Design and Evaluation of a Gesture-Controlled System for Interactive Manipulation of Medical Images and 3D Models

Édimo Sousa Silva, Maria Andréia Formico Rodrigues
Programa de Pós-Graduação em Informática Aplicada (PPGIA)
Universidade de Fortaleza (UNIFOR)
60811-905 Fortaleza–CE Brasil
{edimossilva, andreia.formico}@gmail.com

Abstract— This work presents the design and evaluation of a gesture-controlled system for interactive manipulation of radiological images and 3D models using the *Kinect* device. Several abstractions have been implemented and refactored to improve the system performance, making the application simpler, at an affordable cost. Additionally, specific gestures to change the visualization settings of 3D models represented by layers were also successfully modeled. Further, we have conducted systematic and detailed usability testings with users to determine quantitative performance measures and qualitative analysis (usefulness, visual quality of the interface, ease of learning, ease of use, 3D spatial perception, level of interactivity, mental and physical fatigue, effectiveness and satisfaction). The results show that the participants are able to perform the tasks of search, selection and manipulation of 2D images (zoom in/out and translations) and 3D models (zoom in/out and rotations), quickly and accurately, demonstrating the usefulness of the system as a possible effective and competitive alternative solution, to the traditional use of the negatoscope.

Keywords – Gesture-Controlled System; Interaction; Manipulation; Evaluation; Images; 3D Models.

I. Introduction

Interactive gesture controlled systems have gained nowadays greater visibility. This was essentially motivated by the evolution of gesture recognition devices, along with the improvement of techniques in Computer Vision. More specifically, the 3D accelerometers currently capture data more accurately, the reflective markers overcame most of the problems with object occlusion and the depth sensors have become increasingly precise [15].

Recently, gesture control devices such as the Microsoft *Kinect* [28] has enabled the development of applications involving gestures at a much lower cost, while offering enough precision (or resolution), when compared to previously existing devices, such as stereoscopic camera pairs (and arrays) and magnetic trackers. In addition, the *Kinect* is a small and easy-to-setup device which suffers less from problems induced by variations in illumination. Moreover, the tracking stage abstraction of the member of interest to perform the gesture is a quite complex task and very often time consuming. In this context, the *Kinect* shows great flexibility for gesture identification, due to its robustness in tracking the major joints of the human body.

Despite the benefits already achieved with the use of modern input devices, the gesture identification is still a challenging area: the same gesture can be performed in different time intervals and may be represented in various ways. Other problems related to joint occlusion (resulting in poor tracking quality), robust hand gesture recognition and skeleton tracking are also important issues. Basically, these issues are associated with manipulation gestures (open trajectories, often requiring continuous feedback, with challenges related to robust tracking of a feature, *e.g.* the user's hand) and command gestures (predefined trajectories, with challenges related to efficient pattern matching).

The gesture identification begins by tracking a certain segment of the human body skeleton, followed by the processing and treatment of the data obtained by tracking. This procedure is usually performed by *Support Vector Machines* (SVM) [1], together with the training of neural networks to identify gestural patterns [7], [9], [17], [33], [3]. However, the use of a SVM is part of a complex process, with low learning curve.

There are different gesture-controlled systems targeting various application areas: digital games [27], physical therapy [35], surgical procedures [42], among others. Recently, a special attention has been devoted to the development of graphical tools for aiding medical procedures.

Generally, the doctor plans in advance all the steps of a particular surgery, including the analysis of patient medical data such as medical records, clinical photographs, medical images (CT scans, MRIs, X-ray films, etc.). Thus, it is a routine procedure to ask patients to bring their medical images on the day of surgery. Such images are generally exposed on a negatoscope (Figure 1), a traditional medical device, similar to a light box, whose handling demands important

care, particularly with respect to maintain an aseptic surgical environment. Considering the fact that these medical images are queried, searched and exchanged several times in the negatoscope (by the doctor or an assistant) to assist the surgeon in the decision making process during surgery, the asepsis in the operating room can be broken due to direct and accidental contact of the sterile gloves with the negatoscope or with the computer mouse/keyboard, putting at serious risk the patient's life. Also, another downside is the surgeon having to move away from the patient to approach the negatoscope, every time the query and manipulation of medical images of the patient is necessary. In practice, the doctor often asks another member of the medical team to handle the images in the negatoscope, so they can be viewed while the surgical procedure is performed. However, it is possible that the medical assistant does not demonstrate the level of precision and agility desired by the surgeon. Besides, this procedure does not prevent the displacement of the medical professional to the negatoscope. There are also times where there may be no other member of staff available, being the sole responsibility of the surgeon to access the images, necessarily requiring the exchange of gloves, followed by a sterilization procedure for cleaning the hands and arms (washing them with sufficient antimicrobial soap to ensure asepsis), after each use of the negatoscope.

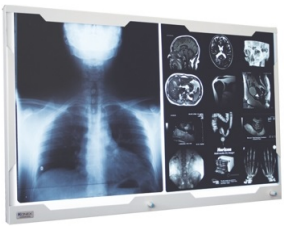


Fig. 1: A negatoscope used for viewing X-ray films.

These routine activities of the surgeon can be significantly optimized with the use of gesture-controlled systems, as an alternative less costly, faster and safer [34], compared to traditional methods. It also offers the advantage of the physician to remain beside the patient throughout the surgery. Currently, there are several studies showing the importance of surgeons directly control the radiological images so they can familiarize themselves and act more quickly and accurately, according to the surgery plan designed to the patient [19].

This work is an extended version of [41]. It presents the design and evaluation of a gesture-controlled system for interactive manipulation of radiological images and 3D models using the *Kinect* device. Several abstractions have been implemented and refactored using the *OpenGL* library to improve the system performance (interactive rates 3 times faster were now achieved), making the application simpler, at an affordable cost. Additionally, gestures (movements of arms and hands) with specific meanings to change the visualization settings of 3D models represented by layers (opaque, transparent or invisible) and to manipulate radiological images in the operating theatres were also successfully modeled. Moreover, this version includes systematic and detailed usability testings with three different groups of users, including a specialist.

The rest of the paper is organized as follows. The next section presents some recent initiatives in generating gesture-controlled applications and systems. Section III presents an overview of our system, including the approaches developed in our research for interactive manipulation of medical images and 3D models. Section IV details the usability testing studies we have conducted with users and the resulting comparative analyses. Section V presents and discusses the results. The last section concludes with future directions of work.

II. Related Work

This section presents some of the main works related to the development of gesture-controlled applications and systems to support medical procedures in general.

Despite lots of previous work, traditional vision-based hand gesture recognition methods are still far from satisfactory for real-life applications [36]. It is mainly due to the ambient lighting dependence that computer vision algorithms still have to deal with. This problem was particularly practically solved with the use of infrared cameras (such as the Kinect's one), which do not depend on the lighting and even offer facilities to image segmentation due to their depth sensor.

Gestural identification has been reported by many authors as a complex task due to the discontinuity of the captured coordinates, emphasizing that a small deviation in the gesture makes it considerably more difficult to be identified [24].

There are several gestural identification approaches based on depth sensors, by tracking human joints [23], [40], [4]. In general, the gestural feedback has still been little explored. Basically, it displays the captured gesture, helping users to adjust their movements with greater accuracy to generate the gesture of interest. Some possibilities of continuous feedback for gesture recognition are described in [21]. Ideally, feedback should be done in real time, but the authors identified the presence of tilt and noise in the captured images, hindering the recognition process. In order to minimize this problem, a filter is applied to normalize the gestures captured before displaying them on the screen. Alternative techniques use the gesture subdivision into several sub-gestures, sorted in ascending order of complexity [26]. If there is a similarity between the captured and the stored sub-gestures, they are considered as identified. In [46], a *B-spline* curve is used to indicate similarity in the gestures captured over time.

Thanks to the recent development of the inexpensive *Kinect* sensor, an interface to minimize the time spent in the medical appointment is described in [6]. The authors propose a system for projecting images onto a touch-sensitive table, in conjunction with the depth information captured by the *Kinect*. A gesture database is generated to identify which gestures are similar. Changes in thresholds can be made graphically to eliminate false positives.

The *Kinect* is also being used as an input device for handling medical imaging systems, such as the *OsiriX* [37] and *MITO* [22]. This is done through the implementation of software layers, especially developed for *OsiriX* [12] and for

MITO [14], which are responsible for making these systems accessible from the *Kinect*. Although our system has similar goals to these works, unlike these, it was developed natively to be manipulated via gestural control. Therefore, as the identification of gestures lies at its core, there is no need of an extra layer to realize the communication between the interpretation and translation of gestures to native system commands.

An interesting study was conducted to verify the feasibility of the existing solutions based on gestural control for the manipulation of images [19]. The medical team members are asked about the implications of using this technology in their routine activities and to produce a survey of the main activities that would benefit from the use of gestural control devices.

Several works use a wide range of more elaborate gestures for manipulation of features in medical systems [13], [12], [14], [38], [43], particularly using fingers combined with hand motion oscillatory gestures around different axes (*e.g*, simulating the rotational movement of a car steering wheel with both hands, etc.).

However, few studies explore aspects of usability using the *Kinect*. An example is a system for manipulating 3D molecules, using various options of input devices [39]. A comparative study is conducted, showing that users have the same level of comfort using those devices, but different levels of performance (*e.g.*, users using the traditional mouse & keyboard combination show a performance four times faster than those using the *Kinect* device alone). Not least, it is worth mentioning that little affinity with non-conventional input devices can generate false positive results, since it has been observed that after practicing with a new input device, the user performance improves significantly.

Finally, the implementation of an optimized number of features to control such systems using distinctly different gestures, with high accuracy rate in recognition, still remains challenging. Besides, when designing interfaces for gesture-based control applications it is necessary to decide whether the gestures are intuitive or not. Intuitive gestures generate a rapid learning curve [20]. An example of a gestural control interface based on intuitive gestures (based on the daily life movements of a person, such as walking, etc.) is described in [5] to manipulate the *Google Earth app*. Basic factors such as muscle strength, agility, dexterity and the space necessary to perform the gestures were taken into consideration during the design phase. The authors suggest that a good practice when activating different functionalities is to avoid repeating the same gestures, however, this in turn, increases the amount of movements that the user needs to memorize. Other systems [42] make use of simple gestures (based on hand gestures oscillatory motion along one axis), rather than intuitive gestures. Ideally, the chosen gestures should not induce hands contact with other parts of the body since this problem can have a negative impact on the gesture recognition process [45].

III. System Overview

The following is a general overview of the gesture-controlled system we have designed, implemented and evaluated for interactive manipulation of medical images and 3D models.

A. Tools and Technologies

In this work, we chose the C# as the programming language due to its high compatibility with the available libraries for the *Kinect* device. We used the *Kinect for Windows SDK v1.8* and the *Kinect for Windows Developer Toolkit v1.8* [30], in order to access the three major physical *Kinect* components (Figure 2) and its main features [18]. More specifically, the first component is the *RGB camera* with a resolution of 640x480 pixels, at a frequency of 30 frames per second, which is mainly used to capture colors. The second and main component is the *3D depth sensor*, responsible for identifying the user, which consists of an emitter of infrared rays with a monochrome sensor. This allows the depth sensor to perform properly in environments without any lighting. The third component is the *multi-microphone vector*, formed by a set of four microphones that are steerable to users, partially eliminating noises that are not produced by them.

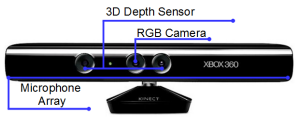


Fig. 2: *Kinect* and its three main physical components.

The main functionality of the *Kinect* is tracking the joints of the human skeleton. An integration with the *Kinect Toolkit 1.8* is required to access this information. In total, 20 joints (Figure 3) are identified and mapped spatially in real time by the device's depth sensor. It is important to emphasize that the *Kinect* does not identify gestures natively, but only through the detection of the major joints of the user, in this work, in accordance with the recognition algorithms described in [41].

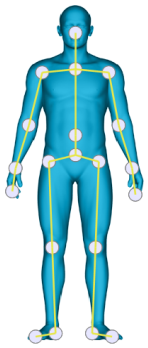


Fig. 3: Mapping of the major joints of the human skeleton.

The *Windows Presentation Foundation (WPF)* standard [29] consists of a number of useful libraries. It is up-to-date and compatible with various types of applications (*Windows Forms, GDI+, Direct3D*, etc.). We decided to use this pattern due to three main reasons: (1) it has full compatibility with

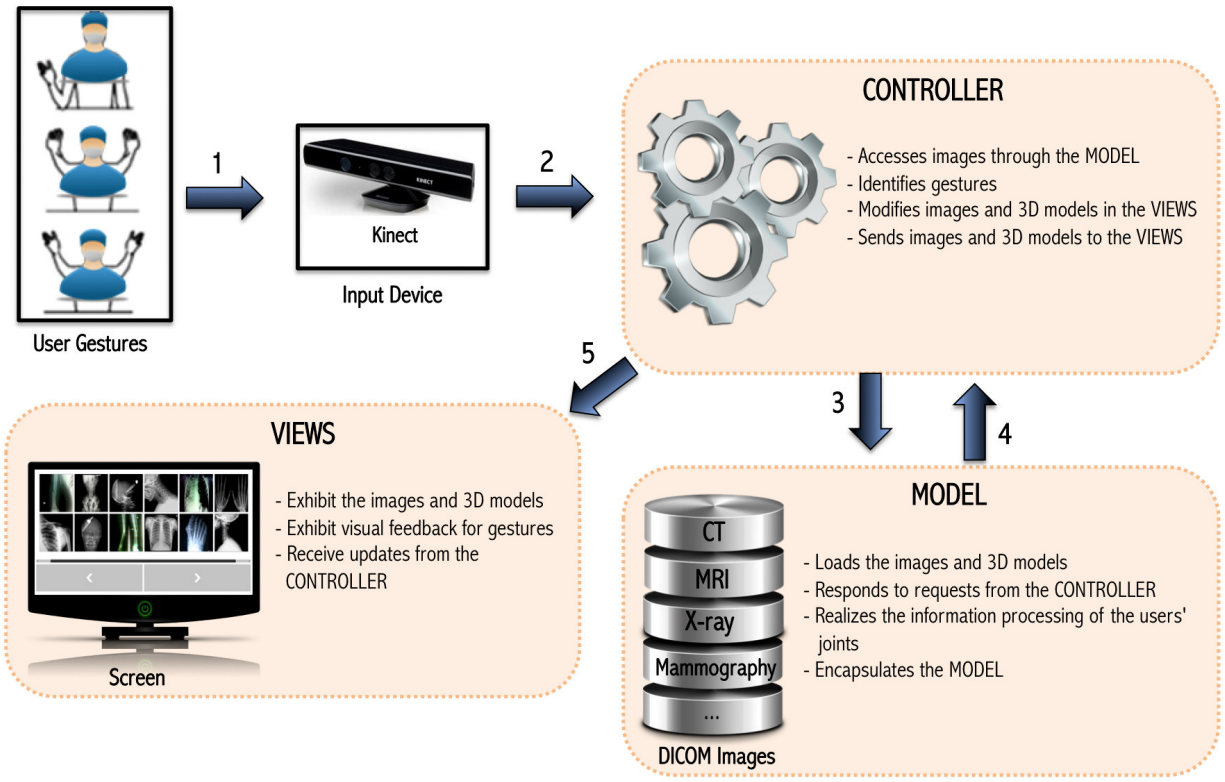


Fig. 4: Architecture of the gesture-controlled system. The user performs a gesture that is identified by the camera of the *Kinect* input device. Through the CONTROLLER, the VIEWS of the MODEL are modified according to the user selected functionalities.

the library created and with the SDK, both used together with the *Kinect* device; (2) there is a vector rendering engine at its core, able to access the graphics hardware resources; and (3) it contains appropriate libraries for the development of 2D and 3D applications.

Additionally, several abstractions have been implemented and refactored using the *OpenGL* library to improve the system performance (interactive rates 3 times faster were now achieved when compared with the results presented in [41]) and to make the whole gesture-controlled process simpler. *OpenGL* was also used to generate the rendering and shading of the 3D models, as well as to set the visualization attributes (transparent, opaque or invisible) of each of the three layers (skin, bones and soft tissues) which make up the 3D model. We chose to use the *Flat Shading* model combined with the ambient light for shading the 3D models, so as to ensure a reasonable tradeoff between performance and realism. Since the surfaces of the 3D models have detailed geometry and to solve the problem of deciding which elements of the rendered model are visible and which are hidden, we opted to use a *z-buffer* to make the *z-culling*. Moreover, we have also used a free software, the *InVesalius* [11], as a converter to extract radiological images in DICOM file format (a standard used worldwide by radiologists) [8], as well as to export these images to the formats *jpeg* and *Wavefront* [44].

B. System Architecture

Our system architecture is based on the principle of separation of data, presentation, and interaction mechanisms, using the Model-View-Controller (MVC) architectural pattern [25]. The MVC pattern is used to break the application into three main parts: the MODEL, the VIEWS, and the CONTROLLER (see Figure 4). In this variation of the MVC pattern, the MODEL is passive (it does not notify the VIEWS, but the CONTROLLER does). The advantage of this is the increased decoupling between the MODEL and the VIEWS. On the other hand, it increases the coupling between the CONTROLLER and the VIEWS, common in rich client applications, as is the case of our system. So, this pattern was chosen because of its simplicity and flexibility to meet the application requirements, keeping it cohesive and without strong coupling.

The flow presented in Figure 4 starts with gestures made by the user, which are captured by the *kinect* (step 1) and identified by the CONTROLLER (step 2). The MODEL manages the data and behaviour of the 2D and 3D graphical objects that are part of the application domain, for example, loading images and 3D models and processing the information from user's joints; responds to queries about its state (from the CONTROLLER); reacts to instructions to change its state (from the CONTROLLER); and encapsulates the state of the system (steps 3 and 4). The MODEL is the static part of the application since it accesses and encapsulates the data that will be handled by the CONTROLLER and displayed by the VIEWS.

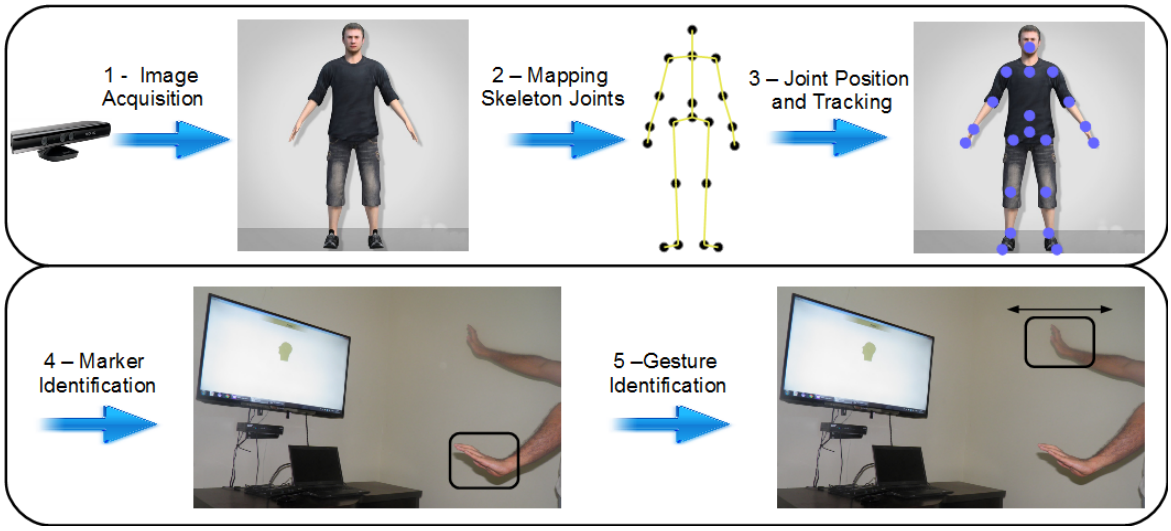


Fig. 5: Gesture recognition process.

The CONTROLLER in turn is the core of the application, responsible for communication between the MODEL (steps 3 and 4) and VIEWS (step 5). It interprets the user gesture inputs from the *Kinect* device and maps these user actions into commands that are passed to the MODEL (step 3) to effect the appropriate change (step 4). In fact, the CONTROLLER is the means by which the user interacts with the application. For example, if the user performs specific gestures (*e.g.*, lifts a hand and pushes it forward to issue a selection command like "select this X-ray film" or positions both arms straight, perpendicular to the trunk, pointing forward to issue a selection command like "change the visibility settings of the skin layer of the 3D Model to transparent"), the CONTROLLER is responsible for translating these actions into invocations of the MODEL's operations. More specifically, the CONTROLLER sets the behavior of the application, accesses images through the MODEL, identifies gestures, alters the images and 3D models to display them in the VIEWS, and sends these visual data to the VIEWS (step 5).

The VIEWS receive constant updates from the CONTROLLER and are responsible for displaying images and 3D models handled by the CONTROLLER. Both components mantain asynchronous contact so that all gestures can be captured in real time and displayed in the VIEWS. In addition, the VIEWS also exhibit a gesture feedback, which assists the user in using the system, *e.g*, by displaying the name of the selected functionality on the top of the screen, currently being activated through a specific gesture performed by the user.

C. Gesture Recognition

Initially, as shown in Figure 5, we acquire the gesture images of the user (step 1), then we perform the mapping of the skeleton joints (step 2) and finally we track the joints (step 3). We have also designed a simple and effective mapping of 2D regions of the human body to optimize the process of gesture recognition, through which we identify the positions of the hands of the user (step 4) and the variation of their positions along an axis (step 5).

In our implementation, regions correspond to abstractions based on 3D mapping of the joints of the user. A 2D projection is calculated, in which the relative coordinates to the axis z are neglected. We define nine regions of interest (Figure 6) which are composed of two identifiers: horizontal and vertical. The horizontal identifier is the region's position on the horizontal axis (L - Left, R - Right, C - Center) and the vertical one represents the region's position on the vertical axis (T - Top, M - Middle, B - Bottom). The joints of both shoulders and the extremities of the hip joints were used as boundary points of the region for using the markers. The identification of a particular region consists of locating the region in which one of the user hands is positioned (or both of them), which is used as a trigger to activate some specific functionality. For example, during interactive manipulation of 2D images, when the user's left hand occupies the region LT (Left - Top), the functionality trigger *zoom-in* and *zoom-out* will be activated.

The variation of the hand along an axis is modeled through the observation of its displacement along a given axis (x, y, z). This analysis can have three returns: *still, increasing the distance*, and *decreasing the distance*. For example, while manipulating 2D images, after shooting the trigger for the *zoom-in* and *zoom-out*, if the variation along an axis has a return value *still*, nothing occurs; however, if the return value is *increasing the distance* or *decreasing the distance*, the *zoom* is incremented and decremented, respectively.

It is worth noting that whenever the *Kinect* device is used, firstly it is necessary to perform some tasks to ensure the correct use of the sensor information. Some examples of tasks include sensor identification, capturing the stream with skeleton information, updating of sensor data, etc. The library we have developed, automatically performs all these tasks. Besides these features, the main one is the identification of gestures quickly and precisely. For example, using only two

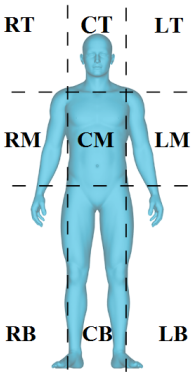
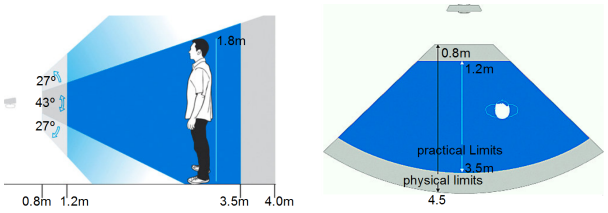


Fig. 6: Regions of interest: RT = {Right, Top}, CT = {Center, Top}, LT = {Left, Top}, RM = {Right, Middle}, CM = {Center, Middle}, LM = {Left, Middle}, RB = {Right, Bottom}, CB = {Center, Bottom}, LB = {Left, Bottom}.

commands we can identify a simple gesture: one command is used to identify the region in which the joint of interest is positioned, and another one to check whether this joint is moving along an axis. On the other hand, the functionalities implemented to identify gestures exhibit a strong coupling to the *WPF* libraries, *Kinect Developer Toolkit v1.8* and *Kinect SDK v1.8*. However, when using the *Kinect* in conjunction with C#, it is common to adopt the previously mentioned libraries, considering that they were created by the *Kinect* developers and thus, have full compatibility.

Another important aspect for correct gesture recognition is the *Kinect* field of view (FOV). In default range mode, *Kinect* can "see" people standing between 0.8 and 4.0 meters away; users will have to be able to use their arms at that distance, suggesting a practical range of 1.2 to 3.5 meters [31]. In addition, large objects located between the *Kinect* sensor and the user should not exist because they can cause partial or total occlusion of the latter. The *Kinect* has a vertical and horizontal FOV of approximately 43° and 57°, respectively (Figure 7).



(a) Vertical FOV. (b) Horizontal FOV.

Fig. 7: In (a) and (b), the *Kinect* vertical and horizontal FOV in default range, respectively [31].

D. Gesture Interface and System's Functionalities

In general terms, when interacting with the system through the *Kinect*, the user must perform a set of specific gestures to control the buttons displayed on the interface. These gestures were chosen targeting three main aspects: asepsis (both hands do not come into contact with any part of the user's body or with objects), simplicity (mostly, they are modeled as an oscillation of one hand along an axis) and simplicity in the recognition process (as previously described in Subsection III-C). For example, in order to control the cursor, the user must lift a hand. When the cursor is positioned over the button of interest displayed on the screen, the user must push that same hand forward. Then, when the application starts, the user should select which tasks will be conducted through the menu option.

All the functionalities were defined after a comparative study we conducted of DICOM (*The Digital Imaging and Communications in Medicine*, a highly recommended standard by radiologists) image manipulation systems [37], [11], [22]. Moreover, the tasks for manipulation of 3D models and the visualization configuration were designed with the aim of providing more detailed views of 3D layered models, cutting away layers of anatomy for a deeper vision of the human body. Besides, all these tasks performed by the user were also presented to a senior consultant surgeon who described them as the most useful ones in the operating room, since many of them are still the most prevalent in use.

A complete description of the data flow diagram of the system's functionalities is shown in Figure 8. In particular, two main display functions are available in the menu through the options: *2D Images* and *3D Models*. Alternatively, the user can also quit the application by selecting the *Exit* button (Figure 9). The option of viewing *2D images* displays a list with several thumbnail medical images on the application screen. Through the *Selection Procedure* option, the user can slide through an image series and choose the target image. As generated by the radiology equipment, the selected image is shown in its original size and resolution. The user can perform gestures for *zooming in*, *zooming out* and translating the image along the y and x axes, as shown in (a), (b), (c), (d), (e), (f) and (g) of Figure 11, respectively, and in the left side of Figure 8.

The standard library *Windows Presentation Foundation WPF* [32] is used to resize the images, as it presents: (1) at its core, an independent mechanism for resolution and vector rendering option, which is able to use the graphics hardware; (2) full compatibility with the library developed in this work for the *Kinect*; and (3) appropriate libraries for developing 3D applications. Furthermore, the *WPF* is compatible with all types of development environments (*microarray Windows form, GDI+, Direct3D, etc.*) [29], previously used by C#, also enabling the accomplishment of resizing for more (*zoom-in*) and for less (*zoom-out*), and image repositioning (translations).

In the visualization module of 3D models, only one object is displayed at a time. Through the gestures identification displayed from (a) to (h) of Figure 12, rotations around the x and y coordinate axes and *zoom in* and *zoom out* movements can be made, as shown in the right side of Figure 8. To this end, the *System.Windows.Media.Media3D* graphics library was initially used due to its ease of use, possibility of abstractions, and because it is part of the standard libraries of

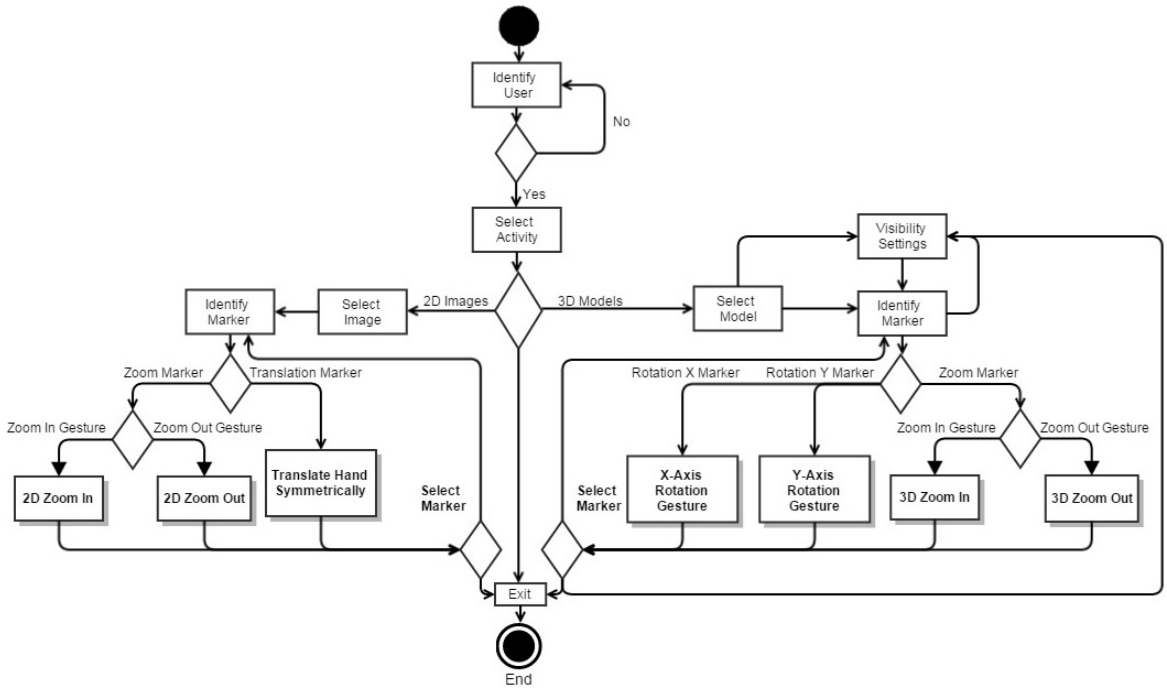


Fig. 8: Data flow diagram of the functionalities of the system.

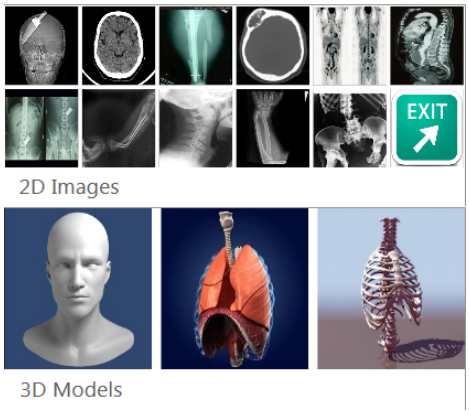
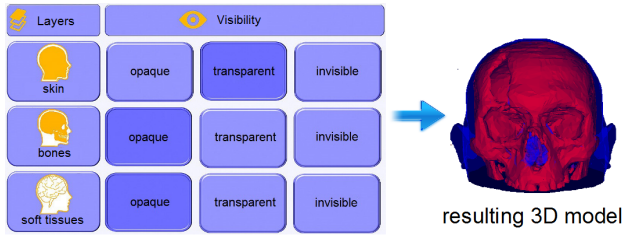
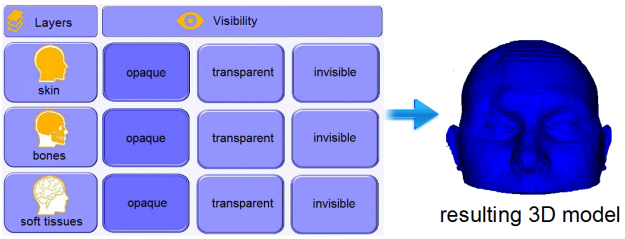


Fig. 9: Menu for selecting the application functionalities.



(a) Two layers opaque (bones and soft tissues) and one transparent (skin).

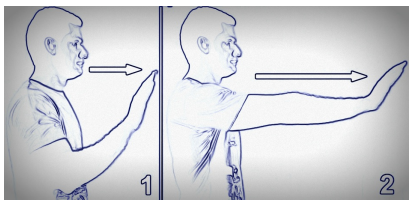


(b) All the layers opaque (bones, soft tissues and skin).

Fig. 10: In (a) and (b), different configurations of visibility for generating 3D models.

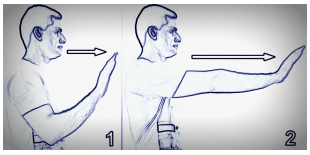
C#. However, using 3D models with a number of faces higher than fifty thousand (50,000), the frame rate per second (FPS) obtained was not interactive. Thus, we decided to improve the design of our existing code by refactoring it using *OpenGL*, resulting this time in interactive frame rates (around 9 FPS, 3 times faster than in the previous implementation using the *System.Windows.Media.Media3D* graphics library).

From MRIS or CTs is possible to extract a 3D model using a software such as the *InVesalius* [11]. However, a 3D model extracted without any image processing treatment does not show all the layers of interest in the human body. For example, using the 3D model of a human head, it is only possible to view its outermost layer, *i.e.*, the skin. Consequently, it is necessary to perform various segmentations to compose the desired 3D anatomical model and organize it into layers. Thus, in order to access the whole information contained in the segmented model, some body layers must become transparent, invisible or opaque, according to the user's needs. More specifically, a 3D human head model can be divided into three main layers, namely: (1) the outer layer, in which the skin is viewed; (2) the intermediate layer, containing the

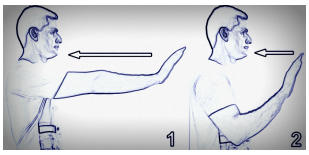


(a) Selecting an image.

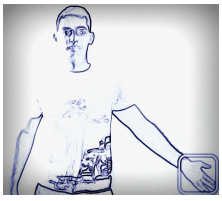
(b) Activating *zoom in* and *out* movements.



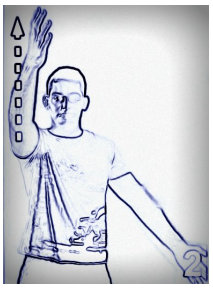
(c) Zooming in.



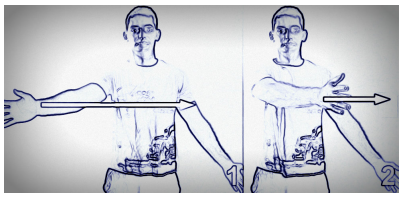
(d) Zooming out.



(e) Activating translation movement.



(f) Translating along y-axis.

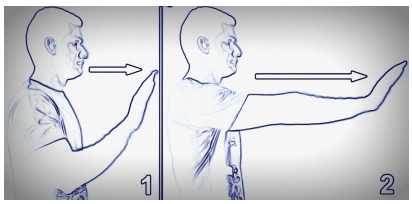


(g) Translating along x-axis.

Fig. 11: In (a), (b), (c), (d), (e), (f) and (g), hand gestures to control 2D images.

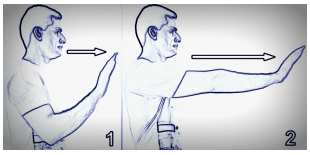
bony parts; and (3) the inner layer, which displays soft tissue structures, such as the brain, tumors, etc.

During the visualization and manipulation of 3D models, we can also trigger the visibility settings menu, as shown in (a) and (b) of Figure 10, performing the gesture for selecting the *Visibility Settings* with both arms straight, perpendicular to the trunk, pointing forward, as shown in (i) of Figure 12. On this screen (Figure 10, the user selects what type of visibility settings (transparent, opaque or invisible) should be displayed in each body layer. Following that step, the user returns to the

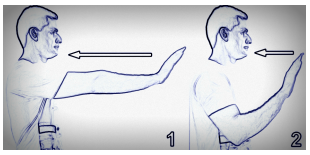


(a) Selecting a 3D model or the Visibility Settings Menu.

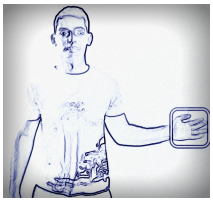
(b) Activating *zoom in* and *out* movements.



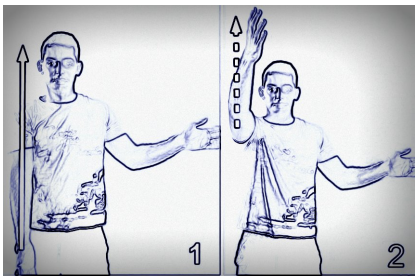
(c) Zooming In.



(d) Zooming Out.

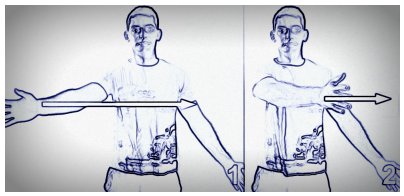


(e) Activating rotational movement around x-axis.



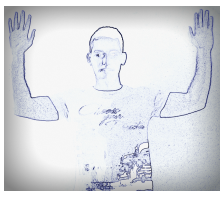
(f) Rotating around x-axis.

(g) Activating rotational movement around y-axis.



(h) Rotating around y-axis.

(i) Selecting visibility settings.



(j) Menu exit.

Fig. 12: In (a), (b), (c), (d), (e), (f), (g), (h) and in (a), (i), (j), respectively, hand gestures to control 3D models and the visibility settings menu.

mode of viewing and manipulating the 3D model (see (j) of Figure 12) and can reconfigure this functionality at any time.

E. Camera Parameters and Lighting

All the features we have implemented for manipulating 3D models depends on the following parameters of the synthetic camera: position, *look at* vector, *view up* vector and *global view up* direction, shown in Figure 13.

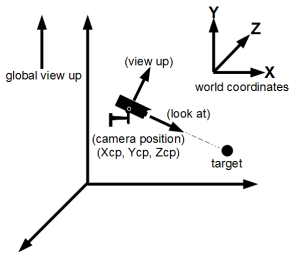


Fig. 13: Camera parameters.

The camera position corresponds to its world coordinates; *look at* is the direction the camera is pointed, which is obtained from the coordinates of the camera position and from the region of interest; *view up* vector is the rotation around viewing direction; and *global view up* direction is the rotation around viewing direction for the camera in the world coordinate system. The camera *position* and *global view up* parameters are numerical input values, however, the *look at* and *view up* are calculated by the system. The camera rotation along the axes x, y and z is made as follows. We recalculate all the camera parameters previously described, with the exception of the *global view up*. The camera position is initially represented by any coordinate which is updated according to the angle and the axis on which it was rotated; the *look at* is recalculated according to the current position of the camera; the *view up* is calculated from two cross products (the first one, between the *look at* and the *global view up*, and the second, between the result obtained from the first cross product and the *look at*). In all calculations, we use the right-hand rule. An oriented bounding box is also used to facilitate basic operations using 3D models, since it is usually less complex than the geometry of the "real" shape and useful to speed up calculations like centralization of objects in the middle of the screen and rotations, for example.

Lighting is also another important factor that influences the rendering of 3D models, because it changes the color perception of objects, influencing the outcome of the generated image: if an object is being poorly lit, the observer perceives its color tending to black or RGB(0, 0, 0); otherwise, to white or RGB(255, 255, 255). In our system, the lighting requires special treatment, because the coordinate system used in the rotation transformations has as reference the camera (and not the object). Consequently, the light position should be changed, according to the camera position. Otherwise, the camera may display model plans that are not being illuminated.

IV. User-Based Evaluation

User-based evaluations were conducted to perform a quantitative and qualitative analysis of the system. The analysis was performed with 16 users (1 specialist and 15 non specialists), 10 males and 5 females, aged between 19 and 36 years, 6 of them accustomed to playing computer games averaging 10 hours per week.

The testing sessions consisted of four steps: (1) a *preliminary questionnaire*, (2) a *calibration phase*, (3) a *testing phase* to evaluate the user performance (quantitative analysis), including a set of specific manipulation tasks of 2D images (search & selection, zoom in, zoom out and translation along the x and y-axes simultaneously) and 3D models (rotation around the x and y-axes, zoom in, zoom out, selection of the visibility settings and menu exit), and (4) a *post testing questionnaire* to produce a qualitative analysis.

As previously described in Subsection III-D, these specific selected tasks performed by the user were presented to a senior consultant surgeon who described them as the most useful ones for this initial test, before testing the application in the operating room, since many of them are the most prevalent in use. Thus, the main objectives of this evaluation were to verify if the participants would be able to perform these gestural control movements and to compare their performance (the time spent on each task, the accuracy and the number of false positives) to the performance of an expert in the field, as well as to collect insights to drive and improve the design of the application through the qualitative opinions of the participants about their experiences and usability in general.

The test setup was defined as follows. The *Kinect* was positioned 1.5 meters from the user and to 1.0 meter from the floor, in a lighted environment. The *pre-test questionnaire* elicited knowledge about previous experience in using gestural input devices and interactive manipulation of 2D images and 3D objects, as well as area of interest. Three groups of participants (each one with 5 users), all of them students, with similar areas of interest were then identified: *Group 1 (G1)*, with participants whose area of interest is Human Sciences, with or without experience using gesture-based systems or devices; *Group 2: (G2)*, with participants whose area of interest is the Exact Sciences, with or without experience using gesture-based systems or devices; and *Group 3 (G3)*, with participants whose area of interest is Computer Science, with experience using gestural control devices and interactive manipulation of 2D images and 3D objects.

The *calibration phase* was used to training the participant during 5 minutes, teaching how to use the system functionalities. The participant was then asked to interactively explore the system freely, as recommended in [10], stimulating the user to perform more natural movements, as well as repeating the gestures that generated greater insecurity or any doubt.

After the *calibration phase*, the *testing phase* was conducted, including ten tasks in number. We measured the time taken for the completion of each successful task; the number of attempts to perform each task (each fault is considered as

one attempt, less attempts indicate higher accuracy); and the number of false positives generated (every time the user plans to activate a feature, but enables another).

In order to evaluate the search and selection tasks of images 2D, the users had to select an image from a set of 30 images and to evaluate the zoom in and out, they had to perform, respectively, two zoom in and two zoom out consecutive operations in the selected image. As for the translation, the user had to perform two translations to position the image in a particular region of the screen. Regarding the 3D model manipulation, the users had to make two zoom in and two zoom out consecutive operations, keeping the model in the center of the screen. Furthermore, the user had to rotate the 3D model in 360° around the x and then y-axes. Finally, to select the visibility settings and to exit the application, the user had to perform these gestures when asked by the expert user.

Finally, the final *post testing questionnaire*, was prepared to collect user's preferences during interacting with our system, including usefulness, visual quality of the interface, ease of learning, ease of use, 3D spatial perception, level of interactivity, mental and physical fatigue, effectiveness and satisfaction. All these items received a score ranging from 1 (poor) to 5 (very good), following the pattern of granularity defined in [2]. With the analysis of the *post testing questionnaire* was possible to measure the overall satisfaction of the users.

The individual user values (including 15 participants and 1 specialist) measured by the evaluator, relative to "time" and "number of attempts", for manipulating 2D images are shown, respectively, in (a) and (b) of Figure 14. In particular, the user 9 (from *G2*) and the user 1 (from *G1*) obtained better performance than the specialist.

The same procedure was repeated for the manipulation of 3D models, and the individual user values are shown, in (a) and (b) of Figure 15. In particular, the specialist had better performance (19.2s), followed by the user 14 (from *G3*) with 19.3s, then by the user 1 (from *G1*) with 23.3s. The average total times for *G1*, *G2* and *G3* were respectively 34.1 $\pm$ 8.3s, 31.7 $\pm$ 6.5s, and 26.8 $\pm$ 4.2s. Again, *G3* had better performance, followed now by *G2* and *G1*. Regarding the 'number of attempts", the average number for *G1*, *G2* and *G3* were 1.1, 1.1, and 1.1, respectively. Again, the variance between groups was very small and their level of accuracy when performing the gestures was quite high. The number of occurrences of false positives was negligible and therefore was not included in our analyzes.

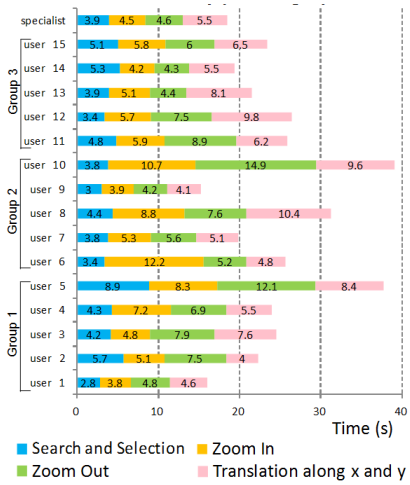
In order to compare the performance of the groups (*G1*, *G2* and *G3*) as a whole, in relation to the set of activities done, all the manipulation tasks of 2D images (search & selection, zoom in, zoom out and translation) and 3D models (rotation around the x and y-axes, zoom in, zoom out, selection of the visibility settings and menu exit) were grouped into *Task1* and *Task2*, respectively. The results are shown in (a) and (b) of Figure 16. More specifically, the average total times for *G1*, *G2* and *G3* were respectively 24.9 $\pm$ 7.9s, 26.2 $\pm$ 9.3s, and 23.3 $\pm$ 2.9s, showing that *G3* had better performance, followed by *G1* and *G2*. Regarding the 'number of attempts", the average values to perform the correct gestures for *G1*, *G2* and *G3* were 1.25, 1.1, and 1.1, respectively. The participants from *G1* had more difficulty to control the gestures than those from the other groups. However, the variance between groups was very small, which leads to the conclusion that the gestures are easy to control.

The final *post testing questionnaire* provided a detailed feedback about the system. More specifically, on average, the users in general considered the system very useful; with very good visual quality of the interface (although some participants commented that in the visibility settings menu, all the components of the interface seem clickable buttons, which is not the case, and thus, means that there is still room for improving the interface design); very easy to learn (although *G2* had more difficulty to memorize the sequence of gestures); very easy to use (*G1* particularly showed great interest and curiosity in using the system); offering a good 3D spatial perception (participants from *G3* reported some discomfort during the zooming in/out movements); very interactive; causing practically no mental or physical fatigue (the users from *G3* mentioned that during some specific gestures of zooming in/out it was necessary to keep more focused on the task to avoid committing mistakes and that depending on the duration of the task, some physical fatigue may occur); very effective; and producing a high level of satisfaction.

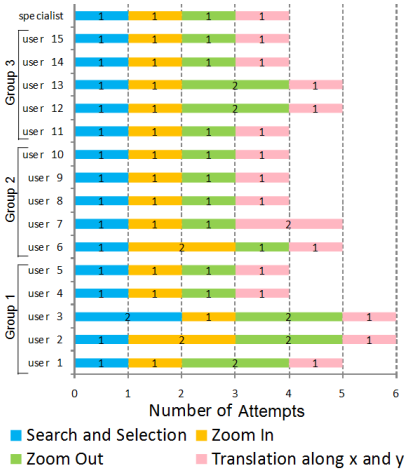
V. Results and Discussion

Although we have used in our evaluation study a small number of user reviews (16), if there were any usability issues, they would have been highlighted. According to Nielsen [16], an analysis of usability with five users has revealed 85% of the popular errors, and it does not matter whether you test websites, intranets, PC applications, or mobile applications, with 5 users, you almost always get close to user testing's maximum benefit-cost ratio.

In our current evaluation study, the different users were behaving in a very similar way and no major usability error was found. Actually, the user experience overlaped between the groups. The results show that the majority of the participants are able to perform all the assigned tasks in the tests, quickly and accurately, despite their age, gender or prior knowledge of gesture-based systems or devices, when compared to the performance of the expert in the field. In practice, without the occurrence of false positives. Interestingly, a variable observed among participants (regardless of which group they belonged) was that users accustomed to playing computer games were more comfortable during the tests and showed the best results. Also, in general, we noted that the users who performed the tasks very quickly, too self-confident, were those who made more often inaccurate gestures (less precise), consequently, tried more times to get the gestures right (more attempts) and generated more false-positives, while the users that spent more time to complete the tasks were those who have failed less. However, we found that as the number of user attempts (the higher the number of attempts to make a gesture correctly, the smaller the accuracy of the gesture made) and the number

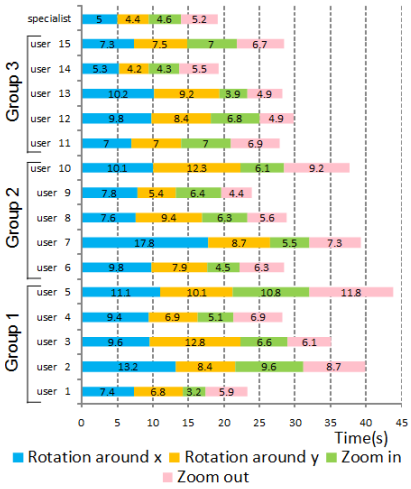


(a)

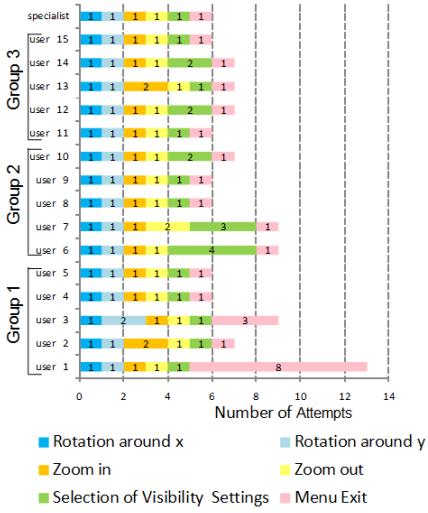


(b)

Fig. 14: In (a) and (b), the time spent and number of attempts to perform different tasks, when interacting with 2D images, respectively.



(a)



(b)

Fig. 15: In (a) and (b), the time spent and number of attempts to perform different tasks, when interacting with 3D models, respectively.

of false-positives decrease, the participants gain experience in how to perform a particular gesture. Thus, having less impact when measuring performance, since we consider the best time obtained from all attempts of a user, as the comparative measure of the user experience relative to the performance of an expert in the field.

Moreover, the test results provided us with new possibilities to improve the design and the quality our gesture-controlled system. During the tasks, some participants have made additional relevant remarks. For example, they commented on the importance of including visual cues in the interface, particularly when handling 2D images, indicating which control feature is currently active, in order to better guide the user and offer a greater sense of confidence (as the visual cues we have already included in the 3D models interface). Users in general do not found the movements of zoom in/out very intuitive and suggested the redesign of these gestures, perhaps by creating a more continuos movement. This indicates that the level of precision of the users can increase even more if we progressively improve our prototype by adding new visual cues in its user interface.

VI. Conclusions and Future Work

We believe that gesture controlled systems seem to be promising, since they allow for a touch-free control of computer systems. In this work, we have successfully presented and evaluated a gesture-controlled system for interactive ma-

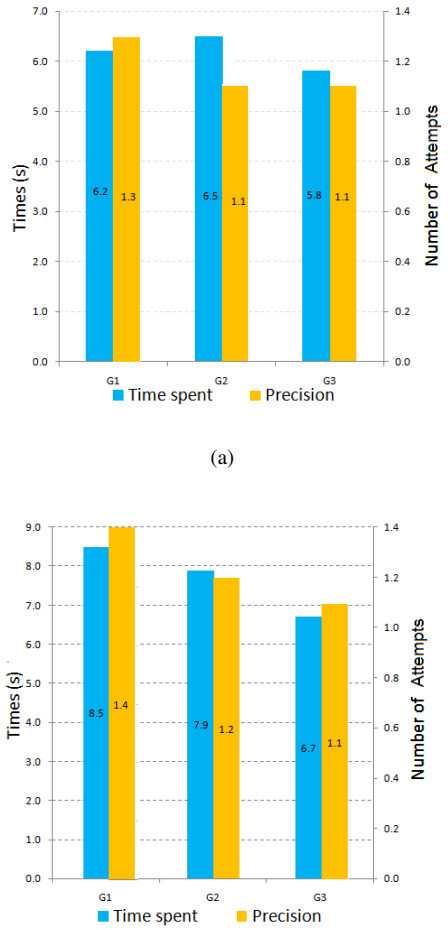


Fig. 16: In (a) and (b), the general performance of 2D images and 3D models (time spent and precision) per group of participants, respectively.

nipulation of 2D images and 3D models using the *Kinect* device.

The results show that the users (the majority of them, having inexperience in the use of gestural control devices) are able to perform the tasks of search & selection and manipulation of 2D images (zoom in/out and translations) and 3D models (zoom in/out and rotations), quickly and accurately when compared to the performance of an expert in the field. In practice, without the occurrence of false positives, demonstrating the usefulness of the system as a possible effective and competitive alternative to the traditional use of the negatoscope. More specifically, taking into account the performance of the expert and comparing it to the average performance of the groups, we observed that the variation is small, considering that the expert accesses the system with great frequency, while the participants had a brief training only during the usability testing. Furthermore, the qualitative analysis overall were very positive and showed a high level of user satisfaction, as well

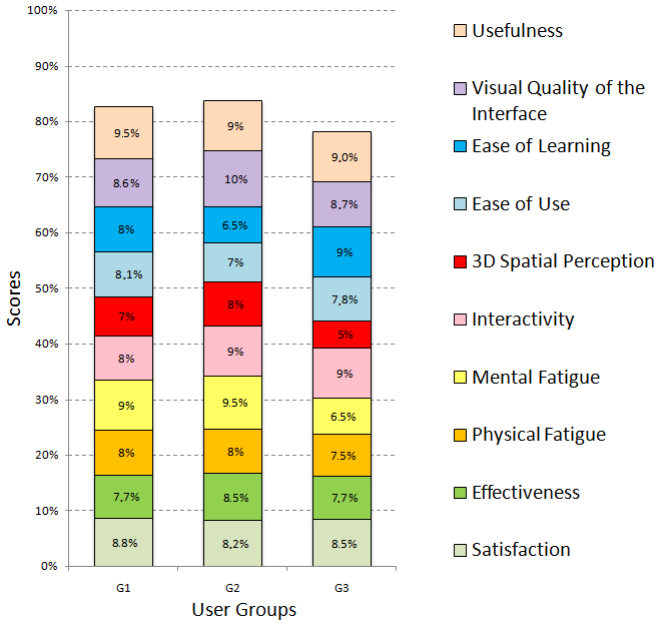


Fig. 17: User-based evaluation (qualitative analysis).

as highlighted interesting points for improving the quality and design of the system.

As future work, there are several possibilities. After this first study with 16 users which has identified the main usability problems of our system, we plan to fix them and improve the system design. Other possibilities are the implementation of more intuitive gestures and the use of new gestural control devices (*e.g.*, the *Leap Motion*), a more in-depth study of the gesture recognition process in users with different heights and positioned at different distances from the *Kinect*, the implementation of tracking for identifying gestures, and the inclusion of new features related to the visualization of 3D models. We can also anticipate that another challenging extension of this work which would be of the utmost interest is to test the system in the operating room in the near future, having surgeons as collaborators. Some initial strategies have already been defined to validate our system based on real cases. This task is indeed not simple, since it involves ethical issues and human beings, therefore it always prevail the expected benefits over the foreseeable risks, following an appropriate methodology and relying on the free informed consent of the subject of the research and/or of his/her legal representative.

Acknowledgments

Édimo S. Silva is supported by CAPES and Maria Andréia F. Rodrigues by CNPq (under grant No. 481326/2013-8), and would like to thank for their financial support. Finally, we are also grateful to the referees for providing constructive comments and suggestions to improve the manuscript.

References

[1] S. Abe, *Support Vector Machines for Pattern Classification*. Springer, 2010.

[2] G. Albaum, "The Likert Scale Revisited", *Journal-Market Research Society*, pp. 331–348, 1997.

[3] L. Baraldi, F. Paci, G. Serra, L. Benini, and R. Cucchiara, "Gesture Recognition in Ego-Centric Videos using Dense Trajectories and Hand Segmentation", in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2014, pp. 702–707.

[4] K. Biswas and S. K. Basu, "Gesture Recognition using Microsoft Kinect®", in *Proceedings of the* 24^{th} *International Conference Automation, Robotics and Applications*. IEEE, 2011, pp. 100–103.

[5] M. N. K. Boulos, B. J. Blanchard, C. Walker, J. Montero, A. Tripathy, R. Gutierrez-Osuna *et al.*, "Web GIS in Practice X: a Microsoft Kinect Natural User Interface for Google Earth Navigation", *International journal of health geographics*, p. 45, 2011.

[6] S. Carvalho and C. M. Presenation, "Augmenting Medical Workspaces with Kinect-based Interfaces", in *Proceedings of the Conference on Human Factors in Computing Systems*, 2012.

[7] W. H. Chen, C. T. Hsieh, and T. T. Liu, "A Real Time Hand Gesture Recognition System Based on DFT and SVM", *Applied Mechanics and Materials*, pp. 3004–3009, 2013.

[8] D. A. Clunie, *DICOM Structured Reporting*. PixelMed Publishing, 2000.

[9] N. Dardas, Q. Chen, N. D. Georganas, and E. M. Petriu, "Hand Gesture Recognition Using Bag-of-Features and Multi-Class Support Vector Machine", in *Proceedings of the 2010 Haptic Audio-Visual Environments and Games*. IEEE, 2010, pp. 1–5.

[10] A. Davidson, "An Evaluation of Visual Gesture Based Controls for Exploring Three Dimensional Environments", Master's thesis, School of Computer Science & Statistics, 2012.

[11] T. F. de Moraes, P. H. J. Amorim, F. de Souza Azevedo, J. V. L. da Silva, H. Pedrini, G. C. S. Ruppert, L. O. Reis, M. R. Moreno, C. A. Rodriguez, A. A. Camilo *et al.*, "InVesalius: An Open-Source Imaging Application", *World Journal of Urology*, pp. 687–691, 2012.

[12] L. Ebert, G. Hatch, and A. G., "You Can't Touch This: Touch-free Navigation Through Radiological Images", *Surgical Innovation 19*, pp. 301–307, 2012.

[13] L. C. Ebert, G. Hatch, M. J. Thali, and S. Ross, "Invisible Touch: Control of a DICOM Viewer with Finger Gestures Using the Kinect Depth Camera", *Journal of Forensic Radiology and Imaging*, pp. 10–14, 2013.

[14] L. Gallo, A. P. Placitelli, and M. Ciampi, "Controller-Free Exploration of Medical Image Data: Experiencing the Kinect", in *Proceedings of the* 24^{th} *International Symposium on Computer-Based Medical Systems*. IEEE, 2011, pp. 1–6.

[15] G. Glonek and M. Pietruszka, "Natural User Interfaces (NUI): Review", *Journal of the American College of Surgeons*, pp. 27–46, 2012.

[16] N. N. Group, "Quantitative Studies: How Many Users to Test, 2006", Available at www.nngroup.com/articles/quantitative-studies-how-many-users/, 2014, last visited December, 2014.

[17] D. Huang, W. Hu, and S. Chang, "Gabor Filter-based Hand-pose Angle Estimation for Hand Gesture Recognition Under Varying Illumination", *Expert Systems with Applications*, vol. 38, pp. 6031–6042, 2011.

[18] A. Jana, *Kinect for Windows SDK Programming Guide*. Packt Publishing Ltd, 2012.

[19] R. Johnson, K. O'Hara, A. Sellen, C. Cousins, and A. Criminisi, "Exploring the Potential for Touchless Interaction in Image-Guided Interventional Radiology", in *Proceedings of the 2011 Conference on Human Factors in Computing Systems*, 2011.

[20] S. Kavanagh, "Facilitating Natural User Interfaces through Freehand Gesture Recognition", pp. 1–6, 2012.

[21] S. Kratz and R. Ballagas, "Unravelling Seams: Improving Mobile Gesture Recognition with Visual Feedback Techniques", in *Proceedings of the 2009 Conference on Human Factors in Computing Systems*. ACM, 2009, pp. 937–940.

[22] H. Lab, "MITO - Medical Imaging TOolkit", Available at http://ihealthlab.icar.cnr.it/index.php/projects/9-mito.html, 2014, last visited December, 2014.

[23] K. LaBelle, "Evaluation of Kinect Joint Tracking for Clinical and In-Home Stroke Rehabilitation Tools", 2011, undergraduate Thesis, University of Notre Dame.

[24] K. F. Li, A. Sevcenco, and E. Yan, "Telerehabilitation Using Low-Cost Video Game Controllers", in *Proceedings of the 2013 International Conference on the Complex, Intelligent, and Software Intensive Systems*. IEEE, 2013, pp. 136–143.

[25] H. Lindberg and P. Rydin, "Model View Controller", 2001.

[26] M. R. Malgireddy, J. J. Corso, S. Setlur, V. Govindaraju, and D. Mandalapu, "A Framework for Hand Gesture Recognition and Spotting Using Sub-Gesture Modeling", in *Proceedings of the* 20^{th} *International Conference on Pattern Recognition*. IEEE, 2010, pp. 3780–3783.

[27] P. Meenan, "User Experiences While Playing Dance-Based Exergames and the Influence of Different Body Motion Sensing Technologies", *International Journal of Computer Games Technology*, 2013.

[28] Microsoft, "Kinect for Xbox 360", Available at http://www.xbox.com/en-US/xbox-360/accessories/kinect, last visited December, 2014.

[29] ——, "Introducing Windows Presentation Foundation", http://msdn.microsoft.com/en-us/library/aa663364.aspx, 02 2014, last visited March, 2014.

[30] ——, "Kinect for Windows Developer Toolkit v1.8", Available at http://www.microsoft.com/en-us/download/details.aspx?id=40276, 2014, last visited February, 2014.

[31] ——, "Skeletal Tracking", Available at http://msdn.microsoft.com/en-us/library/hh973074.aspx, 09 2014, last visited September, 2014.

[32] ——, "Windows Presentation Foundation", Available at http://msdn.microsoft.com/en-us/library/ms754130.aspx, 02 2014, last visited February, 2014.

[33] J. Oh, T. Kim, and H. Hong, "Using Binary Decision Tree and Multiclass SVM for Human Gesture Recognition", in *Proceedings of the 2013 International Conference on Information Science and Applications (ICISA)*. IEEE, 2013, pp. 1–4.

[34] K. O'Hara, G. Gonzalez, A. Sellen, G. Penney, A. Varnavas, H. Mentis, A. Criminisi, R. Corish, M. Rouncefield, N. Dastur *et al.*, "Touchless Interaction in Surgery", *Communications of the ACM*, pp. 70–77, 2014.

[35] L. Omelina, B. Jansen, B. Bonnechere, S. Van Sint Jan, and J. Cornelis, "Serious Games for Physical Rehabilitation: Designing Highly Configurable and Adaptable Games", in *Proceedings of the* 9^{th} *International Conference on Disability, Virtual Reality & Associated Technologies*, 2012, pp. 195–201.

[36] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, "Robust Part-based Hand Gesture Recognition Using Kinect Sensor", *IEEE Transactions on Multimedia*, vol. 15, no. 5, pp. 1110–1120, 2013.

[37] A. Rosset and L. Spadola, "OsiriX: An Open-Source Software for Navigating in Multidimensional DICOM Images", *Journal of Digital Imaging*, pp. 205–216, 2004.

[38] G. Ruppert, P. Amorim, T. Moares, and J. Silva, "Touchless Gesture User Interface for 3D Visualization Using Kinect Platform and Open-source Frameworks", in *Proceedings of the Fifth International Conference on Advanced Research in Virtual and Rapid Prototyping*, 2011, pp. 215–219.

[39] K. Sabir, C. Stolte, B. Tabor, and S. I. O'Donoghue, "The Molecular Control Toolkit: Controlling 3D Molecular Graphics Via Gesture and Voice", in *Proceedings of the 2013 IEEE Symposium on Biological Data Visualization*. IEEE, 2013, pp. 49–56.

[40] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore, "Real-time Human Pose Recognition in Parts from Single Depth Images", *Communications of the ACM*, pp. 116–124, 2013.

[41] E. S. Silva and M. A. F. Rodrigues, "Um Sistema de Controle Gestual de Apoio a Procedimentos Cirúrgicos", in *Proceedings of the XVI Symposium on Virtual Reality and Augmented Reality*, 2014, pp. 287–296.

[42] B. G. Strickland M., Tremaine J., "Using a Depth-sensing Infrared Camera System to Access and Manipulate Medical Imaging from Within the Sterile Operating Field", *Canadian Journal of Surgery*, vol. 56, pp. E1–E6, 2013.

[43] Z. M. Tan J., Chao C., "Informatics in Radiology: Developing a Touchless User Interface for Intraoperative Image Control During Interventional Radiology Procedures", *Radiographics, volume 33*, pp. 61–70, 2013.

[44] W. Technologies, "Wavefront", Available at http://www.wvfront.com/, 02 2014, last visited June, 2014.

[45] J. D. Tremaine, G. O. Brigley, and L. Strickland, "Gesture Operated Control for Medical Information Systems", Mar. 28 2012, uS Patent App. 14/008,179.

[46] X. Wang, M. Xia, H. Cai, Y. Gao, and C. Cattani, "Hidden-Markov-Models-Based Dynamic Hand Gesture Recognition", *Mathematical Problems in Engineering*, 2012.