CtrlMouse et TouchCtrl : Dupliquer les Délimiteurs de Mode sur la Souris

Thomas Pietrzak, Sylvain Malacria, Gilles Bailly

CtrlMouse et TouchCtrl : Dupliquer les Delimiteurs de Mode sur la Souris

Thomas Pietrzak
Universite Lille 1
thomas.pietrzak@lifl.fr

Sylvain Malacria
University College London
sylvain@malacria.fr

Gilles Bailly
CNRS, Telecom ParisTech
gilles.bailly@telecom-paristech.fr

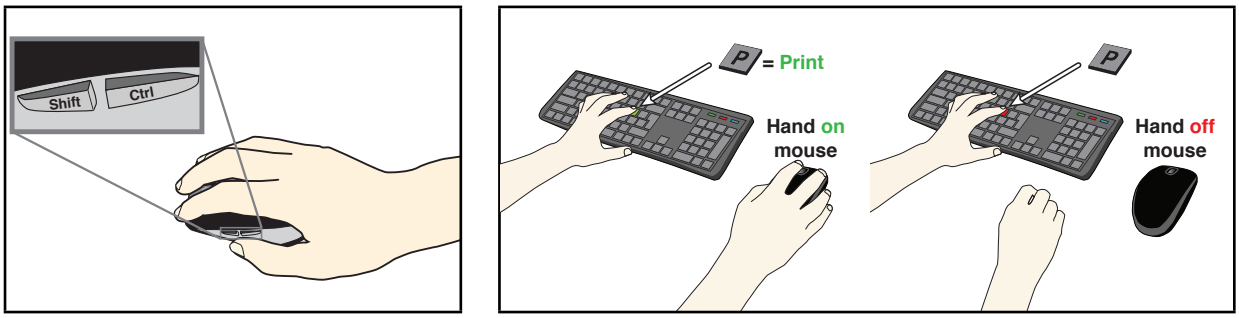

Figure 1: Gauche : CtrlMouse duplique les touches `Ctrl` et `Shift` sur la souris pour délimiter l'entrée de texte de la sélection de commandes et réduire le besoin de coordination des doigts de la main gauche. Droite : TouchCtrl déclenche automatiquement la touche `Ctrl` lorsque l'utilisateur a sa main posée sur la souris.

RESUME

Les touches modifieur du clavier comme `Ctrl` ou `Cmd` (⌘) sont utilisées pour délimiter l'entrée de texte de la sélection de commandes (raccourcis claviers). Dans cet article nous étudions l'impact de la localisation de ces délimiteurs sur la performance et la charge musculaire en les dupliquant sur la souris. À cet effet, nous dérivons deux techniques d'interaction : CtrlMouse duplique les touches `Ctrl` et `Shift` en les associant aux boutons de la souris sous le pouce ; TouchCtrl déclenche automatiquement la touche `Ctrl` lorsque la main est posée sur la souris. Deux expériences en laboratoire révèlent 1) que ces techniques sont d'autant plus adoptées par les utilisateurs que la tâche demande du pointage, 2) le coût temporel des modifieurs sur la sélection de commandes est de $0,21s$, ce qui correspond à $11,9\%$ du temps d'exécution et 3) le temps d'exécution avec CtrlMouse avec un ou deux modifieurs est similaire. Nous avons également déployé ces techniques pour valider de manière écologique les résultats obtenus lors des évaluations en laboratoire. Enfin, nous présentons différents scénarii applicatifs élaborés à partir de CtrlMouse et TouchCtrl.

Mots Cles

modifieurs, délimiteurs de mode, raccourcis clavier

ACM Classification Keywords

H.5.2 Information Interfaces and Presentation (e.g. HCI): Input devices and strategies.

http://dx.doi.org/10.1145/2670444.2670447

INTRODUCTION

Les claviers sont à la fois utilisés pour saisir du texte et pour sélectionner des commandes à l'aide de raccourcis clavier. Le passage entre deux mode se fait à l'aide de *délimiteurs*. Typiquement, les utilisateurs passent en mode commande en pressant des touches *modifieur* telles que `Ctrl` ou `Cmd` (⌘). Ces deux modes ont été amplement et indépendamment étudiés ([3, 11, 12, 29, 45]) mais il y a étonnamment peu de travaux sur la *transition* entre ces deux modes. Pourtant, l'action d'appuyer sur une ou plusieurs touches faisant office de délimiteur de mode influence directement la performance de l'utilisateur en termes de vitesse, précision et confort. En effet, appuyer sur une touche ralentit l'interaction. D'après le modèle GOMS [8], appuyer sur ce modifieurs compte comme une opération d'appui de touche séparée comptabilisant pour $0,28s$ par sélection. Bien que cette pénalité puisse être modeste, elle peut s'avérer importante pour des utilisateurs experts sélectionnant plusieurs centaines voir milliers de commandes par jours. D'autre part, exécuter un raccourci clavier peut introduire une charge musculaire inconfortable pour l'utilisateur [37]. En effet, certains raccourcis (ex : `Ctrl` + `Shift`) nécessitent une coordination délicate des doigts pour réaliser des accords [37].

Dans ce papier, nous étudions l'impact de la localisation des modifieurs sur la performance et le confort de l'utilisateur. Plus précisément, nous étudions l'utilisation de la souris comme alternative pour délimiter l'entrée de texte de la sélection de commandes. Pour cela, nous proposons CtrlMouse, une technique d'interaction qui duplique les modifieurs `Ctrl` et `Shift` sur les boutons du pouce de la souris, (Figure 1-gauche). Les utilisateurs peuvent donc basculer *explicitement* entre les modes de saisie de texte et raccourcis claviers grâce à deux boutons dédiés. Nous présentons également TouchCtrl, une méthode alternative qui déclenche *implicitement* le mode sélection de commandes lorsque la main est posée sur la souris (Figure 1-

droite). Ces deux techniques peuvent être déployées sur un grand nombre d'applications avec une souris ayant des boutons sous le pouce (comme des souris standard Logitech [26]), ou une souris tactile comme l'Apple Magic-Mouse [1] ou la Microsoft TouchMouse [33].

Nous présentons deux évaluations en laboratoire étudiant l'adoption (étude 1) et la performance (étude 2) de CtrlMouse et TouchCtrl. Les résultats 1) montrent que ces techniques sont d'autant plus adoptées par les utilisateurs que la tâche nécessite des opérations de pointage plutôt que de l'entrée de texte ; 2) estiment le coût temporel des modifieurs en fonction de sa *localisation* (souris et clavier), de son mode d'*activation* (implicite ou explicite) et du nombre de modifieurs utilisés.

Nous avons également réalisé une étude qualitative préliminaire sur le terrain pour confirmer de manière écologique les avantages de CtrlMouse et TouchCtrl lors des évaluations en laboratoire. Nous avons déployé nos deux techniques d'interaction auprès de 6 utilisateurs. Les résultat qualitatifs sont partagés parmi les utilisateurs. CtrlMouse est perçu comme un avantage dans le cas des raccourcis à plusieurs modifieurs ou pour les applications graphiques. Inversement, l'implémentation logicielle et matérielle a un impact sur l'utilisabilité.

Enfin nous décrivons 3 extensions tirant parti de CtrlMouse et TouchCtrl : 1) Nous étendons le principe de TouchCtrl au pavé tactile. 2) Nous discutons l'utilisation de nos techniques dans d'autres contextes (sélection d'objets) et 3) avec d'autres techniques d'interaction.

En résumé nos contributions incluent 1) un nouveau regard quantitatif et qualitatif sur les délimiteurs de modes a travers deux évaluations en laboratoire et une évaluation préliminaire sur le terrain ; 2) la conception, la réalisation et l'évaluation de deux techniques d'interaction ; 3) 3 exemples d'applications pour généraliser les concepts introduits par CtrlMouse et TouchCtrl.

ETAT DE L'ART

Ce projet contribue dans le domaine des délimiteurs entre le mode d'entrée de texte et le mode de sélection de commandes. Étant donné que notre approche consiste à modifier le comportement d'un dispositif d'entrée traditionnelle (la souris), nous discutons aussi les travaux qui augmentent les souris et claviers avec de nouvelles capacités.

Delimiteurs et selecteurs de modes

Les *délimiteurs* ont été étudiés à plusieurs reprises [14, 17, 21, 36, 40, 47, 24, 42, 39] en particulier dans le contexte de l'interaction gestuelle et au stylet. Ils permettent aux systèmes interactifs de déterminer la structure lexicale d'une phrase d'entrée [7] et de distinguer différents modes. La plupart des applications que l'on utilise sur nos ordinateurs sont par défaut en mode de saisie de texte : lorsque l'on presse une touche du clavier, la lettre correspondante est saisie et aucune commande est activée. Plusieurs touches du clavier sont alors utilisées comme délimiteurs. Typiquement les touches *modifieurs* permettent de changer l'action d'une touche à l'aide de pseudo-modes. Cependant d'autres touches spéciales permettent de changer de mode. Ainsi `Shift` (pseudo-mode) et `Caps` (mode) permettent de basculer entre minuscules et majuscules. La touche `Ctrl` permet de délimiter le mode de saisie de texte et le mode de sélection de commandes. D'autres touches génériques (`Alt`) ou spécifiques à la plate-forme (`Win`, ⌘) permettent aussi de changer de mode. Les touches modifieurs sont également utilisés comme sélecteur de mode du pointeur souris. Typiquement, la touche `Shift` peut-être utilisée pour sélectionner plusieurs objets à la souris. De manière surprenante, nous n'avons trouvé aucune étude proposant de changer la position du délimiteur différenciant entrée de texte et sélection de commandes. Une exception est le mode de changement d'outils dans certaines applications *orientées souris* telles que des logiciels de montage vidéo (Adobe Premiere), de retouche (Adobe Photoshop), de modélisation 3D (Autodesk 3D Studio), etc. Dans ces applications où le besoin d'entrer du texte est relativement rare, l'utilisateur peut sélectionner un outil fréquent en appuyant sur une lettre (sans appuyer sur un modifieur). Le mode texte n'étant alors activé que dans certains contextes particuliers, généralement en fonction du composant graphique qui a le focus à un certain instant. TouchCtrl est une technique d'interaction inspirée par ces raccourcis clavier à activation *implicite*, que nous cherchons à généraliser à un plus grand nombre d'applications incluant les éditeurs de texte dont les tâches de formatage sont souvent réalisées à la souris ou à l'aide de raccourcis clavier.

Selection de commandes au clavier

Les raccourcis clavier sont des alternatives efficaces à la sélection de commandes à l'aide d'un pointeur [22], en particulier pour les actions répétitives (*e.g.*, copier/coller à répétition) [22, 23, 34]. Plusieurs travaux ont facilité l'utilisation des raccourcis clavier [3, 12, 20, 29, 30, 43]. Par exemple, Grossman *et al*. augmentent l'exposition des raccourcis clavier soit en augmentant la saillance des raccourcis clavier, soit en augmentant le coût de la sélection à la souris [12]. Récemment, Malacria *et al*. ont proposé ExposeHK [29], qui affiche les raccourcis clavier au dessus des boutons de la barre d'outil leur correspondant lorsque la touche `Ctrl` est pressée. Ainsi les utilisateurs n'ont pas besoin de se souvenir des raccourcis clavier mais juste de les reconnaître après avoir pressé le modifieur.

Ces systèmes augmentent la performance des utilisateurs en favorisant l'*apprentissage* des raccourcis clavier. Cependant, ces travaux utilisent toujours les touches modifieurs du clavier pour basculer de mode. De plus certains accords requis pour réaliser ces raccourcis clavier peuvent nécessiter des hyper-extensions, hyper-flexions, écartement des doigts difficile à réaliser et donc entraîner une charge musculaire statique inconfortable [37]. À l'inverse, nous étudions le coût d'appuyer sur la touche `Ctrl` pour améliorer l'*exécution* des raccourcis clavier en minimisant la demande physique.

Claviers et souris avances

Alors que les claviers et souris n'ont pratiquement pas changé conceptuellement ces 30 dernières années, quelques études et produits commerciaux ont récemment proposé des approches différentes [3, 6, 32, 35]. Dans le

contexte de la sélection de commandes, il existe des claviers ayant un petit écran sur chaque touche afin d'y afficher des icônes correspondant à des raccourcis clavier [6, 32, 35]. Des claviers comme celui du Xerox Star [5] ou le Microsoft's Office Keyboard [31] proposent des touches dédiées à des commandes particulières. Cependant ces solutions ne peuvent être appliquées que dans des cas spécifiques, sont limités à des ensembles de commandes réduits et nécessitent de modifier le clavier.

Des études portent aussi sur l'augmentation ou l'utilisation différentes de souris [4, 19, 44, 9]. Par exemple, PadMouse permet à l'utilisateur de sélectionner des commandes en réalisant des gestes sur un touchpad attaché à une souris. Plus spécifique, le système d'exploitation X11 [46, 10] associe par défaut la commande *coller* au bouton du milieu d'une souris, mais cette solution se limite à une unique commande. Cependant ces approches consistent à réaliser la sélection de commande uniquement à la souris, et perdent donc l'expressivité du clavier.

Enfin, certains logiciels permettent de changer les associations entre les touches du clavier ou les boutons de la souris avec les actions à executer. Par exemple, des claviers pour joueurs incluant des touches programmables (*e.g.* Logitech G105 [27]). Le logiciel X-Mouse Button Control permet aussi d'associer les boutons de la souris à des combinaisons arbitraires de touches claviers. Bien que ces logiciels permettent d'implémenter CtrlMouse et TouchCtrl, ils n'informent pas sur l'efficacité des différentes associations. Ce projet vise à explorer une partie de cet immense espace de conception en assignant les modifieurs `Ctrl` et `Shift` sur la souris.

Transition de la main entre le clavier et la souris

Ce travail est aussi lié aux techniques qui combinent les interactions avec le clavier et la souris simultanément [41, 6, 38, 15]. Le trackpoint d'IBM [41], un périphérique de pointage isométrique, a été placé au centre du clavier pour éviter les pertes de temps et les distractions. Des travaux plus récents ont remplacé les périphériques de pointage en intégrant des capteurs capacitifs dans le clavier [6, 15]. Enfin Thumbsense conserve le touchpad, mais utilise les touches du clavier comme boutons de souris [38]. Tout comme ces études, notre projet vise à réduire les transitions de la main entre le clavier et la souris. Cependant, contrairement à ces études, nous n'essayons pas de *déplacer* le pointage dans le clavier, mais nous déplaçons les modifieurs sur le périphérique de pointage. En conséquence nous réduisons le besoin de déplacer la main vers le clavier quand les utilisateurs ont la mains sur la souris et souhaitent saisir un raccourci clavier.

En résumé, à notre connaissance il n'y a pas de travail sur des méthodes alternatives pour délimiter le mode d'entrée de texte du mode sélection de commandes. La touche `Ctrl` est placée historiquement à côté de la touche espace sans que ce choix n'ait été remis en cause suite aux évolutions de l'usage des ordinateurs.

ESPACE DE CONCEPTION

Les claviers sont utilisés à la fois pour saisir du texte et invoquer des commandes à l'aide de raccourcis clavier. Les utilisateurs exécutent généralement des commandes en pressant des touches modifieur dédiées telles que la touche `Ctrl`. Cette approche utilise comme délimiteur une action *explicite* sur le *clavier*. Nous explorons ici des approches complémentaires et étudions comment la souris peut être une *position* alternative de délimiteur, et présentons deux techniques d'interaction qui diffèrent par leur type d'*activation*. *CtrlMouse* duplique les délimiteurs classiques du clavier sur les boutons d'une souris localisés sous le pouce, que l'utilisateur peut alors utiliser pour changer *explicitement* de mode. A contrario, *TouchCtrl* active *implicitement* le mode commande quand la main de l'utilisateur est en contact avec le dispositif de pointage. Ces approches sont résumées en table 1.

		Position	
		Clavier	Souris
Activation	Explicite	Application *textuelle*	*CtrlMouse*
	Implicite	Application *graphique*	*TouchCtrl*

Table 1: Différentes alternatives pour passer du mode entrée de texte au mode commande. Les applications *textuelles* utilisent une action *explicite* de l'utilisateur pour changer le mode du clavier. Les applications *graphiques* disposent de raccourcis *implicites*, sans modifieurs. CtrlMouse duplique les modifieurs sur la souris et demande une action *explicite* de l'utilisateur. TouchCtrl change *implicitement* de mode quand l'utilisateur a sa main sur la souris.

CtrlMouse

CtrlMouse est une technique d'interaction qui facilite l'utilisation des raccourcis clavier quand la main de l'utilisateur est située sur le dispositif de pointage. *CtrlMouse* duplique les délimiteurs classiques du clavier sur les boutons (localisés sous le pouce) d'une souris (Figure 1). Le mode du clavier passe d'*entrée de texte* à *commande* dès que l'utilisateur presse un de ces boutons, et retourne au mode *entrée de texte* quand ces boutons sont relâchés. Comme les touches modifieur sont dupliquées sur la souris, et peuvent être activées avec la main dominante, l'utilisateur n'a plus besoin d'effectuer d'accords sur le clavier, ni de déplacer sa main dominante sur le clavier pour saisir des raccourcis compliqués. Plus précisément, dupliquer les touches de modification sur la souris présente les avantages suivants :

Expressivité. CtrlMouse ne remplace pas les touches de modification du clavier, mais offre une solution alternative à l'utilisateur. Ainsi, l'utilisateur peut choisir d'utiliser les touches de modification qu'il souhaite (souris ou clavier) en fonction de ses préférences.

Coordination des doigts. Les raccourcis clavier traditionnels nécessitent de réaliser des accords avec les doigts. Selon la position des touches et du nombre de modifieurs, les raccourcis clavier peuvent nécessiter une coordination des doigts subtile pour leur exécution, ainsi qu'une charge musculaire statique inconfortable [37]. En divisant les modifieurs et les touches de symbole, CtrlMouse réduit la coordination des doigts en utilisant les deux mains pour réaliser l'exécution des raccourcis.

Association des raccourcis clavier. D'une manière générale, les raccourcis clavier sont associés à une touche du clavier qui va reposer sur une association mnémonique pour faciliter la mémorisation. Cependant, les raccourcis clavier fréquents tendent également à être positionnés sur la partie gauche du clavier pour réduire la distance avec la touche `Ctrl` gauche. En conséquence, il est parfois nécessaire d'augmenter le nombre de modifieurs pour les maintenir dans cette zone. Comme CtrlMouse réduit les contraintes d'accords, les raccourcis claviers peuvent être distribués de manière plus uniforme sur le clavier. Cela donne plus de flexibilité aux concepteurs pour définir une association de commande efficace et réduire les raccourcis nécessitant plusieurs modifieurs [3].

Plusieurs modifieurs. Quand il y a plusieurs modifieurs, CtrlMouse offre un avantage significatif. Il permet d'activer deux modifieurs à la fois en utilisant un appui unique du pouce sur les deux boutons simultanément.

Transitions souris/clavier. Certains raccourcis clavier sont difficiles à exécuter à cause de la position de la touche symbole, en particulier si elle est située sur la partie droite du clavier [3]. Alors que les utilisateurs peuvent réaliser des raccourcis clavier avec la touche `Ctrl` de la partie droite du clavier, nous avons observé que dans les applications orientées souris, ils conservent leur main gauche sur la touche `Ctrl` gauche et utilisent leur main droite pour la touche symbole. Les boutons de CtrlMouse suppriment ces transitions Souris/Clavier avec la main droite.

Compatibilité avec fonctionnalités existantes CtrlMouse écoute l'état des boutons de la souris situés sous le pouce uniqusement lorsqu'une touche du clavier est appuyée. Cela signifie que CtrlMouse est compatible avec les actions usuelles des boutons de la souris situés sous le pouce comme *précédent* et *suivant* pour naviguer dans l'historique d'un navigateur.

Implémentation matérielle. CtrlMouse ne nécessite pas de modifications matérielles car plusieurs souris commerciales (*e.g.*, Logitech M705 [25] ou Logitech M905 [26]) ont déjà deux boutons sous le pouce. Techniquement un appui inférieur à 200*ms* déclenche les actions usuelles, et un appui plus long lorsque une touche du clavier est enfoncée correspond au maintien d'un modifieur.

En résumé, CtrlMouse diffère des raccourcis claviers traditionnels par la *position* du modifieur : les utilisateurs peuvent presser des boutons dédiés sur la souris au lieu de touches modifieur du clavier. Cependant, l'utilisation d'une souris comme nouvelle modalité de basculement en mode sélection de commande offre de nouvelles possibilités illustrées par TouchCtrl.

TouchCtrl

TouchCtrl est une variante de CtrlMouse, limitée à un seul modifieur (typiquement `Ctrl`) mais qui ne requiert pas d'appui *explicite*. Elle est basée sur l'hypothèse que les utilisateurs n'utilisent pas ou peu le clavier pour saisir du texte quand la main droite tient la souris. TouchCtrl bascule *implicitement* le clavier en mode commande dès que la main droite est sur la souris. TouchCtrl a les mêmes avantages que CtrlMouse (meilleure coordination des doigts, moins de déplacements, raccourcis plus simples à associer et moins de transitions clavier/souris) mais requiert également moins d'efforts de la main droite.

Compatibilité. TouchCtrl se base sur l'hypothèse que les utilisateurs ne saisissent pas ou peu de texte quand la main droite est sur la souris. Comme nous l'avons observé, cela correspond en effet à des cas rares. Cependant, s'ils souhaitent saisir du texte (quelques mots dans un champ de recherche par exemple), il leur suffit de lever légèrement la main de la souris pour saisir le texte de la main gauche.

Plusieurs modifieurs. TouchCtrl est plus sensible aux modifieurs multiples que CtrlMouse car l'information binaire (main en contact avec la souris ou non) ne permet de remplacer qu'un seul modifieur. Il est donc toujours nécessaire de presser explicitement un modifieur supplémentaire (sur le clavier, ou éventuellement sur la souris dans un contexte qui combinerait TouchCtrl et CtrlMouse) pour effectuer un raccourci clavier à plusieurs modifieurs. Cependant cette technique d'interaction permet à l'image de CtrlMouse d'encourager les concepteurs à utiliser plus de raccourcis à modifieur unique en exploitant plus la partie droite du clavier.

Implémentation matérielle. TouchCtrl est supporté par les souris multitouch du commerce comme Apple MagicMouse [1] ou Microsoft TouchMouse [33]. De plus, TouchCtrl peut aussi être utilisée avec un touchpad et activer les commandes quand au moins un doigt est en contact avec la surface.

Relocaliser les modifieurs sur la souris peut apporter des bénéfices à la fois qualitatifs et quantitatifs par rapport au clavier. Dans les sections suivantes, nous présentons deux expériences contrôlées et un déploiement préliminaire en situation écologique pour mettre en avant les bénéfices de CtrlMouse et TouchCtrl.

ETUDE 1 : ADOPTION

Cette étude vise à étudier et comprendre l'impact de la *demande d'entrée de texte* (DET) d'une application sur les performances de CtrlMouse et TouchCtrl. Intuitivement, les applications *graphiques* nécessitant beaucoup d'actions au curseur (DET nulle) exploitent au mieux les avantages de nos techniques car la main est déjà posée sur la souris. Au contraire, CtrlMouse et TouchCtrl semblent moins adaptées aux éditeurs de texte (forte DET) car les participants doivent entrer du texte avant chaque sélection de commandes, les deux mains se retrouvant alors positionnées sur le clavier. Cependant, nous ne savons pas à partir de quelle demande d'entrée de texte, les utilisateurs vont préférer adopter une technique plutôt qu'une autre et s'ils sont prêts à abandonner les raccourcis claviers classiques.

Methodologie

Participants et équipement. Nous avons recruté via listes de diffusions douze participants (3 femmes), âgés entre 24 et 35 ans (*moyenne* $= 27,8$, $\sigma = 3,9$). Tous étaient familiers avec les raccourcis clavier. L'expérience a été réalisée sur un PC sous Windows 7 avec un écran LCD (960×1280 pixels). Le clavier était un clavier standard DELL AZERTY. Nous avons utilisé une souris standard

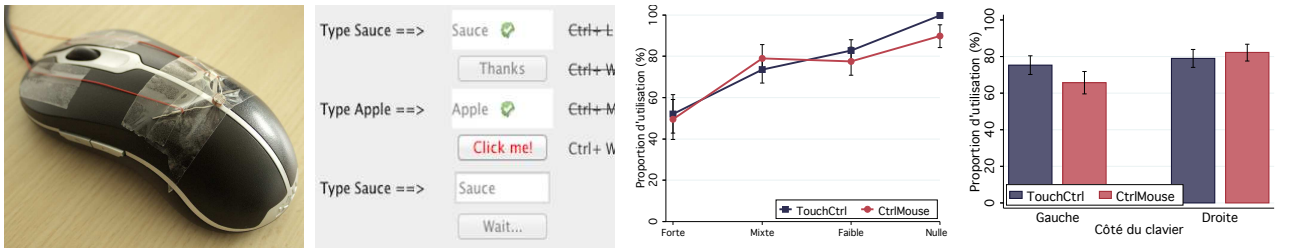

Figure 2: Souris avec des boutons sous le pouce et une photorésistance sur le dos (gauche) ; Fenêtre expérimentale pour la condition mixte (milieu-gauche). Proportion d'utilisation des modifieurs sur la souris par DET (milieu-droite) et par côté du clavier (droite)

DELL avec deux boutons sous le pouce, et augmentée d'une photorésistance pour capter la main de l'utilisateur lorsqu'elle se trouvait dessus (Figure 2, gauche). La photorésistance était interfacée avec Arduino.

Tâche. La tâche est adaptée de la *Home task* de [31] et consiste à faire varier la *demande d'entrée de texte*. Pour chaque essai, les utilisateurs doivent éventuellement saisir du texte, cliquer sur un bouton avec la souris puis exécuter un raccourci clavier affiché dans un label. Le bouton, le(s) champ(s) de texte et le label s'affichaient en début d'essai (Figure 2, milieu-gauche). Ce dispositif expérimental simule le comportement d'un utilisateur expert qui saisit du texte, effectue une sélection et invoque une commande qu'il connaît à l'avance. L'essai se termine et passe au suivant une fois que le raccourci clavier saisi et tous les modifieurs sont relâchés. Ceci empêche un utilisateur de maintenir un modifieur pendant toute l'expérience. Un son informe les utilisateurs lorsqu'ils ont fait une erreur de sélection.

Nous avons simulé l'entrée de texte en demandant à l'utilisateur de saisir des mots simples (sauce, cloak ou apple) avec les deux mains. Ces mots utilisent des lettres des deux côtés du clavier, ce qui assure que les utilisateurs utilisent leurs deux mains lors de l'entrée de texte. Dans la condition à forte demande d'entrée de texte, le pointage servait uniquement à donner le focus au prochain champ de texte dans lequel taper du texte. Nous avons ignoré la condition où les participants ne réalisent aucune tâche de pointage (et entraient uniquement du texte) car les utilisateurs n'auraient aucune raison d'abandonner les raccourcis clavier.

Demande d'entrée de texte (DET). Il y avait quatre niveaux différent :

- forte : tous les essais nécessitent d'entrer du texte avant de sélectionner une commande. L'utilisateur devait néanmoins toujours utiliser la souris pour donner le focus à un champ de texte avant de pouvoir y entrer du texte dedans.
- mixte : la moitié des essais nécessitent d'entrer du texte avant de saisir la commande.
- faible : un tiers des essais nécessitent d'entrer du texte avant de sélectionner la commande.
- nulle : aucun essai ne nécessite de saisir du texte avant de sélectionner une commande.

Technique. Nous avons comparé les deux *techniques* d'interaction CtrlMouse et TouchCtrl.

Raccourcis claviers. Les raccourcis clavier utilisaient un seul modifieur (`Ctrl`). La moitié $\{w,x,f\}$ des 6 lettres utilisées étaient situées sur le côté gauche du clavier. L'autre moitié $\{p,l,m\}$ était située sur le côté droit. Ceci permet d'étudier l'impact du facteur *côté du clavier* sur l'adoption des techniques.

Procédure. Les expérimentateurs expliquaient d'abord la tâche et les deux nouvelles techniques d'interaction (CtrlMouse et TouchCtrl) au participants. Ces derniers avaient pour instruction de réaliser la tâche aussi rapidement et précisément que possible. Il était indiqué également que les participants étaient *libres* de réaliser les sélections avec les raccourcis clavier classiques ou avec la nouvelle technique proposée dans la condition. Ainsi avec CtrlMouse ils pouvaient soit presser le modifieur sur la souris soit sur le clavier. Avec TouchCtrl ils pouvaient soit mettre leur main sur la souris et réaliser le raccourci sans retirer leur main de la souris ou utiliser la touche `Ctrl` du clavier. Les essais étaient chronométrés depuis l'apparition du stimulus jusqu'à ce que la commande soit saisie.

Plan expérimental. L'expérience suivait un plan à mesures répétées $2 \times 8 \times 4 \times 2$ avec les facteurs intra-sujets *Technique* (*CtrlMouse* et *TouchCtrl*), *Bloc* (niveaux 1-8), *demande d'entrée de texte (DET)* (*forte*, *mixte*, *faible*, et *nulle*) et *côté du clavier* (droite ou gauche). L'ordre des *Techniques* et *DET* était balancé entre les participants à l'aide d'un carré latin. Chaque bloc était composé de 3 essais pour chaque *côté du clavier*. Les raccourcis claviers étaient présentés dans un ordre aléatoire. La principale mesure est la proportion de raccourcis claviers classiques utilisés pour chaque condition. Un essai était considéré comme effectué au clavier si la touche `Ctrl` du clavier était pressée au moment de sélection de la commande. L'expérience durait à peu près 20 minutes avec un total de 384 essais par participant. Les participants étaient libre de prendre une pause tous les deux blocs.

Resultats

La proportion d'utilisation de CtrlMouse et TouchCtrl globale était de 75.5% ($\sigma = 37$). La Figure 2 résume les proportions de sélections de commande utilisant les modifieurs sur la souris pour chaque *Technique* en fonction de la *demande d'entrée de texte* (milieu droite) et du côté du clavier (droite). Une analyse de variance montre un effet significatif sur les facteurs suivants : *DET* ($F_{3,33} = 11.43, p < .001$), *Block* ($F_{3,33} = 3.1, p < .05$) et *Côté du clavier* ($F_{1,11} = 8.67, p < .05$).

Les résultats confirment que la proportion d'utilisation des modifieurs sur la souris est inversement proportionnelle

à la demande d'entrée de texte. Celle-ci est de 50.8% ($\sigma = 46.8$) pour une demande *forte* et est significativement inférieure à une demande *mixte* (76.2%, $\sigma = 32.4$), une demande *faible* (80.2%, $\sigma = 29.6$) et une demande *nulle* (94.8%, $\sigma = 19.6$). Nous n'avons pas détecté de différence significative entre les trois dernières demandes. Ces résultats montrent que les participants adoptent nos deux techniques alors qu'ils étaient bien plus entraînés avec les raccourcis claviers classiques. Ils adoptent Ctrl-Mouse et TouchCtrl d'autant plus que la demande d'entrée de texte est faible.

La position des raccourcis claviers (*côté du clavier*) influence également l'adoption de nos techniques. Ctrl-Mouse et TouchCtrl sont significativement plus utilisés lorsque les raccourcis sont sur la droite du clavier (80.5%, $\sigma = 33.4$) plutôt que sur la gauche du clavier (70.5%, $\sigma = 39.8$). Ce résultat s'explique probablement par le fait que les raccourcis claviers classiques avec une lettre sur la droite du clavier nécessitent des aller-retours fastidieux entre la souris et le clavier.

ETUDE 2 : PERFORMANCE

Cette étude vise à mieux comprendre les différences entre CtrlMouse et TouchCtrl dans le contexte où ces techniques s'avèrent le plus utiles, c'est à dire un contexte avec une faible demande d'entrée de texte (voir étude précédente). Nous étudions en particulier l'influence de la *Position* du délimiteur (*Clavier* vs. *Souris*), du mode d'activation (CtrlMouse vs. TouchCtrl) et du *nombre de modifieurs* (1 ou 2) sur les performances.

Cette expérience est similaire à la précédente avec quelques modifications décrites ci-dessous :

- Tâche. Les utilisateurs n'avaient pas de texte à saisir (voir Figure 3, gauche). L'objectif est de simuler des tâches réalisées à la souris interrompues par des raccourcis clavier.
- Raccourcis claviers. Les participants utilisaient soit un (`Ctrl`) soit deux modifieurs (`Ctrl`+`Shift`)
- Techniques. Les participants ne testaient qu'une seule modalité à la fois car nous nous intéressions ici à la performance de chaque technique et non à l'adoption de celles-ci par les utilisateurs. Les participants ne pouvaient donc pas utiliser les touches `Ctrl` et `Shift` du clavier pendant les conditions CtrlMouse et TouchCtrl. Inversement les boutons de la souris étaient désactivés dans la condition *Clavier*.

12 participants (3 femmes) ont été recrutés pour cette expérience.

L'expérience suivait un plan à mesures répétées $3 \times 8 \times 2 \times 2$ avec les facteurs intra-sujets : *Technique* (Ctrl-Mouse, TouchCtrl et Clavier), *Bloc* (niveaux 1-8), *Nombre de modifieurs* (`Ctrl` and `Ctrl`+`Shift`) et *Côté du clavier* (Gauche et Droit). L'ordre de *Technique* était balancé parmi les participants à l'aide d'un carré latin. Il y avait 10 blocs par techniques (dont 2 pour l'entraînement écartés de l'analyse). Chaque bloc comprenait 12 essais (6 lettres x 1 ou 2 modifieurs). Les variables dépendantes sont le temps de complétion d'un essai (essais avec erreurs supprimés), le taux d'erreur, la charge de travail (mesurée avec un questionnaire NASA-TLX [16]) et les préférences utilisateurs. L'expérience durait environ 20 minutes, avec un total de 288 essais par participant.

Resultats

Le temps moyen pour réaliser un essai était de $1.82s$ ($\sigma = 0.59$), avec un taux d'erreur de 3%. La Figure 3 résume les temps de complétion pour les 3 *Techniques* par *Bloc* (milieu-gauche), *Côté du clavier* (milieu-droite) et *Nombre de modifieurs* (droite). Une analyse de variance montre un effet significatif pour tous les facteurs : *Technique* ($F_{2,22} = 9,46, p <,001$), *Bloc* ($F_{7,77} = 9,05, p <,001$), *Côté du clavier* ($F_{1,11} = 11,95, p <,01$) et *Nombre de modifieurs* ($F_{1,11} = 47,9, p <,001$).

Le temps moyen de complétion d'un essai avec TouchCtrl ($1,68s$, $\sigma = 0,51$) était significativement plus rapide que CtrlMouse ($1,88s$, $\sigma = 0,52$; gain de 12,5%) et les raccourcis clavier ($1,89s$, $\sigma = 0,69$; gain de 11,9%). Cependant, il n'y a pas de différence significative entre Ctrl-Mouse et les raccourcis clavier. La sélection de raccourcis avec un seul modifieur ($1,68s$, $\sigma = 0,46$) était significativement plus rapide que les raccourcis à deux modifieurs ($1,96s$, $\sigma = 0,66$; gain de 16,5%).

Il y avait aussi une interaction *Technique* $\times$ *Modifieur* ($F_{2,22} = 6,09, p <,01$, Figure 3 (droite), causée par un temps de sélection plus lent pour les commandes à deux modifieurs avec TouchCtrl et les raccourcis clavier. L'analyse post-hoc montre une différence significative entre la sélection à un seul modifieur avec TouchCtrl ($1,47s$, $\sigma = 0,37$) et toutes les autres conditions. Il est intéressant de noter que CtrlMouse est la seule technique pour laquelle il n'est pas significativement plus long de sélectionner des commandes à deux modifieurs qu'à un seul.

Enfin le temps de sélection moyen global pour les touches du côté gauche du clavier($1,76s$, $\sigma = 0,55$) était significativement plus court que pour les touches du côté droit ($1,87s$, $\sigma = 0,62$), mais le gain (6%) demeure faible.

Preferences subjectives

Nous avons analysé les questionnaires NASA TLX avec des tests de Kruskal Wallis. Seule la *demande physique* a montré un effet significatif entre les *Techniques* ($\chi^2 = 14,8, p <,001$). Les comparaisons pairées de Wilcoxon montrent que TouchCtrl ($4,5/100$, $\sigma = 10,2$) a une demande physique significativement plus faible que CtrlMouse ($13/100$, $\sigma = 16,9$) et les raccourcis clavier ($39/100$, $\sigma = 19$). La différence entre CtrlMouse et les raccourcis clavier était aussi significative.

Cette observation contribue à valider nos choix de conception. Les participants nous ont confirmé lors des entretiens après l'expérience que les sources principales de fatigue sont les allers retours entre le clavier et la souris et la charge musculaire statique. La charge physique moindre de CtrlMouse et TouchCtrl est un signe d'amélioration.

Discussion

Les résultats de cette expérience montrent que l'activation implicite du mode de sélection de commande à la souris avec TouchCtrl a un intérêt. Bien que la différence temporelle par sélection peut sembler petite ($0,21s$), le

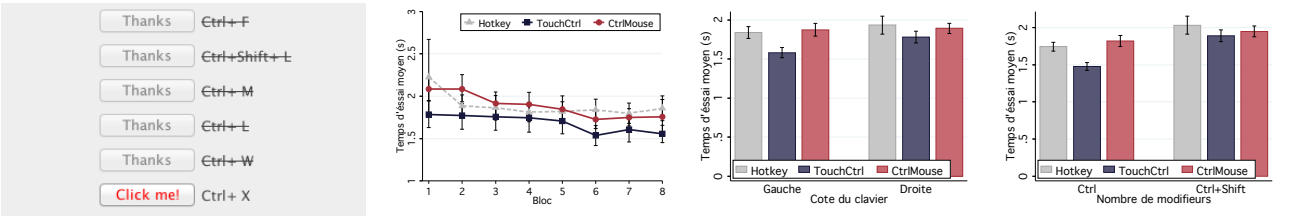

Figure 3: Fenêtre de l'expérience (gauche) : Les participants devaient cliquer sur un bouton, puis réaliser une sélection de commande. Résultats : temps moyen de complétion par bloc (milieu-gauche), temps moyen de complétion par côté du clavier (milieu-droit) et temps moyen de complétion par nombre de modifieurs (droite).

gain est important (11,9%) et s'avère utile lorsque les utilisateurs sélectionnent un grand nombre de commandes. Cette différence est proche de la prédiction par le modèle GOMS [8] si on considère la différence comme un appui sur une touche séparée ($0,28s$). TouchCtrl est donc plus rapide que les raccourcis clavier les plus courants (avec un seul modifieur). De plus CtrlMouse et TouchCtrl ne pénalisent pas la sélection de commandes avec deux modifieurs.

Le manque de différence significative entre CtrlMouse et les raccourcis claviers peut paraître surprenante. Cependant, CtrlMouse demande toujours une action *explicite* de l'utilisateur pour activer les modifieurs, action à laquelle nos participants n'étaient absolument pas entraînés au début de l'expérience (contrairement aux raccourcis clavier). Par conséquent, il est possible que plus d'entraînement ait été nécessaire avec CtrlMouse pour atteindre le seuil de performance de nos participants.

D'autre part, nos résultats n'ont pas révélé d'interaction *Technique* $\times$ *Côté du clavier*, ce qui est assez surprenant. D'un point de vue théorique CtrlMouse (et dans d'autre proportions, TouchCtrl dans le cas d'une commande à un seul modifieur) devraient être aussi rapides pour les touches sur la gauche et la droite du clavier. En effet, il n'a plus à reposer sa main sur le coté gauche du clavier pour accéder à la touche `Ctrl` et pourrait reposer sa main au centre du clavier et ainsi optimiser la distance avec les différentes touches. Nous avons cependant observé que nos participants continuaient à positionner leur main gauche sur la partie gauche du clavier, probablement par habitude.

Notre expérience n'a pas soulevé de différence significative sur la performance de CtrlMouse, quel que soit le nombre de modifieurs (1 ou 2), contrairement à TouchCtrl et raccourcis claviers traditionnels. Cette observation peut s'expliquer par le fait que les deux modifieurs sont positionnés sur la souris, sous le pouce, et que les actions moteurs sont quasi-identiques qu'il s'agisse d'appuyer sur un seul ou les deux boutons.

Nous pouvons également discuter ces résultats au regard des études portant sur l'interaction bi-manuelle. De nombreux travaux s'appuie sur la théorie de la chaîne cinematique de Guiard [13] : « la main non-dominante précède la main dominante et fournit le contexte de l'action ». Nos techniques ne suivent pas ce principe car la main droite définit d'abord le contexte sur la souris avant que la la main gauche agisse sur le clavier en appuyant sur une touche. En ne s'appuyant pas sur le principe de la chaîne cinématique, on pouvait craindre un effort cognitif plus important et donc un temps d'execution plus long. Cependant, les résultats positifs tendent à montrer que la baisse de la complexité bio-mécanique (en simplifiant les mouvements de la main gauche) est plus importante que la hausse de la complexité cognitive. D'autres expériences devront être entreprises pour quantifier précisément l'effort cognitif associé à nos techniques.

Ces expériences en laboratoire nous informent sur l'impact de la localisation et du type d'activation des modifieurs, l'impact de la demande d'entrée de texte ainsi que sur le coût du nombre de modifieurs. Cependant, elles ne nous informent pas sur le comportement des utilisateurs en situation réelle. La section suivante apporte quelques éléments de réponse préliminaires sur cette question.

DÉPLOIEMENT

Nous avons implémenté CtrlMouse et TouchCtrl afin de (1) faciliter leur intégration dans les systèmes existants et (2) confirmer de manière écologique les résultats positifs obtenus lors des évaluations en laboratoire. Dans cette section nous décrivons l'implémentation des deux techniques d'interaction et les premiers résultats qualitatifs de leur déploiement sur le terrain.

Implementations

TouchCtrl

TouchCtrl est une application pour MacOSX qui s'exécute en tâche de fond et doit être utilisé avec un dispositif de pointage multi-points d'Apple comme la Magic Mouse[1] ou le Magic Trackpad[28]. Elle a été réalisée en Objective-C sous les API Cocoa et Carbon et repose sur deux composants logiciels principaux : (1) détection du contact de la main sur la souris et (2) émulation de l'exécution des raccourcis claviers.

Notre implémentation utilise le framework privé d'Apple *MultitouchSupport* pour identifier si la main est en contact avec la souris. Ce framework permet à un programme d'accéder à tout instant à la liste des points en contact (position, taille et orientation du point de contact) avec un dispositif de pointage multi-points d'Apple. Notre implémentation considère que la main est en contact avec la souris si au moins un point de contact existe.

Nous avons utilisé les *event taps* d'Apple pour émuler l'exécution des raccourcis claviers. Notre implémentation intercepte les *event taps* d'appui de touches et modifie sa propriété champs *modifierflags* pour prétendre que la touche ⌘ est pressée si la main est en contact avec la souris. Nous injectons alors le champs modifié dans les événements qui sont renvoyés au système. Ceci a pour effet d'activer le raccourci clavier correspondant et d'éviter

de simuler que la touche ⌘ est pressée en permanence, typiquement pour éviter de modifier les cliques souris.

CtrlMouse

CtrlMouse peut être déployé avec n'importe quelle souris ayant au moins deux boutons à proximité du pouce. Nous avons implémenté la technique sous Windows avec AutoHotkey [2]. Cet outil permet de manipuler les événements d'entrée, et en particulier de les rediriger et en générer. Ainsi nous déclenchons les événement de la touche `Ctrl` en réaction aux événements du premier bouton du pouce de la souris et `Shift` sur le deuxième.

Cet outil permet soit de remplacer le comportement habituel des touches et boutons, soit d'y ajouter des actions. Certains utilisateurs souhaitent conserver leur utilisation classique, par exemple pour passer aux pages précédentes et suivantes dans un navigateur. D'autres ne souhaitent pas ce comportement. Nous avons donc réalisé deux implémentations.

Resultats qualitatifs du deploiement

Nous avons déployé nos techniques auprès de 6 utilisateurs (4 CtrlMouse et 2 TouchCtrl) pendant 2 mois. Les applications étaient lancées au démarrage de la session.

Les utilisateurs ont globalement apprécié le fait de pouvoir effectuer `Ctrl` et `Shift` sur la souris, surtout au fur et à mesure qu'ils découvraient ce qu'ils pouvaient en faire. Un utilisateur de CtrlMouse et un de TouchCtrl soulignent le confort pour les raccourcis à deux modifieurs. Deux participants disent apprécier l'utilisation des modifieurs sur la souris de CtrlMouse pour réaliser des sélections multiples. Un utilisateur de TouchCtrl apprécie pouvoir copier et coller des fichiers dans Finder à l'aide d'une simple touche, et plusieurs suggèrent l'intérêt de CtrlMouse pour le contrôle de paramètres de commandes à la souris (aimantation sur grille, contraintes, zoom, ajouter/supprimer à la sélection). Ce potentiel n'était pas exploité par tous les utilisateurs, deux de nos utilisateurs n'utilisant pratiquement pas d'application orientées graphique.

Faux positifs. L'intérêt de TouchCtrl est l'activation implicite du mode commande quand la main est sur la souris. Cependant les actions implicites possèdent des risques de faux positifs. Nos utilisateurs ont signalé avoir rencontré le problème quelques fois. Il est important d'éviter les faux positifs sur les commandes critiques (⌘-`Q`) et de pouvoir facilement recouvrir des autres. Ainsi les commandes critiques doivent conserver une activation explicite, par exemple avec un modifieur sur le clavier ou la souris. La posture kinesthésique servant à maintenir le quasi-mode n'évoque pas ce changement de mode, car les utilisateurs sont habitués à cette posture et à ce qu'elle n'influe pas sur leurs interactions. Il faut donc veiller à fournir un retour visuel ou haptique supplémentaire, suggérant le mode courant. Cependant, un de nos utilisateurs a déclaré s'être habitué à terme à TouchCtrl.

Implémentation. Les utilisateurs ont soulevé des problèmes d'implémentation qui ont influencé l'utilisation de ces techniques. Deux utilisateurs de CtrlMouse on désactivé fréquemment le système car ils utilisent les boutons du pouce de la souris pour naviguer dans l'historique de leur navigateur web. L'implémentation permet de réaliser les deux fonctionnalités en même temps, cependant la page change quand les boutons sont utilisés. Une implémentation idéale ne déclencherait les modifieurs si et seulement si aucune action n'est associée aux boutons au moment où ils sont utilisés.

Matériel. Un utilisateur a signalé que les boutons du pouce de sa souris (Logitech M500) ne sont pas à la position de repos du pouce. Cela permet d'éviter des activations accidentelles avec une utilisation classique de ces boutons. Cependant dans notre cas ces boutons activent des quasi-modes et non des commandes, donc ce risque est moindre. Avoir les boutons dans une position confortable reste souhaitable pour des sélections multiples, qui nécessitent des clics gauche avec l'index et des mouvements de la souris en plus du maintien d'un bouton du pouce.

Ce déploiement nous a appris tant pour l'utilisation que pour l'implémentation. Au niveau de l'utilisation, les utilisateurs ont commencé à s'approprier les techniques et à y trouver leurs propres usages, au delà des raccourcis clavier, ce qui est particulièrement encourageant. Nous avons aussi constaté que l'implémentation n'était pas aussi triviale que prévue, ce qui est difficile à déceler avec des expériences en laboratoire. Nous avons mis en avant des problèmes ergonomiques qui peuvent nuire à l'utilisabilité des techniques, ainsi que des solutions pour y remédier.

DISCUSSION

Nous discutons dans cette section d'autres usages potentiels de CtrlMouse et TouchCtrl.

Application aux touchpads : TouchCtrlPad

De plus en plus d'utilisateurs d'ordinateurs, et en particulier de portables, utilisent un touchpad. Les études présentées plus haut ont montré certains avantages à l'utilisation de CtrlMouse ou TouchCtrl avec une souris. Cependant, il n'était pas évident que l'on puisse transposer ces techniques aux touchpads.

Inspirés par [38, 18], nous proposons TouchCtrlPad. Lorsque les utilisateurs posent un pouce sur le pavé numérique, le mode `Ctrl` est activé. Les autres doigts sont alors libres pour appuyer la lettre d'un raccourci clavier. Cette technique a plusieurs avantages. Tout d'abord, activer le mode `Ctrl` ne nécessite pas de flexion du pouce, ce qui est potentiellement moins contraignant que l'utilisation de les touches `Ctrl` ou ⌘. Ensuite, tous les doigts longs peuvent rester sur la seconde rangée de lettres, la position par défaut. C'est une différence avec les raccourcis claviers sous Windows, pour lesquels la touche `Ctrl` est souvent utilisée avec l'auriculaire alors que la touche ⌘ sous Mac est plus utilisée avec le pouce. Cependant, même sous Mac, selon la position de la touche symbole du raccourci, la main peut être amenée à tourner dans une position inconfortable. Enfin, sur un touchpad multi-points, l'utilisateur peut poser un ou deux doigts pour différencier le mode `Ctrl` et le mode `Ctrl+Shift`.

Nous avons implémenté TouchCtrlPad sur MacOS. Les premiers tests effectués par les auteurs pour exécuter des

raccourcis claviers étaient positifs. Cependant, de nombreuses activations accidentelles ont été observées lors des premiers déploiements. La raison est que la position de repos des pouces est très proche et parfois sur les coins du pavé numérique. Pour véritablement bénéficier des avantages de cette technique, il s'agira de filtrer les événements du touchpad venant des coins supérieurs pour éviter les activations accidentelles. Une autre solution consisterait à revoir la position des touches modifieur sur le clavier.

Interactions avancees

L'objectif de notre étude était de comprendre l'impact de la position et du type d'activation des délimiteurs de mode du clavier. Nous avons donc conçu et expérimenté CtrlMouse et TouchCtrl séparément. Cependant il est tout à fait envisageable de combiner CtrlMouse et TouchCtrl. Par exemple `Ctrl` serait activé en posant la main sur la souris, et les deux boutons dupliqueraient les comportements de `Shift` et `Alt`.

Au delà de la selection de commandes

Les touches `Ctrl` et `Shift` ne sont pas seulement utilisées pour la sélection de commande. Elles permettent aussi de contrôler des paramètres de commandes réalisées à la souris. Suite au déploiement de notre technique, nous avons observé des usages de CtrlMouse et TouchCtrl pour ce genre d'application. Par exemple lors du déplacement d'un objet sur le bureau ou dans un dossier, les modifieurs servent à basculer entre les modes de copie, déplacement ou création de lien. Avec un outil de loupe, on peut choisir entre zoom et dé-zoom. Il est aussi possible d'inclure ou d'exclure des éléments d'une sélection multiple. Toutes ces interactions utilisent typiquement deux mains : une pour la manipulation du pointeur et une pour la sélection des modifieurs. CtrlMouse et TouchCtrl permettent de réaliser ces opérations avec une seule main.

CONCLUSION

Nous avons étudié l'impact de la *position* et du *mode d'activation* des délimiteurs de modes d'entrée de texte et de sélection de commandes. Pour cette étude nous avons présenté CtrlMouse et TouchCtrl, deux techniques d'interaction qui dupliquent les touches modifieurs du clavier sur la souris. CtrlMouse utilise deux boutons sous le pouce qui dupliquent les touches `Ctrl` et `Shift` lorsque l'utilisateur les presse *explicitement*. TouchCtrl active de manière *implicite* le modifieur `Ctrl` lorsque la main est posée sur la souris. Nous avons présenté deux expériences expérimentales. La première montre que l'adoption de la technique est supérieure quand la tâche nécessite peu d'entrée de texte. La deuxième montre que TouchCtrl procure un gain de performance par rapport aux raccourcis claviers classique, et que CtrlMouse est efficace lorsqu'il y a plusieurs modifieurs. Nous avons déployé ces techniques sur une longue durée chez 6 utilisateurs. Nous avons observé des comportements intéressants au delà de la sélection de commande, telle que la manipulation de paramètres de commandes effectuées à la souris avec une seule main. Nous avons aussi détecté des problèmes ergonomiques pouvant impacter l'utilisabilité de ces techniques, et proposé des solutions.

BIBLIOGRAPHIE

1. Apple Magic Mouse. **http://www.apple.com/magicmouse/**.
2. AutoHotkey. **http://www.autohotkey.com/**.
3. Bailly G., Pietrzak T., Deber J. & Wigdor D. Métamorphe: Augmenting hotkey usage with actuated keys. In *CHI '13* (2013), 563–572.
4. Balakrishnan R. & Patel P. The padmouse: facilitating selection and spatial positioning for the non-dominant hand. In *CHI '98* (1998), 9–16.
5. Bewley W. L., Roberts T. L., Schroit D. & Verplank W. L. Human factors testing in the design of xerox's 8010 star office workstation. In *CHI '83* (1983), 72–77.
6. Block F., Gellersen H. & Villar N. Touch-display keyboards: transforming keyboards into interactive surfaces. In *CHI '10* (2010), 1145–1154.
7. Buxton W. Lexical and pragmatic considerations of input structures. *SIGGRAPH Comput. Graph. 17*, 1 (1983), 31–37.
8. Card S., Newell A. & Moran T. *The Psychology of Human-Computer Interaction*. L. Erlbaum Associates Inc., 1983.
9. Cechanowicz J., Irani P. & Subramanian S. Augmenting the mouse with pressure sensitive input. In *CHI '07* (2007), 1385–1394.
10. Chapuis O. & Roussel N. Copy-and-paste between overlapping windows. In *CHI '07* (2007), 201–210.
11. Evans A. & Wobbrock J. Taming wild behavior: the input observer for text entry and mouse pointing measures from everyday computer use. In *CHI '12* (2012), 1947–1956.
12. Grossman T., Dragicevic P. & Balakrishnan R. Strategies for accelerating on-line learning of hotkeys. In *CHI '07* (2007), 1591–1600.
13. Guiard Y. Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. *Journal of Motor Behavior*, 19 (1987), 486–517.
14. Guimbretiére F. & Winograd T. Flowmenu: combining command, text, and data entry. In *UIST '00* (2000), 213–216.
15. Habib I., Berggren N., Rehn E., Josefsson G., Kunz A. & Fjeld M. Dgts: Integrated typing and pointing. In *INTERACT '09* (2009), 232–235.
16. Hart S. & Staveland L. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Human mental workload* (1988), 139–183.
17. Hinckley K., Baudisch P., Ramos G. & Guimbretiere F. Design and analysis of delimiters for selection-action pen gesture phrases in scriboli. In *CHI '05* (2005), 451–460.
18. Hinckley K., Czerwinski M. & Sinclair M. Interaction and modeling techniques for desktop two-handed input. In *UIST '98* (1998), 49–58.
19. Kim S., Kim H., Lee B., Nam T.-J. & Lee W. Inflatable mouse: volume-adjustable mouse with air-pressure-sensitive input and haptic feedback. In *CHI '08* (2008), 211–224.
20. Krisler B. & Alterman R. Training towards mastery: overcoming the active user paradox. In *NordiCHI '08* (2008), 239–248.
21. Kurtenbach G. & Buxton W. Issues in combining marking and direct manipulation techniques. In *UIST '91* (1991), 137–144.
22. Kurtenbach G. P. *The design and evaluation of marking menus*. PhD thesis, Toronto, Canada, 1993.
23. Lane D., Napier A., Peres C. & Sandor A. The Hidden Costs of Graphical User Interfaces: The Failure to Make the Transition from Menus and Icon Tool Bars to Keyboard Shortcuts. *IJHCI 18* (2005), 133–144.
24. Li Y., Hinckley K., Guan Z. & Landay J. A. Experimental analysis of mode switching techniques in pen-based user interfaces. In *CHI '05* (2005), 461–470.
25. Logitech 705 mouse controller. **http://www.logitech.com/en-nz/product/7108**.
26. Logitech Anywhere 905 mouse controller. **http://www.logitech.com/product/anywhere-mouse-mx**.

27. Logitech G105 gaming keyboard. **http://gaming.logitech.com/en-us/product/g105-gaming-keyboard**.
28. Apple magic trackpad. **http://www.apple.com/magictrackpad/**.
29. Malacria S., Bailly G., Harrison J., Cockburn A. & Gutwin C. Promoting hotkey use through rehearsal with exposehk. In *CHI '13* (2013), 573–582.
30. Malacria S., Scarr J., Cockburn A., Gutwin C. & Grossman T. Skillometers: Reflective widgets that motivate and help users to improve performance. In *UIST '13* (2013), 321–330.
31. McLoone H., Hinckley K. & Cutrell E. Bimanual interaction on the microsoft office keyboard. In *INTERACT'03* (2003), 49–56.
32. Microsoft Adaptive Keyboard. **http://www.microsoft.com/appliedsciences/content/projects/uist.aspx**.
33. Microsoft Touch Mouse. **http://www.microsoft.com/hardware/en-us/p/touch-mouse**.
34. Nielsen J. *Usability Engineering*. Morgan Kaufmann Publishers Inc., 1993.
35. Optimus Maximus Keyboard. **http://www.artlebedev.com/everything/optimus/maximus/**.
36. Pook S., Lecolinet E., Vaysseix G. & Barillot E. Control menus: excecution and control in a single interactor. In *CHI EA '00* (2000), 263–264.
37. Putz-Anderson V. *Cumulative trauma disorders: A manual for musculoskeletal diseases of the upper limbs*. Taylor & Francis London, 1988.
38. Rekimoto J. Thumbsense: automatic input mode sensing for touchpad-based interactions. In *CHI EA '03* (2003), 852–853.
39. Ruiz J., Bunt A. & Lank E. A model of non-preferred hand mode switching. In Proc. *GI '08*, Canadian Information Processing Society (2008), 49–56.
40. Ruiz J. & Li Y. Doubleflip: a motion gesture delimiter for mobile interaction. In *CHI '11* (2011), 2717–2720.
41. Rutledge J. D. & Selker T. Force-to-motion functions for pointing. In *INTERACT '90* (1990), 701–706.
42. Saund E. & Lank E. Stylus input and editing without prior selection of mode. In *UIST '03* (2003), 213–216.
43. Tak S., Westendorp P. & van Rooij I. Satisficing and the use of keyboard shortcuts: Being good enough is enough? *Interacting with Computers 5*, 25 (2013).
44. Villar N., Izadi S., Rosenfeld D., Benko H., Helmes J., Westhues J., Hodges S., Ofek E., Butler A., Cao X. & Chen B. Mouse 2.0: multi-touch meets the mouse. In *UIST '09* (2009), 33–42.
45. Wobbrock J. O. & Myers B. A. Analyzing the input stream for character- level errors in unconstrained text entry evaluations. *ACM ToCHI 13*, 4 (2006), 458–489.
46. Fondation X.org. **http://www.x.org/wiki/**.
47. Zhao S. & Balakrishnan R. Simple vs. compound mark hierarchical marking menus. In *UIST '04* (2004), 33–42.