Procedia Computer Science

Volume 51, 2015, Pages 592–601

ICCS 2015 International Conference On Computational Science

Interactive 180° Rear Projection Public Relations

Ricardo Alves[1,2], Aldric Negrier[1], Luís Sousa[1,2], João M.F. Rodrigues[1,2,3], Paulo Felisberto[1,2], Miguel Gomes[4], and Paulo Bica[4]

[1] University of the Algarve, Instituto Superior de Engenharia, Faro, Portugal
aldricnegrier@gmail.com
[2] LARSyS, University of the Algarve, Faro, Portugal
{rmalves,lcsousa,jrodrig,pfelis}@ualg.pt
[3] CIAC, University of the Algarve, Faro, Portugal
[4] SPIC - Creative Solutions, Loulé, Portugal

Abstract

In the globalized world, good products may not be enough to reach potential clients if creative marketing strategies are not well delineated. Public relations are also important when it comes to capture clients attention, making the first contact between them and companies products while being persuasive enough to trust that the company has the right products to fit their needs. A virtual public relations is purposed, combining technology and a human like public relations capable of interacting with potential clients located 180 degrees in front of the installation, by using gestures and sound. Four Microsoft Kinects were used to develop the 180 degrees model for interaction, which allows tracking and recognition of gestures, sound sources, voice commands, extract the face and body of the user and track users positions (heat map).

Keywords: Visualization, Applications, Kinect, Natural Interaction

1 Introduction

Maintaining regular clients and capturing attention of new clients is one of the very first concerns a company deals with. Through well trained public relations (PR), companies reach their clients persuading that their products and tools are a major asset and can be beneficial to their business and problems. New clients are not often familiar with the company concept and have many questions on what products are available and what services can be provided to them, and normally a website can clear them up, but with many companies in the same working field, it might be difficult to excel them.

A real sized installation with stunning visual effects is more likely to capture a new client attention. There are several techniques for the projection of real size PR persons (movies) or avatars. Frontal projection is most common, but the worse solution, once the user can conceal the projection. Other techniques including Pepper's Ghost technique (see e.g. [6]), requires a lot of space. Rear projection technique using a ultra short throw projector occupies a very

 Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2015

doi:10.1016/j.procs.2015.05.327

small/limited space, and consists on a projector projecting onto a retention film, enabling a group of people to see what's projected on the other side of the retention film without them seeing the projector and with the advantage of creating an image not limited to size.

To develop the interaction with an installation, there are many sensors and cameras that can be used, nevertheless, three-dimensional (3D) sensors, such as the Microsoft Kinect [9], the Asus Xtion [2], the Leap Motion [11] or the Structure Sensor [13] are gaining popularity due to their capability of capturing Natural Interaction (NI) enabling hands-free control of devices and menus, while tracking user's skeleton and recognizing joints and gestures of one or several users. One of the most well-known 3D sensors is Microsoft Kinect, due to the game industry.

There are many applications where this sensor is used, as for instance enabling interaction with art installations [1], in robotics [5], for head pose classification [14], applied in assistive technologies, as the enhancing visual performance skills on older adults [3] or for the operation of wheelchairs [10]. More challenges and applications can be found e.g. in [4, 7, 8, 12]. Of course, interaction can be done with other 3D sensors, such as the mentioned Leap Motion, an example can be found in [6] where the interaction is done with holograms for teaching technical drawing.

In this paper a real-sized humanoid character using rear projection technique is presented. The main contribution is the developed model, which is capable of tracking users position, gestures and sound, for up to 180 degrees in form of the installation. By using 4 Microsoft Kinect sensors, all the users in front of the installation are tracked on-the-fly, and selected the user and respective sensor in which the gesture intersection will occur. In the case of the absence of a user in front of the installation, it is searched for the highest sound source, and the best suitable sensor is used to detect voice commands (words ou small sentences). Every interaction, tracking, information extracted from the users (e.g., biometric information) and requested to the installation is recorded in a database, creating statistical data (for the company analysis), such as users actions, favourite menus, etc. Although there are already a huge amount of applications that uses NI, as far as we know none have all the characteristics mentioned above.

2 Concept and Installation

As already mentioned, the main goal is to develop an interactive installation with rear projection to project a real size human figure plus menus, videos, etc. The installation consists, on present case, of a transparent acrylic (to optimize cost) with a retention film and an ultra throw short projector to minimize distance from the retention film. Figure 1 top left, shows the physical installation layout, i.e., the projector, with the projection surface in front, and the sensors in front of the surface. In the present installation the sensors were almost in the ground, but the sensor group can be moved in the vertical to any position, provided it meets the layout.

The application is divided in 4 modules: (a) Users Data, responsible for handling and manipulating information received from each individual sensor Kinect (S0 to S3), reading gestures, sound and creating statistics, (b) Global Management, responsible to convert spatial information of users from each individual sensor to a global reference and disambiguate users that were detected by multiple Kinects. (c) The Database is responsible for storing menu options and configurations (collected from the Backoffice), as well as storing information collected from (a) and (b). (d) The Interface is responsible for processing visual information, displaying menus and the virtual character. The modules (a) and (b) consists in the 180^o model for NI, the development of modules (c) and (d) is out of the focus of this paper, nevertheless they will be very briefly explained for system comprehension purpose in Section 2.3.

The interaction model, consists in 4 Microsoft Kinect sensors (not Kinect 2), to capture a total field of view of 180^o (degrees), positioned in a way that maximizes overlap areas of

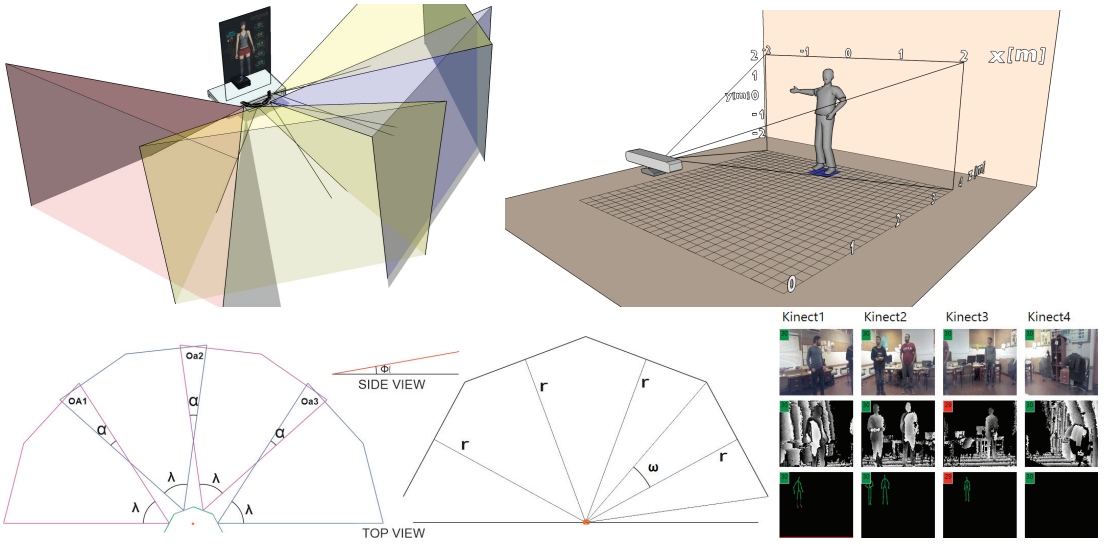

Figure 1: Left to right, and top to bottom: the physical installation layout, Kinect coordinate system, the physical coordinates and angles relative to each Kinects and a real time data streaming from 4 Kinects: RGB, depth and skeletons.

neighbour Kinects (see Fig. 1 top left, for the layout ilustration). For this, three physical properties of Kinects are needed: (a) their position $C_l(x, y, z)$ in a global reference (the red dot in Fig. 1 bottom left is the origin), with l being the Kinect index, $l = \{0, .., 3\}$, which represents Kinect positions relative to each other, (b) their horizontal rotation β_l and (c) their vertical angle ϕ_l with both representing the direction each one of them are facing. Note that the vertical angle doesn't need to be physically calculated, since Kinect SDK has methods that returns Kinect vertical angle, taking advantage of Kinect's built in accelerometer.

To determine the Kinect' position C_l it is necessary to calculate first the horizontal angle they are facing. Since the installation captures 180^o degrees, visible in Fig. 1 second left row, their intersection angle α can be calculate with $180 = 4 \times \lambda - 3 \times \alpha$, with λ being Kinect's horizontal field of view angle ($\lambda = 57^o$)[9], returning an angle $\alpha = 16^o$. The horizontal angles β_l can now be calculated, starting with the Kinect on the right and counter-clockwise, Fig. 1 left second row: $\beta_0 = \lambda/2 = 28.5^o$ and $\beta_i = \beta_{i-1} + \lambda - \alpha$, with $i = \{1, ..., l\}$, returning $\beta_1 = 69.5^o$, $\beta_2 = 110.5^o$ and $\beta_3 = 151.5^o$.

Ideally, positions $C_l(x, y, z)$ would all be the same for all Kinects but this is not possible, since Kinect has a width of $w_{KS} = 28$ cm (centimetres) [9], making it impossible to be placed all in the same point. Due to this, all Kinects need to be distanced of this point by a distance of r in the direction of the angle they are facing, visible on Fig. 1 left second row, and can be calculated with $r = w_{KS}/(2 \times \tan(\omega))$ and $\omega = (\beta_1 - \beta_0)/2$, giving an $r \approx 36$ cm (centimetres). The positions C_l can be calculated with $C_l(x, y, z) = (r \times cos(\beta_l), 0, r \times \sin(\beta_l))$, thus $C_0(x, y, z) \approx (32, 0, 17)$ cm, $C_1(x, y, z) \approx (12, 0, 34)$ cm, $C_2(x, y, z) \approx (-12, 0, 34)$ cm and $C_3(x, y, z) \approx (-32, 0, 17)$ cm. Having the physical positions of the Kinect sensors, it is now possible to compute the remaining components of the NI model.

2.1 Users Data Module

As stated in the previous Section, this module is responsible for handling and manipulating received data from each single Kinect, with all steps described in this Section being replicated

for each of the 4 Kinect sensors used. Each sensor provides sound extraction, colour (RGB) and depth frames, 25 joints of 2 users and can track up to 6 users. The above data is manipulated in 4 main phases: (a) spatial information, (b) gestures recognition, (c) body and face extraction and (d) sound extraction.

2.1.1 Spatial Information

Spatial Information (x, y, z) of user's joints including the global position (P), are stored using two FIFO (First In, First Out) lists, one for the current (detected) users (Pc_l) and another for lost users (Pl_l) (as mentioned, l is the Kinect number). Every time the Kinect has new information available about users, 3 different situations may occur: (i) *new users are available*, they are added to the end of the current users list (Pc_l) alongside all joints informations and entry time (t) and assigned an internal ID, which is then incremented, but (ii) *if the user already exists*, then the new information is stored with the previously obtained information on the current users list, alongside the entry time of the new information. (iii) *If the user is lost*, then all information is updated in the database and the user is discarded from the current users list and added to the lost users list.

2.1.2 Gestures Recognition

Gestures recognition is responsible for ("the visual") inputs from the users. After experimenting with different gestures, the "swipe gesture" and the "pose gesture" were chosen and implemented due to their intuitive nature for the users. Figure 2 left, illustrates those gestures, in the left the "pose" and on the right side of the image the "swipe". These gestures were chosen to be somewhat intuitive while not being too frustrating.

For the ***swipe gesture*** implementation, given minimum swipe distance d_s and the minimum swipe velocity v_s, a time window Δt_s can be calculated with $\Delta t_s = d_s/v_s$. By analyzing only user information acquired from current instance t to $t - \Delta t_s$, a swipe was made if in any other sub-interval Δt_k, defined between t and $t - \Delta t_s$, with k the latest instance of Δt_k, all hand positions (h) minus its previous hand position, taking only into account the x component for the horizontal swipe, or the y component for the vertical swipe, has the same signal along the whole interval Δt_k, $(|\sum_{j=k-\Delta t_k}^{k-1} \mathrm{sgn}(h_{j+1,\{x/y\}} - h_{j,\{x/y\}})| = k-1)$, and if the total distance travelled, taking only into account the x/y component respectively for the horizontal and vertical swipe, between the first and the last point of that sequence is greater or equals to d_s, that is, $d_s \geq |h_{k,\{x/y\}} - h_{k-\Delta t_k,\{x/y\}}|$. Figure 2 second column, illustrates an horizontal swipe. The minimum distance used was $d_s = 30$ cm and the minimum speed used was $v_s = 200$ cm/s.

The ***pose gesture*** can be detected using the vector defined by the user's body $\overrightarrow{V_b} = (x_b, y_b, z_b)$, which is computed by subtracting the *shoulder centre* [9] position, P_{sc}, from the *spine* position, P_s, and using a vector defined by the arm, $\overrightarrow{V_a} = (x_a, y_a, z_a)$, computed by subtracting the *hand (right/left)* position, P_h, from the *elbow (right/left)* position, P_e. The user is performing the menu pose if the two following conditions are true: (1) As the user must be facing the Kinect, the distance between the *elbow* and the Kinect must be approximately the same as the *hand* and the Kinect. For this reason, only if $|z_h - z_e| \geq 0.6 \times \mathrm{d}(P_h, P_e)$ is considered that the user is performing the gesture. (2) An angle θ between $\overrightarrow{V_b}$ and $\overrightarrow{V_a}$ can be measured with $\theta = acos((\overrightarrow{V_b} \cdot \overrightarrow{V_a})/(|\overrightarrow{V_b}||\overrightarrow{V_a}|))$. If the angle is $20^o \leq \theta \leq 160^o$, than the user is doing the pose gesture. To increase the reliability of the pose gesture, it is verified if at least 85% out of the previous detections made in the past 1 s (second) are according to the above conditions. If the user is performing the pose gesture, it is considered - up - if the angle θ is less than 90^o, and - down - otherwise.

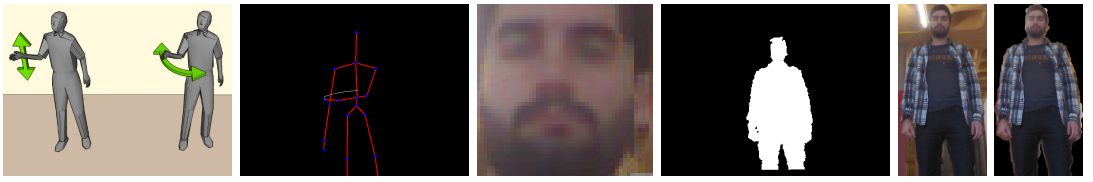

Figure 2: From left to right, the "swipe" and "pose" gestures, a "swipe" gesture being performed, as viewed from a single Kinect sensor, the extracted face of current user, the boolean mask of detected user, the cropped image of a user and the extracted user with no background.

2.1.3 Face and Body Extraction

Face and Body extraction is done and shown to the user as soon as he begins to interact with the installation. While the Kinect SDK does all of the job extracting the user face, Fig. 2 third column, full body photo is not done directly. Every time a new depth frame is available, the first three bits represent which user that pixel belongs to, from 1 to 6 and with 0 meaning it belongs to no user. A boolean mask for each user is constructed, visible in Fig. 2 fourth column. For that image the value for the highest and lowest $x_{min/max}$ and $y_{min/max}$ is found. The colour image is then cropped (U) using this points. An example can be seen in Fig. 2 fifth column. Alternatively a Gaussian filter is applied to the boolean mask ($\sigma = 2$), and again the image is cropped and a bitwise AND logic is performed with U, obtaining Fig. 2 right. These images (face and body) are used to insert the user in different backgrounds/situations and then offered to the user as "promotional gifts".

2.1.4 Sound Extraction

The sound extraction for each sensor is done using the Kinect SDK, each Kinect is equipped with a built in microphone array, composed by 4 microphones. The microphone array is capable of sound localization also called beam forming, that is the ability to estimate the audio source direction by calculating differences between audio streams captured. The beam angle range covers 100^o in front of the Kinect, 50^o from centre to each side [9]. The beam direction is an integer value multiple of 10, in all containing 11 different values. These values represent sound source direction, in addition the Kinect SDK estimates the confidence level that the audio source was located, represented by a value between 0 and 1. Speech recognition is also supported by the API, having only to provide words or small sentences that will be recognized.

2.2 Global Management Module

Global Management module is responsible for managing users and sound detected by each Kinect, creating a global reference, selecting the user that is interacting (gesture/sound), and solving the problem where a user is detected by two neighbour Kinects due to overlap on their field of view. The Global Management works similar to Users Data Module, Sec. 2.1.1, consisting in two FIFO lists, one for current detected users Gc and other for lost users Gl.

2.2.1 Global Reference Conversion

Once all the Users Data modules are updated, all users from Pc_l, are transformed to a global reference and put into an array of potential users Uc. This conversion transforms the coordinates returned by Kinect (Fig. 1 top right), to a global reference, visible in Fig. 3 left first row, being the lighter area the interaction zone with a radius of 4 m (meters) counting from the centre of the Kinects setup. Each user is represented by a circle, occupying 50 cm. To convert a Kinect coordinate (x axis is represented from left to right, the y axis from up to down and the z axis the depth distance, Fig. 1) to global reference, the physical layout of all Kinect cameras are

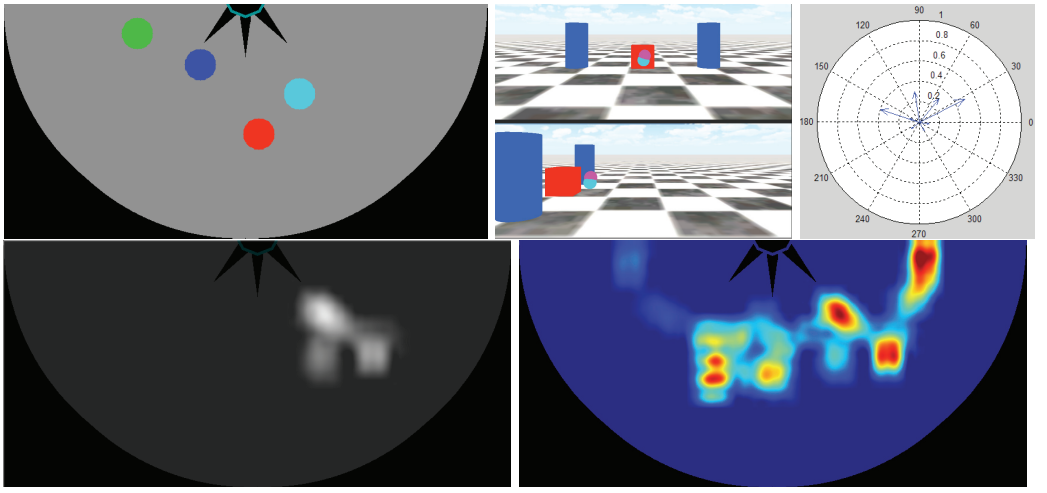

Figure 3: Top to bottom, left to right: an aerial view with 4 detected users, 2 different 3D views of the same user being detected by two different Kinects. A real time sound source beams from all 4 Kinects, in a polar plot, and a Heat Map of a single user, and the Heat Map from multiple user, on JET colour map.

necessary, keeping in mind their distance between each other, as well as their rotation in the horizontal plane β_l and rotation on the vertical plane ϕ_l, see Section 2.

The first step consists in finding the directive unitary vectors of each Kinect sensor, one for each axis: $\overrightarrow{V_a}$, representing the vector of the axis a Kinect is facing, $\overrightarrow{V_b}$, representing the axis with an horizontal 90^o to $\overrightarrow{V_a}$ and $\overrightarrow{V_c}$, representing the third axis with 90^o degrees to both $\overrightarrow{V_a}$ and $\overrightarrow{V_b}$ pointing upwards in relation to Kinect sensor. $\overrightarrow{V_a}$ can be found with $\overrightarrow{V_a}(x,y,z) = (\cos(\beta_l) \times \sin(\frac{\pi}{2}+\phi_l), \cos(\frac{\pi}{2}+\phi_l), \sin(\beta_l) \times \sin(\frac{\pi}{2}+\phi_l))$ and $\overrightarrow{V_b}$ being normal in the horizontal plane to $\overrightarrow{V_a}$ and calculated as $\overrightarrow{V_b}(x,y,z) = (\cos(\beta_l+\frac{\pi}{2}) \times \sin(\frac{\pi}{2}+\phi_l), \cos(\frac{\pi}{2}+\phi_l), \sin(\beta_l+\frac{\pi}{2}) \times \sin(\frac{\pi}{2}+\phi_l))$. Finally, $\overrightarrow{V_c}$ is calculates with $\overrightarrow{V_c}(x,y,z) = (\cos(\beta_l) \times \sin(\phi_l), \cos(\phi_l), \sin(\beta_l) \times \sin(\phi_l))$. Given that the Kinect SDK gives the coordinates in meters $\hat{J}$, a joint position J of a user made by Kinect l in centimetres is calculated with by $J_{x,y,z} = \hat{J}_{x,y,z} \times 100$. The new position in the global reference Jg can now be calculated with $Jg(x,y,z) = (J_x \times \overrightarrow{V_b} + J_y \times \overrightarrow{V_c} + J_z \times \overrightarrow{V_a} + C_l)/100$ m, with C_l being the global position of the Kinect. Now Jg can represent the conversion of any joint or a user global position from any sensor to the global reference, by using this transformation we denote Pg as the global position of the user in the global reference. All users are then mapped on the same reference, with an aerial view visible in Fig. 3 top left.

2.2.2 Recognition of Multiple Kinects Detections of an User

Since all Kinects overlap their field of view with their neighbours (see Fig. 1 top left), a method was developed to determine if a user was detected by multiple (2) Kinects. All potential users on Uc are compared with each other to find if they might be the same user, using two criteria: (a) (a.1) if the euclidean distance between global positions of the two users are compared in Uc, for instance Pg_1 and Pg_2, is less then 50 cm (centimetres) and (a.2) the two users were detected by different Kinects, then (a.3) if the last T skeletons information are available, with $T = 5$, it is compared if the global position variation on all axis (x,y,z) are similar: $(|Pg_{1/2,x}(\delta) - Pg_{1/2,x}(\delta)| < 15\text{cm}) \wedge (|Pg_{1/2,y}(\delta) - Pg_{1/2,y}(\delta)| < 15\text{cm}) \wedge (|Pg_{1/2,z}(\delta) - Pg_{1/2,z}(\delta)| < 15\text{cm})$, with $\delta = \{t, ..., t-T\}$, then it is assumed that both users are in fact the same user. Finally, (a.4) if the relation between the users was found, the newest user will be forced to change his/her

internal ID to the internal ID of the oldest detected user.

(b) In case the two users already share the same internal ID, then they are automatically related as the same user meaning that the relation was already found in a previous situation by method (a). In both (a) and (b) cases, the user closer to its Kinect will be marked for addition to the end of the list Gc, with the other discarded of that list. Figure 3 top middle illustrates, using 2 views, the situation where two sensors detected the same user, with both converted to the global reference, the solids in the image were only used to get a better comprehension of the 3D space.

2.2.3 Updating, Adding and Removing Users

Once the previous step (Sec. 2.2.2) is complete, all marked users will be added to the current users list Gc, with a detected user being either completely new to this list or the user being new information of an already existing user of Gc. For all users marked for addition on Uc, with the user being u_i and i ranging from 0 to the size of array Uc, are compared to each of the users already on Gc, with the user being gc_j and j ranging from 0 to the size of list Gc. If both users u_i and gc_j share the same internal ID, then the user gc_j will be updated with the last skeleton (on global reference) of the user u_i. In case user u_i does not share the same internal ID with any user of Gc then the user u_i is new and is added to the end of the list Gc. On the other hand, if a user gc_j is not sharing the same internal ID with any marked user of Uc, then user gc_j is discarded from Gc list and added to the end of the list Gl.

Since users can sometimes block the Kinect's view, a simple occlusion detection was built to recover a lost user. When a new user is detected, it is verified if, in the lost users list Gl, there are lost users less than Δt seconds ago. For all of the last positions of all lost users (k) in the past Δt seconds, Gl_k, the current position of a detected user (u), Gc_u, is considered to belong to a previous user if the Euclidean distance $\mathrm{d}_{i,u}$ between Gl_i and Gc_u is less than 50 cm. If more than one lost user is closer than 50 cm to the position P_u then the closest distance d is chosen, recovering all information to the current users list Gc (it was used $\Delta t = 5$ s).

The *selected user* to interact with the installation is the one in the Gc list that is closer to the installation, and remains the *selected user* until this leaves the 180^o space analysed by the system. If the user leaves the space or does not have any type of movement during 30 s then the *selected user* is the next in the Gc list closest to the installation.

2.2.4 180^o Heat Map

One important feature of the model is the Heat Map of users, very useful for statistics about the most active locations (and time spend at that location) of a user or a group of users. By using the users global position (Pg) (see Sect. 2.2.1) the Heat Map can be calculated. In this application it is assumed that the physical limits of the Kinect is approximately 4 m in length and in width [9] (in fact it is a bit less), and thus the heat map generated represents a region of approximately 8×4 m. In reality it is more or less a circle of 4 m radius considered from the middle of the Kinect group (see Figs 1 and 2 top left). A matrix HM with size of $2N \times N$ can be created, dividing the map in squares of approximately $(4/N) \times (4/N)$ m. In the present case, it was considered $N = 26$ px (pixels), consequently each square as the area of 0.024 m^2. A user's position obtained after coordinate transformation (Sec. 2.2.1) on instance t is mapped to matrix HM using $(x_t, y_t) = (-x_{k_t} \times (N/4) + N, z_{k_t} \times (N/4))$, with x_k and z_k the coordinates using the global reference. Starting with every position of the matrix HM equals zero, every detection made at instance t increments its value, as well as its 8 neighbours: $M(x_t + i, y_t + j) = M(x_{t-1} + i, y_{y-1} + j) + 1$ with i and $j = \{-1, 0, 1\}$. After this step, the matrix M holds values of the positions of a user, or a group of users (optional). The matrix is then normalized: having the highest value h_m of matrix HM with $h_m = max(M(x, y))$, a grey

scale image H_g is calculated with $H_g(x, y) = (\tau/h_m) \times HM(x, y)$, with $\tau = 255$ in this case, visible on the bottom left row for the case of 1 user Fig. 3. Finally, the map can be converted to JET colour map, visible on the right (same figure) in this case with several users (4) moving during 5 minutes.

2.2.5 Sound

Interaction with the installation thought sound is still possible even if a user is not detected/present in front of it, the installation is capable of detecting the sound source angles and interacting with users via sound. As mentioned in Section 2.1.4, speech recognition can be done by a single Kinect, however in a 4 Kinect configuration (180^o) we have a total of 16 microphones (4×4), so the Kinect that is most frontal to the sound source location must be selected. A single Kinect can determine the audio source direction from within 100^o range in front of itself. In a 4 Kinect configuration, where the sensors are positioned side by side and misaligned from each other, the total audio source direction range is calculated by adding 100^o from each angle range from each Kinect minus the 3 overlapped audio source direction (Oa) (see Fig. 1 bottom left) created by the close position of the Kinect from each other (see Section 2). In this 4 Kinect configuration the total audio source direction is $Sd \approx 223^o$ ($Sd = 4 \times 100^o - 3 \times Oa$), which is greater then the field of view of the combined Kinects RGB and depth data of 180^o degrees. So in order to detect sound from within the 180^o range, a transformation was applied to all audio beam angles, transforming into 180^o, $\varphi_t = \varphi_o \times 180/Sd$, where φ_t represents the transformed and φ_o the original audio beam angles. This transformation will distorts the actual location from the sound being captured, however because we just want to select witch Kinect will be responsible for handling speech recognition this distortion will not influence this selection. A different possible solution would be to clip all information/sound that comes from audio source inside the range $[-(Sd - 180)/2, 0]$ and $[180, 180 + (Sd - 180)/2]$ degrees.

When a user is interacting with the system using their voice or when environmental sound is present, to determine witch Kinect will handle speech recognition the follow algorithm is applied: (a) Capture the sound beam and confidence levels from all 4 Kinect sensors. (b) Sum and store the result of the individual audio beam angles multiplied by the respective confidence levels. (c) In order to filter out isolate sound locations, step b) is repeated several times (it was used 10). (d) In the end of step c) (all repetitions/sums) is determined which of the 4 Kinects has the greatest sound source locations times confidence level stored. (e) In case of a tie or close tie (less than 5% of the highest value), the chosen value will be the one that has the sound beam closest to 0^o (corresponding to the most frontal sound in respect to a Kinect position). (f) Activate Kinect speech recognition from the selected Kinect correspondent to the maximum value calculated.

Figure 3 top right shows real time sound source beams from all 4 Kinects, in a polar plot. In this case the Kinect selected is the right most one. As mentioned, the Kinect works by analysing statically stored keywords that can be detected with a degree of certainty every time a user speaks the keywords. When a keyword is identified and confirmed the application triggers a response using a stored answer (using an implemented text-to-speck functionality) that are replayed using a synthesized voice. The keywords can also trigger actions similar to the gesture interaction. If there is sound and no word is recognized in the followed 30 s, then the installation return a audio personalized message to call the attention to itself (this is only done if no user is interacting using gestures).

2.3 Database and Interface Module

The database module stores two types of information: (i) menus configurations and (ii) statistics. The Interface module is responsible for reading information stored in the database and

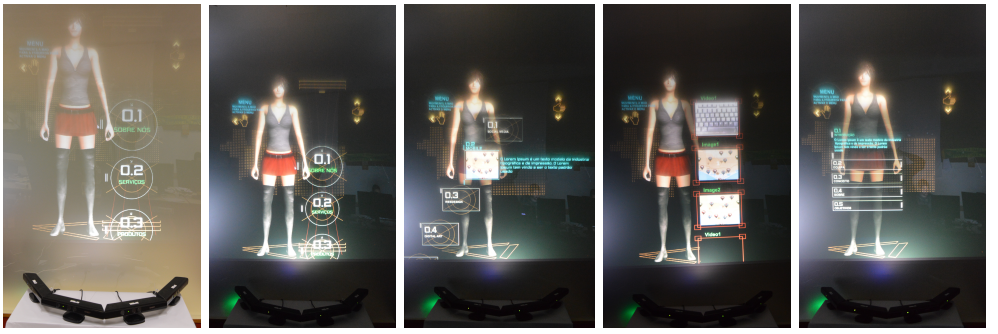

Figure 4: Several examples of the system interface (see text).

automatically generate layouts, as well as generating responses to the user input. In resume, the application can generate the following layouts: (a) titles of menu contents useful to serve as bridge to other menus, (b) similar to a), a layout that can also fit small description if needed. In specific cases, (c) media content as image or video with description is useful to give a user the idea beyond a concept, generating a layout similar to b). Alternatively, the same layout can be used but displayed in e.g. diagonal instead of vertical. (d) One final option is available to display only images or video. All configurations are loaded when the application starts. The avatar or video trigger responses to inputs of the user: (a) If the interacting user with the installation changes, the avatar/video waves while showing that user's photo. (b) If the user selects an option, the avatar/video touches it's content triggering a menu change. All interaction, time spend interaction with the menus, request asked to the installation, etc. are saved in the database. Figure 4 shows some examples of the prototype installation.

3 Discussion

In this paper an interactive installation was presented, based on a model for natural interaction. An avatar work's as a public relations giving the first contact between a company and a potential client, capable of recognizing sound sources, voice commands and interacting with sound and gestures. The fully customization of the application is a major asset, which can also extract statistics about users reached, positions at any instance of time, the most requested contents as well as time spent by users in each of the contents. The installation works in 180^o environment, recognizing up to 24 users, with a maximum of 8 users fully tracked. When a user is interacting with the installation and gestures' joints are tracked, since Kinect returns the three dimensional position of the joints relative to it, the distance of the joints to Kinect, as well as more users on the interaction zone, does not seem to interfere with the success rate at the specified distance and speed parameters (see Sec. 2.1.2). The results of swipe gestures were good, as previous hand positions were taken into account. The detected successful rate was 98%, being too difficult to tell exactly what did fail. Usually it was due to the user not making the gesture correctly, due to a lack of speed or gesture extend (distance). Users could navigate easily through the menus, although some users were not familiar with the pose gesture. The results of recognition of multiple detections of a user showed also good results, even in complex environments (e.g. Fig. 1 bottom right). In about 45 minutes of continuous with multiples users entering and leaving the field-of-view of the installation two conclusions were taken: (a) If the user progressively changes his/her position and is facing the installation then no errors (0) occur, but (b) if the user either changes abruptly (e.g. run) his/her position or is not facing the centre of the installation, then the results drastically decrease. These errors are due majorly to the overlap area of two Kinects being small, in case of an abruptly change of position, and due to the limitations of Kinect not

detecting well users when they are sideways to it, situation that cannot be controlled. On the other hand, results on recovering users were satisfactory. If a Kinect is obstructed and a user does not move, then it works as expected, as long as the obstruction doesn't take to long, but if it is not the case, i.e., the user moves to a different location, the results are not reliable. Also, if another user switch place with the obstructed user, then he/she will be recognized as the user they switch places with.

Future work includes improving obstruction and overlap problems, using biometric information, such as distance between joints, and face recognition (when possible) to minimize the problem where two users switch places and, if the results are good enough, use the same landmarks to try and search for an obstructed user on a predicted area based on his past movements.

Acknowledgements. This work was supported by projects PRHOLO QREN I&DT, nr. 33845, POPH, FEDER and FCT, LARSyS (UID/EEA/50009/2013) and CIAC (PEstOE/EAT/UI4019/2013). We also thank our project leader SPIC - Creative Solutions [www.spic.pt].

References

[1] Ricardo Alves, Marisa Madeira, Jorge Ferrer, Susana Costa, Daniel Lopes, Bruno Mendes da Silva, Luis Sousa, Jaime Martins, and João Rodrigues. Fátima revisites: an interactive installation. In *Int. Multidisciplinary Scientific Conf. on Social Sciences and Arts*, pages 141–148. SGEM, 2014.

[2] Asus. Asus Xtion Pro. `http://goo.gl/HxQcli`, 2014. Retrived: Nov. 10, 2014.

[3] I-Tsun Chiang, Jong-Chang Tsai, and Shang-Ti Chen. Using Xbox 360 Kinect games on enhancing visual performance skills on institutionalized older adults with wheelchairs. In *IEEE Int. Conf. on Digital Game and Intelligent Toy Enhanced Learning*, pages 263–267, 2012.

[4] Leandro Cruz, Djalma Lucio, and Luiz Velho. Kinect and RGBD images: Challenges and applications. In *IEEE Conf. on Graphics, Patterns and Images Tutorials*, pages 36–49, 2012.

[5] Riyad A El-laithy, Jidong Huang, and Micheal Yeh. Study on the use of Microsoft Kinect for robotics applications. In *IEEE/ION Position Location and Nav. Symp.*, pages 1280–1288, 2012.

[6] Mauro Figueiredo, Luis Sousa, Pedro Cardoso, João Rodrigues, Cesar Goncalves, and Ricardo Alves. Learning technical drawing with augmented reality and holograms. In *Int. Conf. on Education and Educational Technology*, pages 11–20, 2014.

[7] Simon Fong, Yan Zhuang, Iztok Fister, and Iztok Fister Jr. A biometric authentication model using hand gesture images. *Biomedical engineering online*, 12(1):111, 2013.

[8] Takuya Kamizono, Hiromichi Abe, Kensuke Baba, Shigeru Takano, and Kazuaki Murakami. Towards activity recognition of learners by Kinect. In *IEEE 3rd Int. Conf. on Advanced Applied Informatics*, pages 177–180, 2014.

[9] Kinect. Kinect for Windows. `http://goo.gl/fGZT8X`, 2014. Retrived: Nov. 10, 2014.

[10] Farid Kondori, Shahrouz Yousefi, Li Liu, and Haibo Li. Head operated electric wheelchair. In *IEEE Southwest Symp. on Image Analysis and Interpretation*, pages 53–56, 2014.

[11] Leap. Leap motion. `https://www.leapmotion.com/`, 2014. Retrived: Nov. 10, 2014.

[12] Mohammad Rahman, Bruce Poon, and Hong Yan Ashraful Amin. Support system using Microsoft Kinect and mobile phone for daily activity of visually impaired. In *Transactions on Engineering Technologies*, pages 425–440. Springer, 2015.

[13] Struture. Struture sensor. `http://structure.io/`, 2014. Retrived: Nov. 10, 2014.

[14] Yixiao Yun, Mohamed H Changrampadi, and Irene YH Gu. Head pose classification by multi-class AdaBoost with fusion of RGB and depth images. In *IEEE Int. Conf. on Signal Processing and Integrated Networks*, pages 174–177, 2014.