A case study on 3D Virtual kitchen design with Kinect sensor

Matias N. Leone, Mariano M. Banquiero, AndresBursztyn

Proyecto de Investigación "Explotación de GPUs y Gráficos Por Computadora", GIGC - Grupo de Investigación de Gráficos por Computadora, Departamento de Ingeniería en Sistemas de Información, UTN-FRBA, Argentina

{mleone, mbanquiero}@frba.utn.edu.ar
andresb@sistemas.frba.utn.edu.ar

Abstract. The industry of kitchen design and furniture fabrication has been largely aided by 3D virtual scene applications and real-time visualization software. Microsoft Kinect provides skeleton tracking and bone orientation features than can be exploited in order to improve the interaction between the user and the 3D virtual scene representation.In this work we present a case study of an interactive application that assists the kitchen design process utilizing Microsoft Kinect hardware. A 3D avatar mesh mimics the user movement in real time, gesture recognition techniques are applied to produce actions on the virtual scene and the user can interact with custom made UI widgets using the hands as mouse cursors.

Keywords: Skeleton tracking, Gesture recognition, Kitchen design, DirectX

Fig.1. Left: 3D kitchen scene rendered by the application. Middle: the user interacts with the software using his hands. Right: the avatar follows the movement of the user in real time.

1 Introduction

Lepton Systems [1] is a software company that provides a desktop application for kitchen design. It let the user to choose appliances from the majority of the industries providers, visualize them, interact with them and set their location in a 3D representation of the kitchen. The application comes with a real time rendering engine that allows the user to envision a virtual representation of the kitchen being designed. It also provides tools for material optimization, costs estimation, reports generation and Autocad [2] model importers.

The company was planning their stand for the industry bigger exhibition celebrated in the country, and they wanted to show a state of the art demonstration of their technological capabilities for kitchen design.

In order to achieve this objective they asked our help to develop a software solution for 3D virtual interaction (Fig. 1). The goal was to build an application that lets the user of the exhibition to interact with the kitchen designed using his own hands. Allowing the user to change the material of the appliances, modify handles and terminations, and navigate through all the corners of the kitchen. The request had the aim to provide a better degree of interaction and immersion than the traditional input approach of mouse and keyboard.

The application developed makes use of Microsoft Kinect [3], which is a hybrid hardware-software solution for 3D skeletal tracking. Its sensors capabilities, its processing power and the low cost of the hardware make it an adequate candidate to fulfill the request. The features of Kinect used in this project are 2D hands tracking for UI navigation and 3D skeletal tracking for gesture recognition and avatar controlling.

The software was designed with three main modes of interaction:

- UI interaction mode: the user controls two mouse pointers with his own hands and navigates in 2D through different UI widgets, like buttons, scrollbars and message boxes, that allow him to interact with the scene.
- Avatar mode: a human 3D mesh is rendered and updated based on the skeleton data tracked by Kinect. The user can move his body and the avatar mimics his movements in real time. The user can also do some simple gestures that are recognized by the application to produce specific actions on the scene, like pulling his hand to open a drawer or waving his hand to close a door.
- 3D navigation mode: the user can move his hands to navigate through the scene in three dimensions. The right hand controls the position of the camera and may also alter the orientation. This allows the user to visualize the kitchen scene from different points of view.

2 Related work

The entertainment and videogame industry has been long trying to find alternatives to the classical joystick input in order to increase the level of realism of their titles. Nintendo Wii [4] was one of the first consoles to introduce a joystick with motion sensing capabilities through the use of accelerometer and optical sensor technology.Many attempts have been made to takes the Wii control to general applications beyond videogames, like the works described in [5][6][7].

Sony also provides PlayStation Move [8] as an alternative for PlayStation 3 console joystick. A complete reference ofvideogame motion controls and its evolution over time can be found in [9].

The Kinect for Windows Human Interface Guidelines [10] has importantremarks that should be taken into account when a new Kinect application is developed. Basic steps to integrate Kinect in a Windows application are described in [11]. A detailed explanation of some internal Kinect algorithms is described in [12], and [13]

explained the smoothing techniques used to filter undesired noise of the hardware sensors. Advanced hands and finger recognitiontechniques can be found in [14] and [15].

The project Ludique's Kinect Bundle [16] allows the user to employ the Kinect camera to interact with Windows UI in many ways. And Kineticspace[17] is a generic tool for gesture tracking and recording using Kinect.

The work in [18] also explores the use of Kinect in a kitchen but not from a design perspective as this paper, but for making use of Kinect in real-life kitchens for a variety of interactions, especially when your hands are full or messy,hence touchableand others traditional interfaces are not an option.

3 Application overview

Three modes of interaction were designed for the software. Each mode allows different kind of actions and makes use of diverse features of Kinect SDK:

3.1 UI Interactionmode

In this mode the user can moves his hands to control two mouse cursors in the screen. In every frame, the application retrieves the 3D position of both handsfrom Kinect, and then converts them to 2D screen positions. Each hand has its own delimited area of interaction (Fig.2). The bounding of the screen are adapted for each hand convenience.

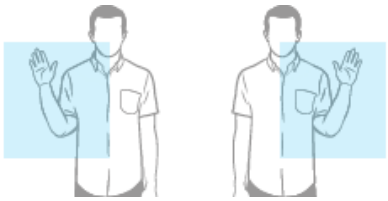

Fig.2. Screen bounding box of interaction computed for each hand.

For each hand, a screen bounding box is computed (physical interaction region) based on the distance between the hip center bone and the hand.Then the 3D position of each hand is converted to screen coordinates.

It should be noted that when Kinect cannot track the hand position, the previous screen position is used in order to avoid noisy movements.

The 2D coordinates of each hand is then used to place a cursor in the screen. The cursor movement allows the user to interact with different UI controls. The actions available are:

- To choose the kitchen scene to interact with from a list of pre-designed scenes.
- To change the material of all kitchen appliances, choosing between different types of surfaces, such as wood, metal and plastic.
- To change the handle of the doors and drawers of the kitchen.
- To change the mode of the application to avatar mode or 3D navigation mode.

The interaction with the UI was designed from scratch to be correctly adapted for Kinect movements. The buttons have a considerable screen size and are placed vertical aligned, in order to be comfortable to the user (Fig. 3).

Fig.3. Application UI widgets specially designed for comfortable Kinect interactions.

3.2 Avatar mode

In this mode the Kinect skeleton is tracked and used to render an avatar in the scene. The avatar is facing to the kitchen and gives his back to the camera. It is aligned with the 3D axis and it is watching towards negative Z.

The avatar is composed of many individual meshes that represent its body parts: arms, legs, torso, hands, feet, neck and head (Fig. 4). The 3D position and orientation of each bone is tracked and used to place the individual meshes correctly.

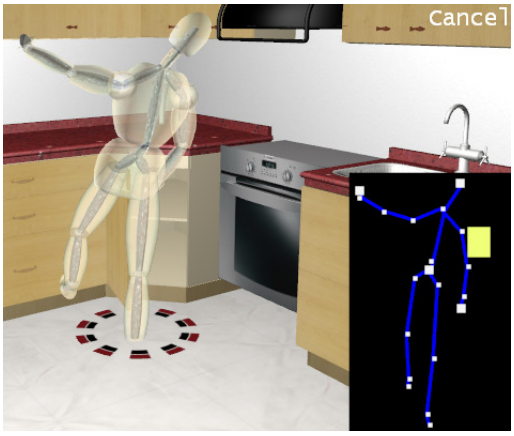

Fig.4. Left: avatar controlled by Kinect tracked skeleton. It is composed by individual meshes for each body part. Right: 2D tracking information of the skeleton.

Each individual mesh is initially located at the origin of coordinates pointing up to the positive Y axis. In every frame the mesh is rotated, translated and scaled according to its associated bone, using the following procedure:

- Restore the original position, orientation and scale of the mesh.
- Rotate the mesh using the orientation matrix tracked by Kinect for the start joint of the bone.

- Translate the mesh to the position of the start joint. This way the center of the mesh is located exactly where the join is, and its direction points to the end join of the bone.
- The mesh is scaled along its local Y axis according to the distance between the start joint and the end joint of the bone.
- To scale the XZ dimensions of the mesh, two approaches were tested. The first one consists of performing a scale proportional to the amount scaled in Y. The second approach is to have a fixed width and length for each mesh and not scale them at all. This second approach provides us with a better control of the avatar visual features.

This strategy allows the avatar to mimic the user movements in real time, which considerably increases the level of integration between the user and the application.

In order to allow the user to interact with the virtual kitchen being designed, the software is capableof detecting some simple hand gestures. The ones recognized by the application are:

- Move the hand forward and backward in a straight line: used to open or close drawers of the kitchen (Fig. 5).
- Wave the hand left to right or right to left: used to open or close doors of the scene.

In each frame,the application tracks both hands 3D positions and stores them in a buffer of one hundred previous frames. All these positions are treated as a cloud of points that need to be processed to extract some useful information. A statistical analysis is performed over this buffer in order to gather indicators that help to recognize the gesture. For each axis (x, y, z) the following indicators are computed: minimum value, maximum value, mean, variance and average of derivatives.

These measures are studied in each frame to detect valid avatar gestures. For example the open drawer gesture is triggered when:

- The average of derivatives in Z is positive
- The variance in X is almost cero
- The variance in Y is almost cero

This way the gestures are defined in a declarative fashion, based only on the values of this statistical indicators, and not on their absolute physical positions.

Once a gesture is detected the next step is to analyze if some object of the scene is close enough to be affected by the action. The application computes the screen distance between the average gesture position and the projected center of the object. The closest object is selected, provided the distance is below a given threshold, and the corresponding gesture action is rendered.

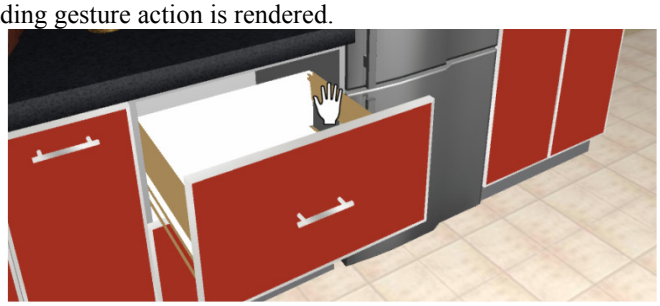

Fig.5. Opening drawer gesture recognized by the application.

Another approached was used to detect more complex gestures. First the required gestures are recorded in a custom made tracking application (Fig. 6). This tool allows the user to start and stop recording and then to trim the undesired parts the record. The tool tracks the movement of the hand for each frame and stores its 2D position and interval in a data file.

These gesture files are then used by the real time application to detect actions from the user and produce accordingly events in the kitchen scene. The application analyzes the last hands positionsof some frames and compares this buffer againstthe data of recorded gestures. The last one hundred frames are used as the size of the buffer being analyzed. Each value is compared against the recorded position of the gesture. If all distance of every tracked position and its corresponding recorded position is less than a given threshold then the application detects a gesture match.

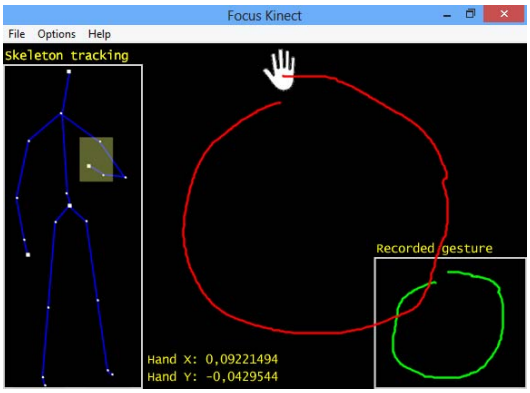

Fig.6. Tool developed for gesture recording.

But in order to accurately compare both gestures they must be previously transformed to some common frame. We callthis frame Gesture-space. First the samples are normalize to a space of [-1,1] x [-1,1], where the (0, 0) is the center of the gesture, which allow us to normalize the gesture scale. Then the first sample is positioned in (-1, 0) and the rest of the samples are rotated according to the angle of this first point. This step normalizes the gesture initial position and orientation. Finally the time of the gesture must be normalized, regenerating samples at regular intervals, which requiresadding or removing points so that both gestures may achieve the same amount of values. The total length traveled by the gesture is taken and divided by the samples count to obtain the segment size. The sequence is reconstructed based on this fixed segment.

Once the samples pass through this normalization steps both gesture can be compare as taking the distance of two functions. If this distance is lower than a given Epsilon, then both gestures can be considered equal.

This way the gesture detection procedure does not depends on absolute positions, speeds or any other physical properties, but on statistical measures, like average and standard deviation.

3.3 Navigationmode

In this mode the user can move around the kitchen scene using his right hand to control the camera. The movement is made in three dimensions. If the user moves up his right hand then the camera goes forward, and if he moves it down the camera goes backward. This axis controls the position of the camera, or look from. When he moves the hand left to right or right to left the user controls the orientation of the camera, or look at. This allows him to turn and watch different angles of the scene.

A map is shown at the right corner of the screen with a top view of the camera location and orientation in the scene.

4 Avatar tracking

The Kinect SDK sometimes cannot accurately capture all bones of the body, especially when one bone is occluded by another. For example this usually happens when the hand is exactlyin front of the shoulder. This kind of problems must be taken into account by the application in order to avoid undesired side effects, such as bad gesture recognition and unnatural avatar poses.

To avoid theseartifacts the software developed has an initial phase of skeleton calibration. During this phase, which last always the same fixed amount of time, all bones are tracked and stored in a buffer. Then some general measures of the skeleton are computed from this buffer, such as distance from hip to floor, length between the head and the hip, length of arms and legs, etc. These anthropomorphic features help the application to smooth and correct the data tracked from Kinect.

It is important that the calibration phase endure some seconds, in order to take an average of all these features. Otherwise incorrect initial measures may be used during all the application life-cycle.

For the Avatar mode the application takes the data tracked by Kinect but do not use it directly into their internal algorithm. A collection of human-constrain heuristics are applied in order to check that positions and orientations retrieved from the sensor areindeed valid for a human being. For example the length of the arms and legs must be equals to the original length captured in the calibration phase. The rotation matrix of knees, feet, elbows and neck cannot overpass allowed angles of motion for a human body. When these situations are encountered, the application ignores the current tracked value and reuses the last valid information for that bone. The same is applied when Kinect cannot track a bone at all.

All the coordinates captured by Kinect aregiven in “Kinect space”, but the Avatar mode needs to show the character interacting with the real dimensions of the kitchen. To achieve this, the application transformed the given coordinates from Kinect space to World space, using the following procedure:

- The hip position computed in the calibration phase is subtracted from all the other bones positions.
- The Z coordinate is inverted to achieve an avatar that gives its back to the user. This way the character can interact with the kitchen scene.
- The values are scaled to the scene measures.

After these steps, the individual meshes that compose the “avatar body” are placed in each bone with the procedure described in the previous sections.

5 UI

The user interface included in the application was specially designed to be adapted for Kinect style of movements and interactions. Widgets like buttons, scrollbars, list box and message box were developed using DirectX projected triangles with texture mapping.

First a low level GUI layer was built with many primitive instructions, such as draw line, draw rectangle, draw polygon, draw rounded rectangle, draw circle, draw disc, draw arc and draw image. All of them were developed with a combination of DirectX triangles fan, vertex color and texture sprites.

Then a higher level layer was built upon the last one to create the following UI widgets: button, displayable menu, frame, circle button, scroll button, message box, and progress bar. For example a button is render combining a draw image primitive with a draw rounded rectangle operation to show when the widget is selected. A frame is a floating panel that can contain other widgets within. A message box is created with a frame and two buttons for the "accept and cancel" options. The progress bar is rendered with many rounded rectangles and one main rectangle that vary its size according to the current progress.

All the buttons and menu items are selected with a timeout approach. The user controls two mouse cursors with his hands and when one of them is over a button a timer is started. If the user let the hand over this button for some time, then the button is selected. A circled progress bar is rendered around the cursor to give feedback about the elapsed time.

One problem with this approach is that the user tends to get his shoulder tired after some minutes of interaction, and then his hands begin to tremble. This flickering produces small movements on the screen and usually the hands get out of the selected button. When this happen, the widget timer is restarted and the task of accurately selecting one specific button becomes a frustrated action.

Our application solves this problem by reducing the amount of movement allowed to the cursor when it stays inside a button. When the hand is over a widget than can be selected, the application automatically centers the cursor to the middle of it (snap to grid). All small movements of the hand are discarded. If the user wants to exit the button then he has do an abrupt change in position. This strategy reduced the flickering and makes the UI easier to use.

6 Implementation and Results

We used Microsoft Kinect SDK 1.7 to track skeleton positions and orientations. These tracking features were integrated in our own real time rendering engine developed with C# 4.0 and Microsoft DirectX 9.

The kitchen scenes and furniture meshes were loaded from the Lepton systems proprietary format used in its software Focus 3D 2013 [1].

The application was run in a notebook with Windows 8, Intel Core i5 1.7 GHz with 8 GB Ram and Intel HD Graphics 4000 GPU.

The notebook also displayed the application through a projector located in the roof, so when one person was interacting with the software others were able to see what he was doing (Fig. 7).

Fig. 7. Top: a user of the exhibition selecting a material with his hands through the application. Bottom: other people watching the projected image of the software.

7 Conclusions and future work

A better degree of interaction between the user and the application waseffectively achieved and Lepton Systems was greatly satisfied with the results. The public at the exhibition was intrigued by the new experience of using his body to move around a virtual kitchen. Some vendors even propose to adapt the idea to other kind of business.

One of the main drawbacks we could observe during development is that Kinect cannot replace the traditional mouse and keyboard solution for many scenarios. Using your hands to move a screen cursor is not as precise as using the mouse. And when an application required a great degree of control the user may get frustrated with Kinect.

We believe that Kinect should be chosen for cases where traditional inputs do not apply. For example when the user cannot use his hands to touch the computer because he is doing something else or when all parts of the body may contribute to the action and not just the hands.

Another problem we encountered at the exhibition was that Kinect sometimes get confused when many people were near the range of the sensor. A common situation was when one user was interacting with the application and suddenly other person crossed behind. In most cases the software lost all the tracking and an application restart was required.

One feature of Kinect that could be integrated to this application is the use of speech recognition. The combination of hands gestures and spoken orders may increase the level of interaction.

Future improvements from the next version of the hardware (Kinect One [19]), specially about fingers tracking, may open many new interesting alternatives of interaction.

8 References

1. Lepton Sistemas SRL Soluciones inteligentes. 2013. Software for kitchen design and material optimization.
2. Autodesk. 2013. AutoCAD. 3D CAD Design Software.
3. Microsoft Corp. 2013. Kinect for Windows SDK.
4. Nintendo Co., Ltd. 2006. Nintendo Wii. Home video game console.
5. NithinSanthanam. 2012. Wii remote as a web navigation device for people with cerebral palsy. In Proceedings of the 14th international ACM SIGACCESS conference on Computers and accessibility (ASSETS '12). ACM, New York, NY, USA, 303-304.
6. David Scherfgen and Rainer Herpers. 2009. 3D tracking using multiple Nintendo Wii Remotes: a simple consumer hardware tracking approach. In Proceedings of the 2009 Conference on Future Play on @ GDC Canada (Future Play '09). ACM, New York, NY, USA, 31-32.
7. Sergio Albiol-Pérez, José-Antonio Gil-Gómez, Mariano Alcañiz, Roberto Llorens, and Carolina Colomer. 2012. Use of the Wii balance board system in vestibular rehabilitation. In Proceedings of the 13th International Conference on Interacción Persona-Ordenador (INTERACCION '12). ACM, New York, NY, USA, Article 11, 4 pages.
8. Sony Computer Entertainment. 2010. PlayStation Move. Motion-sensing game controller.
9. Sung, Kelvin. 2011. Recent Videogame Console Technologies. IEEE Computer 44.2: 91-93.
10. Microsoft Corp. 2013. Kinect for Windows Human Interface Guidelines v1.7.0. Microsoft. URL http://msdn.microsoft.com/en-us/library/jj663791.aspx.
11. Catuhe, David. 2012. Programming with the Kinect for Windows Software Development Kit. O'Reilly Media, Inc.
12. Shotton, Jamie, et al. 2013. Real-time human pose recognition in parts from single depth images. Communications of the ACM 56.1 (2013): 116-124.
13. MehranAzimi. 2012. Microsoft Advanced Technology Group. Skeletal Joint Smoothing White Paper.
14. Zhou Ren, JingjingMeng, Junsong Yuan, and Zhengyou Zhang. 2011. Robust hand gesture recognition with kinect sensor. In Proceedings of the 19th ACM international conference on Multimedia (MM '11). ACM, New York, NY, USA, 759-760
15. Oikonomidis, Iason, NikolaosKyriazis, and Antonis A. Argyros. 2011. Efficient model-based 3D tracking of hand articulations using Kinect. BMVC.

16. Ludique's Kinect Bundle (LKB). 2012. Open source UI manager integrated with Kinect. URL https://code.google.com/p/lkb-kinect-bundle/
17. Kineticspace. Training, Analyzing and Recognizing 3D Gestures. 2011. URL https://code.google.com/p/kineticspace/
18. Galen Panger. 2012. Kinect in the kitchen: testing depth camera interactions in practical home environments. In *CHI '12 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '12). ACM, New York, NY, USA, 1985-1990.
19. Microsoft Corp. Xbox One. 2013. Successor video game console to the Xbox 360.