Quinze Ans de Recherche sur les Menus : Critères et Propriétés des Techniques de Menus

Gilles Bailly [1,2] *Eric Lecolinet* [2] *Laurence Nigay* [1]

[1] LIG-IIHM, Université de Grenoble 1, B.P. 53, F-38041 Grenoble Cedex 9, France
[2] GET/ENST – CNRS LTCI UMR 5141 46 rue Barrault 75013, Paris, France

{gilles.bailly, elc}@enst.fr {gilles.bailly, laurence.nigay}@imag.fr

RESUME
Le corpus de publications décrivant des techniques à base de menus s'enrichit d'année en année. Mais il n'est pas toujours facile de comprendre les différences et les similitudes entre ces techniques d'interaction. Cet article propose un espace de caractérisation fondé sur plusieurs critères et propriétés pour mieux appréhender ces différentes techniques. Cet espace met en particulier l'accent sur les différences importantes que présentent ces techniques en mode novice et en mode expert.

MOTS CLES : Menus, techniques d'interaction, mode novice, mode expert.

ABSTRACT
Many different kinds of menu techniques have been proposed so far and their number is continuously increasing. This paper proposes taxonomy to facilitate understanding of their differences and similarities. The taxonomy provides several criteria and properties for classifying menu systems and focuses in particular on the differences in using menu techniques in expert versus novice modes.

CATEGORIES AND SUBJECT DESCRIPTORS: H.5.2 [Information interface and presentation]: User Interfaces. I.3.6 [Methodology and Techniques]: Interaction techniques.

GENERAL TERMS: Design, Human factors

KEYWORDS: Menu systems, Interaction techniques, novice mode, expert mode.

INTRODUCTION

Le corpus de publications décrivant de nouveaux menus s'enrichit d'année en année. Mais il n'est pas toujours facile de comprendre précisément les différences et les similitudes entre ces différentes techniques de menus. Un état de l'art très complet [32] présente les systèmes de menus élaborés avant 1991. Depuis, une vingtaine de nouvelles techniques ont été développées, comme le montre la chronologie de la figure 2, et de nombreuses propriétés ont été proposées pour améliorer ces techniques.

L'objectif de cet article est de recenser les techniques de menus en 2D présentées depuis 1991 en mettant en évidences leurs principales propriétés par rapport aux critères d'utilisabilité définis dans [39] (rapidité, précision, apprentissage et mémorisation) ainsi qu'à deux critères supplémentaires d' « utilité » (le nombre de commandes et l'occupation spatiale).

Bien que le terme "menu ou technique de menus" soit largement utilisé dans la littérature, il n'en existe pas une définition de référence. Deux caractéristiques importantes des menus sont proposées dans [32]: 1) les menus réduisent l'effort mental de l'utilisateur en lui présentant explicitement les commandes du système. Ils permettent ainsi d'explorer les commandes et d'éviter des erreurs de syntaxe; 2) les menus réduisent également l'effort physique en séparant des fonctions d'entrée de l'affichage. Dans le cadre de cette étude, un menu est un instrument d'interaction [5] non rémanent permettant le choix d'un élément prédéterminé (et d'un seul) dans un domaine fini (appelé domaine de sélection) via une interaction modale (c'est-à-dire non interruptible). Cette définition écarte donc les "combo-box", les "listes" et les "barres de défilement" qui sont rémanents (affichés en permanence à l'écran) ainsi que les "champs textuels" dont la valeur n'est pas prédéterminée. De plus, cette étude se limite aux menus dont l'opération se fait au moyen de dispositifs conventionnels (clavier et dispositif de pointage 2D). Elle ne considère pas les menus 3D souvent conçus pour la réalité virtuelle (dont nous trouvons un état de l'art dans [13]) ni les menus multimodaux (bi-manuels, tactiles, audios, oraux) comme les Toolglass [7] ou les Earpod [47].

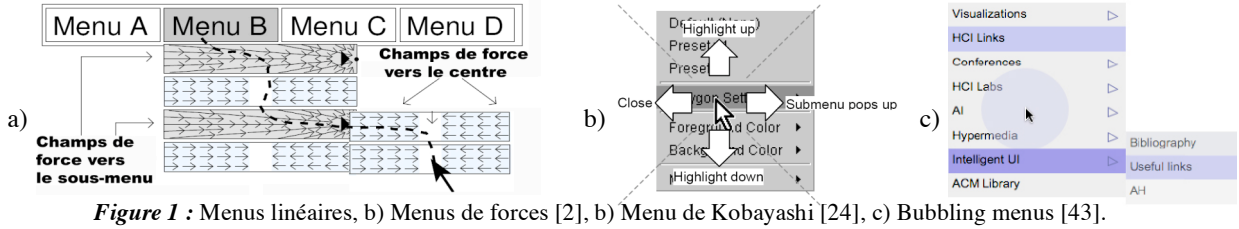

Figure 1 : Menus linéaires, b) Menus de forces [2], b) Menu de Kobayashi [24], c) Bubbling menus [43].

Nous distinguons deux grandes catégories parmi les techniques publiées avant 1991. La première inclut les menus linéaires, qui sont encore largement prédominants dans les systèmes actuels. Ils interviennent dans les systèmes de menus cascadés (Pulldown ou Pullright menus) associées à une barre de menu ainsi que dans les menus contextuels (également appelés Popup menus ou plus rarement Walking menus). La seconde catégorie concerne les menus circulaires, dont des versions préliminaires sont apparues dès 1969 [17, 44] jusqu'à l'introduction des Pie menus [10] en 1988. Ceux-ci organisent les items autour d'un disque centré sur le curseur de la souris de manière à ce que toutes les commandes soient équidistantes du curseur et que la direction suffise à les différencier. Il est indiqué dans [10] un gain de temps de 15% par rapport aux menus linéaires.

Bien que les études effectuées depuis 15 ans portent toujours sur ces deux catégories de menus, nous constatons cependant une attention particulière sur les améliorations des menus circulaires, comme nous le soulignons dans les sections suivantes. Cette différenciation est liée à un autre point important qui concerne le mode de fonctionnement du mode expert des systèmes de menus, celui-ci étant radicalement différent dans le cas des améliorations des menus circulaires. Aussi, nous commençons par décrire les différences des modes novice et expert en les mettant en relation avec les comportements de l'utilisateur. Cette section préliminaire nous permettra ensuite de décrire les propriétés des techniques de menus par rapport aux critères d'utilisabilité et d'utilité précités.

MODES NOVICE & EXPERT

La variété des comportements des utilisateurs et leur impact sur l'utilisation des techniques de menus constituent un aspect important à considérer. Nous présentons trois types de comportements typiques (novice, intermédiaire et expert) que nous mettons en relation avec les modes novice et expert des menus. Cette notion de mode novice/expert est ensuite exploitée pour décrire les différentes propriétés des techniques de menus.

Comportements. L'utilisateur peut se comporter de diverses manières lorsqu'il interagit avec des menus [22, 35]. On observe un **comportement novice** lorsqu'il cherche à atteindre un but qu'il n'a jamais réalisé auparavant. Dans le cas des menus, cela signifie qu'il ne sait pas où se trouve la fonctionnalité recherchée ou qu'il cherche comment elle est libellée. Le comportement novice se caractérise par la navigation dans l'arbre de commandes avec plusieurs retours arrière.

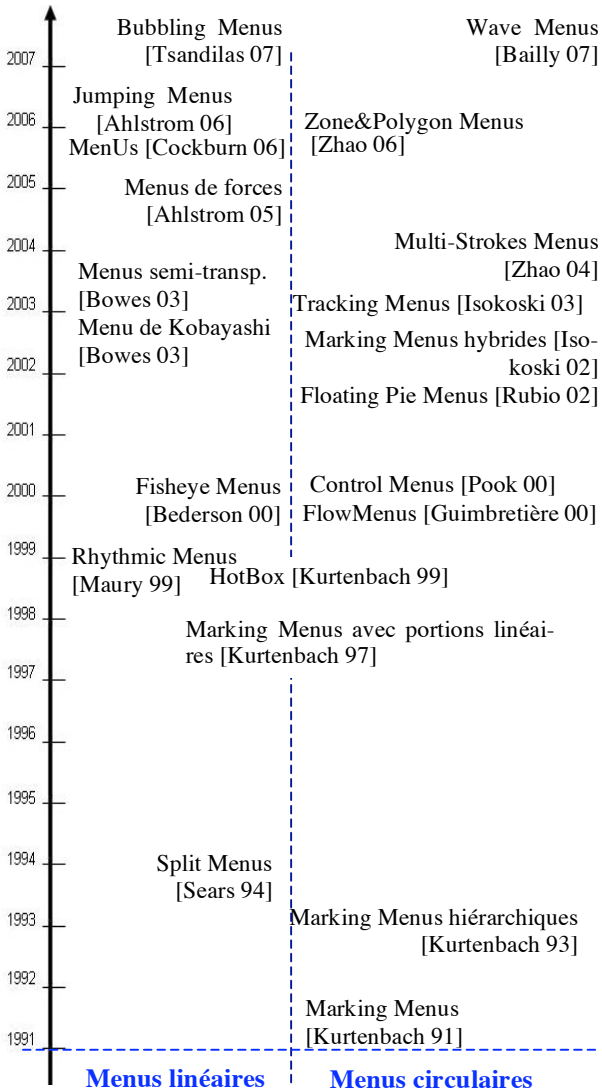

Figure 2 : Historique des menus linéaires et circulaires depuis 1991.

L'utilisateur a un **comportement intermédiaire** lorsqu'il a déjà accompli son but préalablement mais sans avoir tout le savoir-faire. Pour les menus, ceci signifie que l'utilisateur a déjà activé la commande désirée mais ne se souvient plus exactement comment y accéder de nouveau. Cependant le nom des catégories l'aide à retrouver la commande qu'il recherche et lui permet de ne jamais faire de retours arrière.

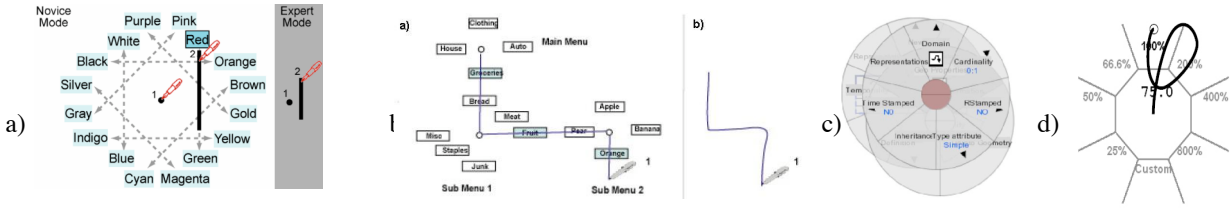

Figure 3 : Menus circulaires a) Polygon Menus [46], b) Marking Menus hiérarchiques en mode novice (gauche) et en mode expert (droite) [26], c) Floating Pie Menus [38], d) FlowMenus [19].

Enfin, l'utilisateur a un **comportement expert** lorsque qu'il sait exactement comment atteindre son but : l'utilisateur sait quelle commande il souhaite utiliser et comment l'activer.

Modes. Les menus comportent généralement deux modes différents d'utilisation : le mode novice, qui est le mode le plus fréquemment utilisé, et le mode expert réservé à des utilisateurs plus expérimentés. Le mode expert typique d'un menu cascadé passe par l'utilisation de raccourcis au clavier.

Les Marking Menus [25] proposent une autre approche. Ils étendent les Pie Menus en laissant l'utilisateur sélectionner un item sans afficher le menu en mode expert. L'utilisateur fait simplement une marque dans la direction du secteur désiré. Il peut toujours afficher le menu s'il le souhaite en attendant 0,3s.

Le mode novice fait appel à la **reconnaissance** : le système présente les commandes à l'utilisateur qui peut les parcourir en phase d'exploration. Au contraire, le mode expert utilise le **rappel** : le système n'affiche pas les commandes, impliquant que l'utilisateur s'en rappelle. Le mode expert a pour objectif de permettre une sélection plus rapide des commandes. Cependant, comme il fait appel à la mémorisation, il est susceptible d'engendrer davantage d'erreurs.

Nous établissons la correspondance entre le mode (point de vue système) et le comportement (point de vue utilisateur). Très naturellement, un utilisateur avec un comportement novice utilise le mode novice. Cependant, un comportement expert n'implique pas forcément le mode expert : c'est le cas si l'utilisateur n'a pas appris le mode expert ou s'il n'existe pas de mode expert. Lorsque l'utilisateur a un comportement intermédiaire, il peut soit utiliser le mode novice, soit commencer l'interaction en mode expert et la finir en mode novice pour confirmation. C'est le cas des Marking Menus [25] où l'utilisateur peut commencer en mode expert en faisant des marques, puis attendre 0,3s que le menu s'affiche. L'utilisateur peut alors finir l'interaction en mode novice.

Il est important de souligner que le mode novice et le mode expert cœxistent en permanence. En effet, une grande partie des commandes d'un menu sont rarement utilisées. Ainsi l'utilisateur expérimenté utilise le mode expert pour ses commandes favorites, mais revient au mode novice pour toutes les commandes qu'il utilise moins fréquemment.

Les modes novice et expert étant définis, nous étudions maintenant les propriétés des techniques de menus au regard des critères d'utilisabilité (section suivante) puis des critères d'utilité. La notion de mode est exploitée dans la définition des propriétés suivantes.

PROPRIETES LIEES AUX CRITERES D'UTILISABILITE

Nous considérons les quatre propriétés d'utilisabilité définies dans [39] : rapidité, précision, apprentissage et mémorisation. Les critères d'utilisabilité *apprentissage* et *mémorisation* sont liés puisque tous deux dépendent de la fréquence d'utilisation. Nous les étudions donc ensemble, dans une même section. De même les critères *rapidité* et *précision* sont corrélés car l'utilisateur fera d'autant plus d'erreurs qu'il essayera d'aller vite. Ces deux critères sont donc traités ensemble dans la section suivante.

Rapidité et Précision

Différents modèles visent à prédire le temps de sélection dans les systèmes de menus [43]. Ces modèles considèrent deux propriétés qui affectent la performance : le temps de recherche visuelle et le temps de pointage. Dans cette section, nous considérons aussi d'autres propriétés en relation avec le type de tâche (tâche répétitive, navigation, etc.) ou du comportement de l'utilisateur (novice, intermédiaire, expert).

Recherche visuelle. Le temps de recherche visuelle dans un menu peut être prédit par la loi de Hick [31]. C'est une fonction du logarithmique du nombre d'items de la liste. Trois stratégies ont été utilisées pour réduire le temps de recherche visuelle :

- **Filtre.** La première stratégie consiste à ne pas afficher toutes les commandes pour accélérer la recherche visuelle. C'est le cas par exemple des menus contextuels qui présentent à l'utilisateur seulement les commandes qu'il peut appliquer à l'objet d'intérêt. C'est le cas également des menus Windows qui masquent complètement les commandes rarement utilisées. L'utilisateur peut cependant visualiser toutes les commandes en patientant un certain délai.

- **Groupement**. La seconde stratégie consiste à réorganiser les commandes par groupe. En conséquence, l'utilisateur recherchera la commande seulement dans les catégories appropriées [31]. Ceci peut être réalisé en réorganisant les commandes hiérarchiquement. A un même niveau, il est fréquent d'utiliser des séparateurs pour distinguer des groupes de commandes.
- **Mise en valeur**. Enfin, la dernière stratégie consiste à laisser toutes les commandes affichées, mais à mettre en valeur certaines commandes. Par exemple, les menus semi-transparents [9] effacent partiellement les items rarement activés ce qui facilite la lecture des autres items. La plupart des menus grisent les commandes non pertinentes pour diminuer le nombre de commandes activables à lire. De plus des icônes sont ajoutées aux commandes fréquemment utilisées (on peut également les retrouver dans la barre d'outils) pour faciliter le parcours visuel des commandes.

Tâche de pointage. Le temps pour pointer une cible peut être prédit par la combinaison de la loi de Fitts [8] et de la loi de Steering [1]. La loi de Fitts définit que le temps de pointage est une fonction linéaire de l'indice de difficulté (ID) où ce dernier est défini comme le logarithmique du ratio entre la distance et la taille de la cible. Lorsque le curseur doit traverser un « tunnel » le temps de pointage est mieux prédit par la loi de Steering qui prend en compte le ratio entre la longueur du tunnel et sa largeur. Ces lois sont appliquées aux menus cascadés dans [3] : il est montré que les mouvements verticaux suivent la loi de Fitts alors que les mouvements horizontaux suivent la loi de Steering [3]. Pour réduire le temps de pointage (dans des menus ou tout autre système), il convient de réduire l'indice de difficulté en diminuant la distance de la cible ou en augmentant la largeur de la cible : par exemple dans le cas des menus linéaires, comme le mouvement est vertical, il s'agit d'augmenter la hauteur des items.

Certains menus ont été conçus pour réduire la **distance** à parcourir pour sélectionner des éléments. Kobayashi rapproche le sous-menu cascadé en l'ouvrant à la position du curseur [24]. Les Pie Menus [10], en organisant les items circulairement autour du curseur, réduisent en moyenne la distance d'activation. Les Split Menus [40] diminuent la distance des commandes les plus utilisées en les remontant dynamiquement dans la hiérarchie. D'autres menus ont considéré la **largeur** (de la cible ou du tunnel) plutôt que la distance. Les MenUs [12] élargissent la zone d'activation des sous-menus. Les Marking Menus [25] permettent à l'utilisateur de faire des marques de n'importe quelle taille : ceci revient à définir des secteurs de largeur infinie.

D'autres mécanismes ont été proposés pour réduire l'indice de difficulté. En particulier, le **pointage sémantique** [8] et le **pointage vectoriel** [18], qui séparent l'espace moteur de l'espace visuel, ont été appliqués à certaines techniques de menus. Par exemple, dans les Menus de forces [2], des champs de forces impactent le mouvement du curseur afin de faciliter l'accès aux sous-menus et d'éviter les erreurs (figure 1.a). Dans les Jumping Menus [3], lorsque l'utilisateur clique sur un item parent, le curseur est automatiquement positionné sur le premier item du sous-menu correspondant. Enfin les Bubbling Menus [43] facilitent l'accès à des items fréquemment utilisés en modifiant la taille du curseur (figure 1.c).

Par ailleurs, d'autres techniques exploitent une stratégie temporelle. Les menus cascadés introduisent un **délai** avant et après l'affichage d'un sous-menu, ce qui a pour effet d'augmenter la « largeur du tunnel ». Le délai des menus cascadés est en général de 0,2s. Dans [7], il est souligné que ce délai est souvent trop court pour les utilisateurs novices et trop long pour les utilisateurs experts.

Enfin, les menus rythmiques [34] ne cherchent pas à réduire la distance ou à augmenter la largeur mais considèrent seulement le **rythme**. Une fois le menu ouvert, les items sont mis successivement en surbrillance à une fréquence donnée. Un clic permet alors de sélectionner l'item courant. Ces menus offrent de bonnes performances pour des menus de petite et moyenne taille (moins de 10 items)**.**

Nous venons d'étudier différentes stratégies qui permettent d'améliorer la tâche de pointage au sein d'un menu. Un autre aspect à considérer est la distance à parcourir entre l'objet d'intérêt et le menu [11].

Transition. Nous appelons transition la distance entre l'objet d'intérêt (zone de travail) et le menu [11]. La transition peut d'abord être de type **objet-commande**, c'est-à-dire de l'objet d'intérêt vers le menu. Les menus **sur place** réduisent alors cette distance de transition. Les menus contextuels et les Tracking Menus [16] sont des menus que l'on peut toujours activer et afficher dans la zone d'attention. Ils vérifient donc la propriété "sur place". A l'opposé, les barres de menus ou les palettes sont dans la plupart des cas plus éloignées de la zone de travail. Le second type de transition est du menu vers l'objet d'intérêt, c'est-à-dire de type **commande-objet** : une fois la commande activée dans un menu, il faut retourner à l'objet d'intérêt. Cette distance peut se révéler importante : elle est par exemple deux fois plus importante pour les Marking Menus hiérarchiques [26] (Figure 3.b) que pour les menus cascadés. Deux solutions ont été proposées pour palier à ce problème : les Multi-Strokes Menus et les menus glissants. Les Multi-Strokes Menus [45] superposent les sous-menus pour ne pas s'éloigner de la zone d'intérêt à la fin de l'activation tandis que les menus glissants [11] font glisser le menu sous le curseur

de la souris au lieu de déplacer le curseur au-dessus du menu. Le curseur reste ainsi à sa position de départ à la fin de l'interaction.

Tâche répétitive. Lorsque l'utilisateur doit appliquer plusieurs fois la même commande à différents objets, il peut être frustrant d'avoir à la réactiver à chaque fois comme c'est le cas avec les menus cascadés. A l'opposé, les palettes permettent généralement de conserver un ou plusieurs modes (par exemple l'outil courant pour un logiciel de dessin). Ceci peut conduire l'utilisateur à faire des erreurs s'il a oublié quel est le mode courant. Les quasi-modes apportent une solution intéressante pour contourner ce problème en ne conservant le mode que le temps de l'interaction. C'est le cas des FlowMenus [19] et des Control Menus [36] qui implémentent la propriété de **fusion de commandes et manipulation directe** [20]. Ce type d'interaction consiste à sélectionner et à contrôler l'opération en un seul geste (par exemple, sélectionner la commande « zoom » et contrôler directement et continûment sa valeur). Dans le cas des Control Menus l'utilisateur doit franchir une certaine distance d'activation pour commencer l'opération tandis qu'il doit repasser au centre avec les FlowMenus (figure 3.c).

Les techniques de menus peuvent parfois être **persistantes**, c'est-à-dire que le menu ne se referme pas automatiquement après la sélection d'une commande. L'utilisateur peut alors sélectionner plusieurs commandes à la suite sans retraverser la hiérarchie. Ainsi les menus détachables (Tear off menus) [32] peuvent se séparer de la barre de menus pour rester ouvert. Plus récemment, les Floating Pie Menus [38] disposent d'un bouton pour laisser le menu ouvert après la sélection (Figure 3.c).

Tâche de navigation. Nous avons vu précédemment que lorsque les techniques de menus étaient utilisées en mode novice, l'utilisateur naviguait dans l'arbre de commandes pour trouver une fonctionnalité. La **prévisualisation** peut alors faciliter cette navigation. Cette propriété consiste à afficher le contenu d'un sous-menu avant de l'activer : c'est par exemple habituellement le cas avec les menus cascadés. La prévisualisation offre un retour visuel proactif qui permet une inspection rapide des possibilités [37]. Il est de plus préférable de laisser les **items parents visibles** surtout lorsque l'utilisateur doit faire des retours arrière fréquents durant l'exploration du menu, comme c'est souvent le cas en mode novice. Plusieurs menus visant à améliorer le mode expert (par exemple [19, 45, 46]) n'implémentent pas ces deux propriétés (prévisualisation et items parents visibles) ce qui peut conduire à une dégradation des performances en mode novice [4].

Sélection non-intrusive. En mode expert, deux propriétés ont été introduites par les Marking Menus [25] : la sélection sans regarder et l'indépendance à l'échelle. La **sélection sans regarder** permet à l'utilisateur de sélectionner une commande alors que son attention est portée sur un autre objet que le menu. **L'indépendance à l'échelle** lui permet de faire des gestes de taille quelconque et donc en quelque sorte d'« échapper à la foi de Fitts » dans la mesure où les marques n'ont pas à pas se terminer dans une cible de taille prédéterminée.

Erreurs et directions. Avec l'introduction des menus circulaires, plusieurs études ont porté sur la relation entre le taux d'erreurs et la direction. Dans [25], il est montré que plus le secteur angulaire affecté à chaque commande est petit, plus le taux d'erreurs est important. Il est ainsi difficile d'avoir plus de 8 items sur le même niveau d'un menu circulaire. De plus les erreurs dépendent aussi de la direction dans le cas des Marking Menus hiérarchiques [45] : les axes diagonaux produisent plus d'erreurs que les axes horizontaux ou verticaux lorsque plusieurs niveaux sont enchaînés.

Apprentissage et mémorisation

L'utilisateur doit être capable d'apprendre rapidement comment fonctionne le menu, d'atteindre rapidement un bon niveau de performance et de le conserver. Ceci implique quatre aspects différents : (1) le temps pour comprendre comment le menu fonctionne, (2) celui pour comprendre comment les éléments sont organisés, (3) le temps nécessaire pour être capable d'utiliser le mode expert, et enfin (4) celui pour pouvoir conserver cette connaissance. La mémorisation est fortement liée au temps d'apprentissage et à la fréquence d'utilisation, laquelle joue également un rôle important.

Apprendre la technique de menus. Les utilisateurs sont plus familiers avec les techniques d'interaction qu'ils ont l'habitude d'utiliser, comme les menus cascadés et, plus généralement, les techniques à base de clics souris. Ils sont par contre généralement moins familiers avec l'interaction gestuelle. Des techniques inhabituelles, comme les Multi-Strokes [45], pourraient ainsi repousser l'utilisateur novice. De même, les Marking Menus hiérarchiques [26] imposent de repasser au centre du sous-menu pour activer une commande en mode novice. Ce mécanisme n'est pas très naturel pour la plupart des utilisateurs, ceux-ci préférant souvent pointer directement sur la commande souhaitée. Enfin, il convient de souligner que les **retours visuels** facilitent significativement l'apprentissage des techniques [27, 42].

Apprendre l'organisation. L'utilisateur doit pouvoir prédire le résultat de ses actions (prévisibilité). Ainsi les menus qui changent les items et leurs positions dynamiquement peuvent être néfastes pour le temps d'apprentissage [14]. Par contre, les menus adaptables, c'est-à-dire des menus où l'utilisateur peut configurer la disposition des commandes, donnent des performances équivalentes aux menus statiques. De plus, l'utilisateur apprécie de pouvoir organiser lui-même ses commandes [14] [30]. Un autre aspect favorisant l'apprentissage de

l'organisation, est de pouvoir naviguer dans l'arbre de commandes grâce à la prévisualisation et l'affichage des items parents (section Rapidité et Précision). Enfin il sera plus facile d'apprendre l'organisation si elle change peu d'une application à une autre comme c'est le cas avec les catégories « Fichier » ou « Edition ». La conception des menus circulaires peut également reposer sur l'opposition sémantique [41] en plaçant par exemple les commandes « Ouvrir » et « Fermer » à l'opposé l'une de l'autre. En se fondant sur la mémoire procédurale [41], les menus circulaires peuvent ainsi faciliter l'apprentissage.

Acquérir le mode expert. Lorsque l'utilisateur acquiert un comportement expert, il souhaite naturellement utiliser le mode expert (section Modes novice&expert). Dans le cas des menus cascadés, l'utilisateur doit changer de technique d'interaction et faire un effort d'apprentissage pour mémoriser les raccourcis claviers. Il y a donc une rupture entre le mode novice et le mode expert. Les Marking Menus proposent une **transition fluide** du mode novice au mode expert. L'utilisateur fait en effet le même geste dans les deux modes, la principale différence étant que le menu ne s'affiche pas en mode expert. Ainsi, c'est en répétant plusieurs fois le geste en mode novice que l'utilisateur apprend implicitement le mode expert. Il n'y a pas d'effort d'apprentissage supplémentaire, et l'utilisateur acquiert donc le mode expert plus rapidement.

Mémoriser les commandes. Si l'utilisateur n'utilise plus la technique de menus pendant un certain temps, il peut ne plus se rappeler comment activer certaines commandes. Dans [28] il est montré que l'utilisateur est capable de réapprendre rapidement les commandes grâce à la mémoire gestuelle. En effet chaque commande est associée à un geste ce qui facilite la rétention.

PROPRIETES LIEES AUX CRITERES D'UTILITE

Comme les critères d'utilisabilité, les critères d'utilité sont des critères de qualité de l'interaction qui permettent de répondre aux besoins de l'utilisateur. Cependant les critères d'utilité sont davantage liés aux fonctionnalités ou au système. Nous identifions deux critères d'utilité pour les techniques de menus : le nombre de commandes que le menu peut contenir et l'occupation spatiale.

Nombre de commandes

Le nombre de commandes a tendance à augmenter régulièrement dans les logiciels. Une solution est d'organiser les commandes hiérarchiquement : les paramètres de conception sont alors la profondeur du menu (nombre de niveaux dans le menu) et la largeur (nombre de commandes par sous-menus). Dans la plupart des cas, il est préférable d'augmenter la largeur plutôt que la profondeur [32, 35]. Par exemple, les Fisheye Menus [6] permettent en mode novice d'accéder à plusieurs dizaines d'éléments en largeur. La HotBox [30] autorise plus de 1000 commandes : lorsque l'utilisateur presse la barre d'espace, l'écran est divisé en 5 zones ; dans la zone centrale, plusieurs barres de menus sont affichées et dans chacune des 4 zones périphériques, l'utilisateur peut configurer ses propres Marking Menus.

La conception des menus circulaires limite souvent le nombre de commandes en largeur à 8 éléments [26] (paragraphe Erreurs et directions). Deux solutions ont été proposées : la première consiste à ne plus considérer seulement des marques droites, mais aussi des **courbures**. Les Marking Menus hybrides [23] divisent les 4 axes principaux en 3 branches. Ainsi, 12 commandes sont accessibles en largeur via une courbure différente. L'autre solution consiste, comme pour les Polygon&Zone Menus [46] à considérer la position relative de l'origine des marques en plus de l'orientation des marques. Les Polygon Menus [46] peuvent alors afficher jusqu'à 16 commandes en largeur et les Zones Menus [46], 32 commandes.
De plus le taux d'erreur croît significativement lorsque la profondeur d'un Marking Menu hiérarchique augmente [26]. En pratique, un Marking Menu hiérarchique peut difficilement comporter plus de deux niveaux. Ces menus utilisent un principe de composition spatiale ce qui fait que les marques deviennent de plus en plus complexes et sont délimitées par des inflexions qui sont parfois difficiles à interpréter. Inversement, les Multi-Stroke Menus [45] sont basés sur un principe de **composition temporelle**. Ils permettent de tracer plusieurs marques élémentaires en séquence (une par niveau) au lieu d'une marque unique mais complexe car composée de plusieurs sous-segments : le taux d'erreurs ne dépend plus de la profondeur ce qui permet aux Multi-Strokes Menus d'autoriser un nombre bien plus important de commandes [45].

Occupation spatiale

L'espace occupé par un menu est susceptible d'occulter la zone de travail de l'utilisateur. De plus, une occupation spatiale excessive peut rendre certaines techniques de menus inutilisables dans certaines conditions comme un écran d'ordinateur de poche.

Maintien du contexte visuel. Le menu est, bien évidemment, rarement l'objet principal de la tâche. Afin de réduire l'encombrement spatial, les menus sont donc affichés "**sur demande**" [21] et se **ferment automatiquement** dès que l'utilisateur a sélectionné une commande pour ne pas occuper inutilement de la place en permanence. Dans le cas des menus hiérarchiques, la branche courante reste affichée pour apprendre la hiérarchie des commandes. Cependant, certaines techniques comme les Multi-Strokes Menus masquent les menus parents pour garder un contexte visuel plus important. Cela va à l'encontre du critère d'observabilité, mais semble un compromis à faire pour garder un contexte visuel plus important.

A l'opposé, les palettes apparaissent en permanence à l'écran et peuvent occulter une partie de la zone d'attention de l'utilisateur. Elles nécessitent souvent l'utilisation d'icônes plutôt que des libellés afin d'occuper moins de place à l'écran.

L'utilisation de la **transparence** constitue une solution intéressante pour permettre à l'utilisateur de partager son attention entre le menu et le contexte de la zone de travail [9, 21, 36, 38, 42]. La transparence peut cependant entraîner des problèmes de lisibilité. Les Fisheye Menus [6] et leur extensions aux menus hiérarchiques [32] utilisent le principe des vues **fish-eye** pour afficher un grand nombre d'items sur une surface réduite. Enfin, les modes expert des menus (raccourcis claviers ou marques) permettent à l'utilisateur d'activer une commande sans masquer la zone de travail.

Contrainte spatiale. Les propriétés précédentes traitent du problème du maintien du contexte visuel. Cependant une occupation spatiale excessive peut rendre certaines techniques de menus inutilisables selon les conditions d'usage. Par exemple, les Marking Menus hiérarchiques de profondeur 3 utilisent en mode novice plus de 3 fois l'espace occupé par un menu cascadé. En effet, ils contiennent deux items sur l'axe horizontal et leurs sous-menus s'affichent forcément dans la direction de l'item parent. Ainsi, un menu activé près d'un bord de l'écran risque de voir ses sous-menus s'afficher en dehors de l'écran. Ce type de menu pourra donc difficilement être utilisé "sur place" car certaines commandes seront inaccessibles en mode novice. Ce problème est fortement lié à la composition spatiale. Les menus basés sur la composition temporelle des marques, comme les Multi-Strokes [45], superposent les sous-menus en mode novice, ce qui évite que l'espace occupé dépende de la profondeur du menu. Cette superposition interdit par contre de pouvoir accéder directement aux items parents. Un compromis est donc nécessaire entre la facilité de navigation et l'occupation spatiale en mode novice. Ce problème n'existe pas en mode expert car les marques sont indépendantes de l'échelle (section Rapidité et Précision).

Une table récapitulative des différentes techniques que nous avons présentées est disponible à l'adresse http://gbailly.free.fr/menus/ .

CONCLUSION

Nous avons exposé un ensemble de propriétés qui permettent de comparer les différentes techniques de menus. Le but de cette étude est de mieux comprendre les différences et les similitudes de ces techniques. Elle vise aussi à permettre aux développeurs d'applications de choisir une technique de menus appropriée aux besoins. Enfin, en permettant aux concepteurs d'améliorer des techniques existantes ou d'en créer de nouvelles en se basant sur nos propriétés, nous soulignons l'aspect génératif de notre espace de caractérisation. Par exemple les propriétés de notre espace nous ont permis d'identifier certains points faibles des Multi-Strokes menus en mode novice. Ce constat nous a conduit à l'élaboration d'une nouvelle technique, appelée Wave Menu [4], qui offre les mêmes performances que les Multi-Stroke en mode expert tout en améliorant sensiblement le mode novice.

REMERCIEMENTS

Les auteurs souhaitent remercier Renaud Blanch, Stéphane Huot, Tahir Muhammad et Anne Roudaut, ainsi que les relecteurs.

BIBLIOGRAPHIE

1. Accot, J., Zhai, S. (1997). Beyond Fitts' Law: Models for Trajectory-Based HCI Tasks. *ACM CHI'97*, p. 295-302.
2. Ahlström, D. (2005). Modeling and improving selection in cascading pull-down menus using Fitts' law, the steering law and force fields. *ACM CHI '05*, p. 61-70.
3. Ahlstroem, D., Alexandrowicz, R., Hitz, M. (2006). Improving menu interaction: a comparison of standard, force enhanced and jumping menus. *ACM CHI'06* , p. 1067-76.
4. Bailly, G., Lecolinet, E., Nigay, l., (2007). Wave Menus: Improving the Novice Mode of Hierarchical Marking Menus. *ACM Interact'07*, à paraitre.
5. Beaudouin-Lafon, M. (2000). Instrumental interaction: an interaction model for designing post-WIMP user interfaces. In CHI '00, p. 446-453.
6. Bederson, B. B. (2000). Fisheye menus. *ACM UIST'00,* p. 217-225.
7. Bier, E. A., Stone, M. C., Pier, K., Buxton, W., DeRose, T. D. (1993). Toolglass and magic lenses: the see-through interface. *SIGGRAPH '93*, p. 73-80.
8. Blanch, R., Guiard, Y., Beaudouin-Lafon, M. (2004). Semantic pointing: improving target acquisition with control-display ratio adaptation. *ACM CHI'04*, p. 519-526.
9. Bowes, J., Dearman, D., Perkins, R. (2003). Transparency for item highlighting. *Poster at GI'03*.
10. Callahan, J., Hopkins, D., Weiser, M., Shneiderman, B. (1988). An empirical comparison of pie vs. linear menus. *ACM CHI'88*, p. 95-100.
11. Cance, J., Collomb, M., Hascoët M. (2006). Accelerating object-command transitions with pie menus. *ACM Enactive'06*, p. 109-110.
12. Cockburn, A., Gin, A. (2006). Faster cascading menu selections with enlarged activation areas. *ACM GI'06*, p. 65-71.
13. Dachselt, R. Hübner, A. (2007). Virtual Environments: Three-dimensional menus: A survey and taxonomy. *Comput. Graph.* 31, 1, p. 53-65.
14. Findlater, L., McGrenere, J. (2004). A comparison of static, adaptive, and adaptable menus. *ACM CHI'04*, p. 89-96.

15. Fitts, P. M. (1954). The Information Capacity of the Human Motor System in Controlling the amplitude of movement. *Journal of Exp. Psych., 47(6)*, p. 381-391.

16. Fitzmaurice, G., Khan, A., Pieké, R., Buxton, B., Kurtenbach, G. (2003). Tracking menus. *ACM UIST'03*, p. 71-79.

17. Forsey, D. (1984) Harmony in Transposition: A Tocatta for Vax and M68000, *M.Sc. Thesis, University of Waterloo, Canada*.

18. Guiard, Y., Blanch, R., Beaudouin-Lafon, M. 2004. Object pointing: a complement to bitmap pointing in GUIs. *ACM GI'04,* p. 9-16.

19. Guimbretiére, F., Winograd, T. (2000). FlowMenu: combining command, text, and data entry. *ACM UIST'00*, p. 213-216.

20. Guimbretière, F., Martin, A., Winograd, T. (2005). Benefits of merging command selection and direct manipulation. *Trans. on Computer-Human Interaction,* p. 460-476.

21. Hinckley, K., Sinclair, M. (1999). Touch-sensing input devices. *ACM CHI'99*, p. 223-230.

22. Howes, A. (1994). A model of the acquisition of menu knowledge by exploration. *ACM CHI'94*, p. 445-451.

23. Isokoski, P., Käki, M. (2002). Comparison of two touchpad-based methods for numeric entry. *ACM CHI'02*, p. 25-32.

24. Kobayashi, M., Igarashi, T. (2003). Considering the direction of cursor movement for efficient traversal of cascading menus. *ACM UIST'03*, p. 91-94.

25. Kurtenbach, G., Buxton, W. (1991). Issues in combining marking and direct manipulation techniques. *ACM UIST'91*, p. 137-144.

26. Kurtenbach, G., Buxton, W. (1993). The limits of expert performance using hierarchic marking menus. *ACM CHI'93*, p. 482-487.

27. Kurtenbach, G., Sellen, A., Buxton, W. (1993). An empirical evaluation of some articulatory and cognitive aspects of "marking menus. *Journal of Human Computer Interaction, 8(1)*, p. 1-23.

28. Kurtenbach, G., Buxton, W. (1994). User learning and performance with marking menus. *ACM CHI'94*, p. 258-264.

29. Kurtenbach, G. (1997). Display and control of menus with radial and linear portions. *US Patent #5,926,178*.

30. Kurtenbach, G., Fitzmaurice, G. W., Owen, R. N., Baudel, T. (1999). The Hotbox: efficient access to a large number of menu-items. *ACM CHI'99*, p. 231-37.

31. Landauer, T. K., Nachbar, D. W. (1985). Selection from alphabetic and numeric menu trees using a touch screen: breadth, depth, and width. *ACM CHI'85*, p. 73-78.

32. Lecolinet, E., Nguyen, D. (2006). Représentation focus+contexte de listes hiérarchiques zoomables. *ACM IHM '06*, p. 195-198.

33. Lee, E. S., Raymond, D. R. (1993). Menu-Driven Systems. *Encyclopedia of Microcomputers, Vol. 11*, p. 101-127.

34. Maury, S., Athénes, S., Chatty, S. (1999). Rhythmic menus: toward interaction based on rhythm. *ACM CHI'99*, p. 254-255.

35. Norman, K. (1991). *The Psychology of Menu selection: Designing Cognitive Control at the Human/Computer Interface*. Ablex Publishing Corporation.

36. Pook, S., Lecolinet, E., Vaysseix, G., Barillot, E. (2000). Control menus: execution and control in a single interactor. *ACM CHI'00*, p. 263-264.

37. Rekimoto, J., Ishizawa, T., Schwesig, C., Oba, H. (2003). PreSense: interaction techniques for finger sensing input devices. *ACM UIST'03*, p. 203-212.

38. Rubio, J.M., Janecek, P (2002). Floating pie menus: enhancing the functionality of contextual tools. *ACM UIST'02* , pp 39-40.

39. Shneiderman, B. (1986). *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley Longman Publishing Co., Inc.

40. Sears, A., Shneiderman, B. (1994). Split Menus: Effectively using Selection Frequency to organize Menus. *Trans. on Computer-Human Interaction*, p. 27-51.

41. Soliz, E., Paley, W. B. (2003). A Re-Interpretation of Marking Menus: The Usage of Gestalt Principles as Cognitive Tools. *ACM UIST'03, poster*.

42. Tapia, M. A., Kurtenbach, G. (1995). Some design refinements and principles on the appearance and behavior of marking menus. *ACM UIST'95*, p. 189-195.

43. Tsandilas, T., Schraefel, M. C. (2007). Bubbling Menus: A Selective Mechanism for Accessing Hierarchical Drop-Down Menus. *ACM CHI'07*, p. 1195-1204.

44. Wiseman, N. E., Lemke, H. U., Hiles, J. O. (1969). PIXIE: A New Approach to Graphical Man-machine Communication. *CAD Conf. Southampton 463 IEEE Conference Publication 51*, p. 463.

45. Zhao, S., Balakrishnan, R. (2004). Simple vs. compound mark hierarchical marking menus. *ACM UIST'04,* p. 33-42.

46. Zhao, S., Agrawala, M., Hinckley, K. (2006). Zone and polygon menus: using relative position to increase the breadth of multi-stroke marking menus. *ACM CHI'06,* p. 1077-1086.

47. Zhao, S., Dragicevic P., Chignell M., Balakrishnan R., Baudisch P. (2007). EarPod: Eyes-free Menu Selection using Touch Input and Reactive Audio Feedback. *ACM CHI'07*, p. 1395-1404.