HAL
open science

Utilisation de la force sur pavés tactiles pour le défilement automatique

Axel Antoine, Sylvain Malacria, Géry Casiez

To cite this version:

Axel Antoine, Sylvain Malacria, Géry Casiez. Utilisation de la force sur pavés tactiles pour le défilement automatique. Actes de la 28ième conférence francophone sur l'Interaction Homme-Machine, Oct 2016, Fribourg, Suisse. pp.264-270, 10.1145/3004107.3004137 . hal-01384315

HAL Id: hal-01384315

https://hal.science/hal-01384315v1

Submitted on 19 Oct 2016

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

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

Utilisation de la force sur pavés tactiles pour le défilement automatique

Axel Antoine
Inria et Univ. Lille 1, France
ax.antoine@gmail.com

Sylvain Malacria
Inria, France
sylvain.malacria@inria.fr

Géry Casiez
Univ. Lille 1
gery.casiez@univ-lille1.fr

© ACM, 2016. This is the author's version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version was published in Actes de la 28ème conférence francophone sur l'Interaction Homme-Machine, 2016.
http://dx.doi.org/10.1145/3004107.3004137

Résumé
Les systèmes d'exploitation supportent le défilement automatique (autoscroll) dans le but de permettre aux utilisateurs de faire défiler une fenêtre depuis le mode glisser : l'utilisateur déplace le pointeur de la souris vers le bord de la vue, ce qui a pour effet de commencer à faire défiler cette dernière "automatiquement". La vitesse de défilement est généralement associée à la distance entre le pointeur de la souris et le bord de la fenêtre. Cette approche souffre de plusieurs problèmes, notamment que l'espace autour de la fenêtre peut être limité, voire inexistant, par exemple quand une vue est maximisée. Un autre problème est que pour certaines opérations, comme le glisser-déposer d'objets, il est possible que la destination se trouve dans une autre vue que celle d'origine, ce qui crée une situation potentiellement ambiguë pour le système. Dans cet article, nous présentons ForceEdge, une nouvelle technique d'autoscroll qui tire parti des capacités des pavés tactiles récents pour s'affranchir des problèmes inhérents aux méthodes conventionnelles d'autoscroll. Nous présentons les fondements théoriques de ForceEdge et l'implémentation d'un démonstrateur qui permet de comparer ForceEdge aux techniques d'autoscroll par défaut de Mac OS X.

Mots Clés
Défilement automatique; pavé tactile; défilement.

Abstract
Operating systems support autoscroll to allow users to scroll a window while in dragging mode: the user moves the pointer near the window's edge to trigger an "automatic" scrolling. Scrolling rate is typically proportional to the distance between the pointer and the window's edge. This approach suffers from several problems, especially when the window is maximized resulting in a very limited space around the window. An other problem is that for some operations, such as object drag-and-drop, the source and destination might be located in different windows, making it complicated for the computer system to understand user's intention. In this paper, we present ForceEdge, a novel autoscroll technique relying on trackpads with force-sensing capabilities to alleviate all the problems related to autoscroll. We present the theoretical foundations of ForceEdge and the implementation of a demonstrator that can be used to compare ForceEdge to the other autoscroll methods.

Author Keywords
Autoscroll; trackpad; scrolling.

ACM Classification Keywords
H.5.2 [Information interfaces (e.g. HCI)]: User interfaces

Introduction
Faire défiler une fenêtre pendant que le pointeur est en mode *glisser* [1] est une tâche fréquente dans les systèmes d'exploitation actuels, typiquement lorsque l'utilisateur souhaite sélectionner une large portion de texte, ou encore lorsqu'il souhaite déplacer un fichier dans un dossier qui n'est pas visible dans la vue courante. Dans ces situations il est nécessaire de faire défiler le contenu d'une fenêtre pour faire apparaître la cible de l'opération de *glisser* à l'écran. Cependant, l'état *glisser* empêche d'interagir avec les contrôles habituels tels que la barre de défilement, et les contrôles spécifiques aux périphériques comme la molette de la souris ne fonctionnent pas toujours.

Pour remédier à ces problèmes, les systèmes d'exploitation actuels supportent l'opération de défilement automatique (autoscroll) : l'utilisateur déplace le pointeur de la souris vers le bord de la vue, ce qui a pour effet de commencer à faire défiler cette dernière *automatiquement*. En réalité, la vitesse de défilement est généralement associée à la distance entre le pointeur de la souris et le bord de la fenêtre [1], ce qui peut poser quelques problèmes de contrôle et d'activation accidentelle, typiquement quand la vue est maximisée et par conséquent que l'espace entre celle-ci et le bord de l'écran est très limité (voir nul).

Certains pavés tactiles récents (typiquement, l'Apple Magic Trackpad 2 qui intègre la technologie Force Touch) permettent non seulement de manipuler le pointeur souris, et par conséquent d'effectuer des opérations de glisser-déposer ou de sélection, mais sont également capables de fournir une approximation de la force exercée en Newton pour chacun des points de contact. Il se distinguent de la génération actuelle de pavés tactiles qui permettent de mesurer ce qui est communément appelé la pseudo-pression, à savoir la surface du doigt en contact avec le pavé tactile. Ce nouveau canal d'entrée pour un périphérique de pointage est particulièrement intéressant dans le cadre de l'autoscroll, en particulier si la force est directement associée à la vitesse de défilement. D'une part, contrôler la force exercée sur le pavé tactile ne nécessite pas de déplacer le pointeur, ce qui permet de limiter le contrôle de l'autoscroll à une zone relativement petite. D'autre part, les périphériques isométriques sont mieux adaptés au contrôle en vitesse

[1] Le *glisser* consiste à appuyer sur un bouton du dispositif de pointage, puis à déplacer le pointeur souris. On reste dans ce mode tant que le bouton du dispositif de pointage reste appuyé.

que les périphériques isotoniques [12]. Par conséquent l'utilisation de la force semble mieux adaptée au contrôle de la vitesse de défilement que l'utilisation de la distance qui sépare le pointeur du bord d'une fenêtre. Enfin, un simple seuil sur la force exercée peut être utilisé pour permettre à l'utilisateur de spécifier s'il souhaite faire défiler la fenêtre.

Dans cet article, nous présentons la conception et l'implémentation de *ForceEdge* (figure 1), une nouvelle technique d'interaction d'autoscroll qui utilise la position relative du pointeur par rapport à la fenêtre pour contrôler la direction du défilement et la force exercée sur le pavé tactile pour contrôler la vitesse de défilement. ForceEdge s'affranchit ainsi des limitations inhérentes aux techniques existantes d'autoscroll.

Nous présentons dans un premier temps l'état de l'art sur l'autoscroll, les connaissances relatives au contrôle de force par les utilisateurs, et les techniques d'interaction exploitant la force pour contrôler le défilement de fenêtres. Nous détaillons ensuite ForceEdge et ses propriétés interactionnelles. Enfin, nous discuterons nos travaux futurs.

État de l'art

Autoscroll

Aceituno et al. ont analysé et comparé le comportement de 19 techniques d'autoscroll intégrées à des systèmes d'exploitation et applications commerciales [1]. Leur analyse montre que la quasi totalité des techniques d'autoscroll utilise un contrôle de vitesse et associe la vitesse de défilement à la distance entre le pointeur de la souris et le bord de la fenêtre, approche qui pose plusieurs problèmes. Tout d'abord, l'espace autour de la fenêtre peut être limité dans de nombreuses configurations, par exemple quand une vue est maximisée ce qui offre peu de contrôle à l'utilisateur sur la vitesse de défilement. De plus, les recherches précédentes ont suggéré que le contrôle de vitesse est inapproprié à une interaction utilisant un dispositif isotonique comme le pavé tactile [12]. Ensuite, pour certaines opérations comme le glisser-déposer d'objets, il est possible que la destination se trouve dans une autre vue que celle d'origine, ce qui crée une situation ambiguë où le système ne sait pas quelle vue faire défiler. Pour lever cette ambiguïté le système doit donc employer des méthodes plus ou moins robustes, souvent en minimisant la taille de la zone de contrôle de vitesse de l'autoscroll [1]. Enfin, les techniques actuelles suggèrent encore très mal comment faire défiler les vues à l'aide de l'autoscroll, et plus particulièrement les position et taille des zones de contrôle de l'autoscroll, ce qui est problématique dans des situations où la cible se situe dans une autre vue que celle d'origine [1].

Plusieurs nouvelles techniques ont été proposées. Bardon et al. proposent dans un brevet de contrôler la vitesse de défilement de l'autoscroll à l'aide d'un contrôle de position, où chaque mouvement de pointeur effectué à l'extérieur de la fenêtre et s'éloignant de celle-ci fait défiler son contenu [3]. Cependant, cette technique souffre toujours d'un effet de bord puisque le pointeur ne peut théoriquement pas s'éloigner indéfiniment de la fenêtre. Malacria et al. ont proposé PushEdge et SlideEdge [8], deux autres techniques d'autoscroll qui utilisent un contrôle de position, mais qui "bloquent" le pointeur lorsqu'il dépasse les limites de la fenêtre pendant la sélection. Tous les événements envoyés par le dispositif de pointage se retrouvent alors convertis en défilement de fenêtre au lieu de déplacer le pointeur souris. Ces techniques ne souffrent plus du problème d'espace limité lorsqu'une fenêtre est maximisée, mais ne permettent pas de résoudre le problème d'ambiguïté potentiel mentionné précédemment. Enfin, Kwatinetz et al. ont proposé d'associer la vitesse de défilement pendant les opérations de glisser-déposer à l'accélération du pointeur, mais cette technique semble difficile à maitriser [1, 6].

D'autres techniques se sont intéressées spécifiquement (et uniquement) au problème d'ambigüité de l'activation [7, 5, 4]. Berry et al. ont proposé d'utiliser un bouton physique dédié sur le dispositif de pointage [5]. Li et al. ont eux proposé d'utiliser une touche du clavier [7]. Enfin, Belfiore et al. [4] proposent d'activer l'autoscroll uniquement lorsque le pointeur souris entre dans la zone de contrôle avec une vitesse supérieure à un certain seuil.

Contrôle de la force
La force maximale que peut exercer un doigt est de l'ordre de 17 N pour une femme et 45 N pour un homme, avec une résolution de 0.3 N, indépendamment de la force exercée [11]. Le Force Touch d'Apple permet de mesurer des forces jusque 10 N, ce qui est inférieur à la force maximale que peut exercer un utilisateur mais suffisant pour des appuis avec des forces modérées. La résolution de contrôle de la force nous informe jusqu'à quel niveau de détail il peut être nécessaire de définir la fonction de transfert de contrôle de l'autoscroll.

Force et défilement de vues
Plusieurs travaux de recherche ont proposé d'utiliser la force ou la pseudo-pression pour contrôler le défilement d'une vue dans un contexte différent de celui de l'autoscroll. Ramos et al. ont notamment proposé une technique de manipulation d'ascenseur de défilement à l'aide d'un stylet de tablette tactile qui modifie la précision du défilement en fonction de la force exercée sur la pointe du stylet [10]. Miyaki et Rekimoto [9] ont eux proposé d'utiliser la force pour permettre à l'utilisateur de contrôler la vitesse de défilement d'une liste sur smartphone : la liste commence à défiler vers le haut (respectivement vers le bas) dès qu'un seuil de force est dépassé sur la partie haute (respectivement basse) de l'écran. Enfin, Baglioni et al. [2] ont présenté Flick-and-Brake, un concept de techniques de

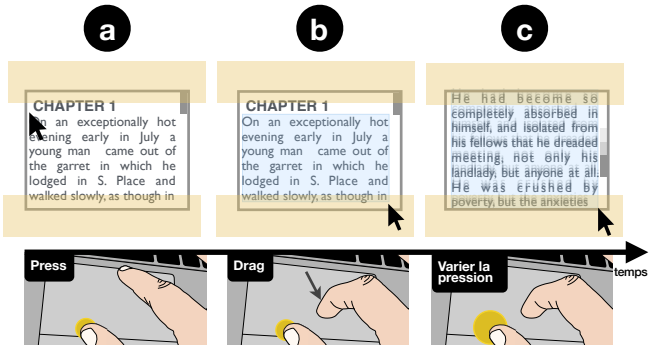

Figure 1: L'utilisateur amorce son opération de sélection de texte. Il appuie sur le bouton physique du pavé tactile (a) puis déplace son doigt sur la surface pour *glisser* le pointeur jusque dans la zone de contrôle (b). L'utilisateur varie la vitesse de défilement de la fenêtre en appuyant plus ou moins fort sur le pavé tactile.

défilement pour smartphones et tablettes qui permet à l'utilisateur de contrôler la vitesse de défilement une fois la liste lancée en appliquant une force (ou pseudo-pression) plus ou moins importante sur l'écran du dispositif. Bien que ces techniques utilisent la force ou pseudo-pression pour contrôler le défilement d'une vue, elles ont été conçues et étudiées dans des contextes d'interaction différents (interaction mobile ou au stylet) et ne se sont pas intéressées au problème spécifique de l'autoscroll, qui diffère grandement des techniques de navigation conventionnelles.

FORCEEDGE

ForceEdge a été conçue pour outrepasser les limitations inhérentes de l'autoscroll, en s'affranchissant des contraintes liées à l'utilisation habituelle de la *position* du pointeur. La vitesse de défilement n'est plus fonction de la distance au bord de la fenêtre mais de la force exercée sur le pavé tactile. Plus celle-ci est importante, plus la vitesse de

défilement sera grande. Enfin, l'effort supplémentaire demandé par ForceEdge comparativement à une opération de *glisser* n'implique pas de déplacements supplémentaires du bras ou de la main mais seulement le contrôle de la force appliquée par un doigt, ce qui ne demande pas d'effort physique considérable et ne change pas les habitudes des utilisateurs de manière radicale.

Comportement
Suggestion. Lorsque l'utilisateur commence une opération de *glisser* (c'est à dire que le bouton du pavé tactile a été enfoncé puis le pointeur déplacé d'au moins un pixel), les vues supportant l'autoscroll affichent leurs zones de contrôle en transparence (figure 1). Ces zones de contrôle, situées à proximité des bords de la vue, indiquent à la fois la vue et la direction du défilement (typiquement le bord du bas pour faire défiler la vue dans cette direction).

Activation. L'utilisateur peut activer ForceEdge en déplaçant le pointeur souris dans une zone active. La force exercée sur le pavé tactile est, par définition, non-nulle lors d'une opération de *glisser* puisque le bouton est enfoncé. Un seuil minimal est donc à atteindre pour démarrer le défilement de la vue. Ainsi, même si le pointeur de la souris survole le bord d'une vue que l'utilisateur ne souhaite pas faire défiler, l'autoscroll ne sera pas activé (sous réserve que la force exercée sur le pavé tactile soit inférieure à la valeur de seuil prédéfinie).

Contrôle de la vitesse de défilement. La vitesse de défilement de l'autoscroll est directement fonction de la force exercée sur le pavé tactile. Plusieurs fonctions de transfert peuvent être utilisées à la fois pour convertir la force exercée sur le pavé tactile en Newton en un déplacement de la vue en mm/s (e.g, une fonction exponentielle figure 2) mais également pour définir le seuil minimal à atteindre pour activer le défilement (e.g, 2N dans l'exemple figure 2).

Terminer l'opération d'autoscroll. La vue défile tant que l'état de *glisser* est maintenu, que le pointeur reste positionné dans la zone active et que la force exercée sur le pavé tactile reste supérieure au seuil prédéfini.

Démonstrateur
Un démonstrateur a été développé mettant en œuvre ForceEdge (figure 2). Ce démonstrateur permet de créer plusieurs vues affichant un très long texte qui peut être sélectionné ou manipulé dans le but de copier la partie sélectionnée dans une autre vue. Un menu affiché dans la barre de titre de chaque vue permet d'associer à la vue la méthode d'autoscroll par défaut du système ou ForceEdge. Enfin, le démonstrateur permet de modifier la fonction de transfert qui convertit la force exercée sur le pavé tactile en vitesse de défilement de la vue grâce à un diagramme interactif dédié (figure 2, droite). Plusieurs fonctions de transfert sont pré-définies, mais l'utilisateur peut également créer sa propre fonction de transfert et en manipuler les points de contrôle directement sur le diagramme.

La visualisation de la courbe représentative de la fonction est particulièrement utile pour concevoir de manière itérative une fonction de transfert. Le concepteur peut en effet réaliser une tâche d'autoscroll tout en visualisant quelle partie de la fonction de transfert est utilisée. Sur la figure 2, cela correspond aux points de contrôle affichés en rouge. Si la vitesse de défilement ne correspond pas à celle qu'il souhaite pour une force donnée, il peut alors modifier la position des points de contrôle et faire un nouvel essai pour vérifier que le nouveau comportement est satisfaisant.

Ce démonstrateur a été développé en Objective-C/Cocoa et nécessite l'utilisation d'un pavé tactile intégré aux Macbook Air sortis après 2014 ou de type Apple Magic Trackpad 2, équipés de la technologie Force Touch. La valeur de force est obtenue grâce à l'API privée multipoints d'Apple.

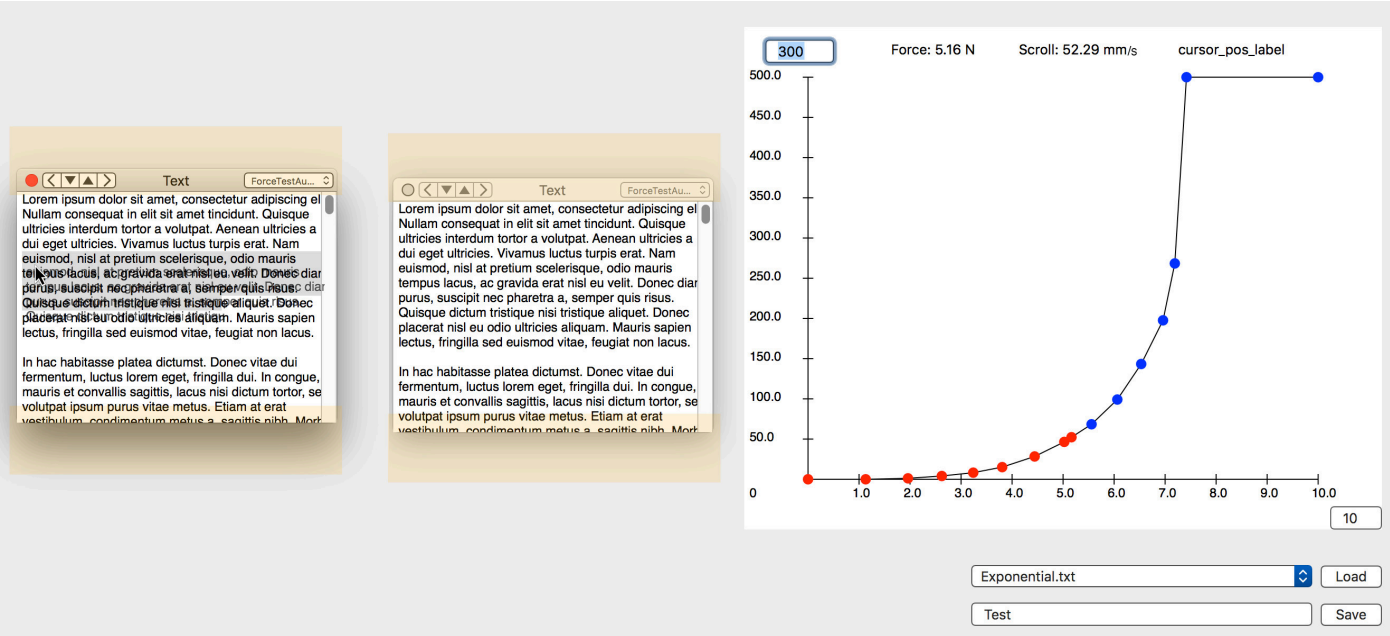

Figure 2: Démonstrateur mettant en oeuvre ForceEdge. Il permet de créer plusieurs vues (ici 2) supportant l'autoscroll via ForceEdge. Le diagramme situé à droite permet de modifier la fonction de transfert convertissant la force exercée sur le pavé tactile en vitesse de défilement.

Conclusion

Nous avons présenté ForceEdge , une nouvelle technique d'interaction d'autoscroll sur pavé tactile qui permet à l'utilisateur de contrôler la vitesse de défilement en variant la force exercée. Un démonstrateur a été développé permettant non seulement de prendre en main ForceEdge et l'essayer avec différentes fonctions de transferts, mais également de la comparer aux techniques d'autoscroll par défaut de Mac OS X. Les fondations théoriques de ForceEdge suggèrent qu'elle n'est pas sensible aux problèmes associés aux fonctions actuelles d'autoscroll. Des tests informels réalisés dans notre démonstrateur semble confirmer ces hypothèses. Malgré cela, nos travaux futurs vont viser à confirmer ces hypothèses au moyen d'expériences contrôlées qui compareront ForceEdge aux méthodes d'autoscroll conventionnelles dans différentes tâches de sélection de texte et de glisser-déposer multi-fenêtres.

Remerciements

Le travail présenté dans cet article a été partiellement financé par l'ANR (TurboTouch, ANR-14-CE24-0009).

Bibliographie

[1] Jonathan Aceituno, Sylvain Malacria, Philip Quinn,

Nicolas Roussel, Andy Cockburn, and Géry Casiez. 2016. The design, use, and performance of edge-scrolling techniques. *International Journal of Human-Computer Studies* To appear. (2016).

[2] Mathias Baglioni, Sylvain Malacria, Eric Lecolinet, and Yves Guiard. 2011. Flick-and-brake: Finger Control over Inertial/Sustained Scroll Motion. In *CHI '11 Extended Abstracts on Human Factors in Computing Systems (CHI EA '11)*. ACM, New York, NY, USA, 2281–2286. DOI:http://dx.doi.org/10.1145/1979742.1979853

[3] Didier D. Bardon, Richard E. Berry, Shirley L. Martin, S. A. Morgan, John M. Mullay, and Craig A. Swearingen. 1997. Method for Auto-Scroll with Greater User Control. *IBM Technical Disclosure Bulletin* 40, 6 (June 1997), 181–182. http://ip.com/IPCOM/000118763

[4] Joseph D. Belfiore, Christopher J. Guzak, Christopher E. Graham, Stepehn M. Madigan, Tandy W. Trower, II, Randall L. Kerr, and Adrian M. Wyard. 1998. Auto-scrolling during a drag and drop operation. (1998). http://www.google.com/patents/US5726687

[5] Richard E. Berry, Stephen S. Fleming, and A. C. Temple. 1991. Auto-Scroll During Direct Manipulation. *IBM Technical Disclosure Bulletin* 33, 11 (April 1991), 312.

[6] Andrew Kwatinetz. 1996. Scrolling contents of a window. (1996). http://www.google.com/patents/US5495566

[7] Shih-Gong Li and Theodore Jack London Shrader. 1998. Scrolling a target window during a drag and drop operation. (1998). http://www.google.com/patents/US5740389

[8] Sylvain Malacria, Jonathan Aceituno, Philip Quinn, Géry Casiez, Andy Cockburn, and Nicolas Roussel. 2015. Push-Edge and Slide-Edge: Scrolling by Pushing Against the Viewport Edge. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 2773–2776. DOI:http://dx.doi.org/10.1145/2702123.2702132

[9] Takashi Miyaki and Jun Rekimoto. 2009. GraspZoom: Zooming and Scrolling Control Model for Single-handed Mobile Interaction. In *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '09)*. ACM, New York, NY, USA, Article 11, 4 pages. DOI:http://dx.doi.org/10.1145/1613858.1613872

[10] Gonzalo Ramos and Ravin Balakrishnan. 2005. Zliding: Fluid Zooming and Sliding for High Precision Parameter Manipulation. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology (UIST '05)*. ACM, New York, NY, USA, 143–152. DOI:http://dx.doi.org/10.1145/1095034.1095059

[11] Hong Z Tan, Mandayam A Srinivasan, Brian Eberman, and Belinda Cheng. 1994. Human factors for the design of force-reflecting haptic interfaces. *Dynamic Systems and Control* 55, 1 (1994), 353–359.

[12] Shumin Zhai. 1995. *Human performance in six degree of freedom input control*. Ph.D. Dissertation. University of Toronto.