Human-Robot Interaction

Elective in Artificial Intelligence

Kevin Munda

Academic year 2020/2021

Professor Luca Iocchi

1 Introduction

Nowadays, with the rapid growth of the traditional industries, intelligent robots and automated systems are widely used in the hospitality industry. In particular the application developed in this project is meant to be used in the catering sector, for example inside a restaurant. The idea behind this project is to automate and personalize the experience of the user inside a restaurant with the help of a robot; in particular the robot used for the development of the application is a simulated version of the known robot "Pepper". Here are briefly defined the main developed functionalities: Pepper, under the will of the user, can register in its database a picture of him and his personal information; if the user is correctly registered, Pepper can perform a recognition through a face-recognition model, in order to provide to him a personalized service based on the registered data; regardless of the fact that the user is registered or not, Pepper has an interface to show the menù and collect the order of the customer, but in this case for registered users there is an additional functionality that allows to speed up the order; the last interface of the application offers a funny way to interact with Pepper through an imitation game performed by the robot. This project wants to show how some operations inside a restaurant can be integrated and personalized through a robotic system, with the advantage of a much more fast and personalized service.

2 Related Work

The main inspiration for this project was taken from the work and results obtained in [1]. In this paper the authors designed and developed a prototype of a robot based on a customer maturity model and then they tested their model with a real robot in a real restaurant. Starting from the Customer's mature model of Sitecore, a customer experience management company, the researchers defined their own customer mature model called "intelligent Robot for Customer eXperience Maturity (iRCXM)" model for hospitality. This model helps the robot in the evaluation of the stage of the customer: the aim is to transform a new customer into a lifetime customer according to the transition conditions of each stage to enhance the user experience. In [1] a new framework was developed composed by four modules: the previously mentioned iRCXM, service process task division (entertain, service, delivery, payment, service feedback, ...), responsibility allocation (payment system management, service robot management, ...) and customer data protection technology (network data protection, customer critical data protection, ...). Grounded on this framework, the first intelligent Service Robotic Waitress in Wales, namely Robot EURKEA Gen-1 is designed and piloted. The robot has been worked in a coffee shop called "Mrs Jones Cater" in Cardiff and interviewed customers who had experienced the robot services. The robot had two main user interaction interfaces: in the welcome interface, the user can control the robot through the buttons on the touch screen; in the face recognition interface, new customers can choose to add

their faces to the face database and enter names. Old customers can choose to perform face recognition directly. These functionalities inspired me for my project in which i tried to implement interfaces for registration and recognition of customers, adding the possibility to them to order directly through Pepper and a particular feature for registered customers.

3 Architecture and Tools

The application was developed using Python, HTML/CSS and Javascript, and is meant to be used inside a Docker container, in which there are all the necessary libraries used to interact with Pepper, in particular the NAOQi module. The libraries used to develop the face registration and face recognition features are *OpenCV*, *dlib* and *face recognition* while the main library used to develop the browser application that simulates Pepper's tablet is *Flask*. The application is started launching the file *main.py* which launchs in parallel the *Flask* application on the browser. These two applications communicate using a shared dictionary of flags, where each flag defines a specific condition in the execution of the user's request. An illustrative scheme of the architecture of the application is showed in Fig. 1.

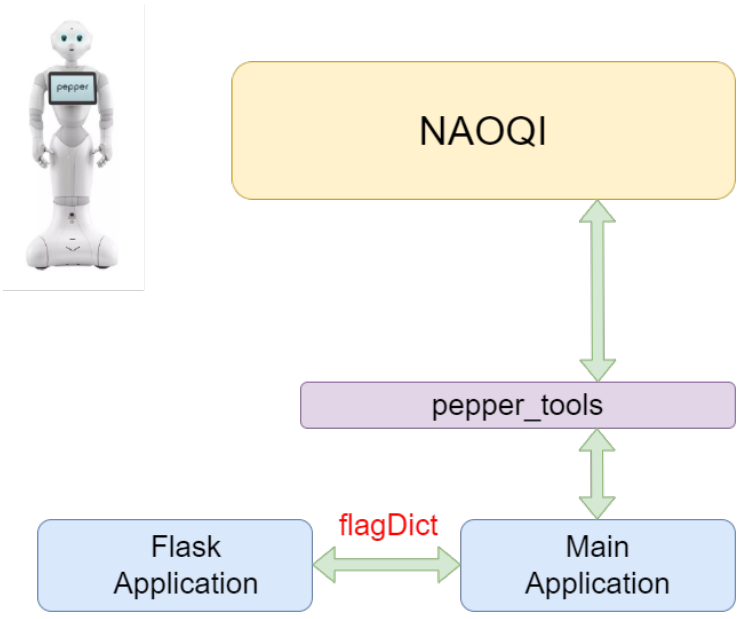


Figure 1: Architecture of the application

4 Functionalities

In this section are defined and explained in detail all the functionalities offered by Pepper to the user. The application has four different services:

- User Registration
- User Recognition
- Take an order
- Play a game with Pepper

At the beginning Pepper welcomes the customer and waits for him to click the "Interaction" button in the browser page that simulates Pepper's tablet, as we can see in Fig. 2.

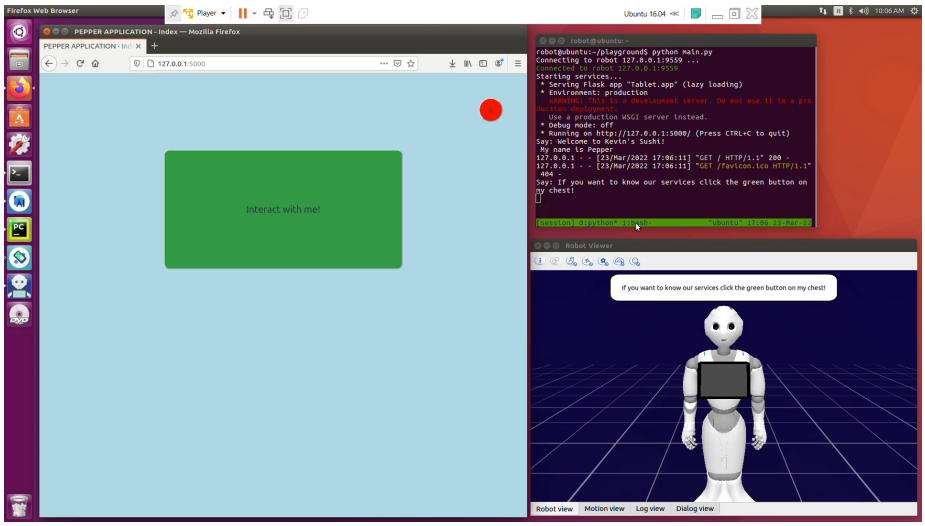


Figure 2: First Interaction

When the button is clicked, the user can choose which kind of service he wants to request from the interface showed in Fig. 3. In this interface the user can request the desired service either by clicking the relative button on the screen or by requesting it by voice. In this case, due to the fact that the application was developed using a simulated robot, the voice of an hypothetical customer is simulated by a Python script in the *pepper tools* folder. If the voice request is not understood, Pepper asks to repeat it. When a a service is selected with one of the two explained modalities, Pepper proceeds with the elaboration of the request.

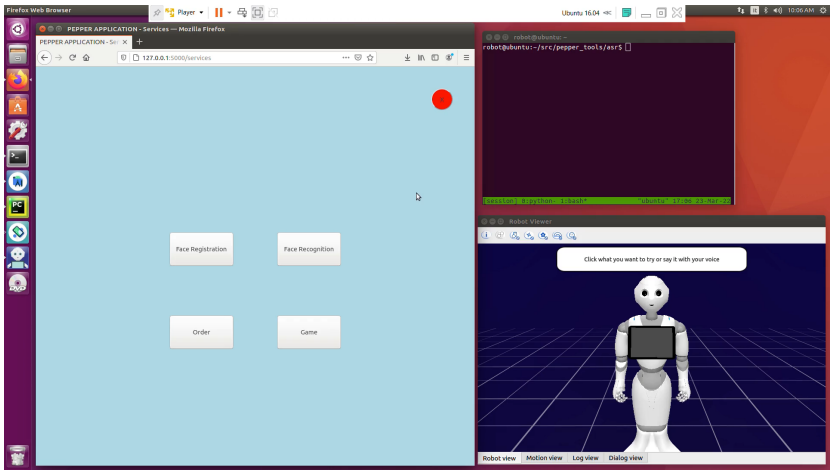


Figure 3: Services menu

4.1 User Registration

In order to give a personalized experience to the customers of the restaurant, Pepper has its own database of registered users. In this project, for the sake of simplicity, the database was simply implemented using a text file where each line corresponds to the data of a single user. In particular data are stored in this way: *NAME # SURNAME # PHOTO PATH # LAST ORDER*. The first three fields will be filled at the end of the registration phase, when the user confirms all the data inserted, while the last one is used to store and eventually recover the last order made by the user. At the end of the registration, it will be initialized as "None" and updated when the user makes his first order. At the beginning of this phase Pepper asks to take a photo of the user, as showed in Fig. 4. We can see that in the interface now there is a video stream of what Pepper is seeing with its camera (in this case the camera of the laptop). On the right side there is the button "Take Photo" through which the user can take a photo of himself. When the button is clicked, the photo taken is showed below the video stream, as we can see from Fig. 5, and Pepper asks to confirm this photo or take another one. At this point a new button appears, the "Confirm Photo" button, which allows the user to confirm the current picture, that will be stored inside the database and linked to the user's profile through the *PHOTO PATH* field. If the user doesn't like the photo taken, he can click again the button "Take Photo" as many times as he wishes until he likes the picture taken and confirms it. Moreover, in the upper right corner of the browser page, there is another button, the "Return to services" button that ends the current operation and go back to the interface in Fig. 3. This button is inserted inside the interface of each of the four services offered by Pepper; in this way the user can end whenever he wants the current operation.

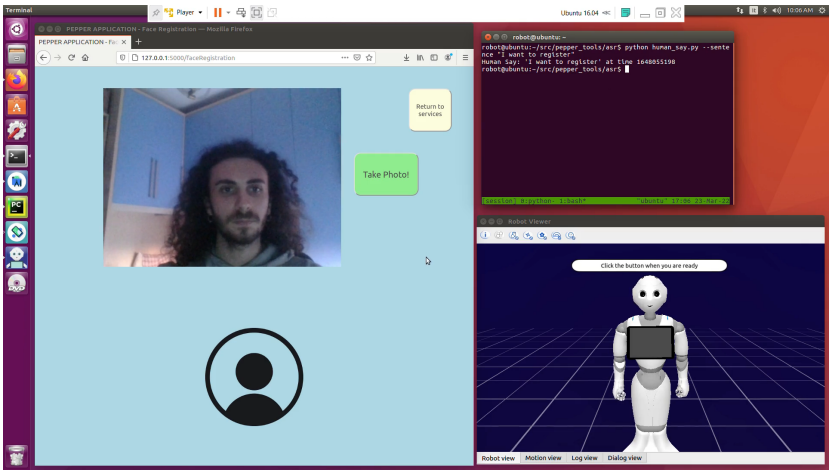


Figure 4: Pepper takes a photo of the user

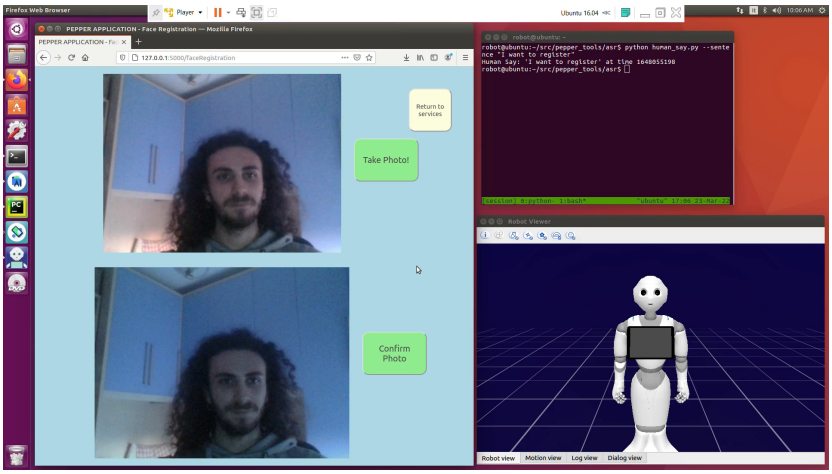


Figure 5: Taken photo of the user

Pepper waits the user choices and when the "Confirm Photo" button is clicked, the next phase of the registration starts, leading to the interface showed in Fig. 6, where the confirmed photo is showed to the user inside a box with two field to be filled. In this second phase, Pepper asks the personal information of the customer, in particular the name and the surname. This should be given by voice one at a time. First the name is requested and when the user responds, Pepper repeats the listened name and asks to confirm it saying "Yes", in case of correct information, or "No", in case something is wrong. If the answer is "Yes", the confirmed name is showed interactively in the respective field in the browser page and Pepper continues the registration asking the surname; if the

answer is "No", Pepper asks the user to repeat again his name. The same operations are repeated when the robot collects the surname and when even this last information is confirmed a new button appears in the interface, showed in Fig. 7.

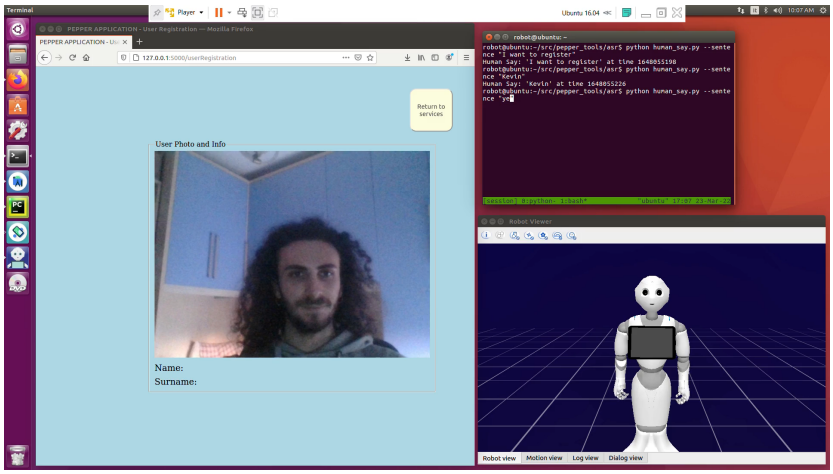


Figure 6: Pepper requests user data

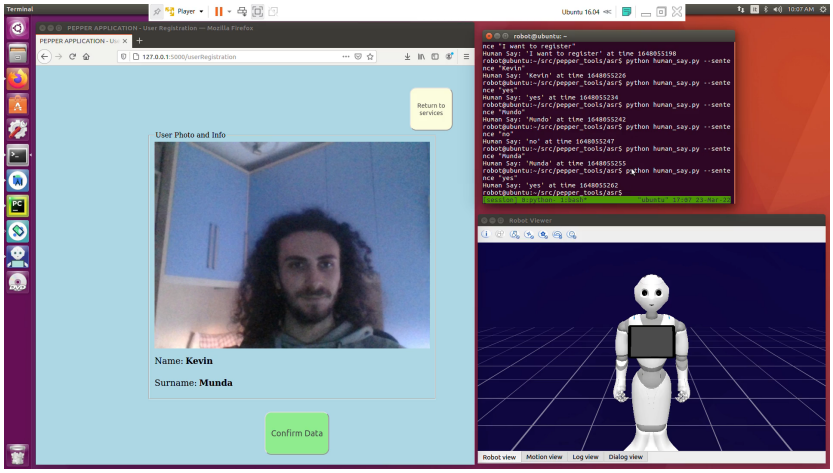


Figure 7: Confirm data

At this point the user should click the "Confirm Data" button to end the registration. When the button is clicked, the data inserted are considered as definitive and a new entry in the database is created with these information. At the end of this operation, Pepper notifies the user that the registration is completed successfully and now the user can choose to try another service.

4.2 User Recognition

The interface that the user will see for this functionality is showed in Fig. 8. There is again a video stream but this one is a bit different from the previous one. In fact, in this case, each frame is not just showed in the browser page, but it's also analysed to perform a face-recognition. This feature is realized through a machine learning algorithm, implemented with a model and functions taken from the libraries *dlib* and *face recognition*.

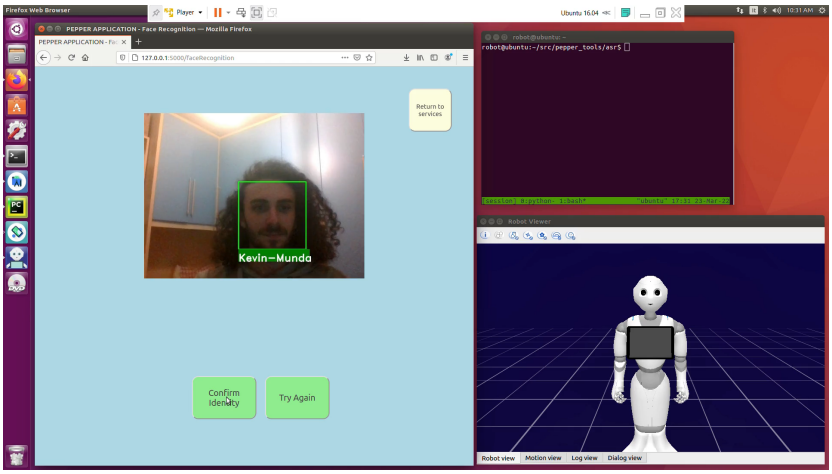


Figure 8: User recognition interface

Before the recognition starts, a database of face embeddings is created: for each user registered in the database of the robot, a function extrapolates the face embedding from the user's photo and associates it to the user's identity. When this face-database is ready, the recognition starts using the current video stream: each frame is analysed by the previous function to take the face embedding of the current photo; then this data is compared to the ones stored inside the face-database with a function that computes distances between the stored information and the current one. If one of the computed distances is below a threshold, specified in advance, the algorithm has found a match with one of the photo inside the face-database and returns the identity of the match, showed in the video stream, under the square that highlights the face of the user. Moreover, Pepper calls the user by the name and surname retrieved with the algorithm and asks him to confirm the found identity, in case is correct, or to try again, if the identity doesn't corresponds. In this last case, the user can click the "Try Again" button to run again the algorithm, and Pepper will search for another match inside its database. If no match is found, in the video stream will be showed "No-Match" and the user cannot complete the recognition operation. Obviously this circumstance happens only if the user is not registered. If the match is correct, the user can confirm his identity clicking the button "Confirm Identity", which ends the recognition phase and goes back to the services menu,

showing the name of the user, as in Fig. 9.

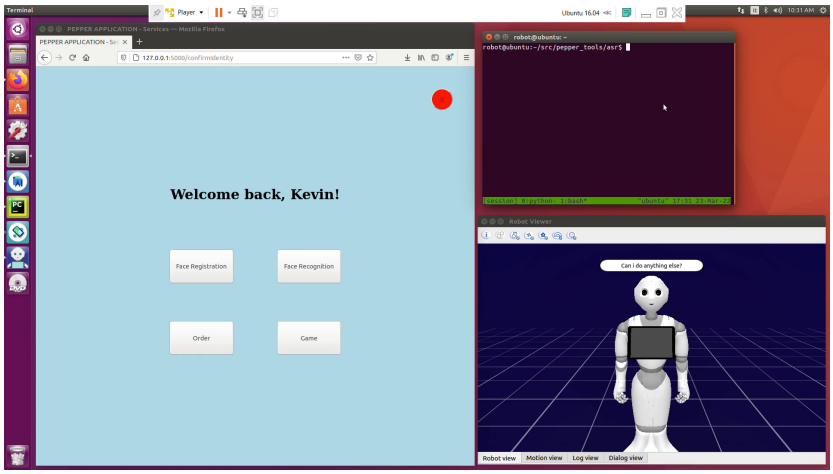


Figure 9: Services menu when the user is recognized

4.3 Take an order

Regardless if the user is registered/recognized or not, the order functionality is available for everyone. In this interface, showed in Fig. 10, the user sees the current menù of the restaurant and has the possibility of choosing what he wants to eat just by selecting the desired dishes.

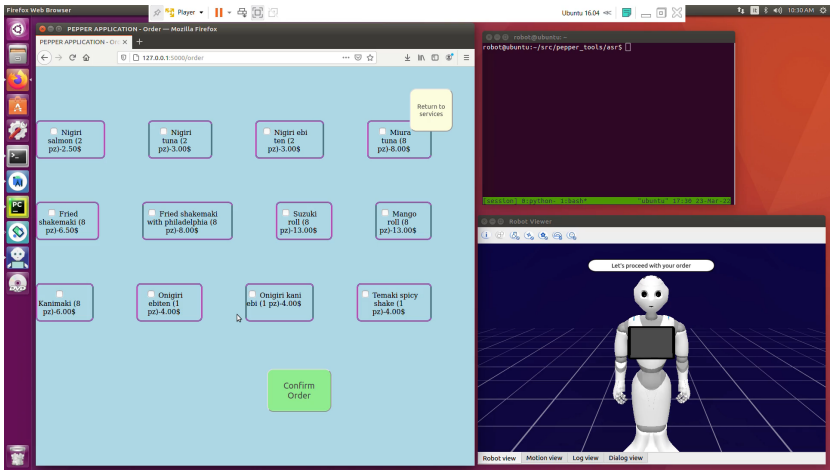


Figure 10: Order interface

One thing to mention is that the menù showed in this interface is not statically defined inside the HTML page of the browser, but is dynamically recovered

from a text file, read when the page is loaded. The data read from this file are then used to create the entries in the order interface. This could be a nice feature for the restaurant staff: in this way, who is in charge of updating the menù, can complete his task just by changing the entries of a simple text file, instead of modifying the HTML file. When the customer selected all the dishes he wants to order, to end the order phase, he just needs to click the "Confirm Order" button: the order phase ends, the order is received by the server and the application goes back to the services menù. However in this project, for the sake of simplicity, the server does not receive this data, because the application doesn't want to focus on this but on the interaction between the robot and the customer. Obviously, for an industrial application, a function that sends this data to the kitchen must be implemented. However if the user is registered and proceeds with the recognition, in this interface he will have an additional feature, as showed in Fig. 11.

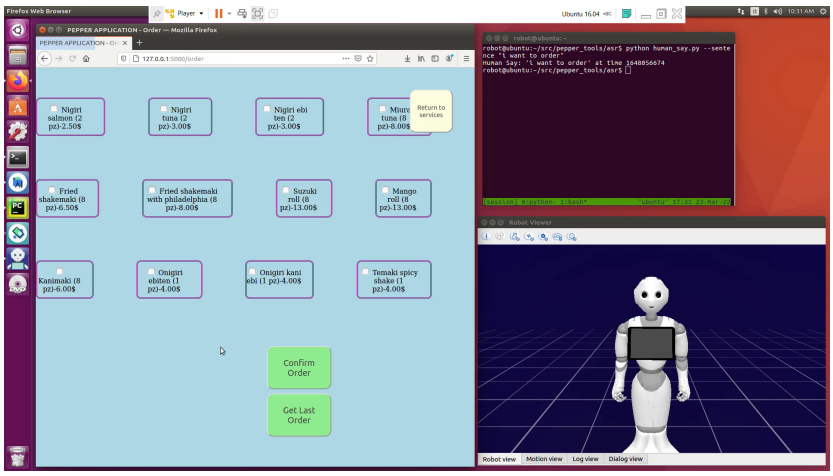


Figure 11: Order interface for recognized user

The button "Get Last Order" allows Pepper to retrieve the last order made by the current user and automatically check the relative entries in the interface, in order to save time during the ordination. This can be done because each ordination made by an user who logged in first through the face-recognition process is stored inside the database and linked to the user. In this way, when the customer returns at the restaurant, if he wishes, he can order the same dishes of the last visit or change something if he desires to try something new. In this last case, the last order field relative to the user is updated in the database with the new choices. This functionality is designed to speed up the ordering process.

4.4 Play a game

While the user waits for his dishes to be ready, he can play a game with Pepper. The game offered by the robot is an imitation game, in which Pepper imitates the pose and the call of an animal and the user has to guess. The interface for this service is showed in Fig. 12.

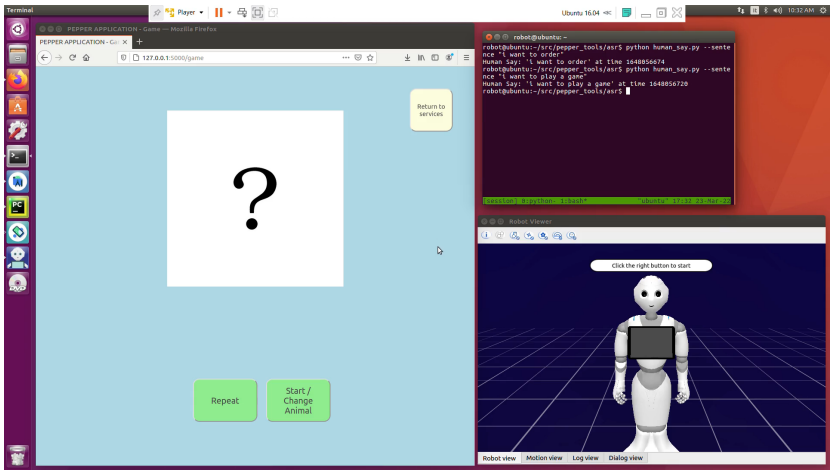
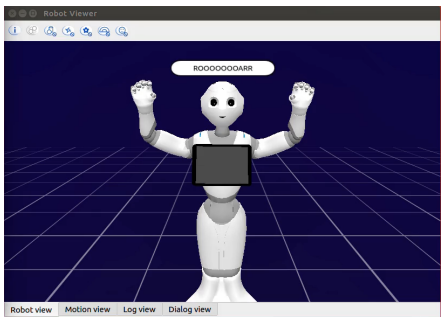
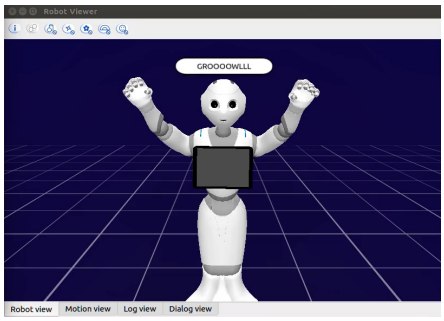


Figure 12: Game interface

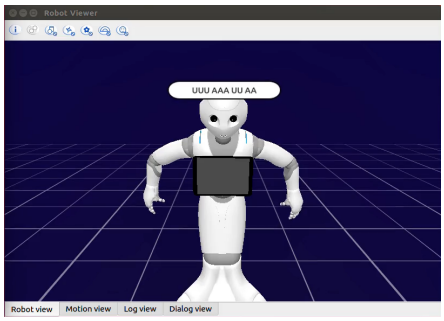
The game starts when the user clicks the button "Start / Change animal": a random animation is selected in the animation pool and executed by Pepper. For this project four animations were developed: lion, bear, monkey and elephant. To each animation is associated a specific pose of the robot joints and a specific (simulated) call. The animations are showed in Fig. 13. Due to the fact that the robot used for the project is a simulated one, the call is just realized through a visual text in the simulator, but in a real robot it can be easily substituted with a sound clip to enhance the imitation. When the imitation ends, Pepper asks to the customer to guess which animal was imitated. At this point the user should say aloud its guess and wait for Pepper to confirm it or not. If the guess is right, Pepper congratulates with the user choosing one of the three possible expression for this case and an image of the animal is showed in the browser page replacing the question mark image; if the guess is wrong Pepper replicates choosing one of the three possible expression for this situation and starts to wait again another answer. During this phase, if the user wants, he can ask to Pepper to repeat again the imitation by clicking the button "Try Again" or he can ask to change the imitated animal clicking the button "Start / Change animal". Whenever the customer wants to end the game he can click the "Return to services" button.



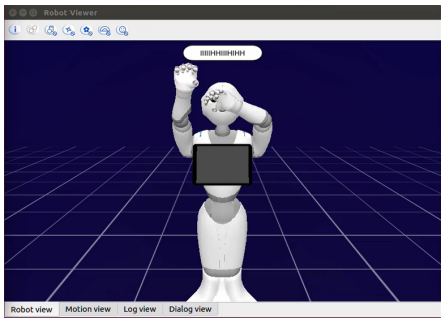
(a) Lion animation



(b) Bear animation



(c) Monkey animation



(d) Elephant animation

Figure 13: Pepper's Animations

5 Conclusions

In this project i developed a robot application for the hospitality industry. In particular the robot offers to the customers different services that can personalize the user experience. In particular each customer can register himself in the robot database, providing his data and by letting Pepper take a photo of him. When a customer is correctly registered, before proceeding to order, he can login in the application by letting Pepper recognize him through the face-recognition interface. If the recognition is successful, the customer can proceed with the order, having the possibility of automatically select the last choices he made; he can confirm them, change some or all of them, and send the ordination that will be prepared as soon as possible. During the wait, he can play a funny game with Pepper, in which he has to guess which animal the robot is imitating. The interactions with Pepper are implemented with a mixed modality, combining button inputs and voice inputs. Some features in this project were realised using simple means, like the database and the order interface: the database was implemented with a simple text file while the order interface doesn't really send the received order but the order is just saved in the database, if it was made by a recognized user. For an industrial application these things should be upgraded

and they could be subject for future works. In particular, the database should be implemented with a real SQL database in order to have a more reliable and better organized way to collect and store data, while the order interface should be improved so that it communicates with the server, sending the customer's order, that will be elaborated and sent to the kitchen to be prepared.

References

[1] J. Yang and E. Chew, "The design model for robotic waitress," *International Journal of Social Robotics*, 2 January 2021.