

Finding hidden objects in large 3D environments: the supermarket problem

Leonardo Fischer, Guilherme Oliveira, Daniel Osmari, Luciana Nedel

Institute of Informatics

Federal University of Rio Grande do Sul – UFRGS

Porto Alegre, Brazil

{lgfischer, gnoliveira, dkosmari, nedel}@inf.ufrgs.br

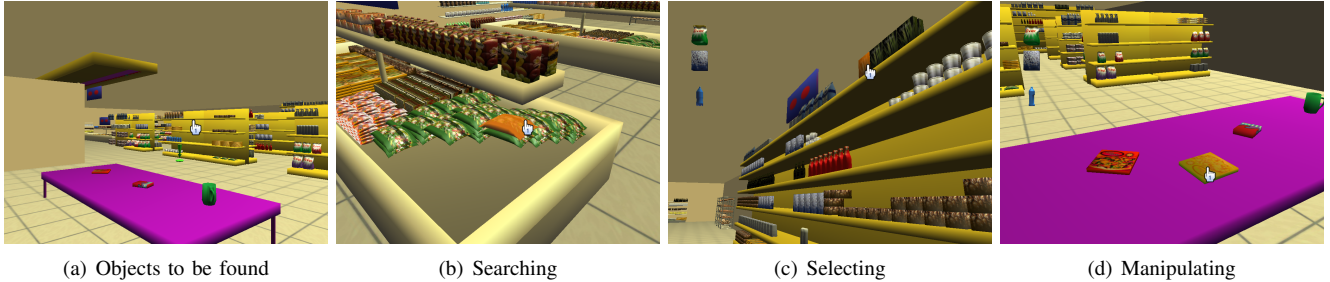


Figure 1. Four snapshots of the supermarket: (a) copies of the target objects on the purple table; (b) and (c) target objects being selected; (d) a selected object being reoriented to match the same orientation of the copy in the table. Notice, in the top-left corner of images (c) and (d), the list of products that the user is carrying with him.

Abstract—Many everyday tasks involve searching for something, selecting, and manipulating it with some cognitive purpose. Even if it is done in a quite automatic manner, due to our experience to deal with real objects, the mapping of these actions to a virtual world is not trivial.

In this paper we present a solution for this problem, which is composed of three parts: a semi-automatic navigation technique based on path-planning; a selection technique comprising the overexposure of nearby objects; and a manipulation technique based on natural gestures. These three techniques were combined in a smooth way to solve the searching, selecting and manipulating problem. A certain degree of parallelism between them was accepted and is handled by the system. We evaluate our results by user testing in a supermarket scenario. Subjects were invited to walk the hallways of a virtual supermarket looking in the shelves for specific products – sometimes hidden by other objects – that they need to pick up and put over a table in a pre-defined position and orientation. Results showed that our solution fulfill the needs of this kind of complex task, in a natural, funny and attractive way.

Keywords—3D interaction; virtual worlds;

I. INTRODUCTION

In the last decade, the popularization of graphics processing units (GPUs) and the emergence of high quality and low cost commodity devices (e.g. Nintendo Wii Remote®, PlayStation Move®, Microsoft Kinect®, and Apple iPhone®) that enable 3D interaction for everyone, motivated the implementation of fully interactive 3D applications.

Examples of this kind of applications may be found in different domains [1], as medicine, for instance. A mini-

mally invasive virtual surgery involves the navigation inside the body, the identification of the target organ, and the manipulation of the tissues (e.g. for suture). In the public safety and military field, the user is invited to walk in a virtual environment, find a bomb and defuse it, precisely manipulating small pieces. Regarding entertainment, thanks to the release of modern video games consoles, many game titles involving natural interaction can be found in the shops shelves.

Although the great part of interactive applications require complete and integrated solutions, research projects usually handle only one interaction technique such as selection, manipulation, or navigation, neglecting the integration with other techniques [2]. The composition of different interaction techniques in a single application is almost as difficult as to propose a new one. An application that require more than one interactive task certainly have dozens of possible solutions. But which technique is really effective to accomplish the task? Combining the best choices for each task that need to be supported may not result in the best solution, because the transition and coupling between the techniques may not be as smooth or natural as expected.

In this paper we propose a solution for a complete interactive 3D application that mimics problems people use to face in real life and that is being explored in industry for simulation software. The user should search for a specific object – sometimes total or partially hidden – in a virtual environment, get it, and put it in a specific position and orientation. We present the interaction techniques we chose

for navigation, selection and manipulation and how we combine them into a unique solution that allows smooth transition between them.

The remainder of this paper is structured as follows. Section II reviews some works on 3D interaction and the best solutions for every kind of task. In Section III we detail the problem that motivates this work. The four following sections present an overview of the proposed solution (Section IV), the semi-automatic navigation technique adopted (Section V), the selection technique with the overexposure of objects (Section VI), and the manipulation technique (Section VII). Section VIII describes the user tests and Section IX the results. Section X discusses the lessons learned and Section XI presents our conclusions and future works.

II. RELATED WORK

Users of immersive virtual environments usually intend to manipulate objects that are part of the scene. Reaching objects may also require navigation through the virtual environment since they might not be close to the user. Manipulation requires a previous selection of an object of interest, and navigation is also frequently based on selecting a target point to indicate the path that the user wants to follow.

Developing simple and comfortable selection and manipulation techniques for 3D environments has been a research issue for many years, and there are several possibilities depending on the application-specific tasks, different devices and interaction metaphors. At a high level, these techniques can be classified in two categories according to the interaction metaphor used: the exocentric and the egocentric metaphors [3]. In the exocentric metaphor, the proportions between the user and the objects are not maintained, assuming that the user interacts with the environment from outside of its reference system. This is known as *god's eye viewpoint*. In the egocentric metaphor, the user is part of the virtual world, maintaining the dimensional coherence between him/her and the objects being manipulated. This class is further subdivided into two metaphors: *virtual hand*, where the user reaches and grabs the object of interest with a virtual hand, and *virtual pointer*, where the user interacts with the target object by pointing at it.

With virtual hand techniques, the users can select and directly manipulate virtual objects by *touching* them with their own hands. These techniques are mainly implemented in two different ways: classical virtual hand technique [3], and Go-Go technique [4] that improves the simple virtual hand by allowing the user to interactively change the length of the virtual arm.

The interaction by pointing allows the selection and manipulation of objects located far from the user reaching area. One of the first pointing-based implementations where developed in the 80s by Bolt [5]. Then, many others

were proposed, differing in the shape of the pointer, the definition of the virtual pointer direction and the methods of disambiguating the object the user wants to select. The most common example of virtual pointer is the ray-casting technique [6] and its variations, like ray-casting with fishing reel [7], spotlight [8], and aperture selection [9].

Due to the difficulty of conceiving a single best technique for all possible scenarios, some hybrid techniques were proposed. Some of them are very popular as: HOMER (Hand Centered Object Manipulation Extending Ray casting) [7], scaled-world grab [8] and Voodoo Dolls [10].

Navigation in three-dimensional environments is a recurring problem and includes both *travel* and *wayfinding*. Travel corresponds to the motor component of navigation, or the control of the user's position and orientation in the virtual world. Wayfinding involves thinking, planning and choosing a path to follow from one point to another. In other words, represents the high-level part of the navigation task.

A large number of applications require some form of egocentric navigation through a simulated three-dimensional environment, either restricted to the ground – degenerating to an effectively planar situation – or involving free movement in the three-dimensional space.

Several devices and techniques have been tried to solve this problem, most of them presenting limitations. Some techniques require the combination of two or more devices to allow the number of degrees of freedom (DOF) needed for effective navigation, as in the use of the traditional mouse usually coupled with a keyboard for additional DOFs. Others techniques impose adaptation problems to potential users, like the three-dimensional mice, spaceballs, etc. Less common techniques, like the ones based on gesture and body motion, suffer from their own drawbacks, often requiring large and/or expensive setups [11], sometimes limiting the user's natural movements not associated with navigation, or being prone to cause user fatigue in long sessions [12].

Other navigation techniques require that the user physically walks through a real environment [13], [14]. These systems achieve a high sense of immersion by tracking the user position and orientation. A restriction is that they require a large available space so the user can walk freely.

A common solution for travel tasks relies on the continuous control of the orientation of the viewpoint and motion by the user. For example, the user may wear a head mounted display (HMD) where the orientation of his head is tracked [13]. In these cases, the user has the hands free to interact with the virtual environment, but the motion of the viewpoint is highly coupled to view direction in cases where the path is not pre-computed.

In automatic techniques, the user indicates where he/she wants to go. The system is responsible to find a path and moves the viewpoint automatically [15], [16]. Mackinlay et al. [15] proposed an interaction technique where the viewpoint position and speed is controlled based on the

user selected point of interest. Hachet et al. [16] proposed improvements by adding some widgets to the technique. In techniques based on sketches, the user draw a path on the environment, and the system controls the motion over that path. Hagedorn and Dollner [17] used a sketch based navigation approach in a touch sensitive display to explore a 3D model of a city.

In this section we addressed the main 3D interaction techniques. However, it is difficult to determine which technique is the best one. In the last few years, many experimental evaluations have been published and some guidelines are accepted nowadays [18]: the interaction technique and the device used should match; one may use pointing techniques for selection and virtual hand for manipulation tasks; when possible, reduce the number of DOFs; consider the use of both natural and magic techniques; use an appropriate combination of technique, display, and input devices; for navigation tasks, avoid teleportation; and consider using multimodal input.

Despite these recommendations, that certainly help the user choose the interactive technique and devices, the achievement of a good interface is not assured. Everytime a new technique, a new device or a new application is proposed, an empirical evaluation should be done to validate the choice.

III. THE SUPERMARKET PROBLEM

A supermarket intuitively illustrates the problem presented in Section I and is explored in this work. A person who goes to the supermarket for shopping is exposed to a large and complex environment full of objects (e.g. shelves, products, and so on). Her tasks involves: walk along corridors, with or without a specific destination, but always looking for something; reach products on shelves, putting it inside the shopping cart; and take the products off the shopping cart and pay for it, where shopping ends. Sometimes, people manipulate a specific product to evaluate if it will be bought or not. This sequence of actions is done repeatedly, for many different products, arranged in very different ways into distinct containers.

To guide the development of this work, we use the same specifications given by the organizers of the IEEE 3DUI (Symposium on 3D User Interfaces) interaction contest of 2010 [2]. The task is to find a set of three objects in a supermarket and move them to a table. Copies of the target objects were put on a purple table (see Figure 1(a)), and the objects itself were hidden behind other objects in the aisles marked with a pane with red dots (red dots can be seen in the Figures 1(b) and 1(c)). Users should be able to move along the corridors, find the object similar to the one on the table, and place it beside its copy in roughly the same orientation (Figure 1(d)).

The supermarket scene model comprises hundreds of distinct products, including milk bottles, soap packages,

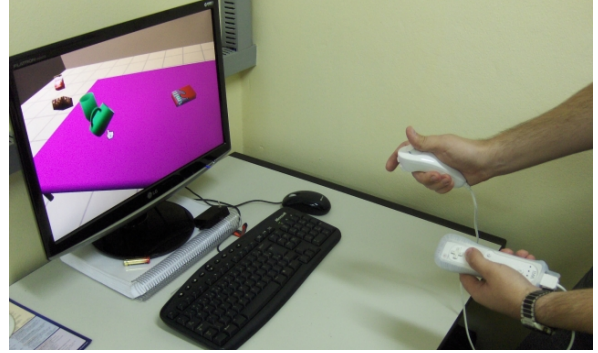


Figure 2. Using the Nintendo Wii Remote[®] and Nunchuck[®] to interact with the supermarket scenario. In this stage the Nunchuck[®] is being used to rotate the selected object so its orientation can match the one of the fixed copy on the table.

pizzas, cereal boxes, and so on. The products that should be found by users are like others in the shelves, but highlighted with a different color.

IV. OVERVIEW OF THE SOLUTION

The kind of action to be accomplished in a virtual environment like a supermarket involves three main tasks: navigation, selection, and manipulation. Since the environment is complex and full of objects, we decided to conceive a solution as simple as possible, that minimizes the use of buttons, and avoids any kind of menu to minimize the cognitive overload to the user.

Since some shelves of the virtual supermarket are full of products and the one we are looking for can be partially occluded by others, we decided to avoid techniques that introduce more visual elements to the scene, as the virtual hand. Likewise, the use of ray-casting-based selection of a volume, if not very well calibrated, can allow the selection of several objects at a time by mistake while trying to select the right object.

In a supermarket, it is common to walk through a corridor looking to the products on the shelves. Then, the user should be capable of navigate in one direction while looking to another one.

Finally, we also considered two additional points: (i) magic solutions are better than the ones that mimic reality [19], (ii) for the sake of efficiency and simplicity, the same input device should be used for navigation, selection and manipulation.

Therefore, our solution is based on the use of the Nintendo Wii Remote[®] and the Nunchuck[®], a spatially convenient device for 3D interaction [20]. The Wii Remote[®] is used as a pointing device – through the infrared (IR) sensors – to indicate objects or positions, allowing navigation and selection. Actions are performed when the Wii Remote[®] buttons are pressed. The manipulation task requires the use of the analog stick of the Nunchuck[®], as well as the

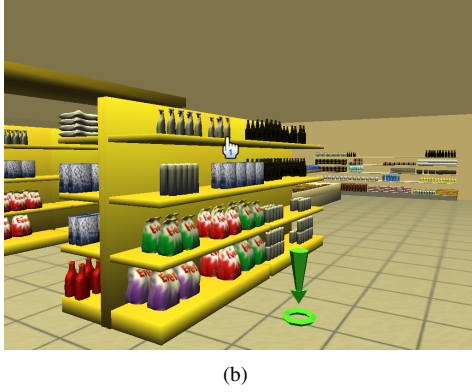
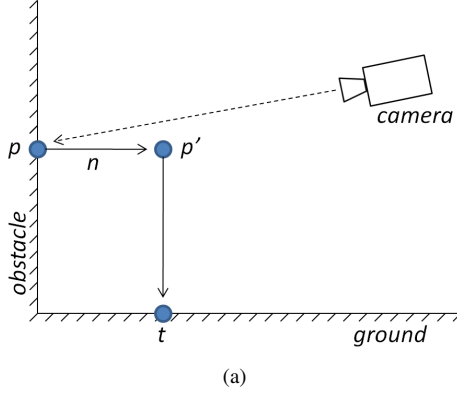


Figure 3. Computation of the target point for automatic navigation: calculus of the target point (a), and an example of a user pointing to a shelf and the resulting navigation target as a green arrow pointing to the ground (b).

accelerometers of the Wii Remote[®]. We implement versions of these interaction techniques based on keyboard and mouse input. Figure 2 shows a user in action with the Wii Remote[®] and Nunchuck[®].

We tried to avoid differentiating between the usage of the two main buttons – A and B – of the Wii Remote[®]. Some users tend to hold the controller as a TV remote (favoring the A button) while others handle it like a gun (favoring the B button). Both buttons perform the same action whenever possible. To confirm and cancel actions we chose the buttons A and B, respectively, as usually assigned in official Wii games. We believe the differentiation on these contexts is more efficient than relying on a menu-based interaction.

The intuition of the solution follows an egocentric metaphor and uses a first person approach, where the camera represents the user eyes. The user takes the Wii Remote[®] in hands and use it to point in the screen where he/she intend to move, confirming that position by pressing the A or B button. The camera is then smooth and automatically moved to the selected position. During the travel, the user has full control on the gaze.

If, instead of pointing to a new position, the user points to a product and press button A or B, this product will be selected, disappearing of its current position and being shown in the left side of the screen, in a kind of virtual shopping cart. If, however, the user is too far from the product he/she is pointing, it is not selected. Instead, the camera moves to a position near the product. In such a way, we are combining navigation and selection tasks in a single command that is sensitive to the context. This behavior avoids errors due to the small size of far objects.

When the user is near a shelf, some products move a little bit from its current place in a *magic* way. This movement, that we called *explosion of products*, helps the user to see products that are behind others.

Finally, to place a product over the table, the user must point to the table and press button A or B. The product

orientation is changed with the help of the Nunchuck[®] stick and rolling the Wii Remote[®].

Our solution was implemented in C++, using OGRE (Open Source 3D Graphics Engine) renderer [21]. We used Blender [22] for modeling.

V. NAVIGATION: SMOOTHLY MOVING THE VIEWER TO THE POINTING OBJECT

In this section, we will detail our proposal for navigation. Also we will overview the traditional FPS (First-Person-Shooter) navigation technique – used in many video games – which we will compare against our own technique.

We propose a point-and-click solution where the user controls a virtual pointer to show where he/she intends to go on the virtual environment. When the desired position is under the virtual pointer, the user must press a button confirming that target position. Then, the system uses an automatic navigation technique to move the camera to the new position.

We use the IR sensor of the Wii Remote[®] to know the screen position where the user is pointing to. The user controls a pointer that moves freely on the screen. When the pointer goes near the edges of the screen, the user is able to control the orientation of the camera. For example, if the cursor is next to the upper edge, the camera looks up, and if the cursor is next to the left edge, the camera turns to the left. Orientating the camera does not change its position in the virtual environment. In the case of using the mouse for pointing (instead of the Wii Remote[®]), the cursor remains always fixed on the center of the screen. Moving the mouse reorients the camera as in a First-Person-Shooter (FPS) game.

With the screen position of the pointer, the camera position and its orientation, the ray-cast pointing technique is used to compute the position p on the surface of the first 3D model in the scene behind the pointer, assuming perspective projection. We also calculate the normal n of that surface. With the position p and the direction n , a point t is computed

by creating an intermediate point p' at a certain distance d from the point p in the direction n , and then projecting orthogonally the point p' on the ground. The point t will serve as the target point for the navigation. Figure 3 shows how the target point t is computed and presents an example in our application.

We compute the target point this way to avoid that the user stops its navigation on the exact point where he is pointing to. In real life, when a person is in the supermarket and wants to buy a product, she goes to a position where the object is reachable within a distance of her arm. If the point p is directly projected on the ground to compute the target point t , the user gets so close to the point p that he will need to turn around and walk back some steps to see clearly the desired point. If the user points directly to the ground, the resulting target position t will be the same as the point p . In this case, the user can control the exact position where he wants to go.

The computation of the target point is done every time the cursor or the camera changes its position or orientation. As a feedback clue for the user, we draw a green arrow on the scene over the target point t . The Figure 3(b) shows a picture of this case, with the cursor controlled by the user and the final target point indicated by the green arrow.

Every time the user press the button A or B on the Wii Remote[®], the current target point t is used as a new target position to guide the camera movement. We use the potential field algorithm proposed by [23] to calculate a path from the current position to the target one, since it generates smooth motions for the camera.

Once a path is computed, the system starts moving the camera over it. The system does not control directly the speed of the camera. Instead, the system controls the acceleration, deceleration and maximum speed, so the camera does not start or stop its movement abruptly. During the camera movement, the system automatically controls the camera position over the computed path. The camera motion is constrained to the ground and its gaze remains free, so the user can look around.

In order to evaluate our approach, we implemented a second camera control system, similar to the ones found in FPS games. In this implementation, the analog stick of the Nunchuck[®] controls the camera position on the virtual environment. Push the analog stick forward makes the camera move forward, parallel to the ground, and pull back it makes the camera do the inverse movement. Push the analog stick to the left or right makes the camera strafe (i.e. walk sideways) to the left or to the right, respectively. Letting the analog stick go back to its original position makes the camera stop its motion.

Like in our technique, the orientation of the camera is controlled with the IR sensor of the Wii Remote[®]. A cursor moves freely on the screen, and when it is near one of the screen edges, the camera turns in that direction.

It is also possible to control the camera with a standard keyboard and mouse. In this case the camera is controlled the same way as it is in most FPS games. The *W* and *S* keys make the camera move forward or backwards, and the *A* and *D* keys make the camera strafe to the left or to the right, respectively. The arrow keys can be used in the same way, with the *up*, *left*, *down* and *right* arrow keys acting as the *W*, *A*, *S* and *D* keys, respectively. The virtual pointer is fixed in the center of the screen, and any motion in the mouse changes the orientation of the camera, i.e., moving the mouse to the right makes the camera also turn right.

VI. SELECTION

The same cursor used to navigate is also used for selection. A product of the supermarket can be selected by just pointing the cursor to it and pressing the A or B button on the Wii Remote[®]. The selected product is removed from the shelf and added to the selected products list, that appears at the left side of the screen. The first product selected is placed at the top-left corner of the screen. The second is placed right below the first one, and so on. Figure 1(d) shows a list of products already selected by the user.

The system lets the user pick a product only if it is close enough. We used a limit distance of 1.2 meters in our tests. This limitation is perceived by the user because we change the color of the products that are in this range slightly. This restriction in the selection range is used to avoid two problems: (i) the first one is to pick products that are too far from the current user position, increasing selection mistakes. In this case the product appears so small that is hard to the user to point exactly to it. (ii) the second problem does not concern the selection itself, but a conflict with navigation. At long distances, it is easy for a user to click in a product instead of a wall or shelf when trying to navigate through the market. So, when the user clicks a product at a distance greater than the threshold, the system interprets this action as a navigation request, instead of selection of that product.

There is another problem inherent of the supermarket scenario that rises in the selection stage. Products vary in shape and size, are placed over shelves of different designs and orientations, and are organized in layers. Thus, several items are occluded by the front most layer of products of the shelf. A customer may want to freely browse the products, even the ones in the back of the shelf, even if it has only the same kind of product. A simple example is to check the product's expiration date. It is a problem to select the desired item if it is in the back layer, or if it has a thin shape, like a pizza box.

To enhance the search for hidden products we defined a behavior for the items based on the idea of *explosion diagrams*. These diagrams are very common in manuals of assembling instructions. They depict the assembling of a complex object by representing the whole object as a set of

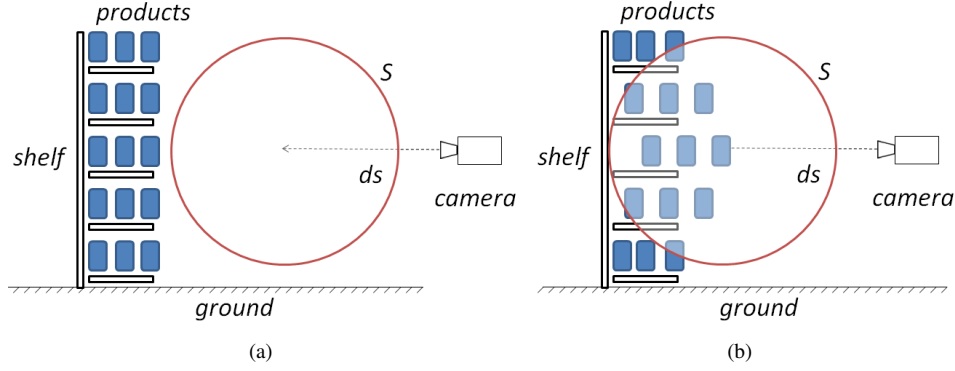


Figure 4. Diagram of the *explosion of products*. (a) Camera far from the products and nothing happens. (b) Camera closer to the products, activating the spreading of products. Observe in the image (b) how the products are moved from its original position (in the figure (a)) to a new position, related to the sphere s .

its minor components, which are slightly moved subcomponents from their original position in the opposite direction from the one that they are attached. This way, the diagram exposes hidden components and make easier for the user to understand how the whole system is composed. Recent works [24], [25] have addressed the automatic creation of such diagrams.

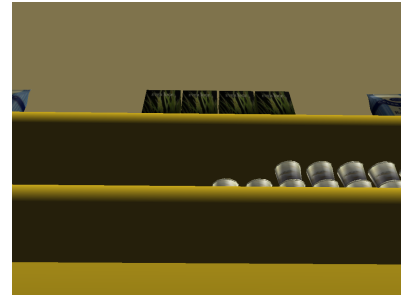
Like an explosion diagram, we move the products from the shelves to expose the hidden items, making the searching process easier. While several different schemes could be used to modify the layout of the items, our choice was to move them along the opposite direction from the one they would normally be placed in the shelf. This drawer-like behavior is quite intuitive and shows good results even when a high amount of items are moved.

First, we construct a sphere s centered in front of the camera, at a distance d_s . We put all products that are inside s in a set p_s . Then, for each product in p_s we compute a direction of displacement. This direction is related to the shelf where the product is placed, so it must be defined to each shelf from the start. That also implies that the items must be logically associated to their shelves. We move the products of p_s , in the displacement direction, with an offset proportional to the distance of the product to the center of s (see Figure 4).

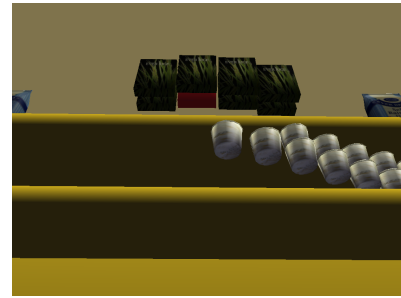
The animation of the position of the products of p_s results in a better distribution of the products on the screen: products that are in front of the shelves still can be easily selected, and products that were hidden by others are now partially visible and selectable. Since the modified layout of the products takes an *explosion shape* we named this process *explosion of products*. Figure 5 shows how some hidden products can be discovered with the *explosion of products* feature enabled.

VII. MANIPULATION

The user needs to point to the purple table and hold down the button A or B on the Wii Remote[®] to start the manipulation task. If the user points to a product that is



(a) *Explosion of products* disabled. Some products are hidden behind the front one.



(b) *Explosion of products* allows occluded products to be seen.

Figure 5. The effect of the *explosion of products* feature on the products position, improving the selection of the hidden ones.

already over the table, the manipulation is activated on that product. Otherwise, if the user points to an empty space over the table and a product has already been selected from the shelves, the first product from the selected products list (in the upper left side of the screen on the Figure 1(c)) is removed from that list and placed over the table, exactly on the position pointed by the user when the button has been pressed. Then, the manipulation is activated on this product. However, if the user presses the button pointing to an empty space over the table, and there is no products selected, the

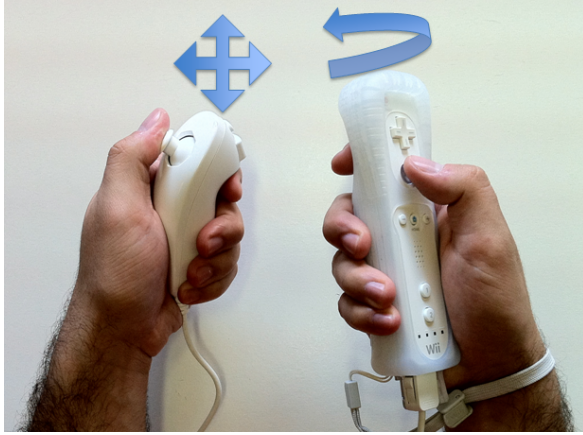


Figure 6. Gestures used to manipulate a product using the Wii Remote® and Nunchuck®.

action is ignored.

Note that the manipulation is enabled only while the user is pressing a button on the Wii Remote®. If the button is released, the system automatically goes back to the navigation/selection mode. In this case, if the user has released the button and is not yet satisfied with the orientation of the product, he/she can again point to the object, select it by pressing the button and manipulate it as he/she want, while the button holds down.

Keeping the button pressed, the orientation of the product can be modified using the Nunchuck® stick and the Wii Remote® accelerometers. The axes of the camera reference system were used as a basis for the product rotation: the camera up vector defines the Y axis of rotation, the direction of view is used as the Z axis, and the cross product between Y and Z is used as the X axis. When the user moves the analog stick of the Nunchuck® to the left-right, or up-down, the product rotates around its Y or X axis respectively. Rolling the Wii Remote® results in a rotation of the product around the Z axis. Figure 6 illustrates the gestures associated.

VIII. USER TESTS

An experiment was designed with the purpose of understanding the possible advantages of our complete solution. More specifically, we were interested in testing the efficiency, acceptance, comfort and adaptability of the proposed technique for applications that involve the selection of objects spread – sometimes partially occluded by others – in large environments. We also wanted to verify if the Wii Remote® really fits to the techniques proposed and to measure how efficient is its use if compared with the mouse.

A. Tasks and subjects

We designed three tasks with the objective of evaluate navigation and selection separately and then both together. The manipulation was not taken into account in these tests.

TASK 1 was conceived to evaluate the navigation system. The user was invited to navigate through the supermarket aisles to reach a series of waypoints (shown as rotating blue stars) positioned over the scene. Only a single waypoint is visible at each time. When the user reaches a waypoint it disappears, and the next waypoint appears. The new waypoint is visible from the position of the previous waypoint, but requires the user to turn around to find it. Users are aware of this fact. The positions of the waypoints are fixed for all tests and users. This test was executed once for each combination of input devices (Wii Remote® or mouse) and interaction technique (point-and-click or FPS). Each user repeated the test four times.

TASK 2 was proposed to test the impact of the use of the *explosion of products* in the selection and how it works with the Wii Remote® and mouse. The user starts the test in front of a shelf and has to select one specific product. The product distinguishes of the others by its color and is out of the user reach. The user needs first to orient himself to face the product, and then to walk a very small distance. Our reasoning is that, when selecting a product the user might also need to walk to achieve a better angle. So the ability to move while trying to select a product is relevant. When the user selects the first product he/she is moved to another shelf for another instance of the test. The test finishes when the user completes three selections. This task was executed twice for each input device (Wii Remote® or mouse), with and without the *explosion of products* feature, totalizing four tests.

TASK 3 is the full task and consists of navigating through the supermarkets shelves, finding and selecting three specific products, navigating back to the purple table, and placing the objects over the table. The test finishes when the three products are placed on the table (the user is aware of the locations of the products). This test was executed twice, once for each input device (Wii Remote® or mouse), with the use of the explosion animation.

The experiment was performed by a group of 38 subjects, 33 male and 5 female, aging from 19 to 37 years old (average = 22.8; standard deviation = 3.9), all of them undergraduate students in Computer Science. Each of them has done the three tasks with the different conditions previewed, resulting in 10 tests per user, and a total of 380 tests.

In a pre-test form, we made a set of questions to the users, asking them to characterize themselves in a scale from 1 to 5, meaning *very little experience* and *highly experienced*, respectively. Each question asked about the user experience with computer games, 3D environments, FPS games, and use of the Wii Remote®. We summarized these answers in the Table I, presenting the average and standard deviation.

B. Procedure and variables

Before the experiment starts, users were invited to fill a self characterization questionnaire, instructed on how to

Table I
CHARACTERIZATION OF THE SUBJECTS, ACCORDING TO THE ANSWERS
IN THE PRE-TEST QUESTIONNAIRE.

Question	Average	Std. dev.
Experience with computer games	4.10	0.83
Experience with 3D virtual environments	3.42	1.00
Experience with FPS games	3.78	1.04
Experience with Wii Remote®	2.02	1.26

operate the devices, and encouraged to perform as many practice trials as wished so that they could feel confident during the tests. Then the three sets of tasks are performed, in order.

For each task, the execution of its tests was randomized in order to try to prevent bias. From each one of the twelve runs we collected the execution times in a log file. The users also filled post-experiment surveys for qualitative analysis including the following data: which technique is more efficient for navigation (point-and-click with the Wii Remote®, point-and-click with the mouse, FPS with mouse and keyboard), which technique is more intuitive and easiest to learn for navigation (point-and-click with the Wii Remote®, point-and-click with the mouse, FPS with mouse and keyboard), which technique is more fun to navigate (point-and-click with the Wii Remote®, point-and-click with the mouse, FPS with mouse and keyboard), which technique is more efficient for selection (point-and-click with Wii Remote® or mouse), which technique is more intuitive and easiest to learn for selection (point-and-click with the Wii Remote®, point-and-click with the mouse, FPS with mouse and keyboard), and which selection technique is more fun (point-and-click with the Wii Remote®, point-and-click with the mouse, FPS with mouse and keyboard). We also offered a space for free comments and opinions.

IX. RESULTS

The tasks described in Section VIII-A and tested with the 38 subjects allowed the capture of objective and subjective data that are presented below. Completion times for the tasks were recorded in log files and used as input for ANOVA (Analysis of Variance) test.

Regarding navigation, we tested four different configurations and measured times for completion. Mean times and standard deviation for the four configurations tested were calculated and can be seen in Figure 7. The use of the mouse is always more efficient than the Wii Remote®, presenting a mean time significantly smaller ($F = 12.4140$; $p < 0.0005$). Comparing the performance with the two navigation techniques tested, we can see that the FPS technique is more efficient than the point-and-click, with a mean time significantly smaller ($F = 25.0174$; $p < 0.0000016$).

From the subjective data, that represents the opinion of the subjects, we can observe that 76% of the users found the use of mouse and keyboard is more intuitive than the

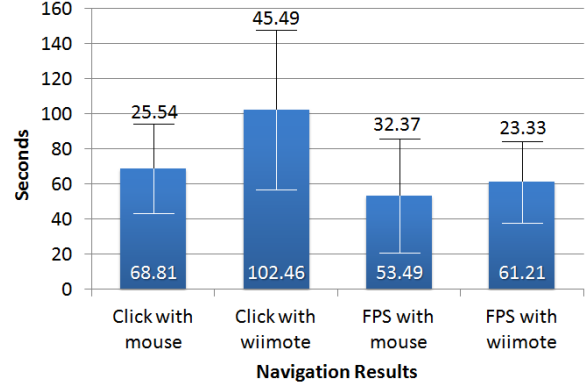


Figure 7. Navigation task performance in four different conditions: point-and-click technique with mouse; point-and-click with Wii Remote®; FPS with mouse and keyboard; and FPS with Wii Remote®. The numbers at the bottom of the bars represents the mean time, while the numbers at the top of the bars means the standard deviation.

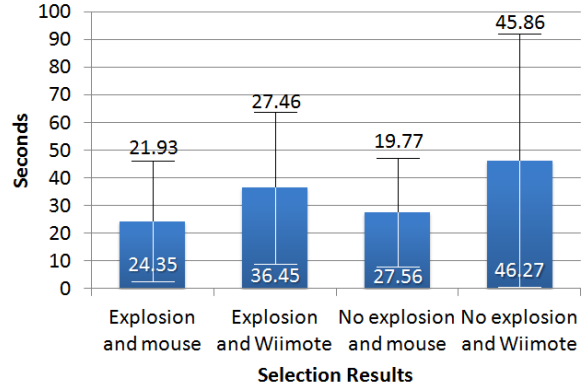


Figure 8. Selection task performance in four different conditions: using the *explosion of products* associated to the mouse or Wii Remote®; without the explosion effect and using the mouse or Wii Remote®. The numbers at the bottom of the bars represents the mean time, while the numbers at the top of the bars means the standard deviation.

Wii Remote®. For the other hand, 86% of these same users declared that the Wii Remote® is more fun than the mouse.

With respect of selection tests, mean times and standard deviation for the four configurations tested can be seen in Figure 8. The use of the *explosion of products* presents the best mean times when associated with the use of a mouse or Wii Remote®. However, these mean times have no statistic significance ($F = 1.6421$; $p < 0.202$).

In the last set of tests, subjects where asked to accomplish the TASK 3, as described in Section VIII-A. The mean times and standard deviation are shown in Figure 9. As it can be seen, the mouse and keyboard is more efficient than the Wii Remote®, presenting a mean time significantly smaller ($F = 8.7025$; $p < 0.0042$).

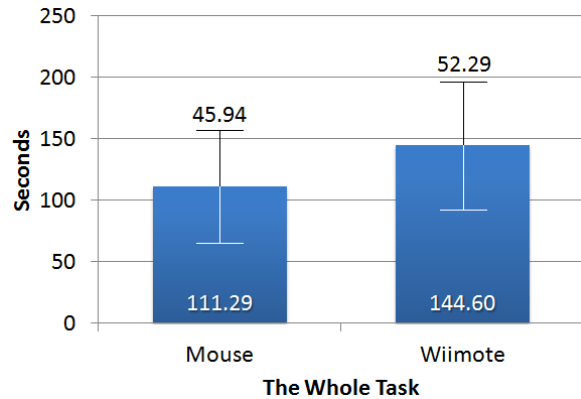


Figure 9. Mean performance achieved in the completion the whole task. The numbers at the bottom of the bars represents the mean time, while the numbers at the top of the bars means the standard deviation.

X. DISCUSSION

Some of the results obtained with the user tests were more or less expected by us. The higher performance of the users with the mouse, if compared with the Wii Remote®, is quite obvious. Mouse and keyboard are working tools as common as pen and paper for computer science students. On the other hand, 72% of the subjects declared to have no or few previous experience with the Wii Remote®. Considering this unfair comparison, we believe that the performance with the Wii Remote® is sufficiently good to consider the use of this device in adverse situation where, for example, the user cannot sit down in front of the computer and use the mouse and keyboard as comfortably as they use to do.

The best performance achieved with the FPS navigation technique, instead of the use of the point-and-click technique was a surprise for us. Observing the subjects during the tests, we noted that they clicked a lot when using the point-and-click technique. In fact, much more than expected. This behavior can be, in such a way, explained by the profile of the users; 62% of the users declared to be hard users of FPS games. We also think that the point-and-click technique is not well suited for the supermarket scenario, because of its characteristics (small aisles with high shelves). In an open-air scenario, where greater distances are seen, the user can take more advantage of the automatic definition of a valid path to follow.

The performance improvement achieved with the use of the *explosion of products* effect for selection were also expected. We could not reach significance with the ANOVA test, but we intend to do more tests with users, varying the position and size of the products to be selected, in order to understand these results.

As mentioned in Section VIII-A, in this study we did not evaluate the manipulation task. However, during the execution of the complete task (TASK 3), the subjects had also the opportunity of manipulate the products over the

purple table. Many users mentioned in the pos-test questionnaire that the Wii Remote® and the Nunchuck® were more comfortable than the mouse (that was considered very uncomfortable), but they also claimed that the Wii Remote® was not sufficiently accurate. In fact, the rotation given by the Wii Remote® accelerometers is not very precise at all, what probably motivated Nintendo to develop another device – the MotionPlus – based on gyroscopes. Unfortunately, we could not use it during the development of this work.

XI. CONCLUSION

In this paper we presented a complete and integrated solution to solve everyday interactive 3D tasks that involves navigation, selection and manipulation of objects that are, frequently, hidden by others. In our proposal, we tried to keep almost the same interactive devices and techniques for all the tasks, avoiding the exchange of devices and reducing the cognitive overload inherent to the interaction.

The results demonstrate that our approach is promising, but more work should be done in order to have a robust solution. As future work, we intend to reimplement the manipulation technique using more robust sensors, as the gyroscopes present in new commodity devices (e.g. iPhone, and Nintendo Motion Plus). New tests should also be done including manipulation and considering subjects with little experience in 3D games and less trained in the interactive techniques commonly used in this kind of application.

ACKNOWLEDGMENT

The authors would like to thank CNPq-Brazil through projects 483947/2010-5, 309092/2008-6 and 580156/2008-7 for partially supporting this work, as well as all the volunteers for their kind participation in the experiments. A special thanks to Leandro Tibola, Leonardo Santagada and Rafael Kovaleski for their help in the design and implementation of the first version of the software, and Vitor Pamplona and Marilena Maule for their comments on this text.

REFERENCES

- [1] A. Craig, W. R. Sherman, and J. D. Will, *Developing Virtual Reality Applications: Foundations of Effective Design*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2009.
- [2] P. Figueroa, Y. Kitamura, S. Kuntz, L. Vanacken, S. Maesen, T. D. Weyer, S. Notelaers, J. R. Octavia, A. Beznosyk, K. Coninx, F. Bacim, R. Kopper, A. Leal, T. Ni, and D. A. Bowman, "3dvi 2010 contest grand prize winners," *IEEE Computer Graphics and Applications*, vol. 30, pp. 86–96, c3, 2010.
- [3] I. Poupyrev, S. Weghorst, M. Billinghurst, and T. Ichikawa, "Egocentric object manipulation in virtual environments: Empirical evaluation of interaction techniques," in *Proceedings of the EUROGRAPHICS'98 Conference*. Eurographics and Blackwell Publishing Ltd., 1998, pp. 41–52.

- [4] I. Poupyrev, M. Billinghurst, S. Weghorst, and T. Ichikawa, "The go-go interaction technique: non-linear mapping for direct manipulation in vr," in *UIST '96: Proceedings of the 9th annual ACM symposium on User interface software and technology*. New York, NY, USA: ACM, 1996, pp. 79–80.
- [5] R. A. Bolt, "'put-that-there': Voice and gesture at the graphics interface," in *Proceedings of the 7th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '80. New York, NY, USA: ACM, 1980, pp. 262–270. [Online]. Available: <http://doi.acm.org/10.1145/800250.807503>
- [6] M. R. Mine, "Virtual environment interaction techniques," Chapel Hill, NC, USA, Tech. Rep., 1995.
- [7] D. A. Bowman and L. F. Hodges, "An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments," in *SI3D '97: Proceedings of the 1997 symposium on Interactive 3D graphics*. New York, NY, USA: ACM, 1997, pp. 35–38.
- [8] M. R. Mine, F. P. B. Jr., and C. H. Séquin, "Moving objects in space: exploiting proprioception in virtual-environment interaction," in *SIGGRAPH*, 1997, pp. 19–26.
- [9] A. Forsberg, K. Herndon, and R. Zeleznik, "Aperture based selection for immersive virtual environments," in *UIST '96: Proceedings of the 9th annual ACM symposium on User interface software and technology*. New York, NY, USA: ACM, 1996, pp. 95–96.
- [10] J. S. Pierce, B. C. Stearns, and R. Pausch, "Voodoo dolls: seamless interaction at multiple scales in virtual environments," in *I3D '99: Proceedings of the 1999 symposium on Interactive 3D graphics*. New York, NY, USA: ACM, 1999, pp. 141–145.
- [11] R. P. Darken, W. R. Cockayne, and D. Carmein, "The omni-directional treadmill: a locomotion device for virtual worlds," in *UIST '97: Proceedings of the 10th annual ACM symposium on User interface software and technology*. New York, NY, USA: ACM, 1997, pp. 213–221.
- [12] F. Sparacino, C. Wren, A. Azarbajani, and A. Pentland, "Browsing 3-d spaces with 3-d vision: body-driven navigation through the internet city," in *International Symposium on 3D Data Processing Visualization and Transmission*. Los Alamitos, CA, USA: IEEE Computer Society, 2002, pp. 224–233.
- [13] G. Bruder, F. Steinicke, and K. H. Hinrichs, "Arch-explore: A natural user interface for immersive architectural walkthroughs," *3D User Interfaces*, vol. 0, pp. 75–82, 2009.
- [14] E. Foxlin, "Pedestrian tracking with shoe-mounted inertial sensors," *IEEE Computer Graphics and Applications*, vol. 25, pp. 38–46, 2005.
- [15] J. D. Mackinlay, S. K. Card, and G. G. Robertson, "Rapid controlled movement through a virtual 3d workspace," in *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1990, pp. 171–176.
- [16] M. Hachet, F. Dècle, S. Knödel, and P. Guitton, "Navidget for easy 3d camera positioning from 2d inputs," in *Proceedings of IEEE 3DUI - Symposium on 3D User Interfaces*, 2008, best paper award. [Online]. Available: <http://iparla.labri.fr/publications/2008/HDKG08>
- [17] B. Hagedorn and J. Döllner, "Sketch-based navigation in 3d virtual environments," in *Smart Graphics*, ser. Lecture Notes in Computer Science, A. Butz, B. Fisher, A. Krüger, P. Olivier, and M. Christie, Eds., vol. 5166. Springer Berlin / Heidelberg, 2008, pp. 239–246. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-85412-8_23
- [18] D. A. Bowman, E. Kruijff, J. J. Laviola, and I. Poupyrev, "An introduction to 3-d user interface design," in *Presence: Teleoperators and Virtual Environments*, 2001, pp. 96–108.
- [19] D. A. Bowman, E. Kruijff, J. J. LaViola, and I. Poupyrev, *3D User Interfaces: Theory and Practice*. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc., 2004.
- [20] C. A. Wingrave, B. Williamson, P. D. Varcholik, J. Rose, A. Miller, E. Charbonneau, J. Bott, and J. J. L. Jr., "The wiimote and beyond: Spatially convenient devices for 3d user interfaces," *IEEE Computer Graphics and Applications*, vol. 30, pp. 71–85, 2010.
- [21] G. Junker, *Pro OGRE 3D Programming (Pro)*. Berkely, CA, USA: Apress, 2006.
- [22] R. Hess, *The Essential Blender: Guide to 3D Creation with the Open Source Suite Blender*. San Francisco, CA, USA: No Starch Press, 2007.
- [23] R. Silveira, F. Dapper, E. Prestes, and L. Nedel, "Natural steering behaviors for virtual pedestrians," *Visual Comput.*, 2009.
- [24] M. Tatzgern, D. Kalkofen, and D. Schmalstieg, "Compact explosion diagrams," in *NPAR '10: Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering*. New York, NY, USA: ACM, 2010, pp. 17–26.
- [25] D. Kalkofen, M. Tatzgern, and D. Schmalstieg, "Explosion diagrams for augmented reality," in *Proceedings of the IEEE Virtual Reality Conference (VR 09)*. IEEE Computer Society, 2009, pp. 71–78.