DESIGN GUIDE FOR 3D POINTING TECHNIQUES

Cédric Dumas, Frédéric Jourdan, Patricia Plénacoste, Laurence Perron, Amine Chellali

DESIGN GUIDE FOR 3D POINTING TECHNIQUES

Internal Report 08/4/AUTO
Ecole des Mines de Nantes
December 2008

This report is the English version of the Technique et Science Informatique journal (TSI) paper:

Guide de conception d'une technique de désignation 3D. Cédric Dumas, Frédéric Jourdan, Patricia Plénacoste, Laurence Perron, Amine Chellali. *Technique et Science Informatiques*, Hermès, TSI 6-7/2009 , 2009.

MINES Nantes

Graduate School of Engineering

Design guide for 3D pointing technique

Cédric Dumas[1,2], Frédéric Jourdan[1], Patricia Plénacoste[3], Laurence Perron[4], Amine Chellali[1,2]

(1) Ecole des Mines de Nantes
4, avenue Alfred Kastler F-44307 Nantes cedex 3
Frederic.Jourdan@emn.fr
Amine.Chellali@emn.fr

(2) IRCCYN
1, rue de la Noë F-44321 Nantes cedex 3
Cedric.Dumas@irccyn.ec-nantes.fr

(3) LIFL & INRIA Futurs
Bâtiment M3 F-59655 Villeneuve d'Ascq cedex
patricia.plenacoste@univ-lille1.fr

(4) Orange France Télécom R&D
2 avenue Pierre Marzin F-22307 LANNION cedex
laurence.perron@orange-ftgroup.com

ABSTRACT. *We present a panel of pointing techniques, both in 2D and 3D. Analyzing them led us to propose here a guide for 3D pointing technique design, dedicated to programmers and designers of virtual environments. An example of 3D semantic pointing is given following this guide.*

KEYWORDS : *pointing, 3D interaction, virtual environments.*

---

1. General introduction

While 3D has become widespread at the technical end of the market since the end of the 20th Century as far as computers are concerned, it is only starting its mass diffusion with applications meant for the general public (Google™'s GoogleEarth), bringing 3D out of specialized professional applications or applets. Virtual worlds, online games and even more probably 3D TV should soon force information flows to systematically take a three-dimensional approach. Concerning interaction devices, except for more mature fields such as virtual reality, wide-ranging achievements are recent, some 3D joysticks (Wii de Nintendo™) or some browser devices (NULOOQ de Logitech™) have appeared, but they are still relatively few.

Input and output devices have indeed evolved but their use is still relatively rare. With marketing (Windows Vista™ Aero) and specialized appplications (Time Machine in MacOS™); these evolutions however promise without any doubt a much richer visualization for the computer work environment. When it comes to interaction means with these representations, things are moving forward even slower. Especially with pointing, since the usual action point for any application is nothing else than a 2D point shown by a static arrow rightly called *pointer*.

However, the exclusive keyboard/mouse couple has to evolve. Computer classical vision is becoming more diverse, minds and expectations are changing following large scale innovative approaches such as Nintendo™ DS touch screen, cameras used as input devices (Sony™EyeToy) or multitouch screens (Apple™ iPhone).

One of the challenges faced by HMI with the industrialization of new interaction techniques (including peripherals) over the last twenty years is starting to be resolved with the development and improving maturity of the computing market, the rise of information technology culture and the expectations that come with it.

In this context, it seems useful to us to provide 3D design methods similar to the numerous ones existing for graphical interface called "WIMP", to promote the development of virtual environments. Here is heuristics that may guide designers through their choices for 3D pointing.

1. 1. *The Notion of pointing*

Collaborative virtual environments (CVE) are based on the use of a three-dimensional interface. With a 3D interface projection on one or several screens, the user can interact with objects - modelled, represented or perceived – in three dimensions. Under these circumstances, the notion of object pointing and selection has to be rethought from our classical 2D interfaces for which the mouse/pointer became imperative. In order to interact in a virtual environment it is necessary to be able to materialize one's intention:

1. to be able to move toward a virtual object,
2. to be able to point at a virtual object,
3. to be able to select, move and act on a virtual object.

These three actions, combined with appropriate environment feedback, allow us to handle 3D objects. In this article, we focus on the second point with an analysis of different pointing techniques, exclusively to act on an object (navigation/selection/handling), in relation to the ergotic function of the gesture (we do not broach here its epistemic or semiotic functions [1]).

This article aims to review the evolution of 2D/3D pointing techniques in recent years and to estimate the consequences on virtual environments. This article is divided into three parts: pointing techniques, comparative review and synthesis as a design guide.

2. Pointing and selection techniques

Different 2 and 3 dimension pointing techniques have been explored, described or implemented, sometimes assessed. They come from various contexts such as the classical WIMP desktop, 2D large display screens or 3D

(semi) immersive virtual reality systems. In this paper we present examples illustrating what already exists in this domain. These examples are grouped according to the techniques used for their completion.

2.1. *Area or volume cursor*

The action point in a virtual environment is usually shown by a pointer, which is a point cursor. An object is designated or selected when the pointer is placed on a point belonging to the object. In order to ease small object selection, different techniques replace the pixel size pointer with a larger size cursor: 2D area or 3D volume. An object is selected when it has a non-empty intersection with the cursor.

In a 2D environment, the term area cursor [2,3] refers to a cursor having the shape of a rectangular or circular area (figure I.a). In the case of a dense environment (including numerous close targets), an area cursor can encompass more than one object, leading to ambiguity about the designated target. To solve this problem, the bubble cursor [4] technique offers a circle-shaped area cursor that dynamically adapts its size so as to always encompass a single target (figure I.b).

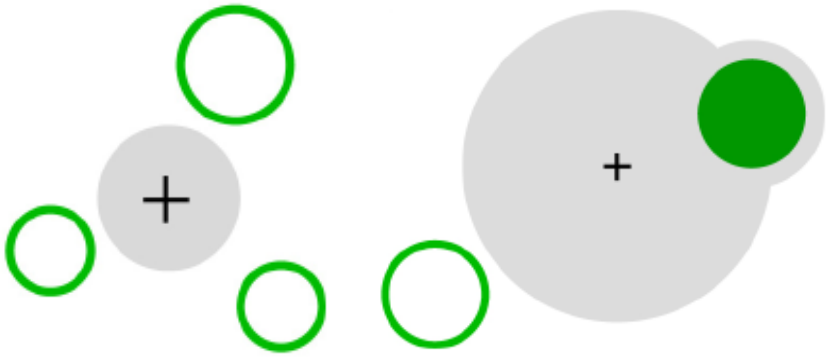


Figure I. ***a)*** *on the left, a circular* ***area cursor****;* ***b)*** *on the right a* ***bubble cursor*** *(targets are green, cursors are grey) – extracted of [4]*

Similar techniques exist for 3D interfaces. Zhai *et al.* proposed the **silk cursor** [5], made of a cubic semi-transparent box. An object placed behind the cursor is seen through two layers of silk, whereas an object placed inside the cursor's volume is seen through only one layer of silk. Because of semi-transparency, the cursor does not hide the objects placed behind it.

In 3D environments, a "ray cursor" based on the laser pointer metaphor is often used. A ray is projected from the user's hand into the 3D space and the closest object it intersects is selected (ray selection). This kind of pointer can be enlarged and become a cone shaped volume cursor, whose ray forms the axis (figure II). In case of multiple selection, the object closest to the axis of the cone is selected (cone selection). With the shadow cone selection technique, the user selects the intended target by moving his hand so as the target always stays inside the cone [6].

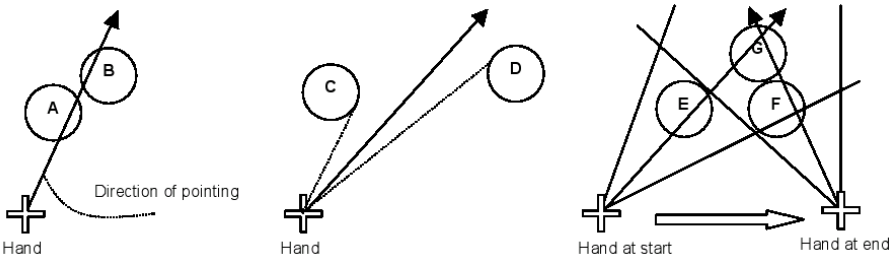


Figure II. *From left to right,* ***ray selection*** *(obj. A is selected),* ***cone selection*** *(obj. D is selected),* ***shadow cone selection*** *(obj. G is selected) – extracted of [6]*

2.2. *Removing ambiguity with multiple selection*

Introduced in the previous section, the issue of ambiguity between several possible targets during a pointing or selection operation is not specific to the use of area or volume cursors. The study of this issue in the context of overlapping 2D windows or an occlusion of 3D objects led to a set of techniques we regroup in this section.

The technique of **target chooser** [7] aims to facilitate the selection of a 2D window. When this technique is activated, a ray is cast perpendicularly to the screen surface and the window whose centre is the closest to the ray is selected. A visual cue shows the user the selected window. The user can move in the windows list using small mouse movements. In 3D environment, the techniques of **depth ray** and **lock ray** [8] also enable the user to cycle through the list of possible targets along a ray pointer, in the context of volumetric displays (cf. figure III).

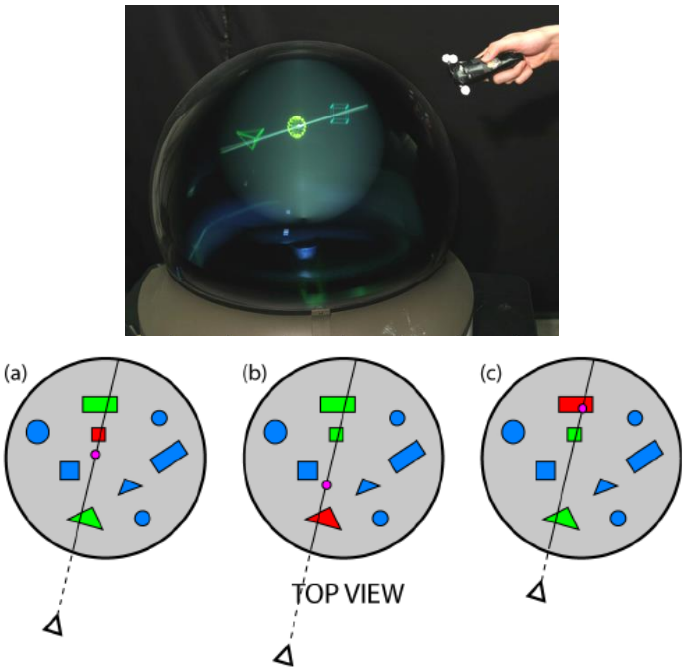


Figure III. *At the top, a volumetric display and a ray pointer. Below, a diagram illustrating the* ***depth ray*** *: the red point can be moved along the ray to select one of the three possible targets - extracted of [8].*

It is also possible, in 2D as well as in 3D to temporarily alter the scene's geometry to provide the user with an overview of potential targets. Using the **splatter** [9], a group of overlapping 2D objects is spread out so that the objects are placed in a circle around the activation point (figure IV). The user then selects the goal target on this view using the pointer. In a 3D environment, a similar spread view is used by the **flower ray** [8] technique with a ray pointer (figure V).

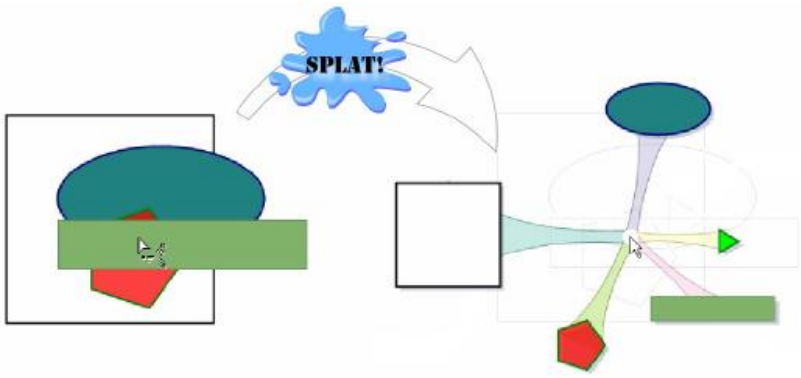


Figure IV. *Illustration of the **splatter** technique on a group of 5 overlapping objects, including one (small green triangle) that was occluded at first - extracted of [9]*

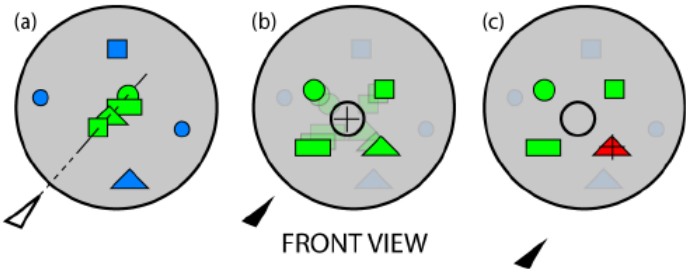


Figure V. *The **flower ray** technique when a volumetric display is used - extracted of [8].*

In the case of overlapping windows, the **fold'n'drop** [10] technique allows the user to fold and unfold the windows with direct handling, just like turning the pages of a notebook (figure VI).

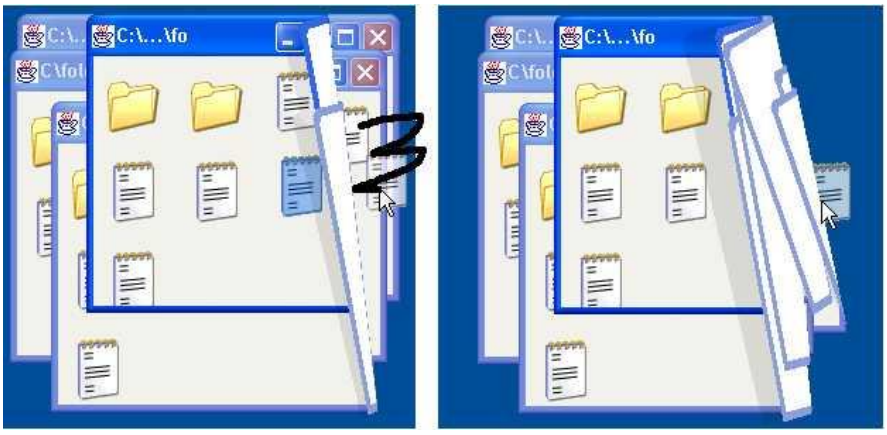


Figure VI. *Direct manipulation of windows with **fold'n'drop** - extracted of [10].*

2.3. *Zoom and Radar*

When pointing a target involves a too precise gesture (targets too small in a virtual environment) or a too wide gesture (too large virtual space), the system can display an enlarged representation of the virtual space (**zoom**) or a narrower one (**radar**) to ease pointing.

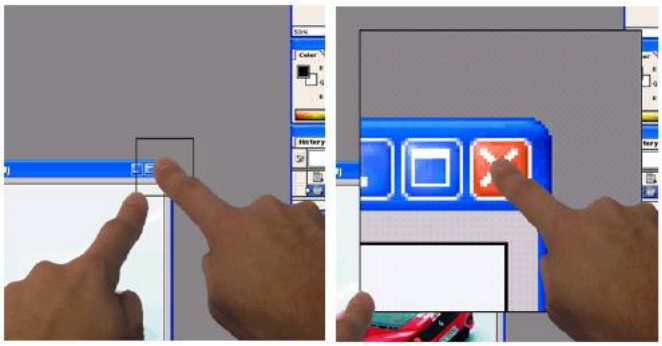


Figure VII. *Using a zoom –extracted of [11]*

We can find recent examples of these techniques with Benko *et al.* [11], who use a zoom activated by bi-manual interaction on a touch screen (figure VII), while Aliakseyeu *and al.* [12] offer with **bubble radar** (figure VIII) a juxtaposition of two interaction techniques, the first one being a radar and the other one a bubble cursor.

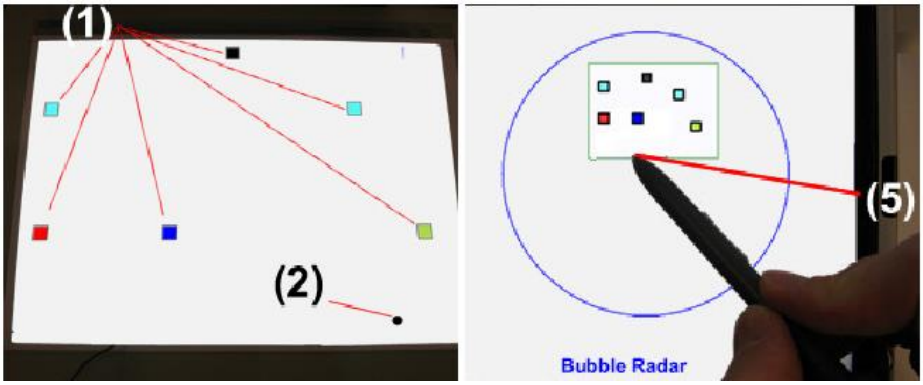


Figure VIII. *The radar (green rectangle on the right) is a narrower view of the left hand side work space where the six targets are represented (1) (shown by red arrows on the left) – extracted of [12].*

2.4. *Accessing remote targets*

In the context of large-surface displays (wall-size displays, tabletop displays…), many solutions have been suggested to facilitate the pointing of targets located far away from the user in the virtual environment. The work discussed in this section takes place in a 2D workspace where (desktop) icons are targets.

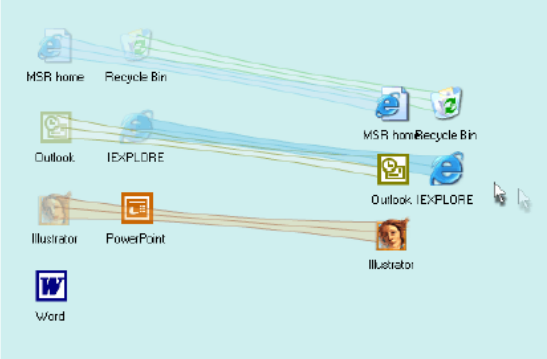


Figure IX. *Illustration of **drag and pick** – extracted of [13].*

In order to reduce the virtual distance between the pointer and the target, two different kinds of strategies have been considered. The first one aims to move the target closer to the pointer and is implemented through the techniques of **drag and pop** and **drag and pick** [13] (figure IX). These techniques are an extension of the traditional drag and drop and are activated in the same way: clicking with the pointer device, then moving it towards graphic elements. The icons placed in the direction the mouse movement points to are seen by the system as potential targets and are duplicated. These targets' doubles move closer to the pointer so that the user only needs a low-amplitude movement to select the intended target. Drag and pop and drag and pick are built on this principle, except that in the first case the movement is initiated from an icon (to move an icon onto another one) while in the second case, the movement is initiated from an empty screen space (selection of a remote target).

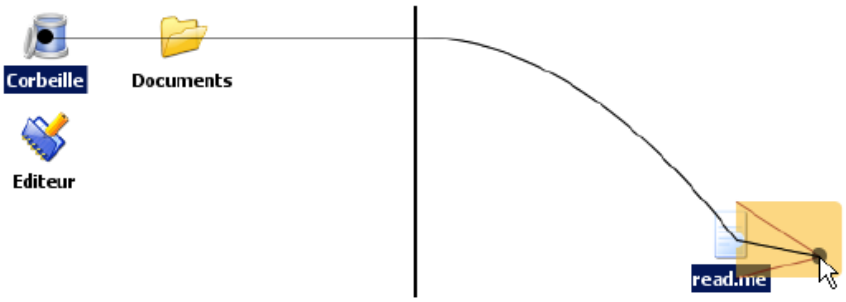


Figure X. *Example of **drag and throw** (archery metaphor) – extracted of [14].*

The second strategy consists in sending the pointer towards the target using a low-amplitude gesture. This is the approach used for **drag and throw** (figure X) and **push and throw** (figure XI) [14], which are techniques for throwing icons adapted to multiple display units. When these techniques are activated, a rectangular "take-off" area appears around the pointer. The pointer movements in this area are interpreted in terms of trajectory for the icon that can be thrown far away from the initial activation point. The difference between these two techniques lies in the metaphor used to determine the trajectory of the thrown icon: the metaphor of archery for one, the metaphor of the pantograph for the other.

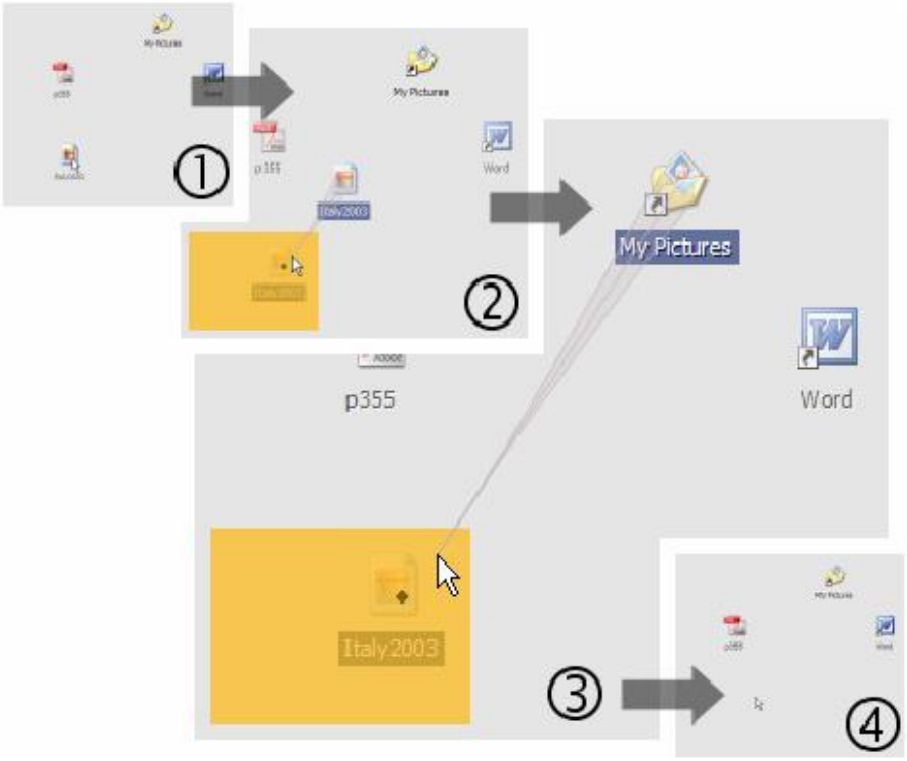


Figure XI. *Course of operations for a **push and throw**. The icon located in the bottom left is moved into the folder " My Pictures" – extracted of [14].*

Inspired by previous examples, Collomb *et al.* recently proposed to combine these two strategies (target towards pointer and pointer towards target) with the technique of **push and pop** [15]. The interaction proceeds as shown in figure XII. (1) The user starts dragging the icon he wants to move, as if to execute a drag and drop. (2) The system surrounds the pointer with a miniature representing the entire display. This "take-off area" is similar to a radar, but the icons keep their original size, unlike a traditional radar that shrinks them. (3) The user lets go of the icon to move over the recycle bin icon. (4) The user just has to release the button of the pointing device.

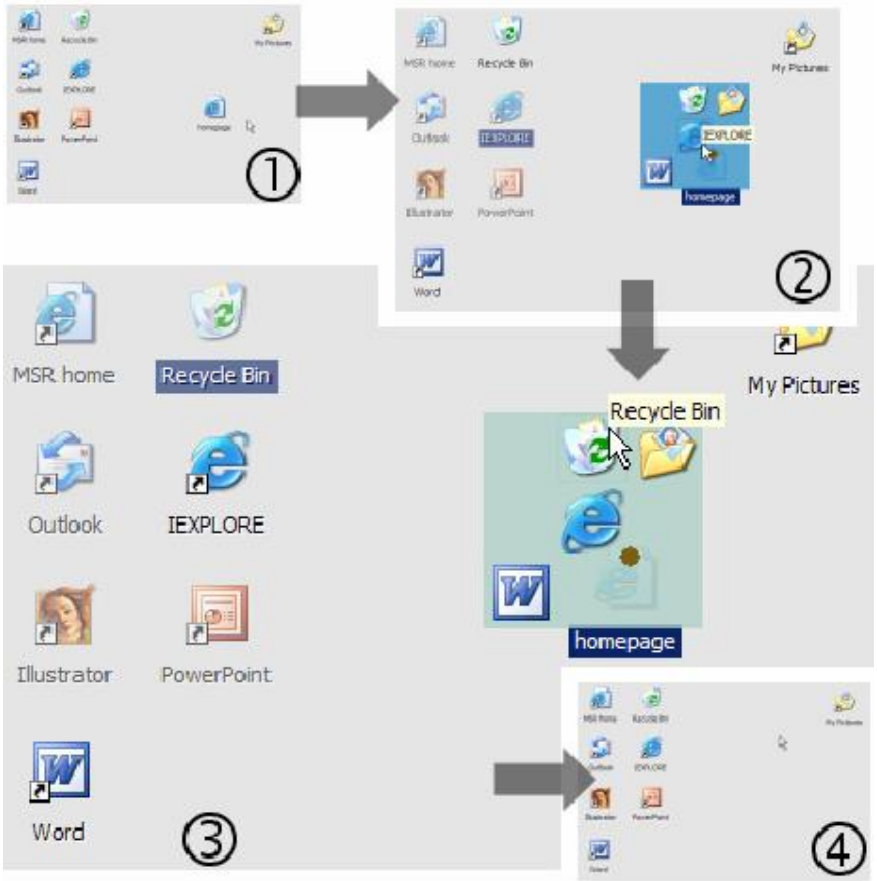


Figure XII. ***Push and pop*** *walkthrough: the icon located in the middle on the right is moved into the recycle bin – extracted of [15].*

Different from other examples is the **tractorbeam** [16]: an exclusively 3D input device (figure XIII), a six degrees of freedom stylus (DOF) is used to interact with a 2D display (a tabletop display). The following illustration explains this device.

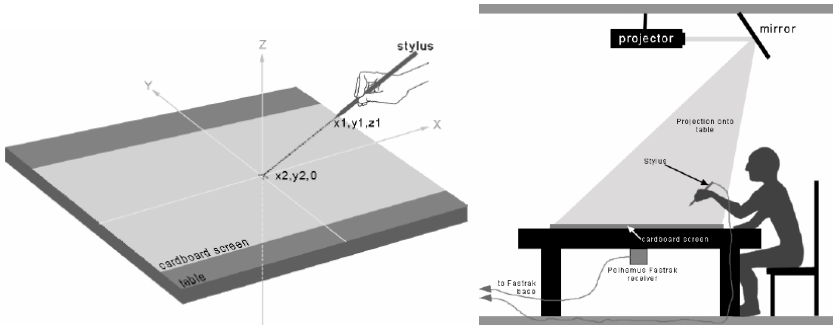


Figure XIII. *The tractorbeam – extracted of [16].*

2.5. *Non-linear control-display ratio*

We discuss here approaches that play on a dynamic modification of the ratio between the pointing device and controlled object movements (control-display ratio or CDR). A well-known example is the « mouse acceleration » for a 2D pointer. These approaches have technical examples in 3D and 2D.

Using the **Go-Go** technique [17], the moving speed of a 3D pointer in a virtual reality environment changes according to the distance between the user's hand and chest. With the same gesture, the user can reach the objects located in his/her immediate environment or far away.

Semantic pointing [18] is a technique in which the CDR is adapted according to the semantics of the displayed objects (figure XIV). More precisely, the pointer speed depends on the pointer position in relation with the displayed objects. The principle is to move the pointer at a slower speed on (or around) potential targets and at quicker pace on empty space. According to the authors, the technique consists in modifying the size of displayed objects in the motor space instead of in the visual space, which enables to avoid the display distortions inherent to the techniques described in the previous section.

Figure XIV. *Semantic Pointing: (a) a dialogue box in the visual space, (b) the same box in the motor space with semantic pointing – extracted of [18].*

The same authors also put forward another version of this principle, called **object pointing** [19], in which the pointer can only be above a target, " jumping" from one target to another, according to the movements of the pointing device.

Finally, and adaptation of their work in 3D has been proposed in [20], extended to any geometric form (not only rectangles), but to point an object on a 3D plane with a 2D device (with the two DOF of a mouse for example). A real-time non-linear CDR algorithm is described, using hardware acceleration facilities.

3. Comparisons and Analysis

We now give a comparative review of the different techniques we have presented, with regard to devices used, the context and the tasks they are meant for.

3.1. *2D Techniques: classification according to implemented devices*

The following chart presents the 2D pointing techniques according to the input/output devices they use. Different types of 2D input devices we consider are:

1. direct absolute: touch screen, light pen...
2. indirect absolute: tablet...
3. indirect relative: mouse...
4. non gestual: trackball...

For each technique, we mark its compatibility wih the different devices:

- ***Orig :*** the device(s) it originally comes with,
- ***Adap :*** the devices we think the technique can be adapted to,
- ***Incpt :*** show the devices incompatibles with the techniques,
- ***Dedic:*** show if the technique is dedicated to the output device used.

	Direct absolute	Indirect absolute	Indirect relative	non gestual	Output device	Dedicated ?
area cursor	Orig	Orig	Orig	Adap	Traditional screen	
bubble cursor			Orig	Adap	Traditional screen	
target chooser			Orig	Adap	Traditional screen	
Splatter			Orig		Traditional screen	
fold'n drop			Orig		Traditional screen	
Zoom	Orig	Incpt	Incpt		Multitouch screen	Dedic
bubble radar	Orig				Tabletop screen+ tablet	Dedic
drag and pop/pick	Orig				Wall-size display	
drag/pick and throw	Orig				Wall-size display	
push and pop	Orig				Wall-size display	Dedic
pointage sémantique		Orig			Traditional screen	

Input devices

The 2D selection techniques we presented in the first section of this paper were all devised for the traditional metaphor of a desktop with windows and/or icons. The fact that there is only one context explains the homogeneity of the devices used. For each case, they are all gestual devices (the gesture made with the device is transmitted to the pointer), that include direct absolute devices (touch screen stylus) and indirect relative devices (traditional mouses).

With a few exceptions (zoom, bubble radar), the work which led to these techniques did not aim to exploit the specific details of particular devices, but to offer wider interaction techniques. For this reason, there is no specialization on input devices: each technique[1] is adaptable for all kind of gestual devices (not specified in the chart to keep it clearer). On the other hand, for many techniques (pointer throw...) the gesture is so important that they are not compatible with non gestual devices.

Output devices

With regard to output devices, the similarity of context between different techniques also explains the lack of existing variety and the number of techniques using a traditional screen. But unlike input devices, there is a family of techniques oriented towards the particularities of a kind of output device. Large displays, that often include several screens, have given rise to numerous works (drag and pop, drag and throw...). Although these techniques are presented as dedicated to such devices (drag and pop), we think they can still be used with more traditional screens.

[1] With an exception for the zoom technique, that is, in the version we present, dedicated to a specific device including several pointers driven by the user's fingers.

3.2. *2D techniques: classification according to the context*

We discuss here the different 2D pointing and selection techniques according to the context in which they have been introduced. We also consider the different kind of selection tasks they are adapted to.

For the selection tasks, we consider these two axes. The first one applies to the number of objects to select one isolated object or a group of objects. The second one focuses on the density of the environment which can be dense (numerous close objects) or scattered (a few objects far away from each other).

As regards context, we make a distinction between four connected groups of techniques, corresponding to the four following problematics:

1. facilitating general selection in a classical graphic user interface (GUI),
2. allowing selection in case of objects overlapping,
3. finding appropriate techniques for large size displays,
4. improving drag and drop technique.

The first of these groups concerns pointing selection in a general context of a classical graphic user interface (desktop, icons), without any specific reference to a situation or equipment. It includes the area cursor technique and its successive by-products bubble cursor and bubble radar. In the original article [3], the objective was to facilitate the selection for the elderly. Although it is first and foremost designed to select an isolated object, this technique of area cursor can be used to select a group of objects. On the other hand, it is not appropriate in a dense environment. This shortcoming is corrected by the technique of bubble cursor, but only for the selection of just one object. The bubble radar is made for tasks such as the selection of isolated objects and also the pointing of any object in free space.

Still in this first group, the technique of semantic pointing aims at improving the efficiency of target selection wherever it is located in a virtual space. This improved entire virtual space exploration capability is based on the use of knowledge about the nature of objects located in the environment. Allowing a quick crossing of area with no potential targets, this technique is appropriate for scattered environments. However it is not only intended for those since the disconnection between visual and motor spaces makes it possible to densify the environment shown (important targets are bigger in the motor space, but smaller on screen).

Of these techniques, three have been presented in a context of objects overlapping (windows or components of a 2D drawing), typical of a dense environment. The target chooser aims at selecting a window among overlapping windows. Using the splatter, it is possible to select an element of a 2D drawing in case of occlusions. Finally, the fold'n'drop allows drag-and-drop operations between overlapping windows.

The developed techniques are numerous for large displays (tabletop displays, wall-size display or even multiple displays). The zoom techniques are essentially intended for the selection of small targets, no matter how dense the environment is. The use of radar is relevant in a scattered environment. Techniques of **drag and pop/pick**, **drag/pick and throw** and **push and pop** are specifically designed for large displays and therefore inappropriate in dense environments.

We note that different techniques aim at improving drag and drop operations :

- in case of windows overlapping with fold'n'drop,

- in case of large displays with drag and pop/pick, drag/pick and throw and push and pop.

The previous comments are all synthesized in the following chart in which the crosses X show the compatibility between techniques and different tasks and environments:

	Context				Tasks			
	Classical GUI	overlapping	*large displays*	*drag and drop*	Isolated object	Group of objects	Dense environment	Scattered environment
area cursor	X				X	X		X
bubble cursor	X				X		X	X
target chooser		X			X		X	
splatter		X			X	(X)	X	
Fold'n'drop		X		X	X		X	
zoom			X		(X)	(X)	X	X
bubble radar	X		X		X			X
drag and pop/pick			X	X	X			X
drag/pick and throw			X	X	X			X
push and pop			X	X	X			X
semantic pointing	X				X		X	X

3.3. *3D Techniques*

In a 3D environment, the selection usually involves a collision between the intended target and the user's hand (virtual or real) or the intersection between the target and a ray cast from the same hand. The techniques of silk cursor and go-go belong to the first category, whereas the techniques of shadow cone, depth/lock ray and flower ray come under the second one. The tractorbeam technique, hybrid between 2D and 3D, falls within the two categories.

Input and output devices

In most cases, the input device used requires a six degrees of freedom sensor (DOF), which can follow the position and the direction of the user's hand. The silk cursor that only involves translation movements only needs a three degrees of freedom data glove. More generally, three DOF are sufficient for pointing by intersection with a 3D pointer, while ray pointing involves five DOF. We will come back to this point later.

Focusing rather on virtual or augmented reality, our benchmark techniques use advanced output devices :

1. non-immersive: stereoscopic CRT display (silk cursor, go-go) ;
2. immersive: head-mounted screen (go-go) ;
3. spatialized immersive: CAVE-type (shadow cone) ;
4. 3D volume rendering devices (depth/lock ray, flower ray) ;
5. interactive table (tractorbeam).

The authors conceived the techniques as dedicated to specific devices. Therefore, these techniques are intended to exploit the particularities of the 3D environments they have been designed for. It is not necessarily possible to use them in other environments, such as the traditional combination of a common screen and a mouse (2 DOF available for requiring 5 DOF techniques).

Context and area of action

While the silk cursor [5] has been presented in a very general context of the exploration of new 3D interaction techniques, the other reference techniques intervene in a more specialized context. The tractorbeam has been specifically designed for interactive tables. The depth ray and flower ray are intended for new tri-dimensional volume rendering devices. As for the shadow cone, it is meant to be used in a multi-user spatialized immersive environment to deal with the fact that the view is correct for only one user.

These techniques have been presented for the same task : unique object selection. The techniques of depth/lock ray, flower ray and shadow cone are specifically designed to be efficient in a dense environment. The purpose of go-go and tractorbeam techniques is fluid integration of interaction with objects close to the user and interaction with objects far away from the user (avoiding navigation).

These two techniques provide two different interaction modes depending on the distance to the target, with a natural transition between the close and remote modes. The silk cursor enables interaction with close objects. With the shadow cone, based on a ray pointer, only one mode allows interaction with the entire environment, but sometimes the handling required to solve problems of target ambiguity can be impractical.

As to the scope of their areas of action, the techniques dedicated to volumetric devices have a special position : because of their small size, all objects can be regarded as close, in this case, the problem comes from the density of the environment.

3.4. Results

Since we are focusing on 3D interaction (and not navigation), we naturally decided to treat 2D and 3D techniques separately, as they present different problematics. For 2D techniques, the main issue is to know to what extent they can be generalized to 3D. Among the techniques we have presented, some 3D techniques can be seen as generalizations. The following couples can be formed:

2D	3D
area cursor	↔ *silk cursor*
bubble cursor	↔ *shadow cone*
splatter	↔ *flower ray*
target chooser	↔ *depth/lock ray*

Among the 2D techniques, semantic pointing, with go-go interaction, share the non linear CDR.

As for 3D techniques, the examples we used were mostly dedicated to a specific device. In order to be able to choose the most appropriate technique according to the task and not to a given device or display, we would like to provide a design method for pointing in a virtual environment.

We will remember for the next parts that all these techniques differ in a few fundamental points:

1. dedicated input/output devices,
2. the displacement mode of the action point (CDR),
3. the definition of their action point, and the object selection algorithm.

A pertinent choice of these three elements is critical in a virtual environment, where the third dimension, the diversity of viewpoints, the variable visibility of objects and all the possible input devices, make the design of a pointing technique harder than in 2D. Each new application has specific requirements that prevent from using a single general solution (as systematic mouse use in 2D).

The (1) argument is task dependant. The (2) argument is known since [17] and formalized since [18]. But the (3) argument needs a full description of what is possible to do before we can summarize all these points in a final guide:

4. Action point modelling

The action point is the user's reality in his/her virtual environment.

As we have seen with the pointing techniques listed in the previous part, the mouse remains the major device for 2D applications as well as for research. Only a few virtual environments differentiate themselves from these two degrees of freedom for pointing.

4.1. *The pointer*

Thanks to their senses and physical abilities, human beings have extensive action abilities in the real world. When studying the technical gesture only, we found a multitude of tool handling abilities in various situations, which allowed humans to produce everything they could (or almost everything). As for computing, the generalization of the WIMP model for all computing applications reduced all these possibilities of action to the mouse's two degrees of freedom. The mouse gets accessories (scroll wheel, multiple buttons...) and other devices become widespread as we noted, but the user's action point, capable of bimanual, tri-dimensional, voice interaction, remains largely a 2D point.

However, this action point is the only materialization of the user in the work virtual environment. We can accomplish highly complex tasks but we are heavily disadvantaged by an excessive factorization of gestures, in the form of a long succession of mouse clicks.

To this acknowledgement, we can object that touch and stylus interfaces (TabletPC, PDA) are coming into ever wider use, that virtual environments never suffered from these limitations thanks to gestual interaction and its numerous 3D devices, or even that synchronous collaborative work or post-wimp applications allowed us to go beyond this limitation. And they are probably not the most common cases.

Action point perception

The action point, when it is a 2D point, is manipulated through gestual devices, with a correlation between the real movement (the hand) and the virtual movement (the pointer) that gives to the user great performance, even if the movement is not in the same plan or if the ratio is non linear (CDR). However, we have noticed in the previous part that using dynamic CDR (or adaptative CDR with semantic pointing) requires a slightly longer training period [18].

The perception of the action point in the environment is much easier if its behaviour corresponds to the user's gestures and if it is easily possible to create a mental pattern of the way the pointer functions according to the gestures (allowing non linear CDR for instance).

Therefore, the choice of navigation techniques is a compromise between direct gestual pointing which is natural to the user and optimized moving of the action point in order to improve its performance in a given environment.

Choosing the action point

The choice of the navigation point largely depends on the kind of device : the pointer for the mouse, the ink point for stylus, the virtual hand for datagloves, ... However, unlike the 2D WIMP interfaces, in 3D a fixed-size 2D pointer is rarely sufficient since the depth and complexity of scenes show objects of very different sizes, including some quite small or composite, for which a fixed-orientation is not suitable. Indeed directing the 3D pointer toward the closest object is often necessary to alleviate the ambiguity in pointing. Similarly, it is recommended to consider the pointer size against the object size so as to get an optimal position.

The action point in 3D has to be a dynamic object that takes the application context into account, which means:

1. of the size of nearby objects,
2. of the object density,
3. of the kind of task the user has to carry out, initiated by pointing (navigation, handling, etc.),

4. of the possible feedbacks of objects involved,
5. of its dynamic features (animation) depending on the type of device and on its CDR,
6. of the user's actions and of the prevention of user's mistakes (dealing with the pointer coming out of the workspace, the visual loss of pointer, ...).

This last point can greatly help the users with feedback on performed actions. The user focuses his/her attention on the action point, if the action point conveys suitable (syntactic, lexical and semantic feedback) and relevant messages, the activity is better perceived.

The 3D application designer can rarely content himself with a default action point (such as the arrow of WIMP toolbox), he has to rethink things well in advance and to integrate a specific design in his development work.

4.4. *The designated objects*

In 3D, there are different strategies to establish which object(s) is/are designed from a point or an action area. They depend on the number of degrees of freedom of the input device and on the selection algorithm for designated objects.

We do not discuss here the possible devices or CDR but only the principles of 3D object selection for pointing in a 3D scene. This is, in our opinion, the key point allowing a consistent classification of techniques, in order to help the designer with his choices.

NB : We present the following diagrams in two dimensions (cross section), but obviously they correspond to 3D pointing techniques.

Simple methods of selection

The pointing algorithm only takes geometric elements of the scene into account. Depending on the modelling quality and the application needs, it can act on different granularities of the scene:

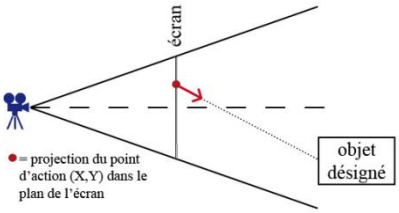
– on base geometric data (usually facets),

– on a precise object in the case of a first structuration of the scene (list of present objects for instance),

– on a sub-object in case of a more structured scene (scene graph, tree, etc.).

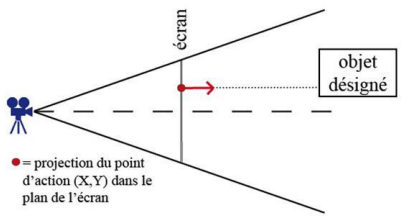
There is a parallel with the lexical level of a 3D scene. Pointing is managed in terms of physical description of the world.

A – Moving + orientation pointing



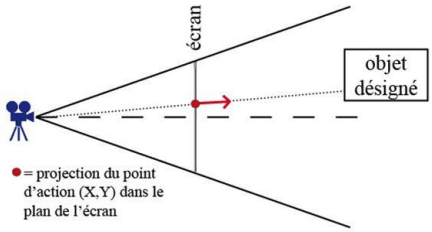
This is a throw of ray oriented by a point moved in the projection plan (or moved in 3D in the case of a 3 DOF device) and oriented by 2 DOF in the space. The ray intersects the target object(s). This technique requires the ray course to be displayed so that the pointer orientation is perceptible.

B –Ray parallel to the axis pointing



A ray is cast from a point moved in the projection plan, orthogonally to this plan. This kind of selection avoids occultation by any pointer in the target object projection plan. It requires the ray to be displayed since the orientation differs from the observer's eye axis (virtual camera). Moving the launch point in depth (3 DOF device) enables the user to designate an object placed behind another one.

C – Ray in the observer's axis view

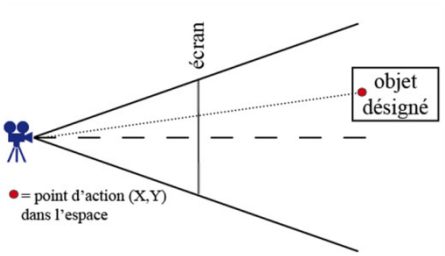


This is the most traditional_pointing technique with a 2 DOF type device (mouse,...) for 3D interaction. The ray is cast from the camera and goes through a point moved in the projection plan. This pointing technique is relatively natural since it follows perspective.

However, this technique is often used with a standard 2D pointer coming directly from a WIMP environment to point in a 3D environment, which has several side effects:

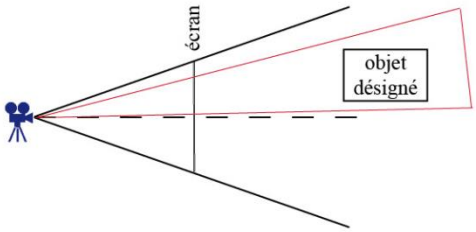
- the lack of feedback for the pointing method (with the display of a ray, the pointer moving in 3D, and so on) in the 3D environment does not help the user to perceive the environment in 3D,
- the 2D pointer is not represented in 3D (flat and static in the projection plan) with the 3D lightening model, this reinforces the feeling of pointing in a plan and is unfavourable to the homogeneity of the work environment and therefore to its general perception,
- on the other hand, the pointing actions seem to be easier but the scene has to be designed like a WIMP type surface (no overlapping, no occultation, enough space between objects...), which is quite rare and restricts the use of 3D.

D – 3D pointing



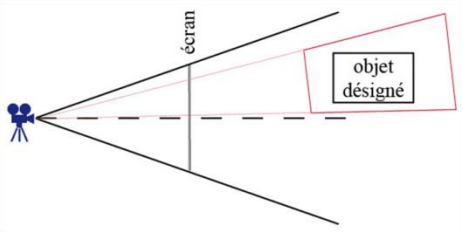
This technique consists in moving the action point directly in 3D using a 3 DOF device. An object in contact with the action point is selected.

E – Cone pointing



The objects included in the pointing cone are selected. The cone is centred on a point moving in the projection plan, it can be manipulated with 2 DOF. With more degrees of freedom, the cone opening or length can be dynamically defined.

F – 3D cone pointing

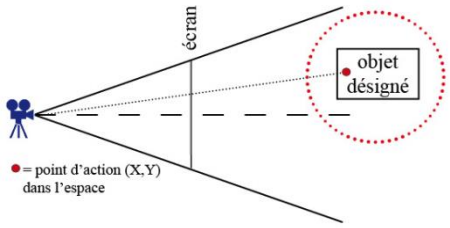


Here the cone is manipulated in three dimensions in terms of position, using a 3 DOF or more (in this case, the orientation is managed by the input device) .

Structural methods of selection

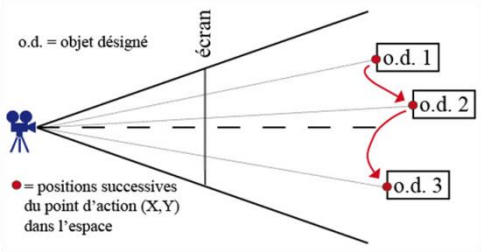
These object selection algorithms take the object geometry into account. They interpret it so as to alleviate pointing ambiguities (close, numerous, small objects, etc). A parallel is possible with the syntax level of a 3D scene: the pointing is performed according to the scene layout.

G – 3D pointing in an encompassing area



This method consists in selecting an object by coming close enough to be in its predefined aera. With this method it is possible to select an object without touching it, which makes reaching the target easier, specially with small or composite objects, or in dense environments. The area can be a box or an encompassing sphere or even a space subdivision. If the objects in the scene are of very different sizes, the area volume has to be a non-linear function of the object size (the volume is enlarged for small objects and close to reality for large objects).

H – Structural pointing

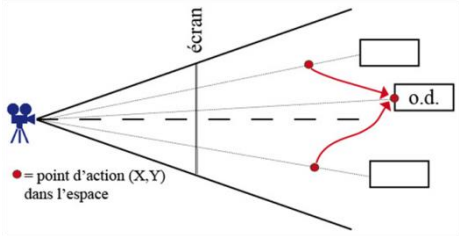


It is knowledge of the geometrical structure that allows us to pass from an object to another. The action point moves according to the device's movements and stops on visible positions (that may be pre-calculated) of the scene's 3D objects, with techniques of the magnetization type. This technique works with absolute and relative devices. Magnetization can occur only at a given distance from each object, in this case a pointer is moved in space.

The advanced pointing methods

These target object selection algorithms take the object semantics into account. They interpret it so as to deal only with objects for which it makes sense in the application. There is a parallel with the semantic level of a 3D scene: the pointing is operated while taking into account the application's semantic representative data.

I – Semantic pointing



This time knowledge of the kind of objects and of their features is used to interpret the action point moving towards one object or another. The action point behaviour highly depends on the kind of task the user has to perform and on the kind of object manipulated. As for the previous method, it can be used with different DOF numbers.

The virtual environments are usually very rich and can distinguish the scenery from avatars, tools, documents, and so on... to allow a selection based on the object semantic.

This method can be associated to a movement of the action point based on a semantic CDR [18], as well as a linear CDR or else. Indeed, there are two different aspects (choosing the method for moving the action point on the one hand, the method for pointing in the other hand) as we will see in the next part.

4.5. *Results*

The 3D object pointing has to respect and capitalize on the environment in which it is used : animations and feedback on the user's actions while ensuring their consistency, context and devices, object density and size, distances, and above all selection algorithm. All the elements we have seen so far enable us to identify different aspects in the pointing problem that, all together, can help in the conception of a pointing technique.

5. Pointing technique design guide

Recently, works have appeared to adapt existing methods in ergonomics, psychology, cognitive science, didactics to the specificities of virtual environments [21, 22, 23]. The assessments carried out generate guides and methods dealing with a specific issue. Each guide or method corresponds to a specific use: from properties of interaction technologies [24, 25] in these environments to scenario design for learning and analysis of these environments. Many disciplines contribute to expanding knowledge in order to improve virtual environment design. However to date there is no design guide to help designers choose the appropriate pointing technique depending on the context of use.

Here we offer such a guide that designers and 3D environment experts can use as heuristics. It can be used during prototyping phases, from design with paper sketch to functional prototypes, to help conception or evaluation of their pointing model.

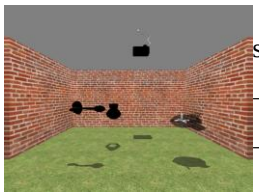
5.1. *Design guide*

A virtual environment design involves a user-centered design that takes tasks to achieve and activity context into account. This kind of design guide is particularly difficult to achieve due to the diversity and complexity of the situations. Beyond design methods based on the user, his/her activity and context, it is possible to provide the designer with overall guides on a specific phase of design, such as the one we propose here on 3D pointing. It generalizes knowledge we have on pointing.

To use this guide you need to have gone through the previous stages of design and to know about the user's activity, 3D environment, what kind of objects are present, what their functions are.

Device choice DOF isotonic / isometric direct / indirect discrete / seamless	*Determining the device and the way it's used depending on the number of required degrees of freedom, on possible haptic feedback. It must be appropriate to the task and the user's characteristics*
Screening absolute / relative Control-to-Display Ratio	*Interpretation of device data, which means transforming the physical device data into data that fits the virtual environment and the pointing mode (see above)*
Action point 3D representation(s) animation model dynamic orientation management (size variation, etc)	*Determining the shape and the behavior of the action point(s), especially if the pointer's appearance changes with the density or the type of objects, or if the pointer's direction or size change with the position and size of the objects.*
Pointing mode ray / point / volume (cone,...) syntactic / semantic	*Choosing the objects' selection algorithm (cf. part 4.4), the pointing rules: how to calculate the distance between objects? How to choose the closest object?*
Direct feedback on objects that can be selected Environment change Action point change Visual / virtual clues	*Display of feedback and clues that help to perceive the 3D space, the pointing mode and the objects about to be pointed. If some objects cannot be selected, it must appear clearly (shaded, part of the scenery, etc.)*
Feedback on selected objects Selected objects visibility Selected objects highlighting	*Clearly showing which objects have eventually been designated (and possibly selected) : change of color, scale, illumination or shape (framing, silhouetting, etc.)*
Deselection mode Techniques to come back to a no selected object condition	*A simple mode to release object(s) currently being selected has to be designed. It must be done in compliance with possible object handling so that the different modes (pointing/ selection/handling) follow on smoothly from one another.*

5.2 *Illustration*



A simple example[2] has been developed in order to illustrate this guide on a 3D semantic pointing technique:

- Choice of device: mouse (2D + Z scroll wheel)
- Filtering: semantic CDR, the action point slows down in the virtual space when it is going through an object.
- Action point: a static 3D model of cross, shadow projection of the cross and pointable objects.
- Pointing mode: an object is selected when the action point is going through its encompassing volume (here a bounding-box parallel to the axis). The object must appear on a given list of objects (cf semantic pointing, part 4.4).
- Feedback on pointed objects: the objects become transparent.

5.3 *Works to come*

The design of different interaction techniques in a virtual environment requires us to consider simultaneously navigation, pointing and manipulation of objects. It must fit the tasks the user wants to achieve. In our future works on these different aspects, we will continue modeling heuristics, with the purpose of offering in the end a coherent set of design guides for virtual environments.

6. Bibliography

[1] Cadoz, C., *Le geste canal de communication homme/machine : la communication "instrumentale"*, Technique et science informatique, 13(1), pp. 31-61, 1994.

[2] Kabbash, P. et Buxton, W., *The "prince" technique: Fitts' law and selection using area cusors*, Proceedings of ACM CHI Conference on Human Factors in Computing systems (CHI'95), pp. 273-279, 1995.

[3] Worden, A., Walker, N., Bharat, K. et Hudson, S., *Making computers easier for older adults to use: area cursors and sticky icons*, Proceedings of ACM CHI Conference on Human Factors in Computing systems (CHI'97), pp. 266-271, 1997.

[4] Grossman, T. et Balakrishnan, R., *The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area*, Proceedings of ACM Conference on Human Factors in Computing Systems (CHI'05), pp. 281-290, 2005.

[5] Zhai, S., Buxton, W. et Milgram, P., *The "silk cursor": investigating transparency for 3D target acquisition*, Proceedings of ACM CHI Conference on Human Factors in Computing systems (CHI'94), pp. 459-464, 1994.

[6] Steed, A. et Parker, C., *3D selection strategies for head tracked and non head tracked operation of spatially immersive displays*, 8th International Immersive Projection Technology Workshop, 2004.

[7] Robertson, G., Czerwinski, M., Baudisch, P., Robbins, D., Smith, G. et Tan, D., *The large-display user experience*, IEEE Computer Graphics and Applications, 25(4), pp. 44-51, 2005.

[8] Grossman, T. et Balakrishnan, R., *The design and evaluation of selection techniques for 3D volumetric displays*, Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'06), p. 3-12, 2006.

[9] Ramos, G., Robertson, G., Czerwinski, M., Tan, D., Baudisch, P., Hinckley, K. et Agrawata, M., *Tumble! Splat! Helping users access and manipulate occluded content in 2D drawings*, Proceedings of the working conference on Advanced visual interfaces (AVI'06), p. 428-435, 2006.

[10] Dragicevic, P., *Combining crossing-based and paper-based Interaction paradigms for dragging and dropping between overlapping windows*, Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'04), pp. 193-196, 2004.

[2] A simple example of 3D semantic pointing : http://www.emn.fr/x-info/cdumas/pointing3D/

[11] Benko, H., Wilson, A. et Baudisch, P., *Precise Selection Techniques for Multi-Touch Screens*, Proceedings of ACM CHI Conference on Human Factors in Computing systems (CHI'06), pp. 1263-1272, 2006.

[12] Aliakseyeu, D., Nacenta, M., Subramanian, S. et Gutwin, C., *Bubble radar; efficient pen-based interaction*, Proceedings of the working conference on Advanced visual interfaces (AVI'06), pp 19-26, 2006.

[13] Baudisch, P., Cutrell, E., Robbins, D., Czerwinski, M., Tandler, P., Bederson, B. et Zierlinger, A., *Drag-and-pop and drag-and-pick: techniques for accessing remote screen content on touch- and pen-operated systems*, Proceedings of the INTERACT' Conference, pp. 57-64, 2003.

[14] Hascoët, M., *Throwing models for large displays*, Proceedings of the British HCI Group Conference (HCI'03), pp. 73-77, 2003.

[15] Collomb, M., Hascoet, M., Baudisch, P. and Lee, B., *Improving drag-and-drop on wall-size displays*, Proceedings of the 2005 Conference on Graphics Interface, pp. 25–32, 2005.

[16] Parker, J., Nunes, N. Mandryk, R., Inkpen, K., *TractorBeam selection aids: improving target acquisition for pointing input on tabletop diplays*, technical report CS-2004-10, université d'Halifax, Nouvelle-Écosse, Canada, 2004.

[17] Poupyrev, I, Billinghurst, M., Weghorst, S. et Ichikawa, T., *The go-go interactive technique, non linear mapping for direct manipulation in VR*, Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'96), pp. 76-80, 1996.

[18] Blanch, R, Guiard, Y., Beaudoin-Lafon, M., *Semantic pointing: improving target acquisition with control-display ratio adaptation*, Proceedings of ACM CHI Conference on Human Factors in Computing systems (CHI'04), pp. 519-526, 2004.

[19] Guiard, Y., Blanch, R, Beaudoin-Lafon, M., *Object pointing: a complement to bitmap pointing in GUIs,* Proceedings of the 2004 Conference on Graphics Interface, pp. 9-16, 2004.

[20] Elmqvist, N., Fekete, J.-D., *Semantic Pointing for Object Picking in Complex 3D Environments*. Proceedings of the 2008 Conference on Graphics Interface, pp. 243-250, 2008.

[21] Dubois E, Mansoux B., Bach C, Scapin D, Masserey G,Viala J., *un modèle préliminaie des domaine des systèmes mixtes*, Actes de la 16ème conférence de l'Association Francophone d'Interaction Homme-Machine (IHM'04), pp. 61-68, 2004.

[22] Bach, C, Scapin D.L *Adapation des critères ergonomiques aux interactions hommes environnement virtuel*, Actes de la 15ème conférence de l'Association Francophone d'Interaction Homme-Machine (IHM'03), pp 24-31, 2003.

[23] Burkhart, J.M., Plénacoste P., Perron, L., *Concevoir et évaluer l'interaction utilisateur-environement virtuel*, dans le Traité de la Réalité Virtuelle, volume 2, éditions Ecole des Mines de Paris, pp. 473-521, 2006.

[24] Hillaire, S., Lécuyer, A., Cozot, R., & Casiez, G., *Using an Eye-Tracking System to Improve Depth-of-Field Blur Effects and Camera Motions in Virtual Environments*, Proceedings of the 15th Annual IEEE Virtual Reality 2008, pp. 47-50, march 2008.

[25] Casiez, G., & Chaillou, C. Effects of DOF Separation on Elastic Devices for the Navigation in 3D Virtual Environments with Force Feedback, Proceedings of IEEE World Haptics 2005, 483-486, 2005.