Adoiraccourcix : Sélection de Commandes sur Écrans Tactiles Multi-Points par Identification des Doigts

Alix Goguey[1], Géry Casiez[2], Thomas Pietrzak[2], Daniel Vogel[3] & Nicolas Roussel[1]
[1]Inria Lille, [2]Université Lille 1, France, [3] Université de Waterloo, Canada
{alix.goguey, nicolas.roussel}@inria.fr, {gery.casiez, thomas.pietrzak}@lifl.fr,
dvogel@uwaterloo.ca

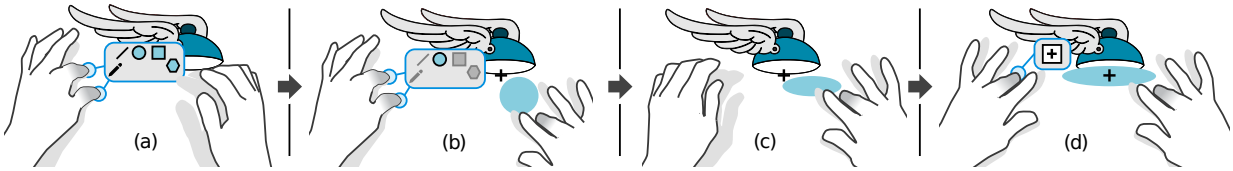

Figure 1: Illustration partielle de la logique d'Adoiraccourcix : (a) un accord de doigts de la main non dominante affiche du *feedforward* précisant les différentes commandes associées aux doigts de la main dominante, pour cet accord. Ici le pouce et l'index affichent les cinq outils de dessin associés aux doigts de la main dominante ; (b) le majeur de la main dominante sélectionne l'outil de création d'ellipses ; (c) le mouvement de la main dominante contrôle les paramètres de la commande, ici la taille et la forme de l'ellipse ; (d) un autre accord de la main non dominante définit une contrainte pour la commande en train d'être exécutée. Ici le majeur de la main non-dominante contraint la création d'ellipse suivant son centre plutôt que son coin supérieur gauche.

RÉSUMÉ

Alors que les raccourcis clavier permettent aux utilisateurs experts des interfaces WIMP d'exécuter rapidement des commandes, il existe aujourd'hui peu d'équivalents pour grands écrans tactiles multi-points. Afin de remédier à ce problème, nous avons conçu Adoiraccourcix, une technique d'interaction tirant parti de l'identification des doigts sur tables interactives, pour la sélection rapide de commandes associée au contrôle continu de paramètres de celles-ci. Après avoir présenté la logique de conception, nous illustrons Adoiraccourcix au travers d'une application de dessin vectoriel. Nous terminons par la présentation d'études préliminaires dans lesquelles nous comparons une mise en œuvre d'Adoiraccourcix à une interface classique. Les premiers résultats suggèrent qu'une fois maîtrisé, Adoiraccourcix permet d'interagir plus efficacement que les interfaces classiques.

Mots Clés

Multi-points ; identification des doigts ; raccourcis ; sélection de commandes ; manipulation directe

ACM Classification Keywords

H.5.2 Information Interfaces and Presentation (*e.g.* HCI): User interfaces

INTRODUCTION

La sélection de commandes, qui permet à l'utilisateur de réaliser ses tâches en agissant sur des objets virtuels, constitue l'un des aspects essentiels des systèmes interactifs. Elle s'effectue généralement à l'aide de menus et de barres d'outils, mais aussi via des raccourcis clavier qui permettent un accès rapide et facile aux commandes les plus usuelles. Si de tels raccourcis sont communs dans les interfaces WIMP traditionnelles, il existe aujourd'hui peu d'équivalents pour l'interaction sur grands écrans multi-points .

Les boutons physiques étant rares sur les dispositifs tactiles, les concepteurs d'interfaces ont généralement recours à des boutons graphiques, à des techniques prenant en compte le nombre de doigts, les gestes effectués ou encore la temporisation, pour déclencher différentes commandes. Ces approches ne permettent cependant d'accéder qu'à un nombre relativement limité de commandes, elles occupent de la place sur l'écran et peuvent aussi poser des problèmes de performance et de fiabilité.

Pour pallier le manque de techniques d'accès rapide à un grand nombre de commandes sur les écrans tactiles ainsi que le permettent les raccourcis clavier, nous proposons Adoiraccourcix, une technique d'interaction tirant parti de l'identification des doigts en contact avec l'écran qui permet, dans une même et unique interaction, le déclenchement rapide de commandes *et* le contrôle continu des paramètres de celles-ci (Figure 1).

Après une présentation de l'état de l'art sur l'identification des doigts et les systèmes de raccourcis, nous détaillons la logique de conception d'Adoiraccourcix. Nous l'illustrons dans une application de dessin vectoriel, puis nous présentons une expérience utilisateurs préliminaire com-

parant une interface traditionnelle à base de boutons graphiques à Adoiraccourcix.

ÉTAT DE L'ART

Nous commençons par présenter les techniques d'identification des doigts en contact avec une surface tactile et les techniques d'interaction qui utilisent cette information. Nous discutons ensuite des avantages de l'interaction bi-manuelle et de la manipulation directe avant de nous concentrer sur les raccourcis clavier et les différentes techniques de sélection de commandes.

Identification des doigts

Les surfaces tactiles utilisant les technologies capacitives et résistives peuvent non seulement fournir la position des contacts mais aussi la pseudo-pression [7] (surface en contact) et l'orientation [25]. Celles qui utilisent une technologie optique, peuvent aussi fournir des informations sur la forme des points de contact [27] ou sur l'utilisateur, la main ou la partie de la main en contact, en ayant recours à des marqueurs [22]. La plupart des techniques d'identification des doigts utilisent la vision par ordinateur.

Les travaux explorant l'identification des doigts ont principalement utilisé cette information pour associer différentes commandes à différents doigts. Sugiura *et al*. utilisent un scanner d'empreintes digitales pour identifier l'index, le majeur, l'annulaire et auriculaire. Ils utilisent ces doigts pour contrôler un lecteur de musique et ajouter des favoris dans un navigateur Internet [23]. En plaçant un électromyogramme sur l'avant-bras, Benko *et al*. identifient chaque doigt en contact avec une précision de 90% [4]. Dans une application de dessin, ils associent une couleur à chaque doigt. Plutôt que de simplement considérer l'utilisation individuelle des doigts, d'autres travaux se sont intéressés à la réalisation de plusieurs contacts simultanément, également appelés accords dans ce qui suit.

Sur des écrans multi-points, Kin-Chung Au *et al*. identifient les doigts en analysant les angles formés par la position des contacts et leur barycentre [1]. Pour correctement identifier les doigts, il est nécessaire de placer les doigts au préalable "*naturellement courbés*" sur l'écran. Dès lors que cinq contacts simultanés sont détectés et les doigts identifiés, un *Palm menu* composé de boutons associés à chaque doigt est alors affiché en dessous du contact correspondant au doigt et l'utilisateur peut ensuite presser le bouton associé à la commande désirée. L'utilisateur peut exécuter une commande à l'aide d'un doigt ou d'un accord. Ce menu a montré de meilleures performances qu'un menu contextuel et une barre d'outils.

Au lieu d'identifier les doigts, *Arpège* utilise des positions prédéfinies de doigts pour reconnaître des accords [8]. Les accords sont formés progressivement à l'aide de *feedforward*[1], ce qui aide à sélectionner la commande voulue. Les noms des commandes possibles sont affichés près des doigts et sont mises à jour à chaque changement d'accord. Les commandes sont exécutées quand tous les doigts quittent la surface. Les auteurs ont aussi montré que les utilisateurs sont capables d'exécuter facilement jusqu'à 480 accords. Lepinski *et al*. peuvent reconnaître précisément 14 accords sur une table interactive. Ils ont utilisé ces accords pour créer un *marking menu* tactile [18]. Les utilisateurs effectuent un accord puis déplacent leurs doigts dans une direction pour exécuter une commande.

Au lieu de simplement déclencher des commandes, les accords peuvent être utilisés pour manipuler des outils. Malik *et al*. utilisent deux caméras en hauteur pour identifier les doigts touchant un fond noir [21]. Chaque doigt ou accord peut être associé à une commande et en contrôler les paramètres. Dans leur exemple ils utilisent ce système pour interagir à distance avec un grand écran, par exemple pour zoomer et contrôler le niveau de grossissement. En utilisant un gant équipé de marqueurs, Marquardt *et al*. identifient le bout des doigts, les jointures et d'autres parties de la main [22]. Les utilisateurs peuvent associer un outil ou une commande à chaque marqueur. Ils peuvent ainsi dessiner avec le bout de leur index et avoir un aperçu de l'outil associé en posant la jointure de leur index.

Holz et Baudisch identifient les utilisateurs à l'aide d'un système reconnaissant les empreintes digitales des doigts en contact avec une table interactive [13]. En considérant que chaque doigt a une empreinte digitale unique, cette technique pourrait être utilisée pour identifier les doigts.

En résumé, différentes approches ont été proposées pour identifier les doigts posés sur une surface interactive et déclencher différentes commandes suivant les configurations détectées. Certaines approches pourraient être suffisamment fiables et pratiques dans un avenir proche. Les techniques d'interaction jusqu'ici proposées se sont concentrées sur l'utilisation de doigts seuls ou d'accords pour déclencher des commandes ou manipuler des outils. La découverte des fonctions associées aux doigts et accords est parfois facilitée par du *feedforward*. Aucun de ces travaux de la littérature n'a cependant proposé d'utiliser le *feedforward* dans des techniques d'interaction qui combinent à la fois la sélection de commande et la manipulation directe.

Interaction bi-manuelle et manipulation directe

Identifier les doigts et les mains en contact avec une grande surface permet de bénéficier de la richesse de l'interaction bi-manuelle. Bier *et al*. proposent par exemple le concept de *Toolglass* où la main non dominante manipule une palette transparente de concert avec la main dominante qui contrôle le curseur de la souris [5]. Cliquer avec la souris sur un outil applique la commande correspondante (*e.g.* une couleur) à l'objet d'intérêt. Dans une expérience comparant les *Toolglasses* à d'autres techniques uni-manuelles Guimbretière *et al*. montrent que la force des *Toolglasses* provient de la combinaison de la sélection de commande et de la manipulation directe. En effet, l'utilisateur peut déclencher une commande et enchaîner avec de la manipulation directe dans un même geste [10]. Vogel et Casiez approfondissent ce concept avec Conté, un crayon digital qui produit différentes actions en fonction

[1] Dans cet article, le terme *feedback* fait référence à l'information fournie par le système pendant ou après une action de l'utilisateur. Le terme *feedforward* fait référence à l'information fournie avant que l'action ne se déroule. Ces termes anglais sont utilisés à la place de "retour d'information" et "retour d'information par anticipation" pour faciliter la lecture.

de la face, de l'arrête ou du sommet qui est en contact avec la surface [24]. Le contact du crayon avec la surface sélectionne l'outil et la manipulation directe commence instantanément. Dans *Manual Deskterity*, Hinckley *et al*. explorent la coordination bi-manuelle entre stylet et doigt sur surface tactile où la main non dominante définit le contexte et les outils utilisables avec la main dominante [12]. Par exemple, en fonction de la façon dont un objet est tenu avec les contacts de la main non dominante et du mouvement relatif entre l'objet et le stylet, un trait de stylet peut couper à la manière d'un couteau, tirer un trait le long d'une arête ou créer une copie de l'objet. Ainsi changer d'outil se fait en changeant la position de la main non dominante et non en changeant explicitement d'outil avec la main dominante.

Dans une étude comparative entre palettes flottantes, *marking menus* et *toolglasses*, Mackay montre qu'aucune technique n'est nettement supérieure et plaide pour l'intégration de ces différentes techniques [19]. L'identification des doigts est un moyen élégant de fusionner la sélection de commande et la manipulation directe en fonction de quels doigts touchent quels objets. De plus, cela permet de changer explicitement d'outil en fonction des doigts utilisés.

Raccourcis clavier et sélection de commandes

Les raccourcis claviers donnent aux utilisateurs un accès rapide aux commandes en pressant une ou plusieurs touches simultanément. Les touches de fonction sont ainsi utilisées dans beaucoup d'applications et de systèmes d'exploitation pour déclencher les commandes les plus fréquentes (*e.g.* `F1` pour afficher le menu d'aide). Les combinaisons de touches utilisent des modificateurs (*e.g.* `Ctrl`, `Shift`) avec d'autres touches pour différencier la sélection de commande de l'entrée de texte (*e.g.* `Ctrl+C` pour copier un objet). Des modificateurs sont aussi utilisés pour contraindre des actions effectuées à la souris. Par exemple, presser un modificateur dans une application de dessin aimante un objet sur une grille ou contraint le rapport hauteur/largeur.

Les principaux problèmes des raccourcis clavier sont leur difficulté à être découverts et retenus. Les utilisateurs ont tendance à ne pas y faire attention, ignorent les indications dans les menus et ne les retiennent pas quand ils ne les utilisent pas fréquemment [17]. Plusieurs techniques permettent d'augmenter leur utilisation, avec un retour visuel [20], auditif [9] ou haptique [3].

La manière la plus courante de déclencher des commandes sur un écran multi-points est d'utiliser les boutons de l'interface graphique. Ces boutons prennent de la place sur l'écran et réduisent la fluidité de l'interaction en obligeant l'utilisateur à effectuer des aller-retours entre les boutons et les objets d'intérêt. Ces deux problèmes peuvent être résolus grâce à l'utilisation de menus contextuels qui sont déclenchés par une action spécifique de l'utilisateur. L'appui long, consistant à maintenir durant un certain temps un ou plusieurs doigts immobiles, est couramment utilisé, par exemple. Il faut cependant que l'utilisateur sache qu'une telle action est possible et le temps d'exécution d'une commande est allongé. Le nombre de doigts en contact avec la surface interactive est aussi souvent utilisé pour sélectionner la commande, et éventuellement contrôler ensuite ses paramètres. *RST* (*Rotate-Scale-Translate*)[2] est un bon exemple d'intégration commande/geste à deux doigts. Le nombre de doigts peut aussi permettre de sélectionner des éléments dans un menu [2]. Une telle méthode permet d'associer jusqu'à 25 commandes, ce qui suffit pour la plupart des applications. Cependant, cette technique rend difficile la découverte des associations, introduit de l'occultation due au nombre de doigts en contact et rend imprévisible l'endroit où sera exécutée la commande .La combinaison de ces techniques avec des gestes permet d'étendre le vocabulaire. Cependant, les utilisateurs doivent d'abord savoir que de tels gestes existent, les gestes sont parfois difficiles à exécuter, leur réalisation prend du temps et leur reconnaissance est sujette à erreurs. Les gestes simples tels que ceux de balayage (*flick*) restent très efficaces quand ils intègrent une transition novice/expert fluide, comme c'est le cas pour les *marking menus* [16].

D'autres informations de contact ont été utilisées pour exécuter des commandes comme la pression exercée, qui est souvent déduite de l'aire de la surface en contact et peut donc être biaisée par d'autres paramètres, comme le nombre de doigts [6]. *Rock & Rails* utilise la forme du contact pour sélectionner des outils [27]. *Tapsense* est un autre exemple qui différencie la pointe, la pulpe, l'ongle et la jointure d'un doigt grâce au son que celui-ci produit lors du contact avec une surface [11]. Cette technique n'occulte pas l'affichage mais elle ne permet pas de distinguer différents doigts.

Dans la section suivante, nous présentons Adoiraccourcix, une technique d'interaction qui utilise l'identification des doigts comme paramètre orthogonal à ceux déjà existants (interactions bi-manuelle, gestes, appuis longs, nombre de doigts) pour la sélection de commandes sur tables interactives.

ADOIRACCOURCIX

Nous détaillons ici la logique de conception d'Adoiraccourcix, créée pour favoriser l'intégration de la sélection de commandes et du contrôle des paramètres associés, en utilisant l'identification des doigts. Dans ce qui suit, nous rappelons qu'un *accord* est considéré comme étant un contact simultané de deux doigts ou plus sur une surface interactive. Une *configuration de contacts*, ou simplement *configuration*, correspond à un contact simple ou à un accord.

Une commande par doigt

L'application la plus évidente de l'identification des doigts est de faire correspondre une commande à chaque doigt [23, 26, 4]. Dans une application de dessin, différents doigts peuvent, par exemple, être utilisés pour créer des lignes, des rectangles ou des cercles en glissant le doigt correspondant sur la surface. D'autres doigts peuvent aussi être utilisés pour déplacer, supprimer ou dupliquer des formes en les touchant. Une telle fusion de la sélection de commande et de ses paramètres (*e.g.* position du doigt pour coller) réduit indéniablement le nombre

[2]Rotation, changement d'échelle et translation

d'actions nécessaires à la réalisation d'une tâche, comparé à l'utilisation de *widgets* qui peuvent être éloignés de la zone d'intérêt. Cependant cette méthode ne permet d'accéder qu'à 10 commandes.

Utilisation des accords

Avec l'identification des doigts, tout accord de k doigts peut théoriquement se faire de $C(10,k)$ façons. Cela signifie aussi que $C(10,k)$ interactions à k doigts peuvent être sélectionnées en un geste simple. La technique *RST* est généralement effectuée à l'aide du pouce et de l'index de la main dominante, ou des deux index. Avec l'identification des doigts, *RST* pourrait être associée à ces deux accords et les $C(10,2)-2=43$ accords restants pourraient être utilisés pour exécuter d'autres commandes.

Le nombre total d'accords disponibles avec 10 doigts est $\sum_{k=2}^{10} C(10,k) = 1013$. La première ligne de la Table 1 montre la distribution de ce nombre en fonction de k. Cette limite est cependant purement théorique. Le nombre de commandes pouvant être confortablement réalisées à l'aide d'accords est vraisemblablement plus petit [8]. Une étude précise de l'effet complexe des contraintes biomécaniques, des capacités motrices et de mémorisation dépasse le cadre de cet article. Cependant, en considérant la valeur de cette borne supérieure, nous pouvons anticiper des difficultés de mémorisation des couples accords/commandes. Il est donc nécessaire de fournir un moyen de les organiser.

k	1	2	3	4	5	6	7	8	9	10	Total
$C(10,k)$	10	45	120	210	252	210	120	45	10	1	1023
$C(5,k)$	5	10	10	5	1						31
$C(4,k)$	4	6	4	1							15
$C(3,k)$	3	3	1								7
$C(2,k)$	2	1									3

Table 1: Nombre de k-combinaisons de n doigts pour différentes valeurs de n.

Approche bi-manuelle inspirée par les raccourcis clavier

Une solution simplifiant la mémorisation et l'exécution d'accords est de réduire le nombre de doigts impliqués et de fournir des moyens de construire les accords de façon incrémentale, en groupant les commandes similaires.

Les *raccourcis clavier* sont en général exécutés à l'aide d'un nombre limité de modificateurs (*e.g.* `Ctrl`, `Shift`) et d'une touche symbolique. Les modificateurs ne déclenchent pas de commande mais définissent un mode dans lequel les touches symboliques peuvent être associées à une commande. Ils sont placés principalement sur le bord gauche des claviers et sont de ce fait principalement utilisés par la main non dominante pour les droitiers. L'ordre et le moment suivant lesquels ces modificateurs sont pressés n'a pas d'importance, ce qui facilite leur utilisation.

A l'instar des *raccourcis clavier*, nous proposons d'utiliser les doigts de la main non dominante comme des *modificateurs* et ceux de la main dominante comme *déclencheurs* de commandes. En considérant les ensembles de configurations S_M de la main non dominante et S_D de la main dominante, le nombre de commandes pouvant être exécutées avec cette approche est :

$$\underbrace{(|S_M|+1)}_{\substack{\textit{avec ou sans}\\ \textit{modificateurs}}} \times \underbrace{|S_D|}_{\substack{\textit{nombre de}\\ \textit{déclencheurs}}}$$

La partie basse de la Table 1 montre le nombre de k-combinaisons pour 5 à 2 doigts. Un total de 31 accords peuvent théoriquement être effectués avec les cinq doigts d'une main, mais ce nombre chute à 15, 7 et 3 en utilisant seulement 4, 3 ou 2 doigts.

Le nombre total de commandes théoriquement accessibles avec cette approche est $(31+1)\times 31 = 992$. Le choix de S_M et S_D est donc crucial pour réduire et organiser l'espace d'interaction. Un grand nombre de modificateurs et un petit nombre de déclencheurs, l'inverse ou un compromis peuvent être préférés suivant le contexte. Pour chaque ensemble, des combinaisons spécifiques de doigts, toutes les combinaisons de certains doigts ou encore une sélection ad-hoc de configurations peuvent être choisies. Par exemple, l'utilisation des combinaisons de 3 ou 4 doigts de la main non dominante et de doigts seuls de la main dominante permet la sélection de $((4+6+4)+1)\times 5 = 75$ commandes, ce qui semble suffisant pour la plupart des applications[3].

Comme pour les raccourcis clavier, nous anticipons l'émergence de conventions dans les couples commandes/accords bi-manuels pour les opérations communes (*e.g.* copier, couper et coller). Cependant pour la plupart des opérations, S_M and S_D peuvent être spécifiques aux applications. Ainsi, nous devons aussi fournir un moyen pour aider les utilisateurs à comprendre et apprendre l'organisation choisie.

Intégration du feedback et du feedforward

S_M et S_D définissent deux vocabulaires : l'un définissant le mode, l'autre précisant les déclencheurs. Pour chaque mode, à la manière des raccourcis clavier, chaque déclencheur appelle, ou non, une commande. Comme l'utilisateur définit le mode explicitement et de manière incrémentale, il est conscient du changement de mode. Il doit aussi être conscient des conséquences que le mode courant aura sur les futures actions de la main dominante. À cet effet, nous proposons une solution qui intègre *feedback* et *feedforward* : *Lilanotix*.

Lilanotix est affichée à chaque fois qu'un utilisateur effectue une configuration de S_M avec sa main non dominante, avant d'utiliser sa main dominante. Elle disparait automatiquement quand les doigts de la main non dominante quittent la surface ou que la commande est sélectionnée avec les doigts de la main dominante. Le but de *Lilanotix* est d'indiquer les commandes du mode courant associées aux doigts de la main dominante. La Figure 1 illustre le concept. Dans cet exemple, un accord à deux doigts de la main gauche définit un mode donnant accès à différents outils de dessin associés aux pouce, index, majeur, annulaire et auriculaire de la main droite. A noter

[3] Malacria *et al.* [20] ont compté de 15 à 112 raccourcis clavier sur 30 applications populaires sur Mac ($\mu = 55.31, \sigma = 20.53$)

qu'idéalement, *Lilanotix* doit non seulement représenter clairement les commandes disponibles mais aussi à quels doigts de S_D elles sont associées. Le but de *Lilanotix* est d'afficher du *feedforward* et du *feedback*. Les informations affichées ne correspondent en aucun cas à un menu avec des boutons destinés à être pressés. Pour déclencher une commande, l'utilisateur doit exécuter la configuration correspondante avec sa main dominante.

La nature des illustrations de *Lilanotix* varie en fonction des commandes disponibles et de la complexité du vocabulaire de S_D. Par exemple, les icônes déjà présentes dans les barres d'outils peuvent être réutilisées de préférence, par souci de consistance. Pour le vocabulaire de S_D, il est aussi possible de réutiliser des représentations de la littérature, *e.g.* [18, 8], ou d'en concevoir de nouveaux.

Contrôle de paramètres

Dès que l'utilisateur a exécuté une commande, il peut en contrôler les paramètres en déplaçant ses doigts. Dans le cas le plus simple, les doigts de la main dominante qui ont déclenché la commande sont utilisés pour ce contrôle, procurant ainsi le bénéfice de l'intégration de la sélection de commande et de la manipulation directe [10].

Les commandes associées à un accord peuvent utiliser la trajectoire d'un seul des doigts composant l'accord, le mouvement relatif entre les doigts ou le mouvement global. Une commande déclenchée par le pouce et l'index peut, par exemple, utiliser la distance entre les contacts pour ajuster un paramètre et le barycentre pour en ajuster deux autres. De nouveaux doigts de la main dominante peuvent aussi être pris en compte pour fournir des contrôles additionnels. A noter que la représentation visuelle des commandes dans *Lilanotix* doit être suffisamment explicite pour que les exemples précédents soient compréhensibles.

De nouveaux doigts de la main non dominante peuvent aussi participer au contrôle d'une commande initiée par la main dominante, si *Lilanotix* n'est pas utilisée. Nous avons choisi d'utiliser ces doigts pour effectuer un contrôle particulier. Dans certaines applications, les modificateurs sont utilisés pour contraindre l'interaction effectuée à la souris, par exemple pour aimanter un point sur une grille, aligner horizontalement ou verticalement des objets, forcer le rapport hauteur/largeur, etc. De façon similaire, dès qu'une commande est en cours d'exécution, la main non dominante peut être utilisée pour contraindre l'interaction courante. Les contraintes sont définies par un troisième ensemble de configurations, S_C. Lorsque le nombre de contraintes applicables à une commande est faible, il est possible de les associer à des doigts individuels, de manière à pouvoir facilement combiner ces contraintes en utilisant plusieurs doigts. Un *feedback* spécifique peut apparaitre près du ou des doigts concernés par la ou les contraintes (Figure 1, image de droite).

Transition novice-expert

Adoiraccourcix emprunte beaucoup aux raccourcis clavier. Nous émettons l'hypothèse que les utilisateurs familiers avec les raccourcis clavier seront capables de comprendre les principes de base de notre technique.

Le *feedforward* permet d'explorer les accords de la main non-dominante et de connaitre les commandes accessibles aux doigts de la main dominante, sans crainte de déclencher une commande par mégarde. Par ailleurs, comme pour les *marking menus*, ce *feedforward* peut être affiché après un certain délai afin de réduire l'occultation de l'écran. Enfin il est envisageable d'offrir à l'utilisateur la possibilité d'afficher explicitement une aide en appuyant par exemple sur un bouton ou en mettant les dix doigts sur la surface. Une fois dans ce mode, l'utilisateur pourrait visualiser et explorer les différentes configurations disponibles avec les commandes associées.

PROTOTYPE D'IDENTIFICATION DES DOIGTS

L'identification fiable des doigts sur une surface multipoints reste un problème technique difficile. L'utilisation d'empreintes digitales est prometteuse [13], mais n'est pas encore opérationnelle pour de grandes surfaces. La vision par ordinateur n'est pas robuste [15, 21], à moins d'utiliser un système de marqueurs attachés sur une paire de gants [22]. Nous présentons ci-dessous le prototype que nous avons mis en place pour obtenir une identification robuste des doigts et ainsi mettre en œuvre Adoiraccourcix.

Notre prototype utilise des *GameTraks* qui sont des contrôleurs de jeu servant à l'origine à simuler des *swings* de club de golf. Cinq *GameTraks* nous permettent de repérer les positions des extrémités de dix doigts dans un espace en 3 dimensions en y attachant les bouts de chacun des fils.

Nous avons écrit une variante d'un serveur TUIO[4] en C++ et utilisant la librairie *libgametrak*[5]. Cette implémentation établit la correspondance entre la position 3D des doigts et la position 2D des contacts sur un écran *32" 3M™ C3266PW multi-touch surface*. À chaque doigt est associé une homographie pour établir cette correspondance. Ces homographies sont calculées dans une phase préalable de calibrage où des points 3D sont associés à une position 2D connue. Une fois le système mis en place, l'identification d'un doigt en contact se fait par la recherche du point 3D le plus proche. Le système génère ensuite un événement TUIO comprenant l'identité du doigt et de la main. Tout le système, ainsi que l'application de dessin vectoriel décrite par la suite, fonctionnent sur un MacBook Pro doté d'un processeur 2.5GHz Intel Core i5 et de 8Go de RAM.

L'architecture de notre système est modulaire, ce qui nous permet d'utiliser d'autres prototypes. Par exemple, nous avons aussi implémenté une identification des doigts avec un *Leap Motion*[6] fixé au dessus de notre surface tactile. Le *Leap Motion* est capable de repérer l'extrémité des doigts en 3D à $100Hz$. La phase de calibrage est similaire à celle précédemment décrite, mais puisque toutes les positions 3D sont données dans le même référentiel, une seule homographie est nécessaire. Cependant, comme la plupart des systèmes basés sur de la vision par ordinateur à un seul point de vue, le *Leap Motion* est sensible aux occultations, ce qui pénalise la fiabilité, même si l'utilisateur

[4] http://www.tuio.org/

[5] https://code.google.com/p/libgametrak/

[6] https://www.leapmotion.com

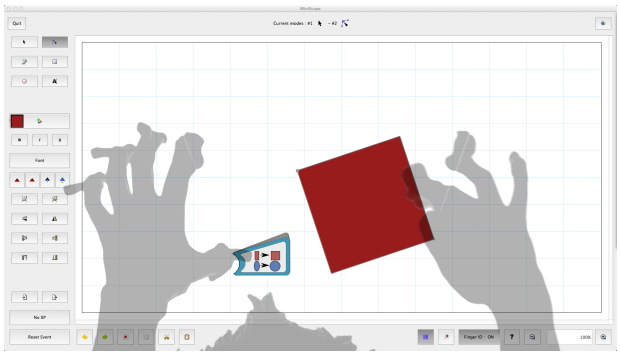

Figure 2: Notre application de dessin vectoriel intègre une *GUI*, mais elle peut être entièrement contrôlée grâce aux interactions Adoiraccourcix (les mains de l'utilisateur sont représentées sous la forme d'ombres semi-transparentes).

n'est plus attaché. Pour ces raisons, la démonstration suivante et les évaluations préliminaires utilisent la version à base de *GameTraks* qui est plus fiable.

APPLICATION DE DÉMONSTRATION

Afin d'expérimenter Adoiraccourcix, nous avons développé une application de dessin vectoriel dans l'esprit d'Adobe Illustrator[7] et InkScape[8]. Ce type d'applications a d'abord l'intérêt d'être connu par beaucoup d'utilisateurs et ensuite l'avantage de posséder de nombreuses commandes afin de créer, éditer et manipuler du contenu, ce qui est intéressant pour tester notre technique.

L'application possède six modes et 26 commandes (Figure 2). Les six modes sont : un mode de *sélection* pour la manipulation d'objets (un doigt pour la translation, deux doigts pour le *RST*), un mode d'*édition* pour modifier certains attributs des objets tels que les points de contrôles, et quatre modes de *dessin* pour créer des rectangles, ellipses, courbes et du texte. Afin de comparer les interactions Adoiraccourcix avec une *GUI* classique, notre application possède des barres d'outils qui permettent l'accès à tous les modes et commandes. En mode expert ces barres d'outils peuvent être cachées afin d'augmenter l'espace de travail disponible à l'écran.

L'application a pour but d'illustrer Adoiraccourcix mais elle met également en œuvre des techniques d'interaction analogues à d'autres techniques de l'état de l'art [22] pour lesquelles aucun mécanisme de *feedforward* n'est disponible. Il s'agit des commandes associées à un seul doigt et des accords uni-manuels.

Par la suite, nous décrirons quels doigts sont utilisés pour accéder aux commandes et aux modes en utilisant la convention suivante : aux noms classiques des doigts nous ajouterons le préfixe 'd-' pour la main dominante et 'nd-' pour la main non dominante. Par exemple, d-index désigne l'index de la main dominante alors que nd-pouce désigne le pouce de la main non dominante.

[7]http://www.adobe.com/products/illustrator.html
[8]http://inkscape.org/

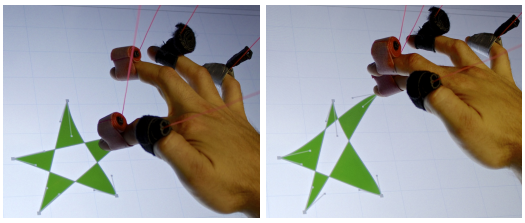

Figure 3: Changement de mode instantané : l'utilisateur déplace une forme avec son d-index et édite ses points de contrôles avec son d-majeur.

Commandes associées à un seul doigt

Sélection, édition et création

Par défaut, l'application est en mode sélection et le d-index est associé à ce mode (Figure 3 gauche) alors que le d-majeur est associé à l'édition de formes et l'ajustement des attributs des objets (*e.g.* points, Figure 3 droite). Quand l'application passe en mode création (*i.e.* en mode rectangle), le d-index est alors associé à la création de formes et le d-majeur est utilisé pour sélectionner et déplacer des objets. L'utilisation de l'index et du majeur (plutôt que l'auriculaire par exemple) nous semble adapté à ce type de tâches qui demandent un certain degré de précision dans l'interaction.

Annuler/refaire

Annuler et *refaire* sont des commandes importantes dans les applications de dessin vectoriel. Nous les avons associées au d-pouce combiné avec un geste : d-pouce balayage gauche pour *annuler* et d-pouce balayage droit pour *refaire*.

Accords uni-manuels

Rotation, échelle et translation

Nous avons observé de manière informelle que dans beaucoup d'applications les utilisateurs se servent de la combinaison pouce, index pour les opérations de RST. Nous avons donc associé cette combinaison aux manipulations d'objets et au grossissement de vue.

Panoramique

La fonction panoramique permet de naviguer dans un document trop grand pour s'afficher dans son intégralité sur l'écran. De façon similaire à l'utilisation de deux doigts sur un pavé tactile pour faire défiler un document, nous associons aux d-index et d-majeur le contrôle relatif du panoramique. Ce type de contrôle est adapté aux déplacements de faibles amplitudes mais peut demander plusieurs débrayages pour des amplitudes plus importantes. Dans de telles situations, l'utilisation d'un contrôle absolu permet d'atteindre plus rapidement une région d'intérêt sans avoir recours au grossissement. Nous avons choisi d'associer aux d-majeur et d-annulaire le contrôle absolu du panoramique, où la vue est centrée sur le point du document correspondant à la position absolue des doigts sur l'écran.

Loupe et tiroir

La combinaison d-pouce et d-index contrôle le niveau de grossissement de la vue sur le document. Cependant il est parfois préférable de n'agrandir qu'une petite partie du document pour y ajuster des détails, sans perdre le contexte général. Pour cela, nous avons implémenté une

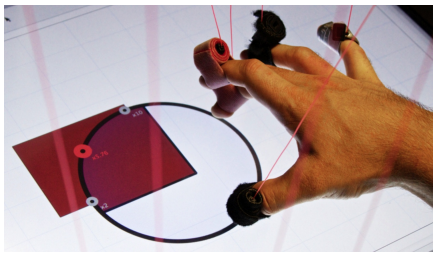

Figure 4: Loupe : grossissement d'une partie de la feuille de dessin. L'utilisateur peut ajuster le niveau de grossissement à l'aide du curseur et déplacer la loupe en glissant les doigts depuis le bord.

loupe (Figure 4), comme détaillé par Käser *et al.* [14] et mis en œuvre dans certaines applications commerciales comme *zoom* dans *Paper*[9]. Cette loupe est associée à la combinaison d-pouce et d-majeur. Les utilisateurs peuvent déplacer et ajuster la taille de la loupe en déplaçant ces doigts. Un curseur situé sur le bord de la loupe et ajustable par n'importe quel doigt de S_D contrôle le niveau de zoom. De plus tous les doigts de S_D peuvent déplacer la loupe en la glissant depuis son bord. Une fois apparue, la loupe reste accessible jusqu'à ce que l'utilisateur la fasse disparaître en réduisant sa taille ou en effectuant une double tape sur celle-ci avec d-pouce et d-majeur.

Nous avons aussi développé un tiroir destiné à stocker des objets graphiques qui pourront par la suite être restitués ou copiés. L'utilisateur peut copier un objet ou un de ses attributs comme la couleur, le type de contour ou le formatage d'un texte. Selon le même principe que la loupe, le tiroir est appelé avec la combinaison d-pouce et d-auriculaire (Figure 5). Les objets peuvent ensuite être glissés dans le tiroir. L'objet est soit déplacé ou copié si le d-annulaire est en contact avec la surface lors de la fin du glissé. De la même manière, déplacer ou dupliquer les objets depuis le tiroir s'effectue en glissant les objets du tiroir vers la feuille de dessin. Pour copier un attribut, l'utilisateur doit changer le mode courant associé au tiroir en maintenant le d-pouce sur la surface et en réalisant une tape sur le tiroir avec le d-auriculaire. Les modes défilent de façon circulaire entre : *forme* (décrit ci-dessus), *couleur*, *type de contour*, *formatage de texte*.

Accords bi-manuels

Les accords uni-manuels sont pratiques pour exécuter les commandes les plus fréquemment utilisées. Pour étendre le nombre de commandes disponibles à l'aide d'accords tout en aidant les utilisateurs à ne pas s'y perdre, nous utilisons Adoiraccourcix avec une mise en œuvre de *Lilanotix*.

Avec un total de 29 commandes et modes restants à associer à des accords, nous avons choisi de les rassembler dans 7 groupes comprenant au maximum 5 commandes (Table 2). Selon la Table 1, 3 doigts de la main non dominante sont suffisants pour différencier 7 catégories ($|S_M| = 3+3+1 = 7$) dans lesquelles chacun des 5 doigts de la main dominante exécute une commande différente (en supposant qu'il n'y a pas d'accords de doigts avec la main dominante). En faisant ce choix, nous pouvons sélectionner $(7+1) \times 5 = 40$ commandes, ce qui est

[9]http://www.fiftythree.com/paper

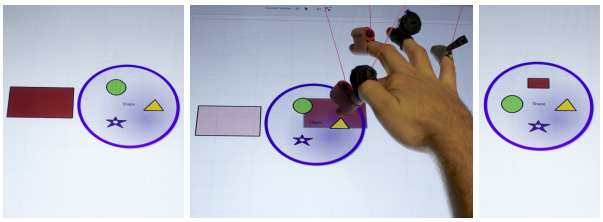

Figure 5: *Tiroir* : les objets peuvent être glissés sur le tiroir pour y être stockés.

suffisant pour notre exemple. Nous utilisons les accords des nd-pouce, nb-index et nd-majeur pour accéder à ces 7 catégories. Une fois un de ces accords joué, du *feedforward* apparaît en fondu après 0.3s et montre les commandes accessibles aux doigts de la main dominante (Figure 6). Le *feedforward* est constitué d'un cercle de couleur affiché en dessous de chaque doigt composant l'accord. Ces cercles sont reliés à un rectangle comportant 5 icônes représentant les doigts de la main dominante. Par exemple, l'accord nd-pouce + nd-index affiche le *feedforward* de l'image de droite de la Figure 6, indiquant que le d-pouce correspond à la multi-sélection, d-index à la commande copier, d-majeur à couper, d-annulaire à coller et d-auriculaire à la commande effacer. Quand les doigts de la main dominante sont associés à des gestes, ceux-ci sont représentés au-dessus de l'icône correspondante. Par exemple, les commandes *annuler* et *refaire* sont aussi accessibles via l'accord nd-pouce combiné à un balayage du d-pouce. Les balayages correspondants sont représentés au dessus de l'icône pouce (Figure 6 image de gauche).

Un des accords (nd-index + nd-majeur) est associé à des groupes contextuels. Les commandes disponibles pour la main dominante dépendent de l'objet sélectionné. Si une ou plusieurs formes sont sélectionnées, les commandes disponibles sont : multi-sélection, contour plein, contour pointillé, contour plus épais et contour plus fin. Si un ou plusieurs textes sont sélectionnés, les commandes disponibles sont : multi-sélection, gras, italique, souligné et police. Quand aucun objet n'est sélectionné ou si les deux types d'objets sont sélectionnés aucune commande n'est disponible.

Sélecteur de couleur

Pour la sélection de couleurs nous avons adapté une palette couleurs classique que l'utilisateur appelle avec les nd-pouce et nd-auriculaire (Figure 7). L'utilisateur ajuste la teinte, lumière, saturation et transparence avec n'importe quel doigt de la main dominante, ce qui met à jour la couleur de l'arrière plan de la roue. L'utilisateur peut ensuite appliquer la couleur courante au remplissage d'un objet en le touchant avec le d-index, et à son contour en

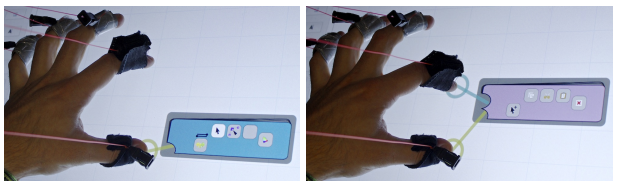

Figure 6: *Lilanotix* : (a) commandes associées aux doigts de la main dominante en association avec nd-pouce ; (b) commandes copier/couper/coller/effacer associées à l'accord nd-pouce et nd-index.

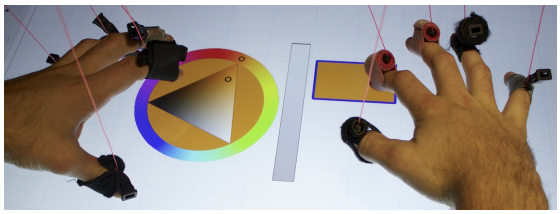

Figure 7: *Sélecteur de couleur* : d-index modifie la couleur de remplissage du rectangle, d-majeur peut modifier celle du contour, d-annulaire permet d'obtenir la couleur de remplissage et d-auriculaire celle du contour.

le touchant avec le d-majeur. Il peut aussi modifier la couleur courante en touchant un objet avec le d-annulaire pour obtenir sa couleur de remplissage et avec le d-auriculaire pour obtenir la couleur de son contour.

Application de contraintes

Quand l'utilisateur interagit avec les doigts de la main dominante, ceux de la main non dominante peuvent définir des contraintes. Lors de la création ou de l'édition de formes, nous avons associé le nd-pouce pour contraindre le rapport hauteur/largeur à des formes parfaites (*e.g.* carré vs. rectangle). Lors de la manipulation, le nd-index est utilisé pour désactiver le changement d'échelle de l'objet. Enfin pour tous les modes, le nd-majeur fixe le centre de la forme et le nd-annulaire aimante les points d'intérêts sur la grille. Comme chaque contrainte est associée à un doigt différent, les contraintes peuvent être combinées. Un *feedback* présenté sous la forme d'une étiquette est affiché près de chaque doigt de la main non dominante en cours d'utilisation (Figure 8). Ces étiquettes sont grisées lorsque la contrainte ne peut pas être appliquée.

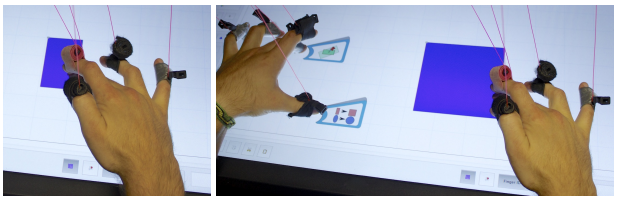

Figure 8: Contraintes de manipulation : elles apparaissent lorsque qu'un doigt de la main non-dominante est en contact avec l'écran alors qu'un ou plusieurs doigts de la main dominante manipulent un objet.

Accord	pouce	index	majeur	annulaire	auriculaire
○○○○●		Sélection	Modification		Couleur
○○○●○		Courbe	Rectangle	Cercle	Texte
○○●○○	S+	Devant	Vers l'avant	Vers l'arrière	A l'arrière
○○○●●	S+	Copier	Couper	Coller	Effacer
○○●○●	S+	Grouper	Dégrouper		
○○●●○	S+	← commandes contextuelles →			
○○●●●	S+	Vsym	Hsym		
●○○○●	← outils couleur →				
○○○○○	●	Zoom	Loupe		Tiroir

Table 2: Liste des associations de *Lilanotix*. L'accord effectué avec la main non dominante est indiqué dans la colonne de gauche où les cercles représentent de gauche à droite les doigts de l'auriculaire au pouce. ● et ○ représentent respectivement des doigts en contact ou pas. Les commandes associées aux doigts de la main dominante sont listées dans les colonnes suivantes où S+ représentes la multi-sélection.

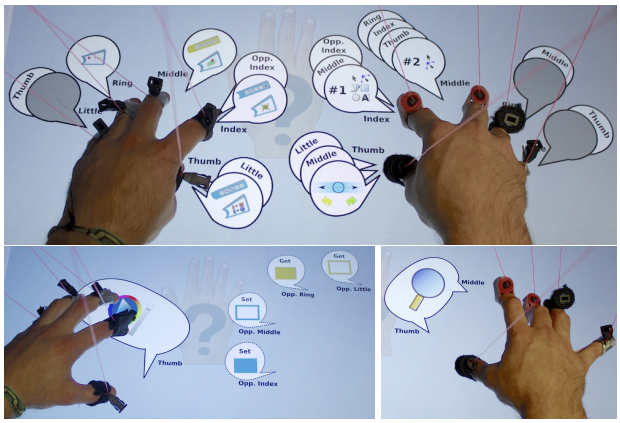

Figure 9: Mode d'aide. Haut : chaque doigt possède des bulles indiquant l'action associée à ce doigt, ou les combinaisons avec les autres doigts. Bas : exemples d'exploration en utilisant certaines combinaisons avec à gauche le sélecteur de couleur et à droite la loupe.

Transition novice-expert

Comme décrit précédemment, nous avons apporté un soin particulier à utiliser du *feedforward* et du *feedback* quand cela est utile à la découverte des associations accords/commandes. En plus de cela, nous avons ajouté des *info-bulles* pour chaque bouton de la *GUI* montrant l'équivalent Adoiraccourcix. Ces *info-bulles* apparaissent à proximité des boutons quand ils sont pressés plus d'une seconde. Enfin, un mode d'aide peut être explicitement appelé par l'utilisateur lorsqu'il presse un bouton d'aide ou lorsqu'il pose ses 10 doigts simultanément sur la surface. Dans ce mode, des *info-bulles* spécifiques à chaque doigt sont affichées près de ceux-ci (Figure 9). Les *info-bulles* affichent la liste des icônes des commandes associées à chaque doigt et la liste des doigts composants les combinaisons (le préfixe "Opp." indique des doigts de la main opposée). Si l'utilisateur souhaite explorer une commande, il lui suffit de jouer l'accord et les *info-bulles* afficheront plus de détails. Pour quitter ce mode, soit l'utilisateur presse le bouton d'aide, soit il effectue une double tape sur la surface avec ses 10 doigts.

EXPÉRIENCES PRÉLIMINAIRES

Nous avons réalisé deux expériences préliminaires, la première pour estimer dans quelle mesure des utilisateurs novices sont capables de découvrir les différentes fonctionnalités offertes par Adoiraccourcix dans notre application de dessin vectoriel, et la seconde pour mesurer les performances d'Adoiraccourcix en comparaison avec une *GUI* classique dans le cadre d'une utilisation experte.

Dans la première expérience, 8 volontaires (20-35 ans, 2 femmes, 2 non informaticiens), tous droitiers, ont participé à une session de pensée à voix haute d'une heure environ comportant 2 parties. La première partie était une période de temps libre pendant laquelle un expérimentateur rappelait aux participants le fonctionnement basique des applications de dessin vectoriel, et où les participants découvraient et interagissaient, sans limite de temps, avec l'application, à la fois avec Adoiraccourcix et la GUI. Durant cette phase, quand cela était nécessaire, l'expérimentateur les aidait à comprendre comment utiliser chaque fonctionnalité et leur montrait celles qui n'avaient pas

été découvertes. Dans la seconde partie, les participants étaient invités à effectuer quatre tâches simples en utilisant la *GUI* ou Adoiraccourcix : créer des rectangles, des courbes et des ellipses selon des dimensions données, dupliquer des objets pour créer un ciel étoilé, changer les couleurs de différents objets et dessiner une maison en partant d'une feuille blanche (les seules indications étaient qu'elle comporte une porte avec une poignée, deux fenêtres et un toit). À la fin, les participants ont répondu à un questionnaire sur chacune des parties dans le but de collecter leurs impressions en répondant à des questions sur une échelle de Likert à 5 points.

Durant la période de temps libre, nous avons collecté les remarques des participants à propos de l'application. Tous les participants ont découvert d'eux-mêmes *Lilanotix*. Nous avons cependant observé que certains participants ont essayé d'appuyer sur les icônes comme si celles-ci étaient des boutons. Pour remédier à ce problème, l'affordance de l'affichage actuel pourrait être amélioré en estompant par exemple l'aspect bouton des icônes. Pour le tiroir, nous avons dû expliquer le rôle des d-annulaire et d-auriculaire. De même certains participants n'ont pas découvert toutes les interactions uni-manuelles d'eux mêmes à cause du manque de *feedforward*. Quant au questionnaire, les utilisateurs ont confirmé la ressemblance de notre application avec celles de dessin vectoriel qu'ils connaissaient (4.4/5). Ils ont réussi à effectuer les tâches qu'ils souhaitaient (4.5/5). Leurs outils préférés d'Adoiraccourcix (min : 3.4/5, max : 4.7/5, médiane : 4.0/5) étaient le *sélecteur de couleur* (4.7/5), Lilanotix (4.5/5) et le *tiroir* (4.3/5). Le moins aimé était le *contrôle absolu du panoramique* (3.4/5). L'impression sur la facilité d'utilisation est au dessus de la moyenne (min : 3.6/5, max : 4.3/5, médiane : 3.8/5). Les participants nous disaient avoir besoin de temps pour apprendre les différents raccourcis mais tous ont affirmé avoir le sentiment qu'ils amélioreraient leur performance avec plus de pratique. Ils ont aussi fait part d'une plus grande fluidité d'interaction qu'avec une *GUI* (4.0/5).

En considérant que notre interface cible prioritairement des utilisateurs experts, nous n'étions pas surpris que les participants n'aient pas atteint un certain niveau d'habileté en une heure. Pour mesurer le gain de performance des utilisateurs experts, nous avons recruté des utilisateurs pour réaliser de manière répétitive des tâches simples en utilisant Adoiraccourcix et la *GUI* : 1) dupliquer un objet et le déplacer à une position donnée, 2) créer un rectangle dont les sommets opposés doivent être à une position donnée, et 3) éditer une courbe et déplacer un des points à une certaine position. Au début de chaque essai l'application était en mode *sélection*. Chaque tâche comprenait 5 blocs de 10 essais où les conditions (*i.e.* positions cibles) à chaque essai étaient identiques. De plus l'expérimentateur imposait aux participants l'utilisation de la façon la plus rapide de réaliser chaque tâche pour chaque technique (*i.e.* utiliser la main non dominante pour presser les boutons de la *GUI* et la main dominante pour interagir). Les conditions favorisaient une utilisation experte des deux SYSTÈMES. Nous avons mesuré le temps de chaque essai, de la première interaction détectée à la fin de l'essai. Nous avons utilisé un plan expérimental intra-sujets et l'ordre des conditions était balancé parmi les participants (Adoiraccourcix et la *GUI*). 12 participants (23-35 ans, 1 femme, 3 non informaticiens), tous droitiers, ont pris part à l'expérience. Le plan expérimental était : 12 PARTICIPANTS $\times$ 2 SYSTÈMES $\times$ 3 TÂCHES $\times$ 5 BLOCS $\times$ 10 RÉPÉTITIONS = 3600 essais.

Comme les temps enregistrés ne suivaient pas une loi normale, les données ont été traitées en utilisant un *Aligned Rank Transform* [28] avant d'y appliquer une ANOVA à mesures répétées. L'analyse des résultats a montré un effet significatif du SYSTÈME ($F_{1,11} = 38.7$, $p < 0.001$) et une interaction significative SYSTÈME $\times$ TÂCHE ($F_{2,22} = 48.8$, $p < 0.001$) sur le temps de réalisation de la tâche. Des analyses post-hoc ont révélé des différences significatives entre les deux systèmes pour la première tâche ($p < 0.001$, *GUI* : 1.87s, Adoiraccourcix : 1.03s) et la troisième ($p < 0.001$, *GUI* : 1.14s, Adoiraccourcix : 0.71s) contrairement à la seconde tâche (*GUI* : 0.91s, Adoiraccourcix : 0.92s). Les différences significatives observées pour les tâches 1 et 3 peuvent être expliquées par le nombre plus élevé d'actions à effectuer dans la condition GUI par rapport à la condition Adoiraccourcix (appuyer sur les boutons copier et coller ainsi que de changement de mode). Dans la seconde tâche, avec les deux systèmes, il n'y avait qu'un changement de mode à effectuer. Adoiraccourcix troque l'appui de bouton et autres actions moteurs de la *GUI* contre la mémorisation et l'exécution d'accords.

LIMITATIONS

Bien que le *feedforward* de *Lilanotix* permette aux utilisateurs de trouver facilement la commande souhaitée, il faut néanmoins savoir quels sont les doigts qui définissent l'ensemble S_M. De même, il n'existe pas de *feedforward* pour les accords uni-manuels. Cependant une fois ces informations connues, le reste des commandes est facile à découvrir. De plus, le mode d'aide et les *info-bulles* favorisent cette découverte.

Notre but n'est pas de donner une association de commandes idéale mais d'illustrer notre logique de conception. Nos choix d'associations sont améliorables. Par exemple l'association de commandes que nous avons proposée suppose que l'utilisateur est droitier. Dans un contexte écologique nous suggérons de permettre à l'utilisateur de configurer sa main dominante, en intégrant par exemple un bouton sur l'interface.

Notre application ne supporte pas la personnalisation des associations accords/commandes. Cependant cette possibilité n'est pas incompatible avec Adoiraccourcix. À la manière des raccourcis clavier, nous suggérons aux développeurs d'applications de permettre aux utilisateurs de configurer l'association des commandes tout en respectant la logique de conception Adoiraccourcix.

CONCLUSION ET TRAVAUX FUTURS

En partant du constat qu'il existe peu d'équivalents des raccourcis clavier sur grands écrans multi-points , nous avons proposé de tirer parti de l'identification des doigts pour favoriser non seulement la sélection rapide de commandes mais aussi le contrôle combiné des paramètres

qui leurs sont associés. Inspiré par les raccourcis clavier, nous avons proposé d'associer les doigts de la main non-dominante à des modificateurs et ceux de la main dominante à des outils. Cette séparation permet d'introduire *feedforward* et *feedback* pour faciliter la découverte des accords existants. Nous avons illustré Adoiraccourcix au travers d'une application de dessin vectoriel et les résultats d'études préliminaires suggèrent qu'Adoiraccourcix offre une alternative prometteuse aux interfaces classiques. Pour les travaux futurs nous envisageons d'explorer l'utilisation de l'identification des doigts dans d'autres contextes que les tables interactives. Nous envisageons aussi de développer des prototypes plus pratiques à utiliser de manière à pouvoir intégrer Adoiraccourcix dans des applications existantes et ainsi mesurer l'intérêt de notre approche de manière plus approfondie. Enfin, nous souhaitons mettre en place des expériences vérifiant si nos hypothèses d'ergonomie sont justes et si *Lilanotix* aide à la mémorisation.

REFERENCES

1. Au, O. K.-C., and Tai, C.-L. Multitouch finger registration and its applications. In *Proc. OZCHI* (2010), 41–48.
2. Bailly, G., Lecolinet, E., and Guiard, Y. Finger-count & radial-stroke shortcuts: 2 techniques for augmenting linear menus on multi-touch surfaces. In *Proc. CHI* (2010), 591–594.
3. Bailly, G., Pietrzak, T., Deber, J., and Wigdor, D. J. Métamorphe: augmenting hotkey usage with actuated keys. In *Proc. CHI* (2013), 563–572.
4. Benko, H., Saponas, T. S., Morris, D., and Tan, D. Enhancing input on and above the interactive surface with muscle sensing. In *Proc. ITS* (2009), 93–100.
5. Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T. D. Toolglass and magic lenses: the see-through interface. In *Proc. SIGGRAPH* (1993), 73–80.
6. Cao, X., Wilson, A., Balakrishnan, R., Hinckley, K., and Hudson, S. Shapetouch: Leveraging contact shape on interactive surfaces. In *Proc. TABLETOP* (2008), 129–136.
7. Davidson, P. L., and Han, J. Y. Extending 2d object arrangement with pressure-sensitive layering cues. In *Proc. UIST* (2008), 87–90.
8. Ghomi, E., Huot, S., Bau, O., Mackay, W. E., and Beaudouin-Lafon, M. Arpège: Learning multitouch chord gestures vocabularies. In *Proc. ITS* (Oct. 2013). 10 pages, tbp.
9. Grossman, T., Dragicevic, P., and Balakrishnan, R. Strategies for accelerating on-line learning of hotkeys. In *Proc. CHI* (2007), 1591–1600.
10. Guimbretière, F., Martin, A., and Winograd, T. Benefits of merging command selection and direct manipulation. *ACM ToCHI 12*, 3 (Sept. 2005), 460–476.
11. Harrison, C., Schwarz, J., and Hudson, S. E. TapSense: enhancing finger interaction on touch surfaces. In *Proc. UIST* (2011), 627–636.
12. Hinckley, K., Yatani, K., Pahud, M., Coddington, N., Rodenhouse, J., Wilson, A., Benko, H., and Buxton, B. Pen + touch = new tools. In *Proc. UIST* (2010), 27–36.
13. Holz, C., and Baudisch, P. Fiberio: a touchscreen that senses fingerprints. In *Proc. UIST* (2013). tbp.
14. Käser, D. P., Agrawala, M., and Pauly, M. FingerGlass: efficient multiscale interaction on multitouch screens. In *Proc. CHI* (2011), 1601–1610.
15. Kung, P., Küser, D., Schroeder, C., DeRose, T., Greenberg, D., and Kin, K. An augmented multi-touch system using hand and finger identification. In *CHI EA* (2012), 1431–1432.
16. Kurtenbach, G., and Buxton, W. The limits of expert performance using hierarchic marking menus. In *Proc. CHI* (1993), 482–487.
17. Lane, D. M., Napier, A. H., Peres, C. S., and Sándor, A. Hidden costs of graphical user interfaces: Failure to make the transition from menus and icon toolbars to keyboard shortcuts. *IJHCI 2*, 18 (2005), 133–144.
18. Lepinski, G. J., Grossman, T., and Fitzmaurice, G. The design and evaluation of multitouch marking menus. In *Proc. CHI* (2010), 2233–2242.
19. Mackay, W. Which interaction technique works when? Floating palettes, marking menus and toolglasses support different task strategies. In *Proc. AVI* (2002), 203–208.
20. Malacria, S., Bailly, G., Harrison, J., Cockburn, A., and Gutwin, C. Promoting hotkey use through rehearsal with ExposeHK. In *Proc. CHI* (2013), 573–582.
21. Malik, S., Ranjan, A., and Balakrishnan, R. Interacting with large displays from a distance with vision-tracked multi-finger gestural input. In *Proc. UIST* (2005), 43–52.
22. Marquardt, N., Kiemer, J., Ledo, D., Boring, S., and Greenberg, S. Designing user-, hand-, and handpart-aware tabletop interactions with the TouchID toolkit. In *Proc. ITS* (2011), 21–30.
23. Sugiura, A., and Koseki, Y. A user interface using fingerprint recognition: holding commands and data objects on fingers. In *Proc. UIST* (1998), 71–79.
24. Vogel, D., and Casiez, G. Conté: multimodal input inspired by an artist's crayon. In *Proc. UIST* (2011), 357–366.
25. Wang, F., Cao, X., Ren, X., and Irani, P. Detecting and leveraging finger orientation for interaction with direct-touch surfaces. In *Proc. UIST* (2009), 23–32.
26. Wang, J., and Canny, J. FingerSense: augmenting expressiveness to physical pushing button by fingertip identification. In *CHI EA* (2004), 1267–1270.
27. Wigdor, D., Benko, H., Pella, J., Lombardo, J., and Williams, S. Rock & rails: extending multi-touch interactions with shape gestures to enable precise spatial manipulations. In *Proc. CHI* (2011), 1581–1590.
28. Wobbrock, J. O., Findlater, L., Gergle, D., and Higgins, J. J. The aligned rank transform for nonparametric factorial analyses using only anova procedures. In *Proc. CHI* (2011), 143–146.