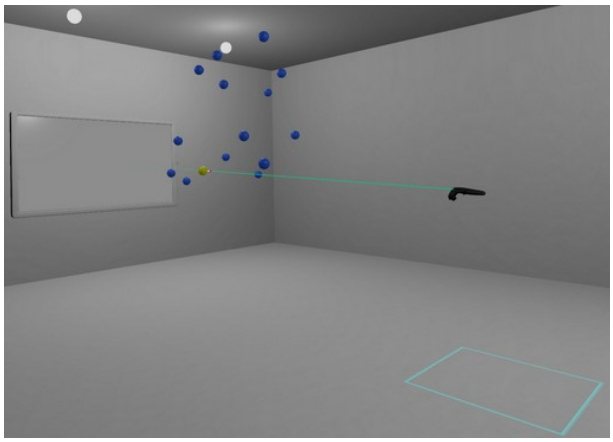


FIGURE II.5 – Scène 3D utilisée pour les expériences. Le participant était debout dans le carré cyan tracé au sol. Les instructions étaient affichées sur l'écran virtuel en face de lui. Les cibles étaient positionnées entre l'utilisateur et l'écran virtuel.

II.2.2 Tâches

Sauf indication contraire, les participants devaient se conformer aux instructions suivantes dans toutes les expériences. Ils devaient se tenir au milieu d'une pièce carrée de 7 m de côté. La position exacte était indiquée par un carré sur le sol de 70 cm de côté. Les instructions de l'expérience étaient affichées sur un écran virtuel devant l'utilisateur (figure II.5). Nous avons demandé aux participants de sélectionner des cibles à différentes positions, tailles et densités, en fonction des conditions expérimentales. Toutes les cibles étaient visibles lorsque l'utilisateur regardait le mur virtuel avec l'écran virtuel. Afin de permettre une comparaison équitable des temps d'exécution et des taux d'erreur entre les conditions, nous avons généré, avant les expériences, une séquence unique de cibles pour chaque condition, utilisée pour tous les participants, toutes les techniques et tous les blocs. L'ordre des techniques était contrebalancé entre les participants en utilisant un carré latin.

Toutes les cibles étaient bleues sauf celle qui devait être sélectionnée, qui était jaune. Le participant pouvait sélectionner une cible en appuyant sur la gâchette. Une vibration

de 10 ms du contrôleur informait le participant lorsque la bonne cible est sélectionnée. Lorsqu'un participant ne sélectionnait pas la bonne cible, le contrôleur vibrait pendant 200 ms, la cible correcte clignotait en vert, la mauvaise cible sélectionnée clignotait en rouge (sauf lorsque l'utilisateur ne sélectionnait aucune cible avec *Raycasting*), et l'essai était marqué comme une erreur. Les participants ne pouvaient pas passer à la cible suivante avant d'avoir correctement sélectionné la cible en cours. Le taux d'erreur a été calculé comme étant le rapport entre le nombre total d'essais pour lesquels la bonne cible n'a pas été sélectionnée à la première tentative et le nombre total d'essais. Nous avons demandé aux participants de maintenir un taux d'erreur d'environ 4 % afin d'équilibrer leur compromis vitesse/précision. Le taux d'erreur était affiché sur l'écran virtuel pendant les pauses.

Les participants répondaient à un questionnaire NASA-TLX pour chaque technique ou fonction testée. Nous utilisons ce questionnaire comme indicateur de la charge de travail de l'utilisateur. À la fin de l'expérience, l'utilisateur devait aussi choisir sa technique préférée.

II.3 Étude des caractéristiques de *RayCursor*

Nous présentons une série d'expériences pour évaluer chacune des caractéristiques de *RayCursor*. Nous commençons par l'étude de différents retours visuels pour indiquer la cible à sélectionner, car elle nous a semblé être la caractéristique la plus susceptible d'affecter les performances. Nous évaluons ensuite les différentes fonctions de transfert que nous avons conçues et évaluons l'effet du filtrage du rayon.

II.3.1 Retours visuels

Nous avons proposé plusieurs retours visuels, basés sur les travaux existants sur des techniques similaires d'interaction 2D et 3D [32, 35, 94]. L'étude de Guillon *et al.* sur le cas 2D a conclu que la mise en surbrillance de la cible la plus proche est le retour le plus efficace, tout en limitant l'encombrement visuel. Nous décrivons une étude similaire qui compare des versions 3D de ces retours visuels, avec *Raycasting* comme condition témoin. Étant donné les travaux existants en 2D, notre hypothèse est que la mise en surbrillance de la cible la plus proche est le retour le plus efficace en 3D (H1). Nous avons utilisé la fonction de transfert *DistDep* pour contrôler le curseur, et le rayon n'a été filtré dans aucune des conditions.

Méthodologie Douze participants (tous droitiers, age moyen = 26, σ = 4,3) ont pris part à cette expérience. Pour deux d'entre eux il s'agissait de leur première expérience de réalité virtuelle.

Nous avons utilisé un protocole expérimental intra-sujet, avec les facteurs : TECHNIQUE, DENSITÉ de cibles, TAILLE des cibles et BLOC. Les 6 techniques sont *Raycasting* (*RC*) en tant que référence, ainsi que *RayCursor* avec les 5 retours visuels décrits figure II.3 : *RopeCursor* (*Rope*), *Surbrillance* (*HL*), *3DBubble* (*Bub*), *3DBubble+Surbrillance*

(*Bub+HL*), *RopeCursor+Surbrillance* (*Rope+HL*). L'ordre des techniques a été équilibré entre les participants à l'aide d'un carré latin. Les 2 tailles de cibles étaient S_{Grande} = 8 cm et S_{Petite} = 4 cm. Les deux densités de cibles étaient de 15 cibles (D_{Faible}) et 40 cibles ($D_{Élevée}$). Toutes les cibles étaient positionnées de manière pseudo-aléatoire dans une sphère de 2 m de diamètre, dont le centre était positionné à 2 m devant le participant [2].

Les participants pouvaient prendre des pauses entre chaque bloc. Le protocole expérimental de l'expérience était donc : 12 participants × 6 TECHNIQUES × 3 BLOCS × 2 DENSITÉS × 2 TAILLES × 10 cibles = 8640 essais au total. L'expérience durait environ 30 minutes par participant.

Résultats Nos variables dépendantes sont le temps de sélection, le taux d'erreur ainsi que les réponses au questionnaire NASA-TLX.

Temps de sélection Dans cette analyse, le temps de sélection correspond au temps écoulé entre deux sélections. Par conséquent, le premier essai de chaque séquence de 10 cibles est écarté, de même que les essais entrainant une erreur. Nous avons appliqué une transformation de Box-Cox avec $\lambda = -1{,}2$ pour corriger les problèmes de normalité des résidus des temps de sélection [12].

Une ANOVA [3] révèle un effet significatif de BLOC ($F_{1,2,12,7} = 21{,}2$, $p < 0{,}001$, $\eta^2_G = 0{,}04$). Les comparaisons par paires montrent des différences significatives entre le bloc 1 et les deux suivants ($p < 0{,}003$, Bloc 1 : 1,45s, 2 : 1,29s, 3 : 1,25s). Nous supposons que cette différence est due à un effet d'apprentissage. Nous supprimons donc le premier bloc du reste de l'analyse.

Les analyses suivantes révèlent un effet significatif de TECHNIQUE ($F_{5,55} = 5{,}4$, $p < 0{,}001$, $\eta^2_G = 0{,}1$). Les comparaisons par paires montrent un effet significatif entre *Raycasting* et toutes les autres techniques sauf *3DBubble* (*RC* : 1,39s, *HL* : 1,19s, *Bub* : 1,42s, *Bub+HL* : 1,22s, *Rope* : 1,23s, *Rope+HL* : 1,17s, $p < 0{,}027$).

Les analyses montrent un effet significatif de TAILLE ($F_{1,11} = 108{,}3$, $p < 0{,}001$, $\eta^2_G = 0{,}07$) et une interaction TECHNIQUE×TAILLE ($F_{5,55} = 28{,}1$, $p < 0{,}001$, $\eta^2_G = 0{,}05$). Les comparaisons par paires ne montrent un effet significatif entre les techniques que pour les *petites* cibles. *Raycasting* est significativement plus lente ($p < 0{,}0004$) que toutes les autres techniques, à part *3DBubble* (*RC* : 1,63s, *Bub* : 1,48s, *HL* : 1,21s, *Rope* : 1,28s, *Rope+HL* : 1,20s).

Nous observons également un effet significatif de DENSITÉ ($F_{1,11} = 176{,}3$, $p < 0{,}001$, $\eta^2_G = 0{,}11$) et une interaction TECHNIQUE×DENSITÉ ($F_{5,55} = 5{,}2$, $p < 0{,}001$, $\eta^2_G = 0{,}01$). Pour la *faible* densité, nous observons une différence significative ($p < 0{,}01$) entre *Raycasting* et toutes les autres techniques, excepté *3DBubble* (*RC* : 1,36s, *Bub* : 1,30s, *Bub+HL* : 1,13s, *HL* : 1,10s, *Rope* : 1,11s, *Rope+HL* : 1,07s).

2. Les fichiers contenant la position précise des cibles, tels qu'utilisés dans nos expériences, se trouvent à l'adresse ns.inria.fr/loki/raycursor

3. Toutes les analyses statistiques des 4 expériences ont été effectuées avec R, avec $\alpha = 0{,}05$ et la correction de Greenhouse-Geisser a été appliquée aux degrés de liberté quand la sphéricité était violée. Nous avons utilisé la correction de Bonferroni, dans laquelle la p-value était multipliée par le nombre de comparaisons. Les analyses statistiques détaillées sont disponibles à l'adresse ns.inria.fr/loki/raycursor.

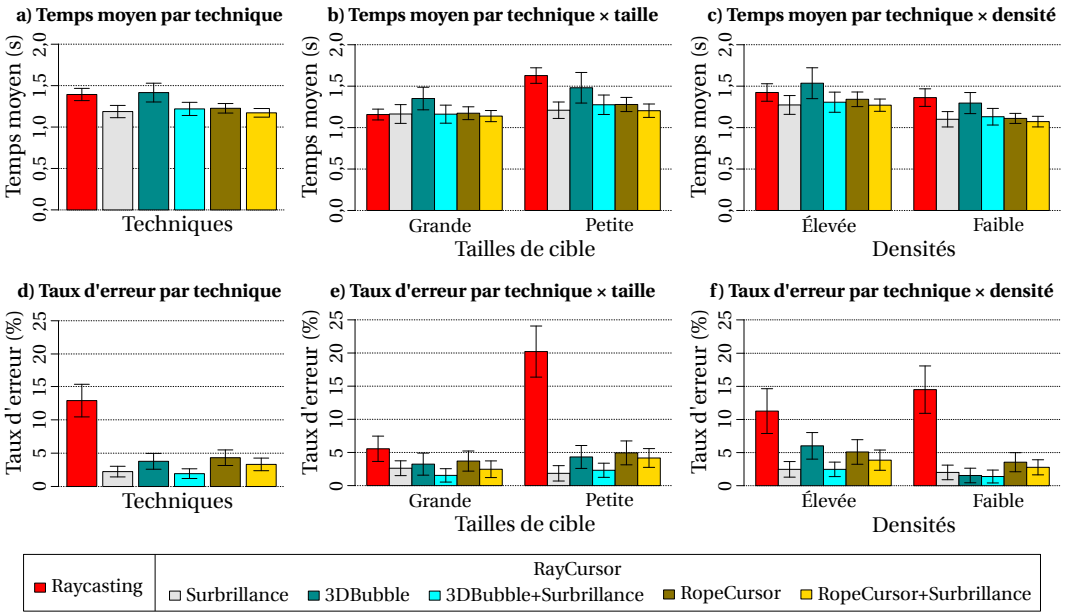


FIGURE II.6 – Temps moyen et taux d'erreur relevés lors de l'expérience sur les retours visuels, avec représentation des intervalles de confiance à 95 %.

Taux d'erreur Le taux d'erreur global est de 4,7%, sachant que les participants avaient pour instruction de rester autour de 4%. Les données ont été pré-traitées à l'aide d'une Aligned Rank Transform (ART) pour tenir compte de leur distribution non normale [100]. L'ANOVA à mesures répétées révèle un effet significatif de TECHNIQUE ($F_{5,829} = 30,3$, $p < 0,001$). Les comparaisons par paires montrent un effet significatif entre *Raycasting* (12,9%) et toutes les autres techniques. (*HL* : 2,2%, *Bub* : 3,8%, *Bub+HL* : 1,9%, *Rope* : 4,3%, *Rope+HL* : 3,3%, $p < 0,006$). Nous observons aussi un effet significatif de TAILLE ($F_{1,829} = 140,7$, $p < 0,0001$) et une interaction TECHNIQUE×TAILLE ($F_{5,829} = 26,4$, $p < 0,0001$). Les comparaisons par paires montrent que le taux d'erreur de *Raycasting* est significativement plus élevé pour des *petites* cibles (20,2%) que pour des *grandes* cibles (5,6%). Pour les *petites* cibles, *Raycasting* a un taux d'erreur plus élevé que pour toutes les autres techniques ($< 5\%$, $p < 0,0001$). Les analyses montrent aussi une interaction TECHNIQUE×DENSITÉ ($F_{5,829} = 5.8$, $p < 0.0001$). Les comparaisons par paires montrent que *Raycasting* a un taux d'erreur plus élevé pour une *faible* densité que les autres techniques ($< 3,6\%$, $p < 0,0001$).

Questionnaire NASA-TLX L'analyse de Friedman sur les réponses au questionnaire NASA-TLX indique qu'il y a un effet significatif pour l'exigence physique ($\chi^2(5) = 18,7$, $p = 0,002$), l'exigence mentale ($\chi^2(5) = 12,1$, $p = 0,03$), l'effort ($\chi^2(5) = 23,8$, $p = 0,0002$) et la frustration ($\chi^2(5) = 25,6$, $p = 0,0001$). L'analyse post-hoc de Wilcoxon révèle que le niveau de frustration est significativement plus élevé pour la technique *Raycasting* (médiane= 14) que pour la technique *Surbrillance* (médiane= 4, $p = 0,03$). Il n'y a pas d'autre différence significative observée pour les tests post-hocs.

Préférences des participants Chaque participant a pu choisir sa technique préférée parmi les 6 testées. Les résultats montrent que les participants ont préféré les techniques avec la mise en surbrillance activée (3 à 4 votes pour chaque) tandis que les autres, *Raycasting* inclus, n'ont reçu qu'un seul vote ou aucun.

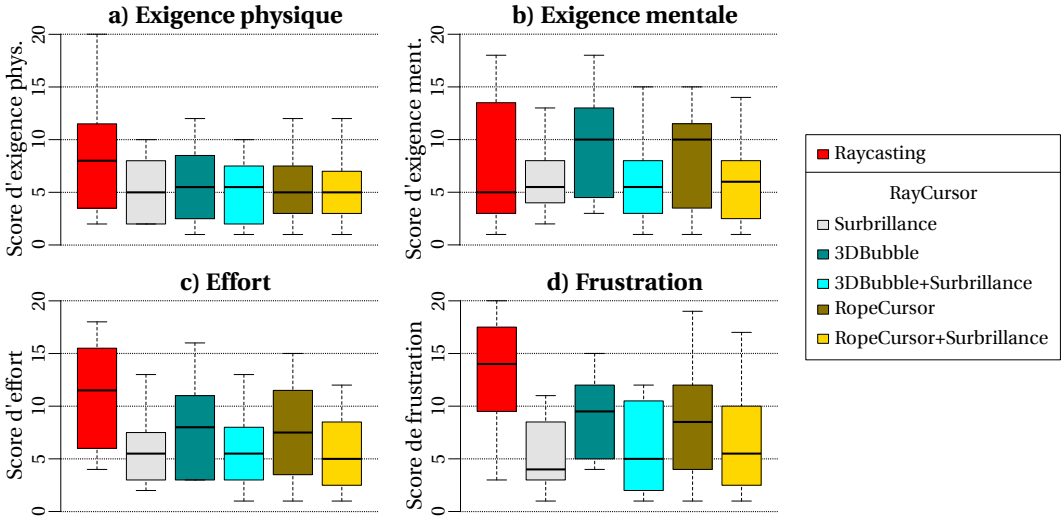


FIGURE II.7 – Diagrammes en boîte des réponses aux critères « Exigence physique », « Exigence mentale », « Effort » et « Frustration » au questionnaire NASA-TLX de la première expérience, pour chaque technique.

Discussion Les résultats montrent que tous les retours visuels utilisés avec *RayCursor* sont globalement plus efficaces que *Raycasting*. En particulier, *RayCursor* est globalement plus rapide pour les petites cibles tout en gardant un faible taux d'erreur. La mise en surbrillance de la cible la plus proche est l'un des effets visuels les plus efficaces (H1). Appliqué à *RayCursor*, il est aussi jugé moins frustrant que *Raycasting* par les utilisateurs. L'affichage d'une bulle ne permet pas d'améliorer les performances de *RayCursor* par rapport au *Raycasting*, en particulier pour les petites cibles ou les environnements denses, certainement en raison de l'encombrement visuel plus élevé qu'elle introduit. Ces résultats sont en accord avec l'étude de Guillon *et al.* concernant les retours visuels en 2D.

II.3.2 Fonction de transfert du curseur

Notre hypothèse est que le choix de la fonction de transfert utilisée influence les performances de RayCursor (H2). Pour vérifier H2, nous avons comparé la performance de *VitLerp*, *DistProp*, et *VitLerp*×*DistDep* (une combinaison entre *VitLerp* et *DistDep*) avec 9 participants (tous droitiers, age moyen=27,7, $\sigma = 6{,}5$). Les paramètres de chaque fonction de transfert ont été ajustés empiriquement pour maximiser les performances. Pour *VitLerp*, les paramètres étaient $k_1 = 30$, $k_2 = 150$, $v_1 = 0{,}05$ et $v_2 = 0{,}15$. Pour *DistProp*, le paramètre était $k = 50$. La combinaison de *VitLerp* et *DistDep* (*VLDD*) consiste à multiplier les deux gains. Pour *VLDD*, les paramètres étaient $k_1 = 20$, $k_2 = 100$, $v_1 = 0{,}05$, $v_2 = 0{,}15$, $k = 1$ et $d = 0{,}55$.

Les variables indépendantes étaient la FONCTION de transfert (*VitLerp*, *DistProp*, *VLDD*), la POSITION de la cible (*Proche*, *Loin*), la DISTANCE par rapport à la cible précédente (*Courte*, *Longue*), et le BLOC (3).

Huit cibles sous forme de cubes étaient positionnées tel que décrit figure II.8. L'utilisateur devait placer son contrôleur de manière à être aligné aux cibles, tout en restant dans le carré au sol.

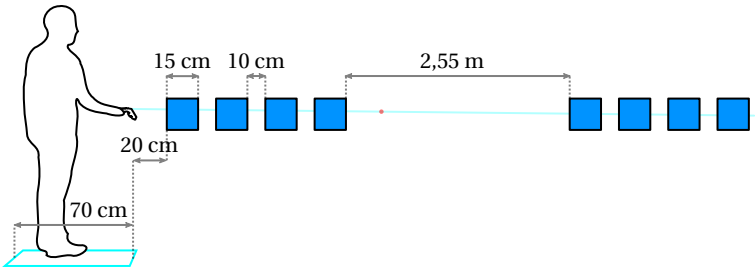


FIGURE II.8 – Positionnement des cibles dans l'espace virtuel pour l'expérience sur les fonctions de transfert.

Pour chaque technique, les participants enchaînaient 3 BLOCS, dans lesquels ils devaient effectuer 4 enchainements de 9 essais. Après chaque enchainement, les cibles disparaissaient et le taux d'erreur depuis le début de la technique courante était affiché à l'écran virtuel. L'utilisateur devait appuyer sur la gâchette pour passer à la suite.

L'ordre des cibles pour les 4 enchainements était défini de manière à ce qu'il y ait le même nombre d'essais pour chaque condition de POSITION×DISTANCE, tel que montré en exemple figure II.9. Les 4 enchainements étaient identiques pour tous les blocs, toutes les fonctions de transfert et tous les participants. L'ordre des fonctions de transfert était contrebalancé entre les participants en utilisant un carré latin.

Cible	POSITION	DISTANCE
1	N/A	N/A
4	*Proche*	*Courte*
2	*Proche*	*Courte*
6	*Éloignée*	*Longue*
3	*Proche*	*Longue*
7	*Éloignée*	*Longue*
5	*Éloignée*	*Courte*
8	*Éloignée*	*Courte*
1	*Proche*	*Longue*

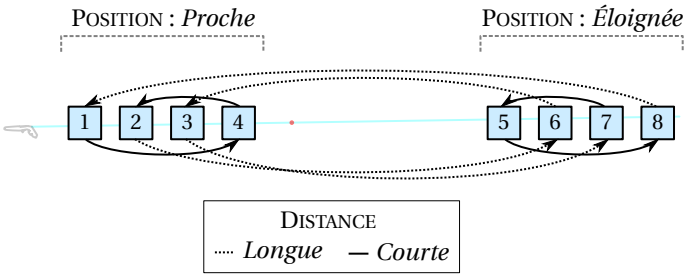


FIGURE II.9 – Exemple d'enchainement de cibles pour parcourir les niveaux des variables indépendantes POSITION et DISTANCE.

Nous avons en résumé un plan expérimental de 9 participants × 3 FONCTIONS × 3 BLOCS × 2 POSITIONS × 2 DISTANCES × 9 essais = 2 916 essais au total. L'expérience durait environ 20 minutes par participant.

Résultats Nos variables dépendantes sont le temps de sélection, le taux d'erreur, les réponses au questionnaire NASA-TLX et les préférences des participants.

Temps de sélection Nous avons utilisé une transformation Box-Cox ($\lambda = -0{,}4$) pour corriger les problèmes de normalité des résidus des temps de sélection. Une ANOVA [3] à mesures répétées montre un effet significatif de FONCTION ($F_{2,16} = 3{,}8$, $p = 0{,}046$, $\eta^2_G = 0{,}07$). Les comparaisons par paires montrent une différence significative marginale entre *DistProp* et *VLDD* (*DP* : 2,79s ; *VLDD* : 2,39s ; $p = 0{,}05$). *DistProp* est légèrement plus lente en moyenne que *VLDD*. Nous observons aussi un effet significatif de POSITION ($F_{1,8} = 223$, $p < 0{,}001$, $\eta^2_G = 0{,}55$) et une interaction de FONCTION×POSITION ($F_{2,16} = 25{,}1$, $p < 0{,}001$,

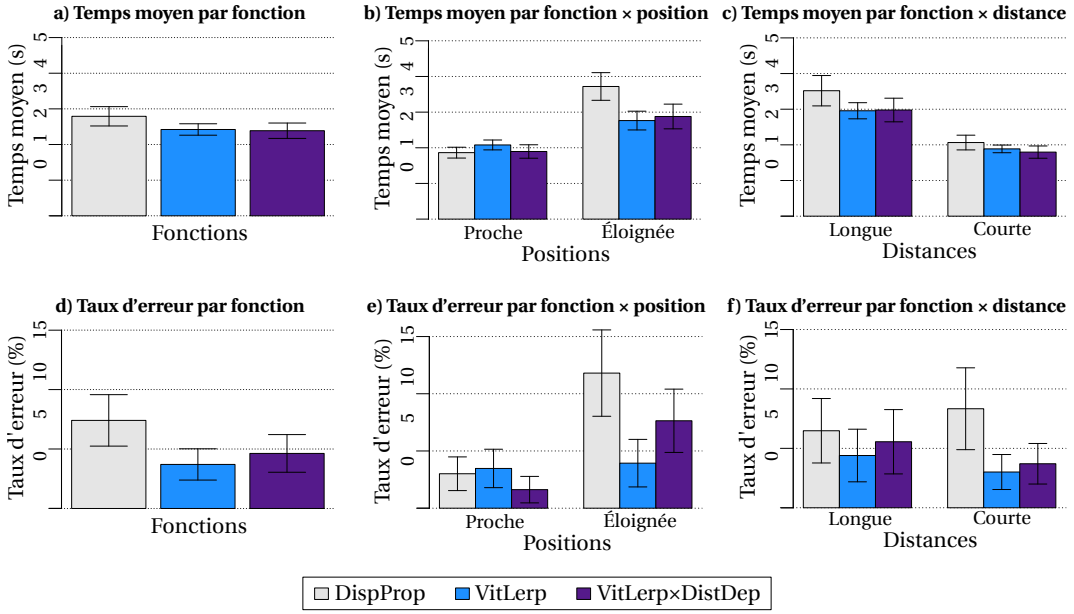


FIGURE II.10 – Temps moyen et taux d'erreur pour l'expérience sur les fonctions de transfert, avec représentation des intervalles de confiance à 95 %.

$\eta^2_G = 0{,}16$). Aucune différence significative n'a été observée pour une position *Proche*. Cependant, pour la position *Loin*, les comparaisons par paires montrent une différence significative entre *DistProp* (3,72s) et les deux autres fonctions (*VL* : 2,76s ; *VLDD* : 2,88s ; $p < 0{,}002$).

Taux d'erreur Nous avons pré-traité les données à l'aide d'une ART pour prendre en compte leur distribution non-normale. Une ANOVA à mesures répétées montre un effet significatif de FONCTION ($F_{2,304} = 4{,}55$, $p < 0{,}0004$). Les comparaisons par paires montrent un effet significatif entre *DistProp* et *VitLerp* (*DP* : 7,4% ; *VL* : 3,7% ; $p < 0{,}002$). Les participants font généralement moins d'erreurs avec *VitLerp* qu'avec *DistProp*. Nous observons aussi un effet significatif de POSITION ($F_{1,304} = 29{,}6$, $p < 0{,}001$), et une interaction de FONCTION×POSITION ($F_{2,304} = 12{,}9$, $p < 0{,}0001$). Les comparaisons par paires ne montrent pas de différence significative entre les fonctions de transfert pour des cibles *Proches*. Cependant, nous observons une différence significative pour des cibles éloignées de l'utilisateur entre *DistProp* et *VitLerp* (*DP* : 11,8% ; *VL* : 3,9% ; $p < 0{,}008$).

Questionnaire NASA-TLX L'analyse de Friedman sur les réponses au questionnaire NASA-TLX indique qu'il y a un effet significatif pour l'exigence mentale ($\chi^2(2) = 8{,}24$, $p = 0{,}01$), l'exigence temporelle ($\chi^2(2) = 7{,}93$, $p = 0{,}02$), la performance ($\chi^2(2) = 8{,}63$, $p = 0{,}01$), l'effort ($\chi^2(2) = 11{,}0$, $p = 0{,}004$) et la frustration ($\chi^2(5) = 12$, $p = 0{,}002$). L'analyse post-hoc de Wilcoxon révèle que le niveau de frustration est significativement plus élevé pour *DistProp* (médiane= 16) que pour *VitLerp×DistDep* (médiane= 11, $p = 0{,}045$). Il n'y a pas d'autre différence significative observée pour les tests post-hocs.

Préférences des participants Chaque participant a pu choisir sa technique préférée parmi les 3 testées. Les résultats montrent que les participants ont préféré les fonc-

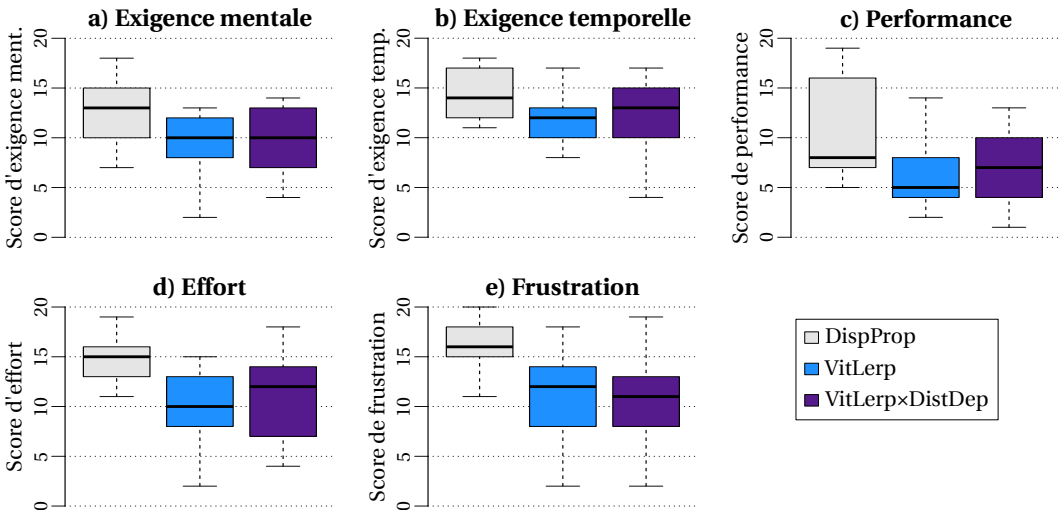


FIGURE II.11 – Diagrammes en boîte des réponses aux critères « Exigence mentale », « Exigence temporelle », « Performance », « Effort » et « Frustration » au questionnaire NASA-TLX de la dernière expérience, pour chaque fonction.

tions de gain *VitLerp* et *VitLerp×DistDep* (4 à 5 votes pour chaque) tandis que la fonction *DistProp* n'a reçu aucun vote.

Conclusion En conclusion, la fonction de transfert influence les performances de *RayCursor* (H2 confirmée) : *VitLerp* est plus rapide et plus efficace que *DistProp* pour les cibles éloignées de l'utilisateur et est globalement la fonction la plus efficace.

II.3.3 Filtrage du rayon

Dans les expériences précédentes, les petites cibles étaient difficiles à sélectionner à cause des tremblements de la main et des entrées bruitées. Dans cette expérience, nous voulons vérifier l'hypothèse que filtrer le rayon réduit les erreurs de sélection pour *Raycasting* (H3). Bien que le filtrage a déjà été utilisé dans la littérature pour filtrer un rayon [97, 45], il était utilisé dans un contexte où ce rayon intersecte un écran physique. Nous n'avons pas trouvé de travaux évaluant formellement les effets du filtrage sur le taux d'erreur et le temps de sélection de *Raycasting*. Vogel *et al.* et Kopper *et al.* ont tous les deux utilisé ce qui correspond à une version primitive du *1€ Filter* [17]. Nous l'avons aussi utilisé car il semble fournir un bon compromis entre un bon filtrage et une faible latence. Dans cette expérience, nous considérons uniquement des cibles éloignées, ce qui est la situation la plus difficile.

Méthodologie Neuf participants (tous droitiers, age moyen = 26,9, σ = 6,25) ont pris part à cette expérience. Tous avaient déjà eu une expérience en réalité virtuelle.

Nous avons utilisé un protocole expérimental intra-sujet. Les variables indépendantes sont : TECHNIQUE et BLOC. Les trois techniques sont décrites section II.1.4 : *Raycasting* non filtré (*RC*), *Raycasting* filtré dans l'espace moteur ($1€_M$) et *Raycasting* filtré dans l'espace visuel et moteur ($1€_{VM}$). L'ordre des 3 techniques a été contre-balancé entre les participants en utilisant un carré latin. Nous avons fixé les paramètres du *1€ Filter*

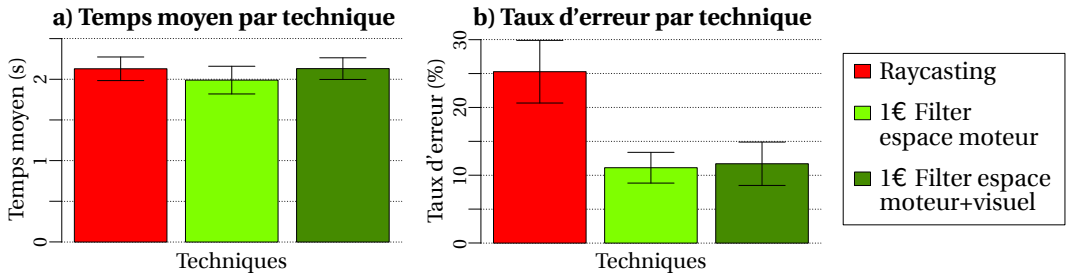


FIGURE II.12 – Temps moyen et taux d'erreur résultants de l'expérience sur le *1€ Filter*, avec représentation des intervalles de confiance à 95 %.

à $\text{min}_{\text{cutoff}} = 0{,}1$ et $\beta = 50$ suite à des évaluations empiriques, de manière à réduire les tremblements tout en minimisant la latence. La cible sélectionnable était mise en surbrillance, en accord avec l'expérience sur le retour visuel. Pour chaque technique, les participants devaient réaliser 5 BLOCS de 20 cibles.

En résumé, le protocole expérimental était : 9 participants × 3 TECHNIQUES × 5 BLOCS × 20 essais = 2700 essais au total. L'expérience durait environ 15 minutes par participant.

Résultats Nous analysons le temps de sélection et le taux d'erreur pour étudier les effets du filtrage. Il y avait une interruption entre les blocs, nous avons donc retiré le premier essai de chaque bloc de l'analyse. Pour le temps de sélection, nous avons également écarté les essais ayant donné lieu à une erreur.

Temps de sélection Nous avons appliqué une transformation de Box-Cox ($\lambda = -0{,}2$) pour corriger les problèmes de normalité des résidus des temps de sélection. Une ANOVA à mesures répétées [3] ne montre aucun effet de BLOC ($F_{4,32} = 2{,}2$, $p = 0{,}09$) ni d'effet de TECHNIQUE ($F_{1,2,9,8} = 2{,}0$, $p = 0{,}18$).

Taux d'erreur Le taux d'erreur global est de 16,02 %, alors que les participants devaient rester autour de 4 %. Nous avons appliqué une transformation logarithmique pour corriger les résidus de taux d'erreur non normaux. Les analyses ne montrent pas d'effet significatif de BLOC ($F_{4,32} = 0{,}94$, $p > 0.05$). Cependant, nous observons un effet significatif de TECHNIQUE ($F_{2,16} = 12{,}2$, $p < 0{,}001$, $\eta^2_G = 0{,}21$). Les comparaisons par paires montrent un effet significatif entre le *Raycasting* sans filtre (25,2 %) et les autres techniques ($1€_{VM}$: 11,7%, $1€_{M}$: 11,1%, $p < 0{,}0001$).

Questionnaire NASA-TLX L'analyse de Friedman sur les réponses au questionnaire NASA-TLX ne révèle pas d'effet significatif pour l'exigence physique ($\chi^2(2) = 15{,}8$, $p = 0{,}0003$), l'exigence mentale ($\chi^2(2) = 11{,}1$, $p = 0{,}003$), l'effort ($\chi^2(2) = 13{,}7$, $p = 0{,}001$) et la frustration ($\chi^2(2) = 11{,}0$, $p = 0{,}004$).

L'analyse post-hoc de Wilcoxon révèle que le niveau d'exigence physique est significativement plus élevé pour *Raycasting* (médiane= 18) que pour $1€_{VM}$ (médiane= 10; $p = 0{,}015$), et que le niveau d'effort est significativement plus élevé pour *Raycasting*

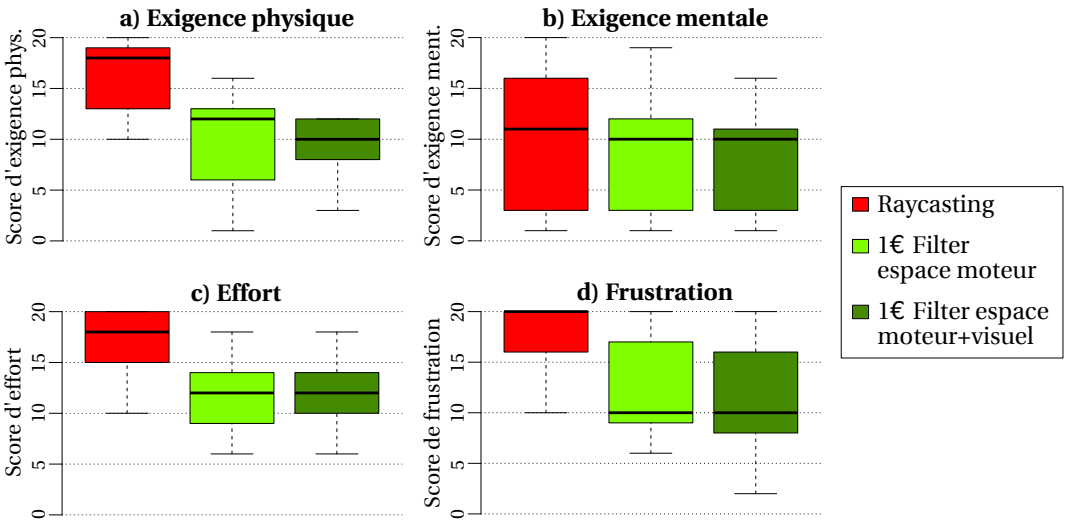


FIGURE II.13 – Diagrammes en boite des réponses aux critères « Exigence physique », « Exigence mentale », « Effort » et « Frustration » au questionnaire NASA-TLX de la deuxième expérience pour chaque technique.

(médiane=18) que pour $1€_M$ (médiane=12; $p=0{,}049$). Les tests post-hocs ne montrent pas d'autre différence significative.

Préférences des participants Chaque participant a pu choisir sa technique préférée parmi les 3 testées. Les résultats montrent que les participants ont largement préféré la technique avec le rayon filtré par *1€ Filter* dans l'espace visuel et moteur (7 votes) suivi par le rayon filtré par *1€ Filter* dans l'espace moteur seulement (2 votes) et aucun vote pour le *Raycasting*.

Discussion Les résultats montrent que filtrer le rayon avec le *1€ Filter* réduit les erreurs de sélection de plus de 50% (H3 confirmée). Lorsque le rayon est filtré dans l'espace moteur, le fait que le rayon soit aussi filtré dans l'espace visuel ou non n'influence pas le taux d'erreur. Cependant, nos discussions avec les participants révèlent que 7/9 préfèrent que l'espace visuel et l'espace moteur soient tous les deux filtrés. Qualitativement, nous avons observé que le filtrage réduit l'exigence physique, ainsi que l'effort nécessaire pour réaliser la tâche.

II.3.4 Discussion

Parmi les retours visuels que nous avons proposés pour *RayCursor*, les plus efficaces d'entre eux montrent la cible qui sera sélectionnée sans ambigüité. C'est le cas notamment pour toutes celles qui utilisent la mise en surbrillance et le Rope Cursor. Bien que plusieurs des retours visuels aient des performances similaires, nous suggérons de simplement mettre en surbrillance la cible la plus proche, puisqu'il s'agit du retour visuel avec le moins d'encombrement visuel. Ceci suggère également que la technique 3DBubble [94] bénéficierait certainement aussi de ce type de retour visuel.

Notre étude sur la fonction de transfert pour contrôler le curseur a montré qu'une fonction non linéaire de la vitesse du curseur était la plus efficace. Ceci est en accord avec

la littérature et les fonctions de transfert courantes pour les souris d'ordinateur et les pavés tactiles de bureau [15]. Nous avons proposé une fonction simplifiée, paramétrée avec deux seuils de vitesse d'entrée et deux gains extrêmes. Le développeur de l'application peut adapter ces paramètres en fonction de la taille du pavé tactile et des distances que l'utilisateur doit parcourir. Nous avons observé de manière informelle que les fréquents mouvements vers l'avant et vers l'arrière du curseur sur de longues distances ralentissent la technique. Cela nous a amené l'idée de combiner *Raycasting* et *RayCursor*, avec un mécanisme pour téléporter le curseur près d'un point d'intérêt, réduisant ainsi le temps de déplacement.

Enfin, nous avons montré que le filtrage du rayon diminue les erreurs de sélection. C'est à la fois bénéfique pour *Raycasting* et *RayCursor*. Bien que nos expériences aient montré que la présence ou non du filtrage dans l'espace visuel n'influence pas les performances lorsque le rayon était filtré dans l'espace moteur, la préférence des utilisateurs tend à suggérer qu'il est préférable de filtrer les deux espaces.

Dans ce qui suit, nous présentons une dernière expérience comparant 1) *RayCursor* utilisant la mise en surbrillance, la fonction de transfert non-linéaire qui utilise la vitesse et le rayon filtré ; 2) la même version de *RayCursor* mais utilisant un positionnement semi-automatique du curseur le long du rayon ; 3) *Raycasting* utilisant un filtrage ; et 4) une technique récente de la littérature, qui utilise un rayon de longueur ajustable [81].

II.4 Étude comparative

Nous avons évalué toutes les caractéristiques de *RayCursor* et proposé des paramètres optimaux. Dans cette section, nous comparons deux variantes de la technique avec *Raycasting* et la technique la plus proche de la nôtre dans la littérature [81]. Les deux variantes sont un contrôle manuel du curseur et un contrôle semi-automatique du curseur. La version semi-automatique (figure II.1, c-f) est un hybride entre *Raycasting* et *RayCursor*. Le contrôle automatique du curseur est activé lorsque l'utilisateur ne touche pas le pavé tactile. Dans ce mode, lorsque le rayon intersecte une cible, le curseur se déplace automatiquement sur le rayon au point d'intersection. Si le rayon sort de la cible, la cible reste sélectionnable à l'aide du mécanisme de sélection par proximité. Si une autre cible est plus proche du curseur ou intersecte le rayon, cette nouvelle cible est sélectionnée. Si l'utilisateur pose son doigt sur le pavé tactile, il peut contrôler la position du curseur comme dans la version manuelle de *RayCursor* (le contrôle automatique est désactivé). Si l'utilisateur relâche le pavé tactile plus d'une seconde, la technique revient au comportement automatique du curseur. Le délai d'une seconde laisse le temps à l'utilisateur de relever et reposer son doigt pour déplacer le curseur sur une longue distance.

Raycasting et les deux variantes de *RayCursor* sont filtrées en utilisant le *1€ Filter* avec les réglages précédemment définis. En effet, lorsque *Raycasting* n'est pas filtré, son taux d'erreur est beaucoup plus élevé que pour les autres techniques selon notre étude dans la section II.3.3. Cette expérience nous permettra de mieux comprendre les améliorations de la performance du *Raycasting* lorsqu'il est filtré. Nous avons aussi implé-

menté la technique de Ro *et al.* car c'est la technique la plus proche de la nôtre [81]. Cela nous permet également de mesurer l'avantage direct de l'utilisation du mécanisme de sélection par proximité. Comme Ro *et al.* n'ont pas détaillé la fonction de transfert utilisée pour contrôler la longueur de leur rayon, nous avons utilisé pour leur technique la même fonction de transfert utilisée pour *RayCursor*. Notre hypothèse est que *RayCursor* avec contrôle semi-automatique du curseur est plus rapide et moins sujette aux erreurs que les autres techniques testées (H4).

II.4.1 Méthodologie

Douze participants (1 gaucher, age moyen=27,6, $\sigma = 5{,}8$) ont pris part à l'expérience. Deux d'entre eux n'ont jamais utilisé de système de réalité virtuelle avant l'expérience. Six d'entre eux ont participé à une des expériences précédentes. Le temps entre chaque expérience était d'au moins 4 semaines, ce qui nous permet de supposer que l'effet d'apprentissage était négligeable. Le plan expérimental intra-sujet a par ailleurs l'avantage de réduire les différences individuelles.

Nous avons utilisé un protocole expérimental intra-sujet, avec comme facteur : TECHNIQUE, DENSITÉ de cibles, TAILLE des cibles et BLOC. Les 4 techniques utilisées sont : *Raycasting* filtré avec *1€ Filter* (valRC$_f$), *RayCursor* avec le contrôle manuel du curseur (*ManRCur*), *RayCursor* avec le contrôle semi-automatique du curseur (*AutoRCur*) et la technique de Ro *et al.* [81] (*Ro*). L'ordre des techniques a été contrebalancé entre les participants grâce à un carré latin. Les 2 tailles de cible étaient $S_{Grande} = 8cm$ et $S_{Petite} = 4cm$ de diamètre. Les 2 densités étaient de 30 cibles (D_{Faible}) et 60 cibles ($D_{Elevée}$). Les cibles étaient réparties équitablement dans 2 sphères de 60 cm de diamètre, en face de l'utilisateur, et centrées à 80 cm du sol. La première sphère était à 1 m en face de l'utilisateur et la deuxième à 4 m. Toutes les cibles étaient dans le champ de vision du participant lorsqu'il regardait en direction de l'écran virtuel. Le participant devait sélectionner alternativement une cible dans la sphère proche et une autre dans la sphère éloignée. Cette condition correspond à la situation la plus défavorable pour *ManRCur* car le participant devait constamment déplacer le curseur sur de longues distances. Ceci a pour but de mettre en avant les limites de *RayCursor*.

Les participants avaient la possibilité de faire une pause entre les techniques. Le design de l'expérience était donc : 12 participants × 4 TECHNIQUES × 3 BLOCS × 2 DENSITÉS × 2 TAILLES × 9 cibles = 5184 essais au total. L'expérience durait environ 30 min par participant.

II.4.2 Résultats

La figure II.14 montre les taux d'erreur et les temps de sélection. Nous discutons aussi des préférences des participants et des réponses au questionnaire NASA-TLX.

Temps de sélection Dans cette analyse, le temps de sélection correspond au temps entre deux sélections. De ce fait, le premier essai de chaque séquence de 10 cibles est ignoré, ainsi que les essais erronés. Nous avons aussi supprimé les valeurs extrêmes,

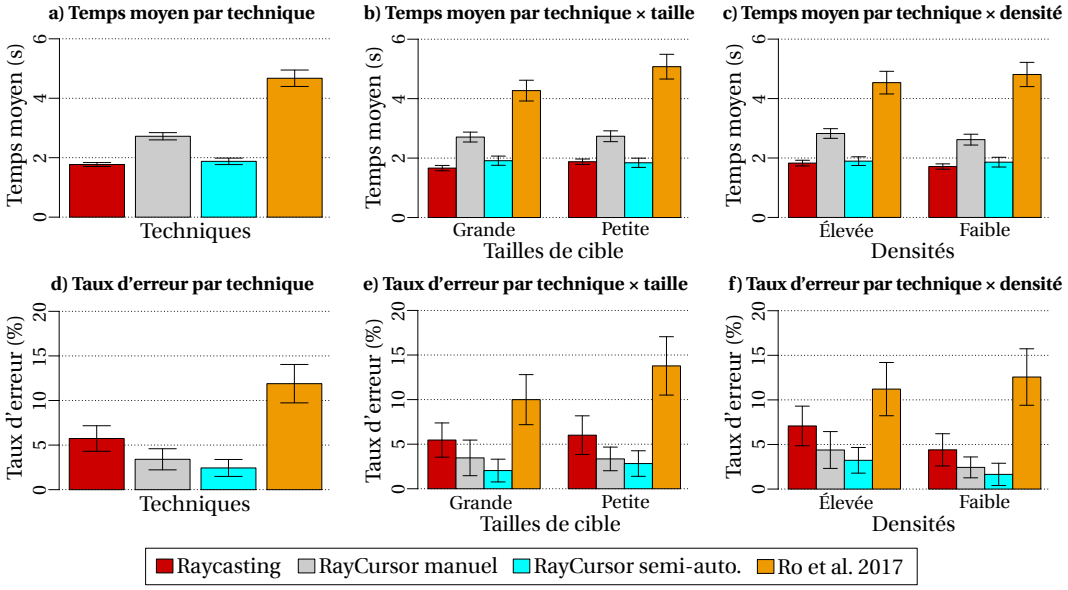


FIGURE II.14 – Temps moyen et taux d'erreur résultants de l'étude comparative, avec représentation des intervalles de confiance à 95 %.

lorsque les temps de sélection étaient au dessus de $mean + 3 \times sd$ pour chaque technique.

Nous avons appliqué une transformation de Box-Cox avec $\lambda = -0{,}3$ pour corriger les résidus de temps de sélection non normaux. Une ANOVA à mesures répétées [3] montre un effet significatif de BLOC ($F_{2,22} = 21{,}3$, $p = 0{,}001$, $\eta^2_G = 0{,}05$), avec les comparaisons par paires révélant une différence significative entre les blocs 1 et 3 ($p < 0{,}016$, Bloc 1 : 2,96s, 2 : 2,72s, 3 : 2,59s). Ces résultats ne suggérant pas clairement d'effet d'apprentissage ou de fatigue, nous avons gardé tous les blocs pour les analyses suivantes.

Nous observons aussi un effet significatif de TECHNIQUE ($F_{1,4,15,9} = 120{,}0$, $p < 0{,}0001$, $\eta^2_G = 0{,}71$). Des comparaisons par paires montrent un effet significatif entre *Ro* et les autres techniques (RC_f : 1,77s, *ManRCur* : 2,72s, *AutoRCur* : 1,88s, *Ro* : 4,67s, $p < 0{,}0001$), et aussi entre *ManRCur* et les autres techniques ($p < 0{,}0001$). La technique de *Ro et al.* est significativement plus lente que *ManRCur*, qui est elle-même significativement plus lente que les deux autres techniques. Avec le *RayCursor* semi-automatique, l'utilisateur a, dans la plupart des cas, juste à viser la bonne cible avec le rayon. Ces techniques sont donc plus rapides que les autres qui obligent l'utilisateur à déplacer le curseur le long du rayon pour atteindre la cible.

L'analyse montre un effet significatif de TAILLE ($F_{1,11} = 11{,}0$, $p < 0{,}007$, $\eta^2_G = 0{,}02$) et une interaction de TECHNIQUE×TAILLE ($F_{3,33} = 11{,}8$, $p < 0{,}0001$, $\eta^2_G = 0{,}03$). Des comparaisons par paires montrent seulement une différence significative pour la technique de *Ro*, entre les *Petites* et les *Grandes* cibles ($Ro_{Petites}$: 5,08s, $Ro_{Grandes}$: 4,27s, $p = 0{,}05$). Nous observons que notre implémentation de la technique de Ro *et al.* est plus lente pour les *Petites* cibles. Nous trouvons aussi un effet significatif de DENSITÉ ($F_{1,11} = 5{,}8$, $p < 0{,}035$, $\eta^2_G = 0{,}01$) et une interaction de TECHNIQUE×DENSITÉ ($F_{3,33} = 3{,}4$, $p < 0{,}03$, $\eta^2_G = 0{,}01$). Une

analyse post-hoc révèle que le temps de sélection augmente pour *Raycasting* avec des densités *Élevée* ($p < 0{,}001$, *Élevée* : 1,82s, *Faible* : 1,71s).

Taux d'erreur Le taux d'erreur global est de 5,9%, sachant que les participants avaient pour instruction de rester autour de 4% d'erreur. Les données ont été pré-traitées avec une ART pour prendre en compte leur distribution non normale. Une ANOVA à mesures répétées ne montre pas d'effet de BLOC ($F_{2,562} = 2{,}9$, $p > 0.05$). Cependant, elle montre un effet significatif de TECHNIQUE ($F_{3,549} = 25{,}5$, $p < 0{,}0001$). Des comparaisons par paires montrent des différences entre la technique de Ro *et al.* (11,9%) et toutes les autres techniques (RC_f : 5,7%, *ManRCur* : 3,8%, *AutoRCur* : 2,4%, $p < 0{,}0001$), aussi bien qu'entre RC_f et *AutoRCur* ($p = 0{,}013$). Le taux d'erreur élevé de la technique de Ro *et al.* est probablement dû au manque de sélection par proximité, comparé au *RayCursor* manuel. Aussi, *RayCursor* semi-automatique a un plus faible taux d'erreur que *Raycasting*. Nous expliquons cet effet par l'utilisation du principe de sélection par proximité lorsque le rayon dévie de la cible visée lorsque qu'on essaye de le sélectionner. Nous relevons aussi un effet significatif de TAILLE ($F_{1,549} = 11{,}9$, $p < 0{,}001$) montrant un taux d'erreur plus élevé pour des petites cibles (*Petite* : 6,5%, *Grande* : 5,2%). Aucun effet de DENSITÉ ($F_{1,549} = 0{,}01$, $p > 0.05$) n'a été trouvé.

Questionnaire NASA-TLX L'analyse de Friedman sur les réponses au questionnaire NASA-TLX indique qu'il y a un effet significatif pour l'exigence physique ($\chi^2(3) = 10{,}7$, $p = 0{,}01$), l'exigence temporelle ($\chi^2(3) = 12{,}6$, $p = 0{,}006$), la performance ($\chi^2(3) = 18{,}9$, $p = 0{,}0003$), l'effort ($\chi^2(3) = 15{,}9$, $p = 0{,}001$) et la frustration ($\chi^2(3) = 23{,}2$, $p < 0.0001$). L'analyse post-hoc de Wilcoxon révèle que le score d'exigence temporelle est significativement plus élevé pour *Ro* (médiane = 14,5) que pour les autres techniques (médianes = 8,5 $p < 0{,}04$). Elle montre aussi que l'estimation des participants de leur performance est significativement plus basse pour *Ro* (médiane = 14) que pour les autres techniques (médianes $\leq 8{,}5$ $p < 0{,}02$). Enfin elle montre que le score d'effort est significativement plus élevé pour *Ro* (médiane = 17) que pour les autres techniques (médianes $\leq 8{,}5$ $p < 0{,}03$) et que le score de frustration est également significativement plus élevé pour *Ro* (médiane = 15,5) que pour les autres techniques (médianes ≤ 5 $p < 0{,}02$). Il n'y a pas d'autre différence significative observée pour les tests post-hocs.

Préférence des participants À la fin de l'expérience, les participants devaient classer les techniques selon leurs préférences. Un test de Friedman révèle un effet significatif de TECHNIQUE pour les préférences utilisateurs ($\chi^2(3) = 25{,}3$, $p < 0.0001$). Une analyse post-hoc de Wilcoxon montre une différence significative entre la technique de Ro *et al.* (rang médian = 4) et toutes les autres techniques (rangs médians : RC_f : 2, *ManRCur* : 2, *AutoRCur* : 1, $p < 0{,}0001$) et entre le *RayCursor* semi-automatique et manuel ($p = 0{,}041$). Ces résultats subjectifs sont en accord avec nos analyses de taux d'erreur et de temps de sélection. Une majorité de participants ont placé le *RayCursor* semi-automatique en premier (9/12 participants).

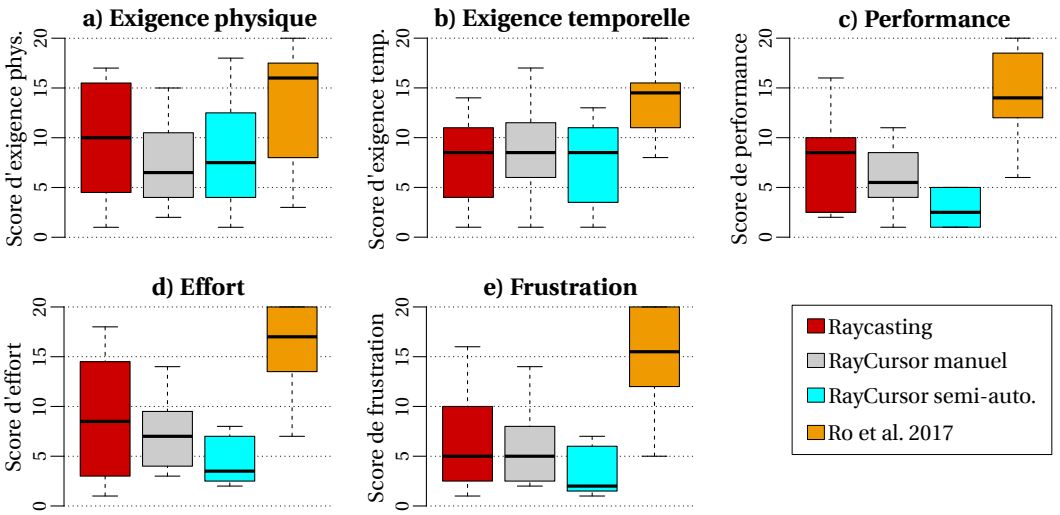


FIGURE II.15 – Diagrammes en boîte des réponses aux critères « Exigence physique », « Exigence temporelle », « Performance », « Effort » et « Frustration » au questionnaire NASA-TLX de la dernière expérience, pour chaque technique. L'échelle de performance du NASA-TLX est inversée afin que les valeurs élevées représentent des contre-performances sur toutes les échelles.

II.4.3 Discussion

Cette expérience montre que *RayCursor* semi-automatique obtient des temps de sélection similaires à ceux du *Raycasting* filtré, pour différentes tailles et densités de cibles. Cependant, le *RayCursor* semi-automatique réduit significativement les taux d'erreur sur le *Raycasting* filtré dans les différentes conditions. Ceci montre l'efficacité 1) du *1€ Filter* pour réduire les mouvements bruités ; 2) de la sélection par proximité pour les cibles en 3D ; et 3) du placement semi-automatique du curseur le long du rayon. Lors de la sélection de cibles distantes, le *RayCursor* semi-automatique aurait pu être affecté négativement par le rayon qui intersectait des cibles proches et faisait soudainement sauter le curseur, mais ce ne fut pas le cas. Si des sauts se produisaient pour des cibles d'éloignement similaire, l'utilisation d'une fonction d'hystérésis aiderait à résoudre le problème. Notre *RayCursor* manuel a obtenu des performances de temps inférieures par rapport aux techniques mentionnées précédemment, certainement en raison du déplacement incessant du curseur nécessaire pour déplacer le curseur en avant et en arrière d'un essai à l'autre. Cependant, en comparant le *RayCursor* manuel à la technique de Ro *et al.*, elle montre clairement les avantages de la sélection par proximité.

II.5 Conclusion

Nous avons présenté *RayCursor*, une nouvelle technique d'interaction pour la sélection de cibles 3D dans des environnements immersifs. Cette technique est une amélioration de *Raycasting* qui utilise un curseur sur le rayon pour sélectionner la cible la plus proche. L'affichage d'une bulle est le retour visuel typique d'une telle technique [32, 94]. Cependant Guillon *et al.* ont montré que, dans un contexte 2D, mettre en surbrillance la cible la plus proche est plus efficace, et produit moins d'encombrement visuel [36, 35]. Nous avons étendu leurs résultats à l'interaction 3D, avec des conclusions similaires.

Malgré l'existence de travaux antérieurs sur le filtrage du rayon, nous avons décrit la première étude d'un *Raycasting* filtré en évaluant formellement ses avantages. Nous avons montré que le filtrage du rayon réduit fortement les erreurs de sélection. Ceci est à la fois bénéfique pour *Raycasting* et *RayCursor*.

Nos résultats démontrent également que les fonctions de transfert comme celles utilisées sur les interfaces de bureau sont efficaces pour le contrôle d'un curseur sur un rayon. Nous recommandons d'utiliser une fonction sigmoïde qui dépend de la vitesse du curseur. Cependant, d'autres manettes de réalité virtuelle n'ont pas de surface tactile mais un stick analogique. Des études plus approfondies sont nécessaires afin d'appliquer *RayCursor* à ce type de contrôleur. En particulier il faudrait dans ce cas déterminer la fonction de transfert adéquate.

Nous montrons aussi qu'une technique hybride entre *Raycasting* et *RayCursor* a le plus faible taux d'erreur, tout en étant aussi rapide que *Raycasting*. Néanmoins, nous n'avons pas évalué l'efficacité de notre technique sur des cibles de formes non sphériques. En effet, avec des formes non sphériques, la manière dont est calculée la distance entre le curseur et chaque cible pourrait affecter les performances de la technique.

Aussi, nos expérimentations utilisant la densité des cibles comme facteur, auraient pu proposer un contrôle plus fin de cette densité. En effet dans nos expérimentations, nous avions positionné aléatoirement un nombre donné de cibles dans un volume donné. Vanacken *et al.* [94] proposent une approche plus précise, en positionnant manuellement les cibles distractives autour de la cible visée, à une distance fixe.

En 3D, la sélection précède souvent la manipulation d'un objet 3D. Un avantage secondaire et important de notre technique est sa capacité à manipuler l'objet le long du rayon une fois qu'il est sélectionné, par exemple pour le rapprocher de l'utilisateur, ce que le *Raycasting* standard ne permet pas de faire. Une évolution possible de *RayCursor* serait donc son extension à la manipulation d'objets 3D dans des environnements immersifs.

Dans ce chapitre, nous avons couvert les travaux réalisés pour améliorer les interactions entre l'avatar et les éléments de l'environnement virtuel. Le chapitre suivant exposera les travaux sur le contrôle de son avatar, pour interagir avec les autres utilisateurs de l'environnement virtuel.

Chapitre III

Techniques de contrôle d'expressions faciales

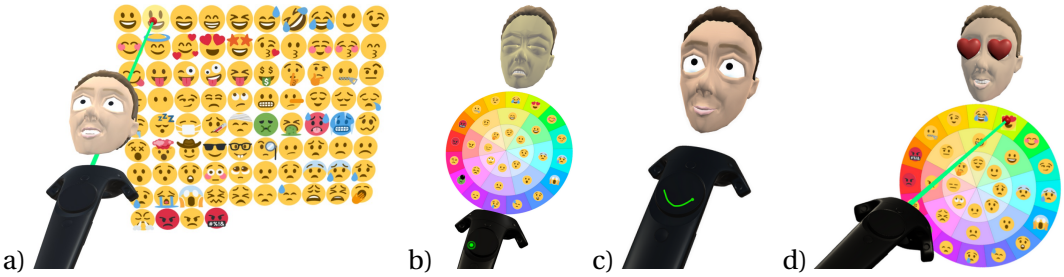


FIGURE III.1 – Illustration de quatre des techniques d'interaction développées : a) *RayMoji* présente une grille d'émoji devant l'utilisateur, et utilise *Raycasting* pour la sélection de l'expression, b) *EmoTouch* présente un menu circulaire au-dessus du contrôleur et la sélection est faite en utilisant le pad tactile, c) *EmoGest* utilise des gestes dessinés sur le touchpad pour définir une expression, et d) *EmoRay* est le résultat de la première expérimentation. Il présente un menu circulaire en face de l'utilisateur et utilise *Raycasting* pour la sélection de l'expression. Toutes ces techniques fournissent un retour visuel de l'expression via une version miniature du visage de l'avatar.

Publication

Marc Baloup, Thomas Pietrzak, Martin Hachet, and Géry Casiez.
Non-isomorphic Interaction Techniques for Controlling Avatar Facial Expressions in VR.
In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST 2021)*, December 2021.

Les réseaux sociaux se généralisent dans les environnements virtuels immersifs. Par exemple, Facebook [27], VRChat [98] et Rec Room [82] permettent aux utilisateurs de communiquer entre eux dans des salons de discussion virtuels. La communication dans les environnements immersifs est fondamentalement différente de la messagerie mobile ou sur ordinateur, car elle repose essentiellement sur la parole en raison de la difficulté à saisir du texte. Cela ouvre également des perspectives intéressantes : elle est plus engageante et favorise la présence sociale [31].

L'utilisation des expressions faciales est un aspect essentiel de la communication dans les environnements virtuels pour enrichir la communication vocale [43, 67], qu'il s'agisse de contextes professionnels comme les réunions et les conférences en réalité virtuelle [63] ou de divertissements comme les jeux de rôle en réalité virtuelle.

Dans ces situations, le contrôle de l'expression faciale d'un avatar doit être une tâche secondaire et ne pas perturber une tâche principale comme parler, manipuler des objets ou naviguer dans l'environnement. À ce titre, les utilisateurs doivent pouvoir contrôler les expressions faciales en temps réel, pour être en phase avec le rythme d'une discussion, par exemple.

Dans l'état de l'art, nous avons vu deux catégories de techniques, pour le contrôle de l'expression faciale : les techniques isomorphiques et non isomorphiques. Les techniques isomorphiques exploitent des capteurs pour détecter ou déduire l'expression du visage de l'utilisateur et la transposer au visage de l'avatar. Ces techniques aident à se concentrer sur la tâche principale car les utilisateurs n'ont pas à contrôler manuellement leur expression faciale. Cependant, les utilisateurs ne peuvent pas choisir l'expression faciale qu'ils souhaitent voir apparaître sur leur avatar. Ils peuvent vouloir exprimer quelque chose de différent de ce qu'ils expriment avec leur visage ou voix réels. Par exemple, les utilisateurs ne voudront pas forcément rire aux éclats pour que leur avatar s'anime en conséquence.

Les techniques non isomorphiques permettent à l'utilisateur de contrôler manuellement l'expression faciale de leur avatar, via des techniques d'interaction et les périphériques d'entrées (les manettes par exemple). Il s'agit d'une solution alternative qui permet de surmonter les limites des techniques isomorphiques. Cependant, les techniques existantes sont limitées à un petit ensemble d'expressions faciales avec un contrôle limité de leurs paramètres [82], ce qui ouvre des possibilités de conception de nouvelles techniques d'interaction.

Dans ce chapitre, nous souhaitons explorer les techniques d'interaction pour le contrôle non isomorphique de l'expression faciale de son propre avatar. Nous proposons un ensemble de techniques d'interaction non isomorphiques pour contrôler les expressions faciales d'un avatar incarné, de la sélection d'une expression au contrôle de son intensité en temps réel. Ces techniques couvrent différentes parties de l'espace de conception décrit dans la section I.2.5, et permettent d'illustrer son pouvoir génératif. Les techniques ont été évaluées à travers deux expériences contrôlées afin d'évaluer leurs performances et les préférences subjectives des participants. Cela nous a permis de

proposer une nouvelle technique qui exploite les points forts de chaque approche. Enfin, nous avons validé cette technique dans une expérience écologique.

III.1 Techniques d'interaction proposées

Nous illustrons notre espace de conception avec un ensemble de nouvelles techniques d'interaction, couvrant différents aspects de la décomposition en sous-tâches, ainsi que le respect des exigences que nous détaillons ci-dessous. Elles permettent le contrôle manuel et non isomorphique des expressions faciales d'un avatar en réalité virtuelle. Ces techniques ont été développées en prenant en compte les entrées utilisateur de la manette HTC Vive, qui sont décrits et illustrés dans l'annexe A.

III.1.1 Contraintes de conception

Nous considérons que ces techniques d'interaction doivent interférer le moins possible avec des tâches principales telles que parler ou interagir avec l'environnement. Cela implique que la sélection et le contrôle d'une expression faciale doivent nécessiter un effort cognitif minimal et que la sélection doit être rapide pour interférer le moins possible avec la tâche principale. De plus, le retour d'information est un aspect important à considérer étant donné que les utilisateurs ne voient pas le visage de leur avatar. Il est important pendant l'étape de sélection pour aider les utilisateurs à choisir une expression appropriée. Le retour d'information est ensuite important pour aider à contrôler l'intensité d'une expression ou rappeler aux utilisateurs l'expression actuelle du visage de leur avatar.

Nous avons choisi d'utiliser des émojis pour représenter les expressions de nos techniques basées sur les menus. Notre conception initiale des menus utilisait des représentations du visage de l'avatar correspondant, mais les expressions n'étaient pas toujours faciles à différencier. Les émojis semblent au contraire plus faciles à distinguer, en raison des attributs bien visibles du visage, et ils peuvent être représentés de manière plus petite, ce qui permet d'en intégrer plus dans l'interface. L'utilisation d'émojis rend aussi nos techniques indépendantes de l'avatar incarné par l'utilisateur. En outre, les utilisateurs sont familiers avec les émojis, ce qui pourrait faciliter le transfert de compétences depuis les applications sociales.

III.1.2 Implémentation d'un visage d'avatar

Nous avons réalisé un système de rendu et d'animation avec le moteur de jeu Unity version 2019.4. Nous avons choisi des techniques de rendu et d'animation nous permettant de mettre en œuvre nos techniques d'interaction tout en fournissant des résultats convaincants et faciles à reproduire.

Nous avons adapté un modèle 3D existant d'un visage [1] pour l'utiliser dans Unity (figure III.2). Nous avons choisi ce modèle pour sa similitude avec le style non réaliste utilisé dans les jeux VR et les applications sociales. L'utilisation de modèles plus réalistes

1. Modèle par *cgcookie*, CC-BY (https://www.blendswap.com/blend/22625)

nécessiterait l'utilisation de techniques de rendu et d'animation plus avancées [65, 84], ce qui dépasse le cadre de ce travail. Le moteur de jeu Unity utilise des *blend shapes* (appelés aussi *shape keys* dans Blender) pour contrôler les degrés de liberté afin d'animer des modèles 3D avec le code C# ou l'interface graphique de Unity. En utilisant Blender, nous avons modifié certaines *shape keys* pour les faire correspondre le mieux possible à 21 des *Action Units* du FACS [25].

Nous avons ajouté des caractéristiques supplémentaires au visage de l'avatar. La couleur de la peau peut être ajustée en fonction de certaines expressions, comme *la colère* 😡, *la nausée* 🤢, *chaud* 🥵 ou *froid* 🥶. Nous avons aussi ajouté des décorations supplémentaires, comme des larmes pour *mort de rire* 🤣, *pleurer* 😢 or *transpirer* 😰, ou des cœurs pour *les yeux en cœur* 😍. La figure III.2 montre un aperçu de notre avatar avec certaines de ces expressions. Nous avons aussi animé certaines expressions, notamment *mort de rire* 🤣 ou *froid* 🥶 pour lesquelles la machoire bouge en conséquence. Tous ces ajouts nous permettent de couvrir les 85 émojis de la catégorie *Smiley* [92]. Enfin, nous lissons la transition d'une expression à une autre en utilisant un filtrage exponentiel, avec $\alpha = 0{,}2$ et un taux d'échantillonnage de 90 Hz. Ces valeurs ont été ajustées lors d'essais informels.

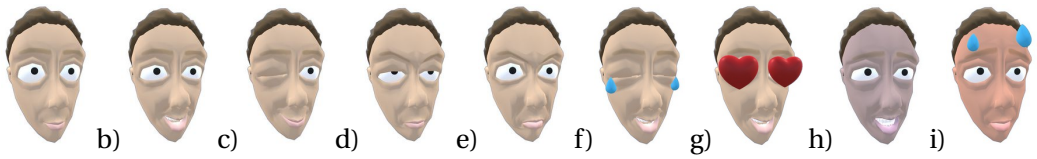


FIGURE III.2 – Exemples d'expressions faciales avec notre implémentation de visage. a) *neutre* 😐; b) *sourire* 😀; c) *clin d'œil* 😉; d) *pas amusé* 😒; e) *en colère* 😠; f) *larme de joie* 😂; g) *yeux en cœur* 😍; h) *froid* 🥶; i) *chaud* 🥵

Pour fournir un retour visuel sur l'expression sélectionnée ou la prévisualiser, nous utilisons une version miniature de ce modèle de visage, que nous appelons *PuppetFace*. Celle-ci est affichée dans le champ de vision de l'utilisateur, par exemple à proximité du menu affiché (figure III.1).

Le code source des techniques ainsi que l'implémentation du PuppetFace sont disponibles à l'adresse ns.inria.fr/loki/AvatarFacialExpressions.

III.1.3 Sélection d'une expression

Nous présentons quatre techniques d'interaction pour la sélection d'expressions. L'objectif est d'explorer un large éventail de modalités, des menus aux gestes et aux commandes vocales. La première technique est *RayMoji*, qui utilise un panneau montrant des émojis que l'utilisateur peut sélectionner en utilisant un *Raycasting*. La deuxième est *EmoTouch*, un menu circulaire s'utilisant avec le pavé tactile du contrôleur. La troisième est *EmoGest*, qui repose sur un ensemble de gestes sémaphoriques que l'utilisateur trace sur le pavé tactile. Enfin la quatrième technique est *EmoVoice*, un ensemble de commandes vocales utilisant la reconnaissance vocale.

III.1.3.1 *RayMoji*

RayMoji montre un panneau à 1 m devant l'utilisateur, et le PuppetFace apparaît au-dessus du contrôleur virtuel (figure III.3). Le panneau est figé dans l'espace 3D, et contient une grille comportant jusqu'à 10 × 9 émojis. La taille du menu et celle des émojis sont un compromis entre la facilité de sélection des émojis et l'obstruction du champ de vision de l'utilisateur. Nous avons ajusté ces valeurs grâce à des études pilotes informelles.

La disposition du menu est similaire à celle dans les messageries instantanées et réseaux sociaux (figure I.5). L'ordre des émojis est similaire, et défini par la norme Unicode sur les émojis [92]. Nous supposons donc que les utilisateurs sont habitués à cette présentation, ce qui peut favoriser le transfert d'apprentissage. Nous avons utilisé 84 émojis, qui appartiennent tous à la catégorie *Smileys et émotion* de la norme Unicode. Le nombre d'émojis affichés peut être modifié en fonction des besoins de l'application. Cette technique permet de fournir facilement un large éventail d'expressions, de celles représentant des émotions à celles moins réalistes.

Pour sélectionner une expression faciale, l'utilisateur vise l'émoji correspondant dans le menu avec la technique *Raycasting*. Le PuppetFace montre l'expression qui est survolée dans l'interface. Lorsque l'utilisateur appuie sur le bouton de déclenchement, le menu disparaît et le visage de l'avatar affiche l'expression sélectionnée.

FIGURE III.3 – *RayMoji* vu par l'utilisateur. Au-dessus de la manette se trouve un aperçu de l'expression lorsque l'utilisateur survole un émoji avec le rayon. Le menu est affiché à 1 m devant l'utilisateur.

RayMoji se place dans notre espace de conception dans « Contrôle non isomorphique », puis dans « Sélection d'une expression ». Pour l'indication de l'expression, la technique utilise des entrées continues qui sont la position et l'orientation de la manette pour le contrôle du rayon, ainsi qu'un menu en grille pour l'affichage des expressions proposées. Pour la confirmation de l'expression, la technique utilise une entrée discrète qui est le bouton de déclenchement.

III.1.3.2 *EmoTouch*

EmoTouch affiche un menu circulaire au-dessus de la représentation virtuelle du contrôleur (figure III.4). Il est conçu pour représenter uniquement les expressions correspondant à des émotions, sous forme d'émojis. L'utilisateur contrôle un curseur sur ce

menu avec le touchpad circulaire du contrôleur pour sélectionner une des expressions faciales, en utilisant une correspondance absolue. Le PuppetFace apparaît au-dessus de ce menu pour indiquer en temps réel l'expression correspondante à l'émoji survolé. L'expression faciale est sélectionnée lorsque l'utilisateur appuie sur la gâchette ou sur le pavé tactile[2]. Le menu disparaît, mais le PuppetFace reste visible. Le visage de l'avatar et le PuppetFace affichent l'expression sélectionnée jusqu'à ce que l'utilisateur relâche le bouton. À ce moment-là, le visage de l'avatar revient progressivement à l'expression neutre et le PuppetFace disparaît.

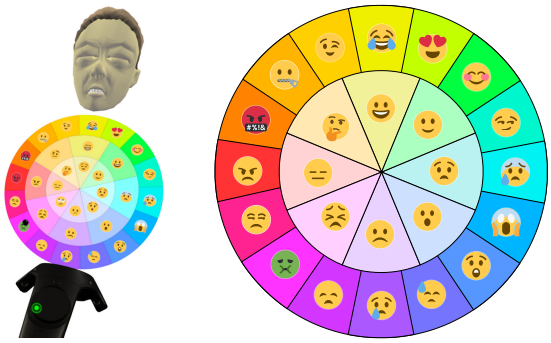


FIGURE III.4 – *EmoTouch* à gauche : Au dessus de la manette, le menu circulaire avec les émojis est affiché, ainsi que le PuppetFace au dessus ; À droite, une vue rapprochée du menu circulaire.

La disposition du menu circulaire est similaire à la roue des émotions de Plutchik [74], avec des émojis pour représenter les expressions. Dans la roue originale, les émotions les plus fortes sont situées au centre [75]. Nous avons inversé le niveau d'intensité de sorte que l'émotion neutre se trouve au centre de la roue, et les émotions les plus fortes à la périphérie. La disposition de ce modèle place judicieusement les émotions similaires les unes à côté des autres, ce qui, selon notre hypothèse, facilite la recherche visuelle et aide à mémoriser l'emplacement de chaque expression. Cependant, le choix des émojis diffère légèrement du modèle de Plutchik, car nous n'avons pas trouvé d'émojis représentant la *vigilance*, la *soumission*, la *confiance*, l'*admiration* et l'*optimisme*. Nous les avons remplacés par des émojis dont la signification est proche de celle des émojis environnants. Nous avons ajusté la taille du menu et le nombre d'émojis suite à des études pilotes informelles. Il s'agit d'un compromis entre un nombre élevé d'éléments et la précision de la sélection. Comme dans la roue de Plutchik, chaque émotion de base utilise une teinte distincte (rouge, orange, jaune, vert, cyan, bleu, violet, rose), et la saturation représente l'intensité.

EmoTouch se place dans notre espace de conception dans « Contrôle non isomorphique », puis dans « Sélection d'une expression ». Pour l'indication de l'expression, la technique utilise des entrées continues qui sont la position du doigt sur le pad tactile de la manette, ainsi qu'un menu circulaire de plusieurs niveaux pour l'affichage des expressions proposées. Pour la confirmation de l'expression, la technique utilise des entrées discrètes qui sont soit la gâchette, soit l'appui sur le pavé tactile.

2. Sur les manettes de RV dotés d'un pavé tactile, le système peut différencier le moment où l'utilisateur touche ou appuie sur le pavé tactile. Voir l'annexe A pour plus de détails.

III.1.3.3 *EmoGest*

EmoGest permet aux utilisateurs de sélectionner une expression faciale avec des gestes simples sur le pavé tactile du contrôleur. L'avatar de l'utilisateur et le PuppetFace affichent directement l'expression faciale sélectionnée. L'expression courante reste active jusqu'à ce que l'utilisateur désactive *EmoGest* avec le bouton associé [3] ou qu'il sélectionne une autre expression. Pour montrer tous les gestes disponibles, nous fournissons un menu d'aide accessible en serrant la manette [4].

Contrairement aux techniques précédentes, celle-ci n'affiche pas d'émojis. Cette technique a donc l'avantage de ne pas solliciter le regard, ce qui permet de se concentrer sur la tâche principale.

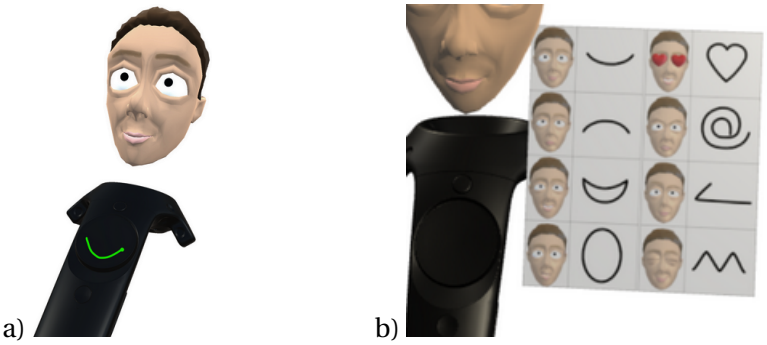


FIGURE III.5 – a) *EmoGest* : Au-dessus de la manette, le PuppetFace illustrant l'expression faciale de l'avatar de l'utilisateur en temps réel. L'utilisateur trace directement un geste sur le touchpad de la manette ; b) Le menu d'aide activé par l'utilisateur, qui illustre les expressions implémentées avec leurs gestes.

Nous avons choisi d'implémenter les 8 expressions comme indiqué dans la figure III.5. Pour décider des gestes à utiliser pour chaque expression, nous avons d'abord mené une étude informelle d'élicitation de gestes avec 6 participants. Nous leur avons demandé de dessiner un geste pour chaque expression. Au moins 2 participants ont dessiné des gestes similaires pour *léger sourire* (6 participants), *triste* (2), *grand sourire* (5), *bouche ouverte* (4), *yeux souriants* (2) et *yeux en cœurs* (5), qui sont aussi inspirés des émoticônes ASCII `:-)`, `:-(`, `:-D`, `:-O`, `^^` et du symbole du cœur. L'expression faciale de la *colère* est représentée par la forme `@`, inspirée de l'émoticône ASCII `:-@` souvent utilisé pour la colère. Comme il n'existe pas d'émoticône ASCII équivalente pour l'expression *pensif*, nous utilisons le geste illustré dans la figure III.6, qui suit la forme de la main dans l'émoji 🤔. L'ensemble des gestes est limité à un petit nombre (8) pour favoriser l'apprentissage et réduire les erreurs de reconnaissance. Aussi, nous avons limité notre technique à des gestes à un seul trait pour réduire le temps de sélection.

Nous avons utilisé une version personnalisée de l'algorithme de reconnaissance de geste $1 [101]. Nous avons désactivé le sous-algorithme de *rotation à zéro* pour différencier les formes qui sont identiques mais qui ont des orientations différentes (comme *léger sourire* et *triste* illustrés figure III.6), mais nous avons maintenu la correction de ro-

3. Le bouton `Menu` sur le HTC Vive, et le bouton `B` sur le Valve Index. Voir l'annexe A pour plus de détails.
4. Les boutons de serrage sur la manette HTC Vive et les détecteurs de suivi des doigts sur la manette Valve Index. Voir l'annexe A pour plus de détails.

tation à ±45° pour permettre des entrées légèrement inclinées. Nous avons également amélioré la normalisation de l'échelle afin de conserver le rapport largeur/hauteur du trait dessiné. Cela permet la reconnaissance de formes telles que les traits horizontaux et verticaux. Enfin, notre version du moteur de reconnaissance tente également de faire correspondre les traits avec leur version inversée (c'est-à-dire que la liste des points qui constitue le tracé est inversée) afin de permettre aux utilisateurs de dessiner des formes dans un sens ou dans l'autre.

Après des études pilotes informelles auprès de 3 participants, pour réduire les erreurs de détection, nous avons adapté *grand sourire* depuis la forme de la lettre `D` (16 % d'erreur) à la forme d'un croissant de lune (6 %), et *bouche ouverte* depuis un cercle (33 %) vers une ellipse verticale (14 %).

EmoGest se place dans notre espace de conception dans « Contrôle non isomorphique », puis dans « Sélection d'une expression ». Pour l'indication de l'expression, la technique utilise des entrées continues qui sont la position du doigt sur le pad tactile de la manette, ainsi que la détection de gestes. Pour la confirmation de l'expression, la technique utilise une entrée discrète qui est le fait de relever le doigt du pad tactile.

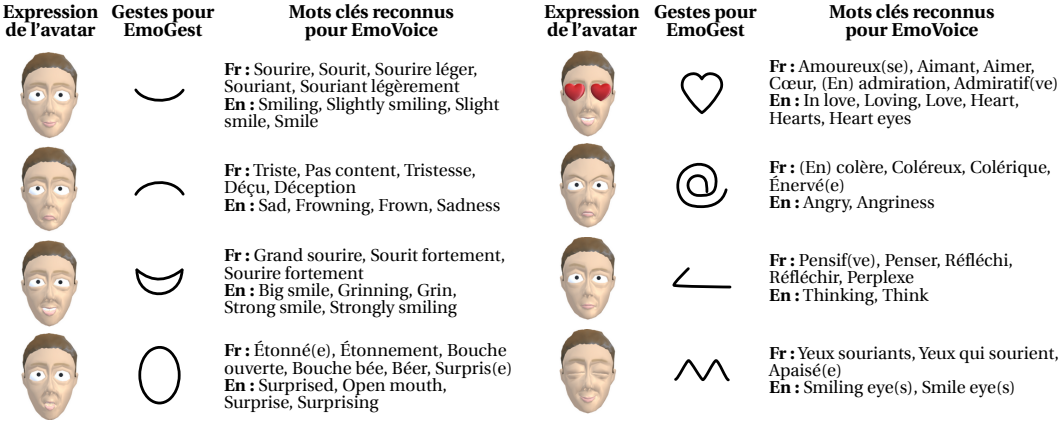


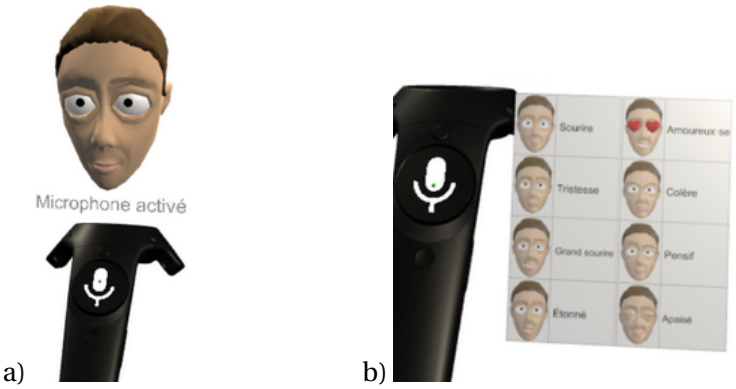
FIGURE III.6 – Gestes implémentés dans *EmoGest* et mots-clés reconnus par *EmoVoice*, associés à leurs expressions faciales respectives.

III.1.3.4 *EmoVoice*

EmoVoice utilise la reconnaissance vocale pour contrôler l'expression faciale de l'avatar. Lorsque l'utilisateur active la technique en appuyant sur le touchpad, le système écoute le microphone jusqu'à ce que le touchpad soit relâché. Pendant ce temps, il détecte des mots-clés spécifiques et active l'expression correspondante sur le visage de l'avatar.

Comme pour *EmoGest*, les utilisateurs peuvent serrer la manette [4] pour accéder à un menu d'aide qui présente les expressions disponibles. Cette technique n'utilise pas non plus d'émojis et présente donc l'avantage de ne pas nécessiter la sollicitation de la vue.

Pour chaque expression implémentée, nous avons associé plusieurs mots-clés qui correspondent à l'expression. Il est également possible de prendre en charge plusieurs



a) b)

FIGURE III.7 – a) l'interface de *EmoVoice* sur la manette ; b) le menu d'aide activé par l'utilisateur, montrant les expressions implémentées avec un mot clé pour chacun.

langues, avec des ensembles de mots-clés et des configurations de reconnaissance vocale différentes. Par exemple, pour l'expression *léger sourire* 🙂, les mots `slight smile`, `smile`, et `smiling` sont reconnus en anglais. Certains des mots-clés ont été suggérés par les participants lors d'études pilotes informelles. La liste complète des mots-clés en français et en anglais se trouve dans la figure III.6.

Nous avons utilisé l'outil Speech-to-text de l'API de Google, car il a montré une meilleure performance (temps de calcul et erreur de reconnaissance) que CMU Sphinx. Nous avons utilisé le script Python SpeechRecognition pour interagir avec ces outils [103].

EmoVoice se place dans notre espace de conception dans « Contrôle non isomorphique », puis dans « Sélection d'une expression ». Pour l'indication de l'expression, la technique utilise une entrée discrète qui est le bouton d'activation de la technique, ainsi que la détection de la voix. Pour la confirmation de l'expression, la technique utilise une entrée discrète qui est le relâchement du bouton d'activation.

III.1.4 Contrôle de l'intensité et de la durée de l'expression

En suivant la décomposition en sous-tâches de notre espace de conception, nous présentons cinq techniques pour contrôler l'intensité, la durée et la fin de l'expression du visage. Nous choisissons de contrôler l'intensité en temps réel, ce qui permet indirectement de contrôler la durée et la fin de l'expression lorsque l'intensité atteint 0. Nous pensons que la manipulation directe de l'intensité est plus appropriée dans les situations sociales où les expressions sont plus spontanées. De plus, cela permet de contrôler facilement l'animation du visage.

Les cinq techniques sont l'utilisation de la gâchette, l'orientation ou la secousse de la manette, la représentation d'un élastique virtuel, et la position d'un curseur sur le menu circulaire.

III.1.4.1 La gâchette de la manette

La première technique associe en temps réel, la position de la gâchette à l'intensité de l'expression contrôlée (figure A.2). La fin de l'expression est déclenchée après un délai de 1 seconde après avoir relâché la gâchette. Ce délai permet à l'utilisateur de régler l'intensité sans perdre le contrôle de l'expression par accident.

Nous avons également implémenté un retour haptique qui fonctionne comme suit : nous avons divisé l'intervalle $[0;1]$ en 10 sous-intervalles de la même taille. Chaque fois que l'utilisateur passe à un nouveau sous-intervalle pendant le contrôle de l'intensité en temps réel, une impulsion haptique est déclenchée dans le contrôleur. L'intensité de l'impulsion haptique est proportionnelle à l'intensité actuelle de l'expression faciale. Nous supposons que le retour haptique aide l'utilisateur à contrôler l'intensité de l'expression.

Cette technique se place dans notre espace de conception dans « Contrôle non isomorphique », puis dans « Contrôle de l'intensité de l'expression » et « Confirmation de la fin de l'expression ». Pour le contrôle de l'intensité, elle utilise une entrée continue qui est la gâchette de la manette. Pour la fin de l'expression, elle utilise un délai au delà duquel l'utilisateur a complètement relâché la gâchette.

III.1.4.2 L'orientation de la manette

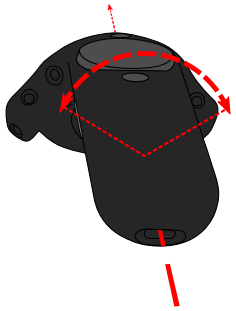


FIGURE III.8 – L'axe de rotation autour duquel l'utilisateur tourne la manette pour ajuster l'intensité.

La deuxième technique consiste à faire correspondre l'inclinaison du contrôleur à l'intensité. L'axe de rotation est celui allant de l'avant de la manette vers l'arrière (de l'anneau du capteur vers le port de recharge, sur la manette HTC Vive). Lorsque le touchpad est orienté vers le haut, l'inclinaison est de 0°, donnant une expression d'intensité nulle. L'intensité augmente lorsque la manette est inclinée vers la droite ou vers la gauche, jusqu'à 50°, correspondant à l'intensité maximale. Nous avons fixé cet angle après des études pilotes informelles. L'expression se termine lorsque l'utilisateur relâche le bouton de sélection. Cette technique utilise le même retour haptique que celui détaillé pour la technique précédente.

Cette technique se place dans notre espace de conception dans « Contrôle non isomorphique », puis dans « Contrôle de l'intensité de l'expression » et « Confirmation de la fin de l'expression ». Pour le contrôle de l'intensité, elle utilise une entrée continue qui

est l'orientation de la manette. Pour la fin de l'expression, elle utilise le relâchement du bouton qui a servi à valider la sélection de l'expression.

III.1.4.3 Secouer la manette

FIGURE III.9 – Contrôle de l'intensité de l'expression en secouant la manette. Le PuppetFace montre l'expression de l'avatar en temps réel.

La troisième technique associe l'intensité du secouement de la manette à l'intensité de l'expression. Plus la secousse est forte, plus l'intensité de l'expression est élevée. Pour mesurer l'intensité des secousses, nous utilisons la norme du vecteur d'accélération du contrôleur. Nous filtrons cette valeur avec une moyenne glissante de 0,5 s pour éviter des oscillations d'intensité indésirables. Nous avons ajusté les paramètres de cette technique après des études pilotes informelles. L'expression se termine lorsque l'utilisateur relâche le bouton de sélection.

Cette technique se place dans notre espace de conception dans « Contrôle non isomorphique », puis dans « Contrôle de l'intensité de l'expression » et « Confirmation de la fin de l'expression ». Pour le contrôle de l'intensité, la technique utilise une entrée continue qui est l'accélération de la manette en temps réel. Pour la fin de l'expression, elle utilise le relâchement du bouton qui a servi à valider la sélection de l'expression.

III.1.4.4 L'élastique virtuel

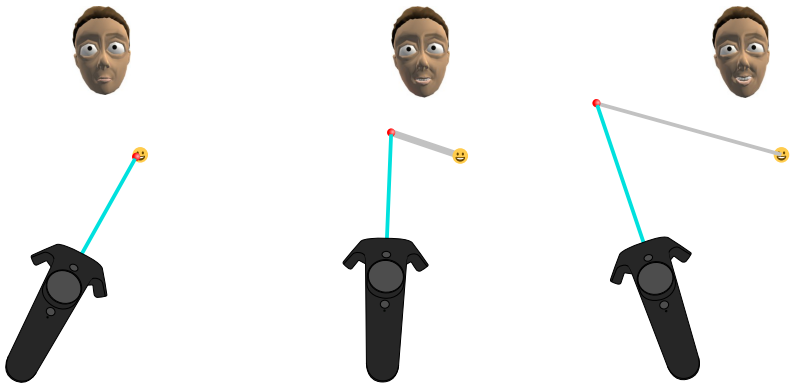


FIGURE III.10 – Contrôle de l'intensité de l'expression en tirant sur un segment virtuel (en gris dans la figure). Ce segment s'affine en fonction de sa longueur, pour imiter un vrai élastique. Le PuppetFace montre l'expression de l'avatar en temps réel.

La quatrième technique associe la longueur d'un segment virtuel blanc à l'intensité. Une extrémité du segment est fixée à la représentation visuelle de l'expression sélectionnée dans le menu. L'autre extrémité est fixée soit au contrôleur virtuel, soit à l'extrémité

d'un rayon (par exemple le rayon dans *RayMoji*), afin que l'utilisateur puisse manipuler la longueur du segment. Nous réduisons l'épaisseur du segment au fur et à mesure qu'il s'allonge, de la même manière qu'un élastique. Cette technique utilise le même retour haptique que celui présenté précédemment.

Cette technique se place dans notre espace de conception dans « Contrôle non isomorphique », puis dans « Contrôle de l'intensité de l'expression » et « Confirmation de la fin de l'expression ». Pour le contrôle de l'intensité, la technique utilise une entrée continue qui est la position et l'orientation de la manette, pour le contrôle d'une des extrémités du segment virtuel. Pour la fin de l'expression, elle utilise le relâchement du bouton qui a servi à valider la sélection de l'expression.

III.1.4.5 Contrôle d'intensité sur menu circulaire

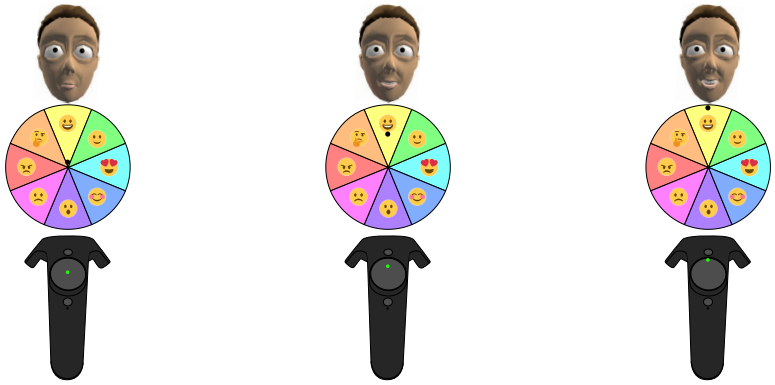


FIGURE III.11 – Contrôle de l'intensité de l'expression en fonction de la position du curseur sur le menu. Ici, le curseur est contrôlé grâce à la technique *EmoTouch*. Le PuppetFace permet d'observer l'intensité de l'expression. À gauche : le curseur étant proche du centre, l'intensité est presque nulle. Au milieu : le curseur est à une position intermédiaire entre le centre et le bord du menu, donc l'intensité est d'environ 0,5. À droite : le curseur est sur le bord, donc l'intensité est maximale.

La cinquième technique nécessite un menu circulaire, tel que *EmoTouch* mais avec un seul niveau (8 expressions). Après avoir sélectionné une expression, l'utilisateur peut ajuster son intensité en déplaçant le curseur dans le menu. L'intensité est nulle lorsque l'utilisateur vise au milieu du menu, et maximale sur le bord (figure III.11). Cette technique utilise le même retour haptique que celui présenté précédemment.

Cette technique se place dans notre espace de conception dans « Contrôle non isomorphique », puis dans « Contrôle de l'intensité de l'expression » et « Confirmation de la fin de l'expression ». Pour le contrôle de l'intensité, la technique utilise des entrées dépendantes de la technique pour le choix de l'expression. Si le choix de l'expression est fait avec *RayMoji*, le contrôle d'intensité utilise les entrées continues de la position et l'orientation de la manette, pour le contrôle du rayon. Si le choix de l'expression est effectué avec *EmoTouch*, le contrôle d'intensité utilise les entrées continues pour la position du doigt sur le pad tactile. Pour la fin de l'expression, la technique utilise le relâchement du bouton qui a servi à valider la sélection de l'expression.

III.2 Études comparatives

Nous avons conçu deux expériences contrôlées pour évaluer la performance et la préférence des utilisateurs des différentes techniques d'interaction que nous avons introduites, pour la sélection d'une expression faciale et le contrôle de son intensité. Ces études ont obtenu l'accord du comité d'éthique d'Inria.

III.2.1 Comparaison des techniques de sélection d'expressions faciales

Dans cette première expérience, nous comparons les techniques de sélection d'expressions faciales non isomorphiques présentées dans la section III.1.3. Nous étudions les hypothèses suivantes :

(H1) Le temps de sélection augmente avec le nombre d'expressions disponibles, les utilisateurs passant plus de temps à trouver l'expression correcte.

(H2) *EmoGest* et *EmoVoice* sont plus sujettes aux erreurs en raison des limites des algorithmes de reconnaissance.

(H3) *EmoGest* et *EmoVoice* nécessitent un entraînement plus poussé que les autres techniques, car elles exigent d'apprendre et d'exécuter des commandes gestuelles et vocales.

(H4) *EmoTouch* est plus rapide que les autres techniques grâce à la correspondance spatiale des expressions qui facilite la recherche de celle voulue.

(H5) Les utilisateurs préfèrent les techniques provoquant moins d'erreurs et nécessitant moins d'apprentissage.

Nous nous sommes également intéressés par la recherche de différences relatives entre les techniques en termes de performances et d'évaluations subjectives.

Matériel Pour cette expérience, nous avons utilisé un ordinateur équipé d'un casque de réalité virtuelle HTC Vive [41]. Les participants manipulaient le contrôleur HTC Vive avec leur main dominante. L'application de l'expérience a été développée en C# avec Unity 3D et la librairie OpenVR.

Tâche Les participants étaient assis sur une chaise placée devant deux écrans dans la scène virtuelle. Le premier était situé directement en face du participant pour afficher les instructions. Le second était situé à droite pour afficher des informations supplémentaires (figure III.12). Les participants ont incarné le visage d'un avatar décrit dans la section III.1.2, et n'avaient pas de corps.

Chaque essai consistait à reproduire une expression faciale donnée en utilisant l'une des techniques. L'expression à reproduire était d'abord affichée à gauche sur l'écran d'instructions à l'aide d'une représentation du visage de l'avatar avec l'expression attendue. Les instructions n'utilisaient pas d'émoji ni de texte, pour éviter de favoriser l'une des techniques. Les participants pouvaient ensuite activer la technique en appuyant sur le touchpad de la manette. Après avoir sélectionné une expression, une représentation

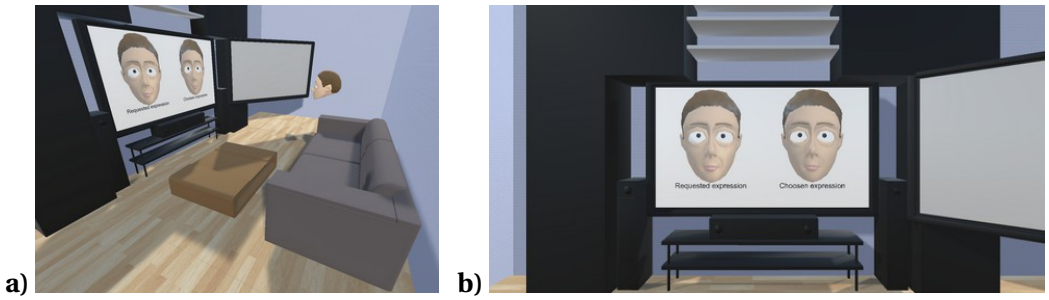


FIGURE III.12 – Scène 3D utilisée pour la première expérience ; **a)** une vue globale sur la scène ; **b)** le point de vue du participant dans la scène.

du visage de l'avatar avec l'expression résultante était affichée à droite sur l'écran d'instruction. Un message « Bon » ou « Mauvais » apparaissait sur l'écran pendant 1,5 s pour indiquer si la sélection de l'expression était correcte. En outre, si la sélection était incorrecte, le contrôleur vibrait pendant 500 ms, et le participant avait la possibilité de réaliser une nouvelle tentative de sélection. Si le participant ne parvenait pas à sélectionner la bonne expression 4 fois de suite, l'expérience passait à l'essai suivant. Les mêmes expressions faciales ont été demandées aux participants, et dans le même ordre pour tous les participants, TECHNIQUES, et BLOCS, pour éviter d'introduire un effet aléatoire dans les mesures.

Méthodologie Douze participants ont pris part à cette expérience (2 femmes, âge moyen = 26,2, σ = 5,3). Deux d'entre eux n'avaient pas d'expérience préalable en réalité virtuelle. Ils devaient sélectionner des expressions faciales en utilisant 6 TECHNIQUES : EmoGest (*EG8*), EmoVoice (*EV8*), EmoTouch avec 8 expressions (*ET8*), RayMoji avec 8 expressions (*RM8*), EmoTouch avec 24 expressions (*ET24*) et RayMoji avec 24 expressions (*RM24*). EmoGest et EmoVoice proposaient tous les deux un ensemble de 8 expressions pour l'expérience. EmoTouch et RayMoji ont tous les deux été testés avec 8 et 24 expressions. Les techniques avec 8 expressions permettent une comparaison équitable par rapport à EmoGest et EmoVoice. Ceux avec 24 expressions permettent de mesurer les bénéfices/coûts d'un plus grand nombre d'expressions proposé. Nous avons utilisé un carré latin pour équilibrer l'ordre des techniques entre les participants. Chaque bloc était composé de 8 essais. Pour chaque technique et chaque bloc, les mêmes expressions faciales suivantes étaient demandées aux participants, dans le même ordre : *pensif* 🤔, *léger sourire* 🙂, *triste* 🙁, *en colère* 😠, *grand sourire* 😀, *yeux en cœur* 😍, *bouche ouverte* 😮 et *yeux souriants* 😊.

Pour chaque technique, les participants ont d'abord effectué un bloc d'entraînement dans lequel ils ont sélectionné les 8 expressions et une seconde phase d'entraînement de 2 minutes, pendant laquelle ils étaient libres d'utiliser la technique. Pendant ces deux phases d'entraînement, l'écran de droite affichait des instructions sur la manière d'utiliser la technique en cours. Enfin, les participants ont effectué 4 blocs de 8 expressions, pour lesquelles les temps de sélection et le taux d'erreur étaient mesurés.

Le protocole expérimental intra-sujet de l'expérience était donc : 12 participants × 6 TECHNIQUES × 4 BLOCS × 8 sélections d'expression = 2 304 essais au total. L'expérience durait environ 1 heure et 15 minutes par participant.

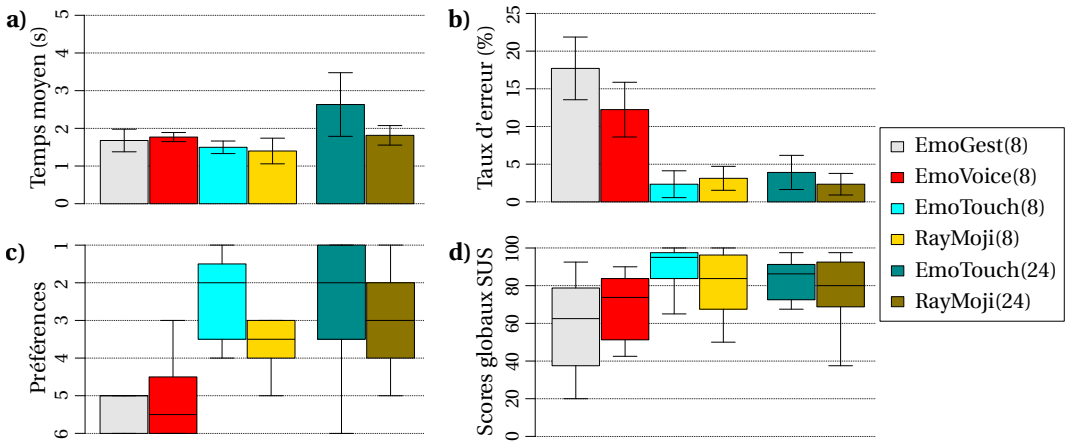


FIGURE III.13 – a) Temps moyens, b) taux d'erreur, c) préférence des participants et d) scores SUS pour la sélection d'expression. Les résultats de temps moyen et les taux d'erreur sont représentés avec un intervalle de confiance à 95 %.

Résultats Les variables dépendantes sont le *temps de sélection* et le *taux d'erreur* pour les mesures des performances, ainsi que le questionnaire SUS [13] et les préférences pour les mesures qualitatives.

Temps de sélection Nous définissons le temps de sélection comme l'intervalle entre l'activation de la technique et la validation de la sélection de l'expression. Les essais comportant des erreurs ont été écartés de l'analyse du temps de sélection.

Nous avons appliqué une transformation de Box-Cox avec $\lambda = -0{,}75$ pour corriger les problèmes de normalité des résidus des temps de sélection [12]. Une ANOVA à mesures répétées montre un effet significatif de BLOC ($F_{3,33} = 15{,}1$, $p < 0{,}0001$, $\eta^2_G = 0{,}13$), mais pas d'interaction avec les autres facteurs, suggérant une progression d'apprentissage identique pour chaque technique. Les comparaisons par paires avec la correction de Bonferroni montrent des différences significatives entre le bloc 1 et les blocs 3 et 4, ainsi qu'entre les blocs 2 et 4. Nous supposons que ces différences sont dues à un effet d'apprentissage, donc nous avons supprimé les deux premiers blocs des analyses suivantes. Les analyses montrent aussi un effet significatif de TECHNIQUE ($F_{5,55} = 6{,}8$, $p < 0{,}0001$, $\eta^2_G = 0{,}28$). Les comparaisons par paires montrent que *RM8* (1,40 s) est significativement plus rapide à utiliser que *RM24* (1,81 s, $p = 0{,}003$) ; *ET8* (1,50 s) est significativement plus rapide que *ET24* (2,63 s, $p = 0{,}01$). Ces résultats confirment H1, mais H3 ne peut pas être confirmée par manque d'interaction significative entre les BLOCS et les TECHNIQUES. Aussi, nous n'avons pas de résultat significatif montrant que *EmoTouch* est plus rapide que les autres techniques, donc H4 ne peut pas être confirmée.

Taux d'erreur Le taux d'erreur est le pourcentage d'essais où l'expression faciale n'a pas été sélectionnée avec succès du premier coup. Le taux d'erreur est calculé et

agrégé pour chaque combinaison de PARTICIPANT, TECHNIQUE et BLOC. Le taux d'erreur global est de 6,9 %. Les données ont été pré-traitées à l'aide d'une Aligned Rank Transform (ART) pour tenir compte de leur distribution non normale [100]. Une ANOVA à mesures répétées montre un effet significatif de TECHNIQUE ($F_{2,1,23,0} = 19,0$, $p < 0,001$, $\eta^2_G = 0,58$). Les comparaisons par paires montrent que *ET8* (2,3 %), *RM24* (2,3 %), *RM8* (3,1 %) et *ET24* (3,9 %) sont significativement moins sujettes aux erreurs que *EV8* (12,2 %) et *EG8* (17,7 %). Ces résultats confirment H2.

Questionnaire SUS Une analyse de Friedman montre un effet significatif de TECHNIQUE sur le score SUS global ($\chi^2(5) = 23,6$, $p = 0,0002$).

Une analyse par paire de Wilcoxon montre que *ET8* ($M_{ET8\ S} = 95$) a un score SUS significativement plus élevé que *EG8* ($M_{EG8\ S} = 62,5$, $p = 0,016$) et *EV8* ($M_{EV8\ S} = 73,75$, $p = 0,036$).

Préférences des participants Après avoir testé toutes les techniques, les participants ont dû les classer en fonction de leur préférence. Une analyse de Friedman montre un effet significatif de TECHNIQUE sur les préférences des participants ($\chi^2(5) = 17,0$, $p = 0,0045$). Une analyse post-hoc de Wilcoxon montre que les participants préfèrent significativement *ET8* (M = 2), *ET24* (M = 2) et *RM24* (M = 3) par rapport à *EG8* (M = 5) et *EV8* (M = 5,5) ($p < 0,030$).

L'ensemble des résultats sur les préférences des participants et des scores SUS tendent à valider H5.

Discussion *EmoVoice* et *EmoGest* ont généralement des performances inférieures à celles des autres techniques. Le taux d'erreur des deux techniques est significativement plus élevé que les autres et *EmoVoice* est plus lent que *RayMoji* (8). En outre, les participants ont déclaré qu'ils sont plus susceptibles d'utiliser *EmoTouch* (8) qu'*EmoVoice* dans leurs expériences sociales de RV et ils considèrent qu'*EmoGest* est plus difficile à utiliser que toutes les autres techniques. Ils se sentent moins en confiance avec *EmoGest* qu'avec *EmoTouch* (8), et *EmoGest* et *EmoVoice* sont moins préférés que *EmoTouch* (8 et 24) et *RayMoji* (24). Selon les commentaires des participants, la précision du système de reconnaissance et la nécessité d'apprendre des mots ou des gestes ont affecté la performance globale d'*EmoVoice* et d'*EmoGest*. Deux participants ont eu des difficultés à utiliser *EmoGest* sur le pavé tactile, en raison de la forme du contrôleur et de la morphologie de leurs mains. Trois des participants ont suggéré que l'utilisation d'*EmoVoice* interrompait leur discours lorsqu'ils parlaient à d'autres personnes, en raison de l'utilisation du microphone. Enfin, deux participants ont expliqué qu'ils ne voulaient pas utiliser *EmoVoice* parce qu'ils se sentaient gênés de prononcer les mots-clés à haute voix. Tous ces résultats montrent que ces deux techniques sont moins adaptées pour être utilisées pour contrôler l'expression faciale d'un avatar.

Les résultats indiquent également que l'affichage de seulement 8 expressions dans l'interface réduit le temps de sélection pour *EmoTouch*, par rapport à l'affichage de 24 puisque l'utilisateur prend moins de temps pour trouver l'expression souhaitée. Malgré

ces résultats, aucune autre différence significative n'a été trouvée entre les techniques avec 24 expressions et leurs équivalents avec 8 expressions. Les commentaires informels des participants montrent que pour *EmoTouch*, l'un d'entre eux préférerait avoir plus de choix, et deux autres ont trouvé qu'avoir moins de choix facilitait la réalisation de la tâche. Pour *RayMoji*, un participant aurait préféré avoir plus d'expressions, mais 3 autres auraient préféré en avoir moins. D'après ces résultats, le nombre et l'ensemble des expressions affichées pour une technique devraient être définis en fonction des besoins des applications cibles.

L'analyse statistique n'a pas montré de différence significative entre *RayMoji* et *EmoTouch*, mais les retours informels des participants nous ont permis d'identifier les caractéristiques des deux techniques qu'ils ont appréciées ou non. Pour *EmoTouch*, trois participants ont trouvé utile l'organisation des expressions sur le menu circulaire et l'utilisation des couleurs, même si trois participants ont trouvé que certaines couleurs étaient incohérentes et nécessitaient quelques ajustements. Un participant a apprécié que *EmoTouch* prenne peu de place dans son champ de vision, comparé à *RayMoji*. L'inconvénient de *EmoTouch*, selon les participants, est qu'ils doivent regarder la manette en dessous de leur champ de vision, ou lever la manette pour pouvoir voir le menu circulaire. *RayMoji* n'a pas ce problème, puisque le menu de la grille apparaît devant l'utilisateur, et que ce dernier pointe le menu avec la manette et un rayon virtuel. De plus, l'un des participants ne pouvait pas atteindre la partie supérieure du pavé tactile de la manette, en raison de ses petites mains, ce qui rendait *EmoTouch* difficile à utiliser. L'utilisation de *Raycasting* dans *RayMoji* permet d'éviter ce problème.

Tous ces retours montrent que les participants préfèrent un menu coloré, organisé sur la base de la roue des émotions de Plutchik, plutôt qu'un menu en grille sans organisation spécifique. Cependant, l'utilisation du touchpad pour indiquer une expression, et la position du menu au-dessus du contrôleur, sont des facteurs limitants pour *EmoTouch*. Au contraire, l'utilisation de *Raycasting* et le menu apparaissant devant l'utilisateur sont de meilleurs choix pour la sélection d'une expression.

FIGURE III.14 – Illustration de *EmoRay*. Le menu circulaire apparait en face de l'utilisateur, avec le PuppetFace au-dessus. L'utilisateur navigue dans le menu avec un *Raycasting*.

Enfin, les commentaires des participants nous ont aidés à concevoir une nouvelle technique que nous avons appelée *EmoRay*, qui combine les forces de *RayMoji* et *EmoTouch*. Elle utilise le menu circulaire de *EmoTouch* avec le *Raycasting* de *RayMoji*. Le

menu est placé plus près de l'utilisateur (de 1 m à 0,6 m) et sa taille est réduite pour prendre moins de place dans le champ de vision. Le PuppetFace est déplacé depuis le dessus du contrôleur vers le dessus du menu, afin que l'utilisateur n'ait pas à regarder le contrôleur pour prévisualiser l'expression pointée. Le placement du menu est basé sur la direction du rayon lorsque le menu est activé, plutôt que sur la direction du HMD, afin d'éviter que le menu ne bloque le centre du champ de vision.

III.2.2 Comparaison des techniques pour le contrôle de l'intensité

Dans cette deuxième expérience, nous comparons les techniques de contrôle non isomorphiques de l'intensité de l'expression, présentées dans la section III.1.4. Nous émettons l'hypothèse que les participants préfèrent utiliser la *gâchette* (H1), car elle permet de valider le choix de l'expression et de contrôler son intensité avec la même entrée. Nous avons également émis l'hypothèse que la technique consistant à *secouer la manette* pourrait être plus fatigante à utiliser et que les participants préféreraient donc les autres techniques (H2). Le matériel expérimental était le même que lors de la première expérience.

Tâche Les participants étaient assis sur une chaise, tout comme leur avatar dans l'environnement virtuel, avec un miroir virtuel en face d'eux pour voir le visage de leur avatar. Le visage de l'avatar est celui décrit dans la section III.1.2, et montrait l'expression du visage que l'utilisateur ajustait avec la technique d'interaction testée. Dans cette expérience, nous avons utilisé *EmoRay* pour la sélection de l'expression faciale.

Pour chaque technique testée, l'expérimentateur expliquait au participant comment l'utiliser. Ensuite, les participants étaient invités à tester la technique à leur convenance. Les participants ont été encouragés à exprimer leur pensée à haute voix, en particulier pour comparer les techniques. Chaque technique était testée pendant 3 minutes.

Méthodologie Dix participants ont pris part à cette expérience (4 femmes, âge moyen = 26,7, σ = 1,75). Nous avons utilisé un protocole intra-sujet, avec comme facteur la TECHNIQUE. Les techniques sont la *Gâchette*, l'*Orientation* de la manette, la *Secousse* de la manette, l'*Élastique* virtuel et le *menu circulaire*. Nous avons utilisé un carré latin pour équilibrer l'ordre des techniques entre les participants.

Le protocole expérimental intra-sujet de l'expérience était donc : 10 participants × 5 TECHNIQUES = 50 sessions de test au total. L'expérience durait environ 45 minutes par participant.

Résultats Nous avons demandé aux participants de classer les techniques testées en fonction de leur préférence. Une analyse de Friedman montre un effet significatif de TECHNIQUE sur les préférences des participants ($\chi^2(2) = 8{,}13$, $p = 0{,}017$). Une analyse post-hoc de Wilcoxon montre que les participants préfèrent significativement *Élastique* (M = 1,5), *Gâchette* (M = 2,5) et *Orientation* (M = 3) à *Secousse* (M = 5) ($p < 0{,}049$). L'analyse montre aussi qu'ils préfèrent *Élastique* au *Menu circulaire* (M = 3,5, $p = 0{,}020$).

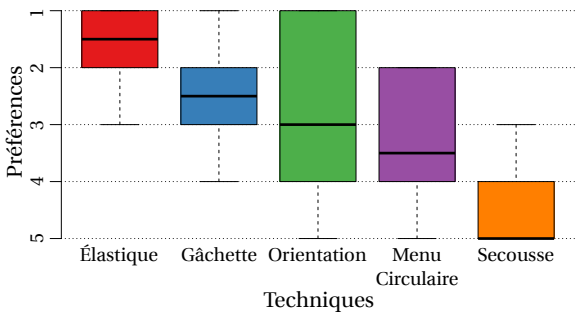


FIGURE III.15 – Préférence des participants pour l'expérience sur le contrôle d'intensité. L'axe des ordonnées du graphique est inversé : une valeur de 1 (en haut) représente un classement en première place.

Discussion Secouer la manette est la technique la moins appréciée. Bien que certains participants aient trouvé cette technique amusante à utiliser, sept d'entre eux ont déclaré que cette technique manquait de précision et quatre participants ont trouvé cette technique fatigante. Les résultats, ainsi que les commentaires des participants, tendent à confirmer H2. Trois participants pensent également qu'elle ne convient qu'à quelques expressions, comme *très énervé* 🤬 ou *rire très fort* 🤣, et qu'elle est perturbante à utiliser pour d'autres expressions.

Bien que la *Gâchette* soit préférée par rapport à la *Secousse* de la manette, celle-ci n'a pas le plus haut classement médian. De ce fait, nous ne pouvons pas confirmer H1. Cinq participants ont trouvé que la *Gâchette* permet un contrôle précis de l'intensité, mais 4 autres ont indiqué que la précision n'était pas suffisante. Deux participants ont aimé la présence d'une temporisation lors du relâchement de la gâchette par erreur, étant donné qu'elle évite l'arrêt instantané du contrôle de l'intensité. Au contraire, deux participants ont plutôt remarqué que la présence d'une temporisation ne permettait pas de quitter le contrôle d'intensité assez rapidement.

Le retour haptique a été implémenté dans toutes les techniques sauf celle où l'utilisateur secoue la manette. Certains participants ont remarqué et trouvé utile la présence d'un tel retour lors de l'utilisation des techniques (4 participants pour l'*Orientation*, 3 pour la *Gâchette*, 2 pour l'*Élastique* et 2 pour le *Menu circulaire*).

Dans l'ensemble, les participants ont préféré contrôler l'intensité avec l'*Élastique*, avec un rang médian de 1,5. Cinq participants ont apprécié le retour visuel de l'élastique virtuel. De plus, quatre participants ont déclaré que cette technique permet un contrôle précis de l'intensité. Enfin, quatre des participants ont remarqué que la ligne virtuelle était une analogie avec un réel élastique (le nom de la technique n'avait pas été révélé au participant), probablement en raison du fait que la ligne blanche est plus fine lorsqu'elle est plus longue, et aussi grâce au retour haptique.

Par conséquent, nous décidons de conserver la technique de l'*Élastique virtuel* pour contrôler l'intensité de l'expression, avec *EmoRay* pour la sélection de l'expression. La combinaison de ces deux techniques est appelée *EmoRayE* dans ce qui suit.

III.3 Étude sur l'utilisation de *EmoRayE* dans un contexte écologique

Cette dernière expérience a été conçue pour évaluer *EmoRayE* dans un contexte écologique, similaire aux applications sociales de réalité virtuelle. Les participants devaient contrôler leur expression faciale pendant qu'ils discutaient avec l'expérimentateur. Nous avons étudié les hypothèses suivantes :

(H1) Notre technique est facile à utiliser.

(H2) Elle est agréable à utiliser.

(H3) La technique ne dérange pas les utilisateurs lorsqu'ils parlent.

(H4) Elle n'interrompt pas les utilisateurs lorsqu'ils parlent.

(H5) La technique ne dérange pas les utilisateurs lorsqu'ils écoutent quelqu'un.

III.3.1 Matériel

Le participant et l'expérimentateur se trouvaient dans des pièces différentes pendant l'expérience. Le participant portait un casque VR Valve Index et tenait deux manettes [93], une dans chaque main. L'expérimentateur portait un casque VR HTC Vive et tenait également deux manettes [41]. Chaque casque utilisait un ordinateur dédié. L'application expérimentale a été codée en C# avec Unity 3D, le plugin OpenVR et l'API réseau Mirror [62]. L'instance de l'application sur l'ordinateur du participant servait d'hôte et de client. L'ordinateur de l'expérimentateur était connecté au serveur par un câble Ethernet, afin de minimiser la latence et de maximiser la stabilité de la connexion. Bien que cette configuration ne représente pas une connexion Internet courante, elle minimise le risque de déconnexion et garantit des conditions expérimentales homogènes entre les participants. L'audio était transmis en temps réel sur le réseau.

III.3.2 Tâche

Les participants étaient assis sur une chaise. Leur avatar était également assis sur un fauteuil dans l'espace virtuel. Ils étaient devant un miroir virtuel, qui leur permettait de se voir. Les participants ont incarné le visage de l'avatar décrit dans la section III.1.2, ainsi qu'un corps complet comme illustré dans la figure III.16.

Le visage et les mains de l'avatar bougeaient en fonction du mouvement de l'utilisateur, en utilisant la cinématique inverse. Les hanches et les jambes de l'avatar étaient fixées dans la scène. L'expression du visage de l'avatar changeait en fonction de l'utilisation d'*EmoRayE*. Les avatars du participant et de l'expérimentateur se faisaient face à 3 m de distance.

La conversation était semi-dirigée, les deux personnes pouvaient donc parler de n'importe quel sujet, comme dans toute situation sociale. Cependant, l'expérimentateur avait préparé des sujets de discussion spécifiques amenant les participants à utiliser des expressions faciales. Par exemple, l'expérimentateur a interrogé les participants sur les plats qu'ils aiment ou non (😊, 😋, 🤢, ...), sur ce qu'ils pensent d'un événement sportif

FIGURE III.16 – La scène virtuelle de l'expérience écologique.

récent (😀, 😢, 😠, ...) ou sur leur travail (🤔). Au cours de la discussion, le participant et l'expérimentateur devaient contrôler leur expression faciale, pour appuyer leurs propres propos ou pour réagir à ce que l'autre disait.

III.3.3 Méthodologie

Neuf participants ont pris part à cette expérience (4 femmes, âge moyen = 25,8, σ = 2,8). Trois d'entre eux n'avaient aucune expérience en réalité virtuelle. La technique testée était *EmoRayE*, la combinaison entre *EmoRay* pour la sélection de l'expression, et l'*élastique virtuel* pour contrôler l'intensité de l'expression.

Au cours de l'expérience, les participants sont passés par trois phases. Tout d'abord, l'expérimentateur a expliqué au participant comment utiliser *EmoRayE*, tandis que le participant a pu utiliser la technique pour explorer ses capacités. Dans la deuxième phase, le participant et l'expérimentateur ont participé à une discussion semi-dirigée, comme décrit dans la section précédente. Enfin, le participant a rempli un questionnaire papier. L'expérience a duré environ 25 minutes par participant.

III.3.4 Résultats

Nos variables dépendantes sont le nombre de fois où les participants ont utilisé *EmoRayE*, la durée de la conversation, le nombre d'expressions différentes qu'ils ont utilisées, les questionnaires SUS [13] et AttrakDiff [39], ainsi que des questions supplémentaires avec des échelles de Likert.

La conversation a duré en moyenne 10 min et 25 s (σ = 1min 56s) et les participants ont utilisé *EmoRayE* en moyenne 26,1 fois (σ = 14,3), soit 2,5 fois par minute (σ = 1,3).

Questionnaire SUS Le score SUS global moyen était de 80,6 (σ = 14,2). Cela correspond au rang *A* d'après le classement par percentile des résultats du SUS de Sauro (≥ 90^e percentile) [85], montrant une très bonne utilisabilité et confirmant H1.

Questionnaire AttrakDiff *EmoRay* a été classé au-dessus de la moyenne (Qualité pragmatique : moyenne = 0,71, CI ±0,60 ; Qualité hédonique : moyenne = 0,86, CI = 0,43). Par conséquent, *EmoRayE* se situe dans la zone entre « neutre » et « désiré » sur l'échelle

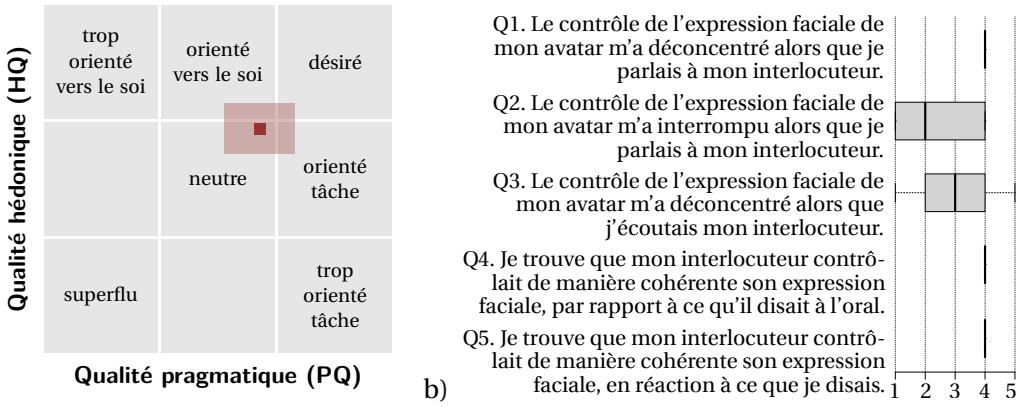


FIGURE III.17 – a) Portfolio des résultats du questionnaire AttrakDiff ; b) Diagramme en boîte des réponses aux questions utilisant l'échelle de Likert, qui vont de 1 (pas du tout d'accord) à 5 (tout à fait d'accord).

AttrakDiff (figure III.17 a). Bien qu'elle ne se situe pas dans la zone « désiré », *EmoRayE* semble globalement agréable à utiliser (H2).

Utilisation lors de la conversation Nous avons posé aux participants cinq questions sur ce qu'ils ressentaient sur l'utilisation de *EmoRayE* pendant la discussion avec l'expérimentateur. Chaque question présentait une phrase pour laquelle le participant devait indiquer son niveau d'accord ou de désaccord, en utilisant une échelle de Likert à cinq niveaux. Les phrases sont listées dans la figure III.17. Dans l'analyse, les réponses sont échelonnées de 1 (pas du tout d'accord) à 5 (tout à fait d'accord).

Les participants étaient majoritairement d'accord avec la première phrase ($M_{Q1} = 4$), nous ne pouvons donc pas confirmer que la technique ne dérange pas les utilisateurs lorsqu'ils parlent (H3). Au moins la moitié des participants n'étaient pas d'accord avec la deuxième phrase ($M_{Q2} = 2$). D'autres expériences seraient nécessaires pour confirmer clairement H4. Les participants étaient soit d'accord soit en désaccord avec la troisième phrase ($M_{Q3} = 3$), nous ne pouvons donc pas conclure sur H5 (La technique ne dérange pas les utilisateurs lorsqu'ils écoutent quelqu'un). Les participants étaient pour la plupart d'accord avec les deux dernières phrases ($M_{Q4} = 4$, $M_{Q5} = 4$), donc la perturbation et l'interruption potentielles de la discussion ne semblent pas être perceptibles pour une personne extérieure.

Discussion Pris ensemble, ces résultats suggèrent que *EmoRayE* semble facile à utiliser dans un contexte écologique et qu'elle peut perturber les utilisateurs pendant qu'ils parlent ou écoutent, mais avec des interruptions qui semblent limitées. De plus, les participants ont trouvé que l'expérimentateur contrôlait son visage de manière cohérente lorsqu'il parlait. On peut donc s'attendre à ce qu'une pratique plus poussée aide les utilisateurs à maîtriser la technique et à réduire les perturbations et les interruptions.

Les observations informelles de l'expérience montrent que quatre participants gardaient souvent le menu ouvert, et ne l'utilisaient que lorsque cela était nécessaire. Il s'agissait probablement de réduire le temps nécessaire à l'activation d'une expression, puisque le menu était déjà ouvert quand on en avait besoin.

III.4 Discussion

Les résultats des différentes expériences nous permettent de faire des recommandations pour la conception de techniques non isomorphiques de contrôle des expressions faciales. Nous avons d'abord conçu deux techniques basées sur des menus et deux autres techniques reposant sur les gestes et la voix. Selon les résultats de la première expérience, les participants ont préféré les techniques basées sur les menus. Ces techniques sont plus performantes que les deux autres, notamment en ce qui concerne le taux d'erreur qui reste élevé pour le système de reconnaissance vocale, malgré l'utilisation d'un système commercial. Même si les techniques basées sur les gestes et la reconnaissance vocale ne requièrent pas le regard de l'utilisateur, elles nécessitent d'apprendre des mots et des gestes. Par ailleurs l'utilisation de la technique reposant sur la reconnaissance vocale peut perturber le flot d'une discussion.

En ce qui concerne l'agencement visuel des techniques à base de menus, les participants ont préféré le menu circulaire au menu en grille. Il permet d'organiser les expressions en fonction des émotions qu'elles représentent. Cependant, les expressions abstraites s'intègrent difficilement dans cette disposition. Dans ce cas, une représentation en grille reste une solution appropriée.

Les participants préfèrent que les menus soient placés devant eux plutôt qu'attachés à la manette. Cette dernière condition oblige les utilisateurs à baisser les yeux vers la manette ou à la lever pour voir confortablement le menu, ce qui les oblige à détourner le regard du principal sujet d'intérêt (par exemple, une autre personne) dans l'application de réalité virtuelle. De plus, notre technique fournit un *Raycasting* pour permettre à l'utilisateur de naviguer dans le menu.

La taille des éléments du menu et la distance avec l'utilisateur ont été ajustées au cours d'études pilotes informelles, afin de trouver un équilibre entre une bonne visibilité des éléments et l'occultation du champ de vision de l'utilisateur. L'affichage du menu à 0,6 m des yeux de l'utilisateur et l'utilisation d'une taille de cibles de 9 cm pourraient être utilisés comme un bon point de départ, compte tenu de leur faible taux d'erreur. Comme attendu, le temps de sélection augmente avec le nombre d'expressions dans le menu, mais pas de manière linéaire, puisque l'augmentation du temps de sélection n'était que de 30 % lors du passage de 8 à 24 éléments pour *RayMoji*. Dans l'ensemble, les concepteurs d'application devraient privilégier un nombre réduit d'éléments dans les menus afin de réduire le temps de sélection. Cependant, un plus grand nombre d'émojis dans l'interface permet un choix plus varié d'expressions faciales.

Pour le contrôle de l'intensité de l'expression en temps réel, nous recommandons l'utilisation de l'*élastique virtuel*, mais l'utilisation de la *gâchette* ou de l'*orientation du contrôleur* sont également de bonnes alternatives. Les participants ont constaté que ces techniques permettent un contrôle plus précis de l'intensité par rapport à *secouer la manette* et au menu *circulaire*.

La technique finale que nous avons conçue, *EmoRayE*, a montré une très bonne utilisabilité dans l'étude écologique. Même si les participants ont eu l'impression que la technique pouvait les perturber pendant qu'ils parlaient, cela n'a pas été remarqué par l'expérimentateur. Une étude écologique avec deux participants qui discutent, aurait permis une mesure qualitative sur la perception de la maîtrise de la technique par un autre participant. L'ensemble des résultats suggère que la *EmoRayE* pourrait être facilement utilisée dans les applications sociales de la RV.

III.5 Conclusion

Dans ce chapitre, nous avons présenté un ensemble de techniques d'interaction non isomorphiques pour contrôler les expressions faciales, qui explore différentes parties de l'espace de conception.

Une première expérience contrôlée nous a permis de comparer leurs performances et les préférences des utilisateurs pour la sélection d'une expression. Ceci nous a permis de combiner des composants des techniques les plus performantes pour proposer *EmoRay*.

Nous avons également proposé différentes techniques pour contrôler l'intensité des expressions, qui ont été évaluées dans une autre expérience contrôlée montrant que l'*élastique virtuel* était parmi les meilleures techniques.

Les résultats de ces expériences permettent la création de *EmoRayE*, une combinaison de *EmoRay* pour la sélection de l'expression faciale, et de l'*élastique virtuel* pour le contrôle en temps réel de l'intensité. Enfin, *EmoRayE* a été évaluée dans une étude écologique, montrant une bonne utilisabilité et une perturbation minimale des participants, confirmant qu'elle peut être utilisée dans des applications sociales.

Dans la dernière expérimentation, les participants avaient le sentiment que l'utilisation de la technique les perturbait dans la discussion. C'est l'une des limites d'une technique non isomorphique, puisque les participants doivent penser activement à une expression, activer le menu, puis sélectionner et contrôler l'expression montrée aux autres utilisateurs. Les techniques isomorphiques n'ont pas cette limitation. D'autre part, comme nous l'avons montré dans la section sur les travaux connexes (section I.2.3), les techniques isomorphiques ont des limites qui sont résolues par les techniques non isomorphiques. Comme travail futur, nous aimerions donc explorer l'utilisation combinée de techniques isomorphiques et non isomorphiques, où les techniques isomorphiques seraient utilisées pour les expressions simples et les techniques non isomorphiques les complèteraient ou les remplaceraient si nécessaire. Nous prévoyons également d'évaluer l'impact de l'utilisation des expressions faciales dans des applications telles que les réunions en RV et les conférences virtuelles.

Conclusion

L'évolution des technologies de réalité virtuelle immersive ces dernières années a permis l'étude de nouvelles interactions impliquant le suivi des mouvements de l'utilisateur et des manettes proposant de nombreux degrés de liberté. L'état de l'art nous a permis d'explorer les différentes techniques couvrant différentes tâches d'interaction en réalité virtuelle : la sélection, la manipulation, le déplacement et la communication avec d'autres utilisateurs. Dans nos travaux, nous avons voulu nous concentrer sur deux pistes permettant l'amélioration des techniques de la littérature.

1 Contributions

Pour la tâche de sélection d'objets 3D, nous avons identifié les limites des techniques couramment utilisées pour la sélection d'objets en réalité virtuelle : la main virtuelle et *Raycasting*. La main virtuelle ne permet pas de sélectionner des objets hors de portée. *Raycasting* règle ce problème, mais la précision de sélection est limitée à mesure que l'objet à sélectionner est éloigné et petit. De plus, *Raycasting* ne permet de sélectionner que l'objet le plus proche de l'utilisateur, intercepté par le rayon, ce qui pose un problème d'occultation. Nous avons donc développé *RayCursor*, une nouvelle technique d'interaction pour la sélection de cibles 3D dans des environnements immersifs. Cette technique améliore *Raycasting* en utilisant différentes approches. Nous avons d'abord appliqué un filtrage sur l'orientation du rayon pour réduire les effets de tremblements de la main de l'utilisateur et les erreurs de suivi de mouvements. Une expérimentation a permis de montrer que le filtrage réduit le taux d'erreur de sélection. De plus, nous avons ajouté un curseur, que l'utilisateur peut déplacer le long du rayon, pour régler le problème d'occultation évoqué plus haut. Nous avons démontré que les fonctions de transfert comme celles utilisées pour contrôler le curseur de la souris sur les interfaces de bureau sont efficaces pour le contrôle du curseur sur le rayon. Ce curseur permet de combiner *Raycasting* avec des techniques de facilitation du pointage par proximité. Une troisième expérimentation nous a permis de comparer différents retours visuels pour la représentation de la cible la plus proche. L'affichage d'une bulle centrée sur le curseur et qui touche le bord de la cible la plus proche [32, 94], est le retour visuel typique pour la sélection par proximité. Cependant Guillon *et al.* ont montré que, dans un contexte 2D, mettre en surbrillance la cible la plus proche est plus efficace, et produit moins d'encombrement visuel [36, 35]. Nous avons étendu leurs résultats à l'interaction 3D, avec des conclusions similaires. Enfin, nous avons développé une technique hybride entre

Raycasting et *RayCursor*. Une dernière expérimentation montre qu'elle est plus rapide et moins sujette à des erreurs de sélections, par rapport à un ensemble de techniques de référence de la littérature.

Pour le contrôle d'expressions faciales, nous avons vu que les techniques isomorphiques, très explorées dans la littérature, peuvent manquer de précision [90] ou nécessiter une installation complexe. Aussi, il est impossible à l'utilisateur de contrôler une expression autre que celle de son visage. Nous avons alors proposé des techniques non isomorphiques, c'est-à-dire des techniques où l'utilisateur peut choisir manuellement l'expression du visage de son avatar en utilisant des techniques d'interaction. Ces techniques reposent sur une décomposition du contrôle d'expressions faciales en sous-tâches que nous avons formalisée dans un espace de conception : la sélection de l'expression faciale, le contrôle de son intensité et de sa durée. L'espace de conception permet d'une part de décrire les techniques existantes pour le contrôle d'expressions faciales, et d'autre part, de guider la conception de nouvelles techniques. Pour la sous-tâche de sélection d'une expression, nous avons réalisé quatre techniques. *RayMoji* et *EmoTouch* proposent des émojis qui correspondent à des expressions faciales, que l'utilisateur peut sélectionner. *RayMoji* dispose les émojis en grille, comme les menus d'émojis dans les messageries instantanées, et l'utilisateur peut faire sa sélection grâce à un *Raycasting*. Dans *EmoTouch*, les émojis sont arrangés en fonction des émotions qu'ils représentent, dans un menu circulaire navigable avec le touchpad de la manette. *EmoGest* reconnaît des gestes symboliques sur la surface tactile, qui activent les expressions correspondantes. Enfin, *EmoVoice* active des expressions sur le visage de l'avatar, en fonction de mots-clés prononcés par l'utilisateur pendant l'appui sur un bouton. Une expérience contrôlée nous a permis de concevoir une nouvelle technique combinant les avantages des techniques précédentes, que nous appelons *EmoRay*. Pour les sous-tâches de contrôle d'intensité et de la durée de l'expression, nous avons conçu cinq techniques, qui exploitent soit la pression sur la gâchette, l'inclinaison ou la secousse de la manette, la longueur d'un élastique virtuel ou la position visée sur un menu circulaire. Une deuxième expérience contrôlée a permis de déterminer que l'élastique virtuel était la technique préférée des participants. Nous l'avons donc combinée avec *EmoRay*, afin de concevoir une technique que nous appelons *EmoRayE*. Enfin, une expérience écologique a montré qu'*EmoRayE* était utilisable dans un contexte qui s'approche des réseaux sociaux virtuels.

2 Perspectives

Les projets que nous avons réalisés ouvrent différentes opportunités d'amélioration.

Tout d'abord, en 3D, la sélection précède souvent la manipulation d'un objet 3D. Une extension possible de *RayCursor* est donc son utilisation pour les tâches de manipulation. Pour le déplacement, une solution simple consiste à faire correspondre la position de l'objet à celui du curseur. En effet, un avantage secondaire et important de notre technique est sa capacité à déplacer l'objet le long du rayon une fois qu'il est sélectionné, par exemple pour le rapprocher de l'utilisateur, ce que le *Raycasting* standard ne permet

pas de faire. Pour la rotation de l'objet, l'utilisation de *RayCursor* en l'état est moins évidente. Une première solution consisterait à faire correspondre exactement l'orientation de l'objet en fonction de l'orientation de la manette. Cependant, comme certaines rotations de la manette changent la direction du rayon, l'objet étant attaché au curseur serait aussi déplacé. Cette technique ne permettrait donc pas une manipulation de la rotation qui est indépendante du contrôle de sa position. Pour régler ce problème, il faut utiliser d'autres techniques. Par exemple, il est possible d'utiliser d'autres entrées utilisateurs proposées par la manette, comme la surface tactile [19], ou la rotation de la deuxième manette pour contrôler celle de l'objet.

Ensuite, concernant le contrôle d'expressions faciales, l'expérience écologique que nous avons conduite a montré que les participants avaient le sentiment que l'utilisation de la technique les perturbait dans la discussion. C'est l'une des limites d'une technique non isomorphique, puisque les utilisateurs doivent penser activement à une expression, activer le menu, puis sélectionner et contrôler l'expression montrée aux autres utilisateurs. Les techniques isomorphiques n'ont pas ce problème. D'autre part, comme nous l'avons montré dans le chapitre état de l'art, les techniques isomorphiques ont des limites qui sont résolues par les techniques non isomorphiques. De ce fait, nous pourrions explorer la possibilité de combiner une technique isomorphique avec notre technique non isomorphique *EmoRayE*. La technique isomorphique serait utilisée pour les expressions simples que l'utilisateur peut réaliser avec son propre visage et que le système est capable de détecter. Pour les autres expressions faciales, l'utilisateur pourrait utiliser la technique non isomorphique pour contrôler manuellement l'expression souhaitée. Cette combinaison permettrait donc, d'une part, un contrôle transparent de l'expression faciale dans la plupart des situations, et d'autre part, un contrôle plus avancé comme le choix d'une expression personnalisée ou le contrôle de l'intensité. Des nouveaux casques de réalité virtuelle sortis récemment [56], faciliteraient la mise en place d'études mettant en œuvre des techniques isomorphiques, puisqu'ils intègrent les capteurs nécessaires à la détection d'expressions faciales.

Enfin, nous prévoyons également d'étudier de manière plus approfondie l'influence du contrôle des expressions faciales dans les applications sociales en RV. Pour ce faire, nous avons commencé à mettre en place une application de réunion en réalité virtuelle que nous avons appelée *VRMeet*[1]. Dans cette application, les utilisateurs incarne le même avatar que celui que nous avons utilisé lors de notre dernière expérimentation. Les utilisateurs communiquent entre eux grâce à leurs voix, aux mouvements de leurs avatars et à leurs expressions faciales. Pour le contrôle des expressions faciales, ils ont à leur disposition la technique *EmoRayE*. Cette application nous permettrait d'évaluer l'impact du contrôle des expressions faciales via notre technique, dans des environnements tels que les réunions en RV et les conférences virtuelles.

1. https://ci.inria.fr/vr-meet/job/VRMeet/

Bibliographie

[1] Karan Ahuja, Chris Harrison, Mayank Goel, and Robert Xiao. MeCap : Whole-Body Digitization for Low-Cost VR/AR Headsets. In *Proceedings of the 32Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '19, pages 453–462, New York, NY, USA, 2019. ACM.

[2] L. Almeida, E. Lopes, B. Yalçinkaya, R. Martins, A. Lopes, P. Menezes, and G. Pires. Towards natural interaction in immersive reality with a cyber-glove. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pages 2653–2658, 2019.

[3] Jessalyn Alvina, Chengcheng Qu, Joanna McGrenere, and Wendy E. Mackay. MojiBoard : Generating Parametric Emojis with Gesture Keyboards. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI EA '19, pages 1–6, Glasgow, Scotland Uk, May 2019. Association for Computing Machinery.

[4] Apple. Animoji. https://support.apple.com/en-us/HT208190. Last accessed September 28th, 2021.

[5] Ferran Argelaguet and Carlos Andujar. A survey of 3d object selection techniques for virtual environments. *Computers and Graphics*, 37(3) :121 – 136, 2013.

[6] Jean Basset, Stefanie Wuhrer, Edmond Boyer, and Franck Multon. Contact Preserving Shape Transfer For Rigging-Free Motion Retargeting. In *Motion, Interaction and Games*, MIG '19, pages 1–10, New York, NY, USA, October 2019. Association for Computing Machinery.

[7] Bernhard Bittorf and Charles Wuethrich. EmotiCon Interactive emotion control for virtual characters. In *Proceedings of the 20th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, WSCG' 2012, 2012.

[8] Renaud Blanch, Yves Guiard, and Michel Beaudouin-Lafon. Semantic pointing : Improving target acquisition with control-display ratio adaptation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 519–526, New York, NY, USA, 2004. ACM.

[9] Doug A. Bowman and Larry F. Hodges. Formalizing the Design, Evaluation, and Application of Interaction Techniques for Immersive Virtual Environments. *Journal of Visual Languages & Computing*, 10(1) :37–53, February 1999.

[10] Doug A. Bowman, Ernst Kruijff, Joseph J. LaViola, and Ivan Poupyrev. *3D User Interfaces : Theory and Practice*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 2004.

[11] Doug A Bowman, Chadwick A Wingrave, JM Campbell, VQ Ly, and CJ Rhoton. Novel uses of pinch gloves(tm) for virtual environment interaction techniques. *Virtual Reality*, 6(3) :122–129, 2002.

[12] George EP Box and David R Cox. An analysis of transformations. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 211–252, 1964.

[13] John Brooke. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194) :4–7, 1996.

[14] Jeffrey Cashion, Chadwick Wingrave, and Joseph J LaViola Jr. Dense and dynamic 3d selection for game-based virtual environments. *IEEE transactions on visualization and computer graphics*, 18(4) :634–642, 2012.

[15] Géry Casiez and Nicolas Roussel. No more bricolage ! : Methods and tools to characterize, replicate and compare pointing transfer functions. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 603–614, New York, NY, USA, 2011. ACM.

[16] Géry Casiez, Nicolas Roussel, Romuald Vanbelleghem, and Frédéric Giraud. Surfpad : Riding towards targets on a squeeze film effect. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2491–2500, New York, NY, USA, 2011. ACM.

[17] Géry Casiez, Nicolas Roussel, and Daniel Vogel. 1€ filter : A simple speed-based low-pass filter for noisy input in interactive systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 2527–2530, New York, NY, USA, 2012. ACM.

[18] Eunhee Chang, Hyun Taek Kim, and Byounghyun Yoo. Virtual Reality Sickness : A Review of Causes and Measurements. *International Journal of Human–Computer Interaction*, 36(17) :1658–1682, October 2020.

[19] Michael Chen. Two-dimensional emulation of three-dimensional trackball, May 1991.

[20] CyberGlove Systems. CyberGlove II. https://www.cyberglovesystems.com/cyberglove-ii, Last accessed September 28th, 2021.

[21] Gerwin de Haan, Michal Koutek, and Frits H. Post. Intenselect : Using dynamic object rating for assisting 3d object selection. In *Proceedings of the 11th Eurographics Conference on Virtual Environments*, EGVE'05, pages 201–209, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.

[22] William Delamare, Céline Coutrix, and Laurence Nigay. Mobile pointing task in the physical world : Balancing focus and performance while disambiguating. In *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services*, MobileHCI '13, pages 89–98, New York, NY, USA, 2013. ACM.

[23] Diane Dewez, Ludovic Hoyet, Anatole Lécuyer, and Ferran Argelaguet. Studying the Inter-Relation Between Locomotion Techniques and Embodiment in Virtual Reality. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 452–461, November 2020. ISSN : 1554-7868.

[24] Henry Been-Lirn Duh, Donald E. Parker, and Thomas A. Furness. An Independent Visual Background Reduced Simulator Sickness in a Driving Simulator. *Presence : Teleoperators and Virtual Environments*, 13(5) :578–588, October 2004.

[25] Paul Ekman and Wallace V. Friesen. *Facial action coding systems*. Consulting Psychologists Press, 1978.

[26] Niklas Elmqvist and Jean-Daniel Fekete. Semantic pointing for object picking in complex 3d environments. In *Proceedings of Graphics Interface 2008*, GI '08, pages 243–250, Toronto, Ont., Canada, May 2008. Canadian Information Processing Society.

[27] Facebook. Facebook Spaces. https://web.archive.org/web/20191005002238/https://www.facebook.com/spaces. Last accessed October 5th, 2019.

[28] Rebecca Fribourg, Ferran Argelaguet, Ludovic Hoyet, and Anatole Lécuyer. Studying the Sense of Embodiment in VR Shared Experiences. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 273–280, March 2018.

[29] Adélaïde Genay, Anatole Lécuyer, and Martin Hachet. Virtual, Real or Mixed : How Surrounding Objects Influence the Sense of Embodiment in Optical See-Through Experiences ? *Frontiers in Virtual Reality*, 2 :74, 2021.

[30] Google. Google Cardboard. https://arvr.google.com/cardboard/, Last accessed September 27th, 2021.

[31] Scott W. Greenwald, Zhangyuan Wang, Markus Funk, and Pattie Maes. Investigating social presence and communication with embodied avatars in room-scale virtual reality. In Dennis Beck, Colin Allison, Leonel Morgado, Johanna Pirker, Foaad Khosmood, Jonathon Richter, and Christian Gütl, editors, *Immersive Learning Research Network*, pages 75–90. Springer International Publishing, 2017.

[32] Tovi Grossman and Ravin Balakrishnan. The bubble cursor : Enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pages 281–290, New York, NY, USA, 2005. ACM.

[33] Tovi Grossman and Ravin Balakrishnan. The design and evaluation of selection techniques for 3d volumetric displays. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*, UIST '06, pages 3–12, New York, NY, USA, 2006. ACM.

[34] Xiaochi Gu, Yifei Zhang, Weize Sun, Yuanzhe Bian, Dao Zhou, and Per Ola Kristensson. Dexmo : An Inexpensive and Lightweight Mechanical Exoskeleton for Motion Capture and Force Feedback in VR. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 1991–1995, New York, NY, USA, May 2016. Association for Computing Machinery.

[35] Maxime Guillon, François Leitner, and Laurence Nigay. Investigating visual feedforward for target expansion techniques. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 2777–2786, New York, NY, USA, 2015. ACM.

[36] Maxime Guillon, François Leitner, and Laurence Nigay. Static Voronoi-Based Target Expansion Technique for Distant Pointing. In Franca Garzotto, Paolo Paolini, Antonella De Angeli, Giulio Jacucci, Alessio Malizia, Maristella Matera, and Rosa Lanzilotti, editors, *Proceedings of the International Working Conference on Advanced Visual Interfaces (AVI 2014)*, pages 41–48, Como, Italy, 2014. ACM.

[37] Aakar Gupta, Thomas Pietrzak, Cleon Yau, Nicolas Roussel, and Ravin Balakrishnan. Summon and select : Rapid interaction with interface controls in mid-air. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces*, ISS '17, pages 52–61, New York, NY, USA, 2017. ACM.

[38] HaptX Inc. HaptX gloves. https://haptx.com/virtual-reality/, Last accessed September 23th, 2021.

[39] Marc Hassenzahl, Michael Burmester, and Franz Koller. Attrakdiff : Ein fragebogen zur messung wahrgenommener hedonischer und pragmatischer qualität. In *Mensch & computer 2003*, pages 187–196. Springer, 2003.

[40] Ken Hinckley, Randy Pausch, John C. Goble, and Neal F. Kassell. A survey of design issues in spatial input. In *Proceedings of the 7th Annual ACM Symposium on User Interface Software and Technology*, UIST '94, pages 213–222, New York, NY, USA, 1994. ACM.

[41] HTC. HTC Vive VR headset. https://www.vive.com/us/product/vive-virtual-reality-system/, Last accessed January 7th, 2019.

[42] HTC Corporation. VIVE Tracker. https://www.vive.com/fr/accessory/vive-tracker/. Last accessed September 28th, 2021.

[43] Jennifer Hyde, Elizabeth J. Carter, Sara Kiesler, and Jessica K. Hodgins. Using an Interactive Avatar's Facial Expressiveness to Increase Persuasiveness and Socialness. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 1719–1728, New York, NY, USA, 2015. ACM. event-place : Seoul, Republic of Korea.

[44] KATVR. KAT Walk C. https://www.kat-vr.com/products/kat-walk-c, Last accessed September 28th, 2021.

[45] Regis Kopper, Felipe Bacim, and Doug A Bowman. Rapid and accurate 3d selection by progressive refinement. In *2011 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 67–74, Washington, DC, USA, March 2011. IEEE, IEEE Computer Society.

[46] Gordon Paul Kurtenbach. *The design and evaluation of marking menus*. University of Toronto Toronto, 1993.

[47] Mikko Kytö, Barrett Ens, Thammathip Piumsomboon, Gun A. Lee, and Mark Billinghurst. Pinpointing : Precise head- and eye-based target selection for augmented reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 81 :1–81 :14, New York, NY, USA, 2018. ACM.

[48] Hao Li, Laura Trutoiu, Kyle Olszewski, Lingyu Wei, Tristan Trutna, Pei-Lun Hsieh, Aaron Nicholls, and Chongyang Ma. Facial performance sensing head-mounted display. *ACM Trans. Graph.*, 34(4), July 2015.

[49] Jiandong Liang and Mark Green. Jdcad : A highly interactive 3d modeling system. *Computers & Graphics*, 18(4) :499 – 506, 1994.

[50] James Jeng-Weei Lin, Habib Abi-Rached, Do-Hoe Kim, Donald E. Parker, and Thomas A. Furness. A "Natural" Independent Visual Background Reduced Simulator Sickness. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 46(26) :2124–2128, September 2002.

[51] James Liu, Hirav Parekh, Majed Al-Zayer, and Eelke Folmer. Increasing Walking in VR Using Redirected Teleportation. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, UIST '18, pages 521–529, New York, NY, USA, 2018. ACM.

[52] J. Lugrin, D. Zilch, D. Roth, G. Bente, and M. E. Latoschik. FaceBo : Real-time face and body tracking for faithful avatar synthesis. In *2016 IEEE Virtual Reality*, VR '16, pages 225–226, March 2016.

[53] Manus VR. Prime X Marker Mocap. https://www.manus-vr.com/mocap-gloves, Last accessed September 28th, 2021.

[54] Manus VR. Xsens Gloves. https://www.manus-vr.com/xsens-gloves, Last accessed September 28th, 2021.

[55] Ali Mazalek, Sanjay Chandrasekharan, Michael Nitsche, Tim Welsh, Paul Clifton, Andrew Quitmeyer, Firaz Peer, Friedrich Kirschner, and Dilip Athreya. I'M in the Game : Embodied Puppet Interface Improves Avatar Control. In *Proceedings of the Fifth International Conference on Tangible, Embedded, and Embodied Interaction*, TEI '11, pages 129–136, New York, NY, USA, 2011. ACM.

[56] Megadodo Games. DecaGear. https://www.deca.net/decagear/, Last accessed October 11th, 2021.

[57] Albert Mehrabian and James A Russell. *An approach to environmental psychology.* the MIT Press, 1974.

[58] 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. *Computer Graphics Forum*, 38(1) :21–45, February 2019.

[59] Meta Quest. Getting started with Hand Tracking on Oculus Quest 2 and Quest. https://support.oculus.com/articles/headsets-and-accessories/controllers-and-hand-tracking/hand-tracking-quest-2, Last accessed January 7th, 2022.

[60] Microsoft. Kinect. https://developer.microsoft.com/fr-fr/windows/kinect/, Last accessed September 27th, 2021.

[61] Mark R Mine. Virtual environment interaction techniques. Technical report, University of North Carolina, Chapel Hill, NC, USA, 1995.

[62] Mirror Networking. Mirror API : Open source networking for unity. https://mirror-networking.com/, Last accessed July 12th, 2021.

[63] Mozilla. Hubs by mozilla. https://hubs.mozilla.com/. Last accessed July 15th, 2021.

[64] Mathieu Nancel, Olivier Chapuis, Emmanuel Pietriga, Xing-Dong Yang, Pourang P Irani, and Michel Beaudouin-Lafon. High-precision pointing on large wall displays using small handheld devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 831–840, New York, NY, USA, 2013. ACM.

[65] M. Obaid, R. Mukundan, M. Billinghurst, and C. Pelachaud. Expressive mpeg-4 facial animation using quadratic deformation models. In *2010 Seventh International Conference on Computer Graphics, Imaging and Visualization (CGIV '10)*, pages 9–14, 2010.

[66] Oculus VR. Oculus Rift. https://web.archive.org/web/20140919131636/http://www.oculus.com/rift/, Last accessed October 6th, 2021.

[67] Soo Youn Oh, Jeremy Bailenson, Nicole Krämer, and Benjamin Li. Let the Avatar Brighten Your Smile : Effects of Enhancing Facial Expressions in Virtual Environments. *PLoS ONE*, 11(9), 2016.

[68] Nicolas Olivier, Ludovic Hoyet, Fabien Danieau, Ferran Argelaguet, Quentin Avril, Anatole Lecuyer, Philippe Guillotel, and Franck Multon. The impact of stylization on face recognition. In *ACM Symposium on Applied Perception 2020*, SAP '20, pages 1–9, New York, NY, USA, September 2020. Association for Computing Machinery.

[69] Igor S. Pandzic and Forchheimer Robert, editors. *MPEG-4 Facial Animation : The Standard, Implementation And Applications*. John Wiley & Sons, Inc., 2003.

[70] Mathias Parger, Joerg H. Mueller, Dieter Schmalstieg, and Markus Steinberger. Human upper-body inverse kinematics for increased embodiment in consumer-grade virtual reality. In *Proceedings of the 24th ACM Symposium on Virtual Reality Software and Technology*, VRST '18, pages 1–10, New York, NY, USA, November 2018. Association for Computing Machinery.

[71] Catherine Pelachaud and Isabella Poggi. Subtleties of facial expressions in embodied agents. *The Journal of Visualization and Computer Animation*, 13(5) :301–312, 2002.

[72] Carole Plasson. *Interaction 2D/3D en réalité augmentée sur table*. Theses, Université Grenoble Alpes [2020-....], May 2021.

[73] Carole Plasson, Dominique Cunin, Yann Laurillau, and Laurence Nigay. A lens-based extension of raycasting for accurate selection in dense 3d environments. In *Human-Computer Interaction – INTERACT 2021*, pages 501–524, Cham, 2021. Springer International Publishing.

[74] Robert Plutchik. Chapter 1 - A General Psychoevolutionary Theory of Emotion. In Robert Plutchik and Henry Kellerman, editors, *Theories of Emotion*, pages 3–33. Academic Press, January 1980.

[75] Robert Plutchik. The Nature of Emotions : Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical

practice. *American Scientist*, 89(4) :344–350, 2001. Publisher : Sigma Xi, The Scientific Research Society.

[76] Henning Pohl, Dennis Stanke, and Michael Rohs. EmojiZoom : emoji entry via large overview maps. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '16, pages 510–517, Florence, Italy, September 2016. Association for Computing Machinery.

[77] Ivan Poupyrev, Mark Billinghurst, Suzanne Weghorst, and Tadao Ichikawa. The go-go interaction technique : Non-linear mapping for direct manipulation in vr. In *Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology*, UIST '96, pages 79–80, New York, NY, USA, 1996. ACM.

[78] Sharif Razzaque, Zachariah Kohn, and Mary C. Whitton. Redirected Walking. In *Eurographics 2001 - Short Presentations*. Eurographics Association, 2001.

[79] Gang Ren and Eamonn O'Neill. 3d selection with freehand gesture. *Computers & Graphics*, 37(3) :101–120, May 2013.

[80] Grégoire Richard, Thomas Pietrzak, Ferran Argelaguet, Anatole Lécuyer, and Géry Casiez. Studying the Role of Haptic Feedback on Virtual Embodiment in a Drawing Task. *Frontiers in Virtual Reality*, 1 :28, 2021.

[81] Hyocheol Ro, Seungho Chae, Inhwan Kim, Junghyun Byun, Yoonsik Yang, Yoonjung Park, and Tackdon Han. A dynamic depth-variable ray-casting interface for object manipulation in ar environments. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2873–2878. IEEE Systems, Man, and Cybernetics Society, Oct 2017.

[82] Rec Room. Rec Room. https://recroom.com/. Last accessed July 15th, 2021.

[83] RootMotion. Final IK. https://assetstore.unity.com/packages/tools/animation/final-ik-14290. Last accessed September 28th, 2021.

[84] Fiorella de Rosis, Catherine Pelachaud, Isabella Poggi, Valeria Carofiglio, and Berardina De Carolis. From Greta's mind to her face : modelling the dynamics of affective states in a conversational embodied agent. *International Journal of Human-Computer Studies*, 59(1) :81–118, July 2003.

[85] Jeff Sauro. *A practical guide to the system usability scale : Background, benchmarks & best practices.* Measuring Usability LLC, 2011.

[86] Greg Schmidt, Yohan Baillot, Dennis G. Brown, Erik B. Tomlin, and J. Edward II Swan. Toward disambiguating multiple selections for frustum-based pointing. In *Proceedings of the 3D User Interfaces*, 3DUI '06, pages 87–94, Washington, DC, USA, March 2006. IEEE Computer Society.

[87] Clemens Sielaff. EmoCoW : An Interface for Real-time Facial Animation. In *ACM SIGGRAPH ASIA 2010 Sketches*, SA '10, pages 40 :1–40 :2, New York, NY, USA, 2010. ACM.

[88] Peng Song, Wooi Boon Goh, William Hutama, Chi-Wing Fu, and Xiaopei Liu. A handle bar metaphor for virtual object manipulation with mid-air interaction. In

Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '12, pages 1297–1306, New York, NY, USA, May 2012. Association for Computing Machinery.

[89] Frank Steinicke, Timo Ropinski, and Klaus Hinrichs. Object selection in virtual environments with an improved virtual pointer metaphor. In *Computer Vision and Graphics : International Conference, ICCVG 2004*, pages 320–326, Warsaw, Poland, 2004. Springer Netherlands.

[90] Katsuhiro Suzuki, Fumihiko Nakamura, Jiu Otsuka, Katsutoshi Masai, Yuta Itoh, Yuta Sugiura, and Maki Sugimoto. Facial Expression Mapping Inside Head Mounted Display by Embedded Optical Sensors. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16 Adjunct, pages 91–92, New York, NY, USA, 2016. ACM.

[91] Ultraleap Limited. Leap Motion Controller. https://www.ultraleap.com/leap-motion-controller-overview/, Last accessed September 23th, 2021.

[92] Unicode. Unicode® Emoji Charts v13.0. https://unicode.org/emoji/charts/. Last accessed April 21st, 2020.

[93] Valve. Valve Index. https://www.valvesoftware.com/fr/index/, Last accessed October 15th, 2021.

[94] Lode Vanacken, Tovi Grossman, and Karin Coninx. Exploring the effects of environment density and target visibility on object selection in 3d virtual environments. In *2007 IEEE Symposium on 3D User Interfaces*, Washington, DC, USA, March 2007. IEEE Computer Society.

[95] Donald Lee Vickers. *Sorcerer's Apprentice : Head-mounted Display and Wand*. PhD thesis, University of Utah, 1972.

[96] Hannes Högni Vilhjálmsson. *Avatar Augmented Online Conversation*. PhD thesis, Massachusetts Institute of Technology, 2003.

[97] Daniel Vogel and Ravin Balakrishnan. Distant freehand pointing and clicking on very large, high resolution displays. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*, UIST '05, pages 33–42, New York, NY, USA, 2005. ACM.

[98] VRChat Inc. VRChat. https://vrchat.com/. Last accessed July 15th, 2021.

[99] Thibaut Weise, Sofien Bouaziz, Hao Li, and Mark Pauly. Realtime Performance-based Facial Animation. In *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, pages 77 :1–77 :10, New York, NY, USA, 2011. ACM.

[100] Jacob O. Wobbrock, Leah Findlater, Darren Gergle, and James J. Higgins. The aligned rank transform for nonparametric factorial analyses using only anova procedures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 143–146, New York, NY, USA, 2011. ACM.

[101] Jacob O. Wobbrock, Andrew D. Wilson, and Yang Li. Gestures without libraries, toolkits or training : A $1 recognizer for user interface prototypes. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*,

UIST '07, page 159–168, New York, NY, USA, 2007. Association for Computing Machinery.

[102] Robert C. Zeleznik, Andrew S. Forsberg, and Paul S. Strauss. Two pointer input for 3D interaction. In *Proceedings of the 1997 symposium on Interactive 3D graphics*, I3D '97, pages 115–120, New York, NY, USA, April 1997. Association for Computing Machinery.

[103] Anthony Zhang. SpeechRecognition. https://pypi.org/project/SpeechRecognition/. Last accessed July 15th, 2021.

Annexe A

Caractéristiques techniques des manettes HTC Vive et Valve Index

Nous avons principalement utilisé dans nos travaux les casques de réalité virtuelle HTC Vive [41] et Valve Index [93]. Dans cette annexe, nous décrivons les caractéristiques des manettes qui accompagnent ces deux casques, pour faciliter la compréhension des techniques d'interaction que nous avons développées.

Les manettes du HTC Vive et du Valve Index restituent les 6 degrés de liberté qui correspondent à leurs positions et à leurs orientations dans l'espace de suivi. Ils se repèrent grâce à des émetteurs infrarouges présents dans la pièce, et de multiples capteurs infrarouges présentes dans la structure des manettes. En plus de ces caractéristiques, ces contrôleurs fournissent des entrées supplémentaires, qui sont décrites dans les sections suivantes.

A.1 HTC Vive

Les manettes HTC Vive sont interchangeables, puisqu'elles sont symétriques et que leur association gauche/droite est déterminée au démarrage de l'application, en fonction de leurs positions dans l'espace. Les commandes possibles sont illustrées figure A.1, et détaillées ci-dessous :

Pavé tactile La manette est pourvue d'une surface tactile circulaire d'un rayon de 2 cm. Elle peut être dans trois états différents :

— *Relâché* lorsque l'utilisateur ne touche pas la surface tactile.

— *Touché* quand il le touche sans appuyer. Le système peut alors accéder aux coordonnées de contact du doigt sur le pad.

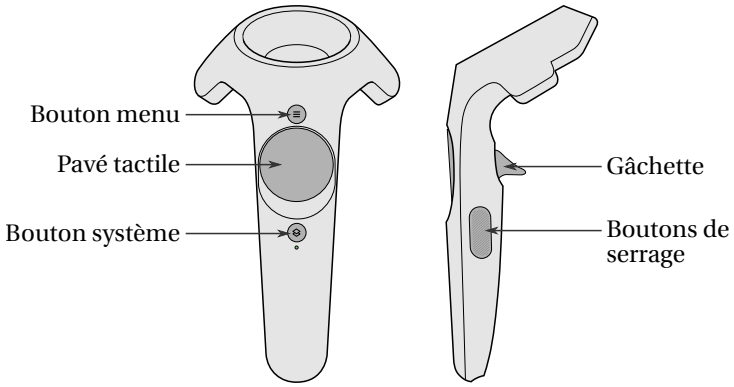


FIGURE A.1 – Entrées utilisateur sur les manettes HTC Vive.

— *Appuyé* quand l'utilisateur appuie plus fortement sur la surface. L'utilisateur peut ressentir l'activation de cet état grâce à un retour haptique et auditif produit par un bouton-poussoir derrière le pad tactile.

Le système a donc accès à quatre degrés de liberté pour le pad tactile : une valeur discrète (booléenne) indiquant si le doigt touche la surface, une autre indiquant s'il appuie fortement, et les deux coordonnées de la position du doigt entre -1 (bas et gauche) et 1 (haut et droite). La surface tactile ne permet pas la détection de plusieurs points de contact.

Gâchette La gâchette est une entrée élastique dont l'amplitude d'appui est transmise au système sous forme d'une valeur continue entre 0 (entièrement relâchée) et 1 (complètement appuyée). Lorsque l'utilisateur appuie complètement, un retour haptique et auditif est produit par un bouton-poussoir.

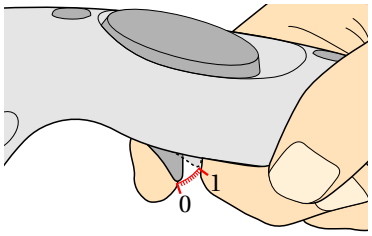


FIGURE A.2 – La gâchette sur la manette HTC Vive. La gâchette est une entrée élastique qui donne une valeur entre 0 et 1.

Boutons de serrage La manette est équipée de deux boutons de serrage, que l'utilisateur actionne en serrant la manette avec les 3 derniers doigts de sa main. Bien qu'il y a deux boutons à la surface de la manette, ceux-ci n'actionnent en réalité qu'un seul bouton-poussoir interne. De ce fait, ces boutons ne correspondent qu'à une seule entrée discrète.

Boutons menu et système Enfin, ces deux boutons sont de simples entrées discrètes sous forme de bouton-poussoir. Le bouton menu est exploité par l'application en cours d'utilisation. Le bouton système est utilisé pour afficher le menu SteamVR, et ne peut donc pas être exploité par l'application de réalité virtuelle.

A.2 Valve Index

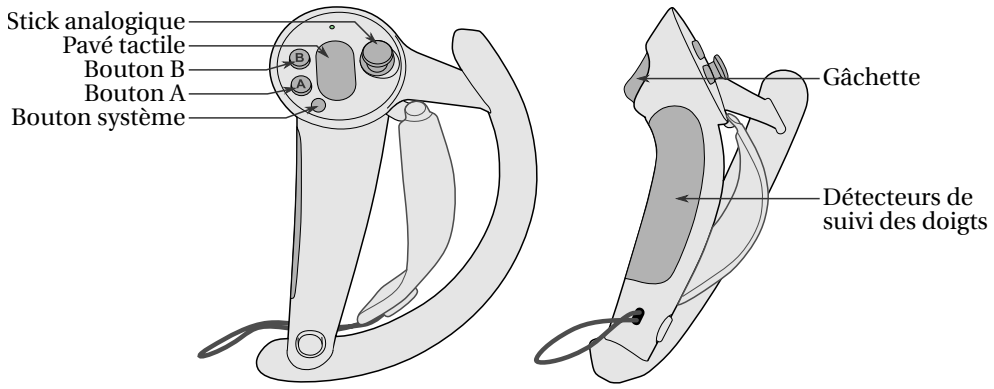


FIGURE A.3 – Entrées utilisateur sur la manette droite du Valve Index.

Les manettes du Valve Index sont, contrairement au HTC Vive, différentes pour les deux mains. Elles sont équipées d'une bande de tissu ajustable permettant de serrer la main contre la manette. Ceci permet à l'utilisateur d'ouvrir complètement sa main sans que la manette ne tombe.

Stick analogique Le stick analogique ou *thumbstick* est une entrée élastique continue à deux degrés de liberté. Le stick est stable en son centre, c'est-à-dire aux coordonnées $(0,0)$. Il peut être déplacé par l'utilisateur, généralement avec son pouce, dans la direction souhaitée. Les valeurs envoyées au système sont donc les coordonnées (x, y) du stick par rapport au centre. L'utilisateur peut aussi appuyer sur le stick : ceci produit un retour haptique et auditif grâce par un bouton-poussoir, et un degré de liberté discret supplémentaire pour le système.

Pavé tactile La surface tactile de la manette Valve Index est de forme oblongue, de 17 mm de large et 30 mm de haut. Elle fonctionne de manière similaire à celle du HTC Vive, sauf que l'état *appuyé* n'est pas déclenché par un bouton-poussoir, mais par un capteur de force. Le retour haptique et auditif est donc fourni par le vibreur de la manette.

Gâchette La gâchette est une entrée élastique dont l'amplitude d'appui est transmise au système sous forme d'une valeur continue entre 0 (entièrement relâchée) et 1 (complètement appuyée). Lorsque l'utilisateur appuie jusqu'au bout, un retour haptique et auditif est produit par un bouton-poussoir.

Détecteurs de suivi des doigts Ces capteurs capacitifs permettent de détecter l'amplitude d'ouverture de chacun des trois derniers doigts de la main.

Boutons A, B et système Enfin, ces trois boutons sont de simples entrées discrètes sous forme de boutons-poussoirs. Les boutons A et B sont exploités par l'application en cours d'utilisation. Le bouton système est utilisé pour afficher le menu SteamVR, et ne peut donc pas être exploité par l'application de réalité virtuelle.