Interactive Imitation Game and Motor Coordination With Pepper

Master in Artificial Intelligence and Robotics

Human Robot Interaction, Electives in Artificial Intelligence

Damiano Brunori and Simone Faricelli

La Sapienza, University of Rome

Abstract

With this project we want to create an *imitation* game mostly designed for children, who can interact with the Pepper robot. The main idea is to make children interact with Pepper using the tablet mounted on the torso of the Pepper; the game is displayed on the tablet and it helps children to improve motor coordination by followings the instructions given by the robot. It could also helps younger children who have to become familiar with the 'concept of right and left'; the game is also designed in such a way as to shorten the time of reasoning thanks to a stimulus aimed at constantly engaging the mnemonic abilities. In order to estimate the pose of the users, we use *PoseNet*, which is a real-time pose estimation in the browser using *TensorFlow.js*; on the tablet will be displayed some geometric figures representing human joints and the player has to assume the position shown one the display of the Pepper. Then the movements are evaluated and a final score is returned. The final aim is to introduce children into the world of robotics thanks to an endearing game based precisely on Human Robot Interaction (HRI).

Keywords: Imitation game, Pepper, PoseNet, HRI

1. Introduction

Pepper is a semi-humanoid robot manufactured by SoftBank Robotics (formerly Aldebaran Robotics), designed with the ability to read emotions; Pepper's ability to detect emotion comes from the ability to analyze expressions and voice tones. The robot's head has four microphones, two HD cameras (in the mouth and forehead), and a 3-D depth sensor (behind the eyes). There is a gyroscope in the torso and touch sensors in the head and hands. The mobile base has two sonars, six lasers, three bumper sensors, and a gyroscope. Pepper (Fig. 1) is not a functional robot for domestic use.

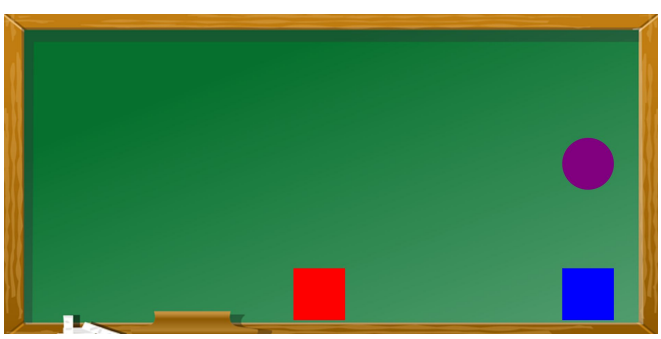
Fig, 1: Pepper robot interacting with a child.

Instead, Pepper is intended "to make people enjoy life", enhance people's lives, facilitate relationships, have fun with people and connect people with the outside world. For this reason we use it as a 'children friend' which is able to make them fun and learn at the same time.

2. Our Game

This project has been mainly inspires by [1] and [2] in which robots are used to interact with elderly people. In our game Pepper invites user to play and then the game is shown on the tablet mounted on the torso of the robot. A screenshot of the game is shown in Fig.2. It consists of a series of rounds within with appear some geometric figures representing human limbs. The instruction to follow in order to get a good score are shown before the first round. The main rules are always the same, but there are some random rules which change at each match; indeed geometric figures are randomly related to two human limbs, which are the wrist and the knee. In addiction the figures referring to the wrists can be blue or red, and also their color varies randomly at every match and it represents the right or left directions; figures referring to the knees are purple, which means that user can use right or left knee.

(a) Semi-random game rules.



(b) A round of a game match.

Fig. 2: Screenshots of the game displayed on the Pepper's tablet.

The pose estimation is perfomed in real-time, while the subsequent assignment of a score is performed off-line at the end of all the rounds of the game; this is mainly due to reasons of computational time. Each correct position is evaluated with one point, and the maximum obtainable score is 9, i.e. 3 points for each of the rounds, which are 3 per match; the final score is returned as a percentage of correct movements.

3. Pose Estimation

The pose estimation performed in order to evaluate the correctness of the position of the user, is executed on-line with *PoseNet*, which is a real-time pose estimation in the browser using *TensorFlow.js*. We tested the pose estimation on a user 182 cm tall and located about 235 cm from the Pepper. This evaluation part is obviously implemented in *Javascript*.

Pose estimation happens in two phases:

- An input RGB image is fed through a convolutional neural network;
- Either a single-pose (which will be our choice) or multi-pose decoding algorithm is used to decode poses, pose confidence scores, keypoint positions, and keypoint confidence scores from the model outputs.

The most important output used in our case are the following ones:

- **Keypoint**: a part of a person's pose that is estimated, such as the nose, right ear, left knee, right foot, etc. It contains both a position and a keypoint confidence score. PoseNet currently detects 17 keypoints.
- **Keypoint Confidence Score**: this determines the confidence that an estimated keypoint position is accurate. It ranges between 0.0 and 1.0. It can be used to hide keypoints that are not deemed strong enough.
- **Keypoint Position**: 2D x and y coordinates in the original input image where a keypoint has been detected.

In the next it is reported how looks like the main output returned by *PoseNet*.

```
{
  "score": 0.32371445304906,
  "keypoints": [
    { // nose
      "position": {
        "x": 301.42237830162,
        "y": 177.69162777066
      },
      "score": 0.99799561500549
    },
    . . .
}
```


4. Workflow and Implementations

We will describe here the workflow and the main steps related to the realization of our project, focusing on the implementation of both Pepper side and remote device (our laptop) side.

4.1. Workflow

The first step of the process of building such a project is the implementation of a **RESTFUL API NodeJS server** that responds to GET and POST requests. Then, through the use of **Pepper Command Tools** we make our robot perform some spoken interactions, movements and camera shots which will be deeply discussed later. Moreover, we will make the Pepper's tablet show our dynamic html page once the game has started. The pictures shot during the game are then stored into our server and a remote call to **PoseNet** is performed in order to get the info about the human position. The score is calculated, through a python script, on our remote device and the total final score is pronounced by the robot.

4.2. Pepper Side

We implemented the main functions such as the ones used to derive the final score and to convert the data points returned by *PoseNet* into areas (e.g. *left center*, *right bottom*, . . .), in such a way to perform an easy comparison with

the actual position shown on the tablet. All these function are implemented in *Python* directly on Pepper side. We used also another function to parse the file generated after running *PoseNet* evaluation; we use this file to derive the different joint areas positions (as we have just said) related to the poses performed by the user.
We remark that another game rule is that if the user has to perform a particular pose, he cannot cross his arms, but can only turn his back to the Pepper. This rule is mainly due to the fact that *PoseNet* can make incorrect evaluations when user is back to the camera.

Fig. 3: User performing a pose turning his back to the Pepper camera.

For this reason and since *Posenet* sometimes inverts the 'right' of the user with his 'left' when he turns his back to the camera, we decided to implement an additional evaluation strategy which acts only when a person is turned away. Actually the latter 'problem' does not happen very often, thus it is

not always a good choice to use this function. Anyhow this 'auxiliary function' evaluates the score assigned to the 'nose', and if this score is less than 0.65 percent, then it establishes that the user is turned away. In addiction *if the user is not in a frontal position* ***and*** *one of his knees has to be in 'center position'* (i.e., raised), *then* we can say that if the score assigned to the considered knee is less than 0.65 percent, thus the user has performed properly the movement, because in a 'reversed position', a raised knee will not be visible (See Fig. 3).

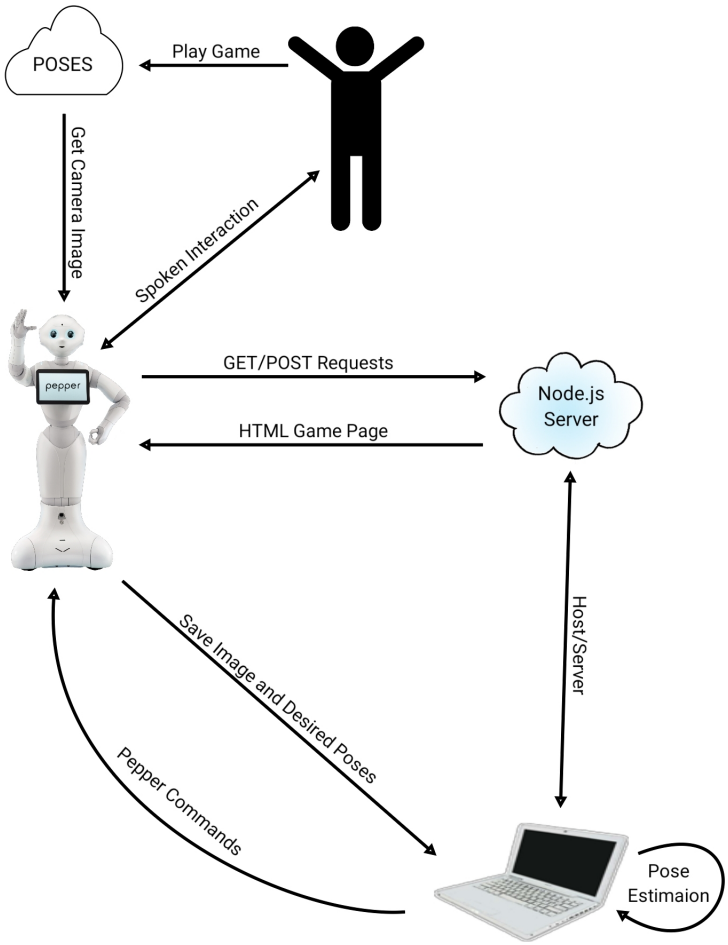


Fig. 4: Project Workflow.

4.2.1. Spoken Interaction

The spoken interaction consists of a series of sentences spoken by the Pepper robot in order to make more enjoyable and pleasant the game experience. Phrases are pronounced in very specific moments in order to make it easier for the user to understand the various phases of the game. There is only one situation in which the user is expected to answer to the robot, namely when it asks you to start the game or not.

4.3. Redirecting Data to The Remote Device

What is shown on the tablet (see Fig.2) are pages written in *html* and *Javascript*. As we said in the previous paragraph, we need to download and parse a file containing the poses performed by the user; we need this file to be downloaded on the remote device (connected to the Pepper), avoiding in such a way to download it directly on the Pepper. In order to do that, we used the Javascript run-time enviroment *NodeJS*.

4.3.1. The Client-Server System

Through the use of NodeJS and the web application framework *ExpressJS*, we built a Client-Server System running on the Pepper's tablet and shown through a GET request. In this way the above-mentioned HTML game page (Fig.2) generates and shows a series of positions to be emulated from the user and then it send them to the *RESTFUL API* NodeJS server, as the body of a POST request. The body has the form:

```
leftWrist    top_left        bottom_left     top_right
rightWrist   bottom_left     top_center      top_center
rightKnee    center          right_center    center
```

For each row we can see:

- on the first column → the body part requested to move;
- on remaining columns → the corresponding positions for each round.

4.4. Server Side

When a POST request is received by the server, its content is stored in a text file, which will be then opened and parsed by a python script which will compare the request's data with the results of the Pose Estimation prediction and will build a score page.

4.5. Score Function

We simply evaluate the final score according to the number of poses properly performed by the user. We simply return a percentage of movements performed correctly and we make them to pronounce to the robot. As we have just said, while the pose estimation is performed in real-time after each round, the final score is return at the end of all the rounds; this is due to the fact that pose estimation takes several seconds and we do not want the user to wait too long between rounds.

5. Conclusions and Future Works

In order to deduce concrete and reliable conclusions we should conduct an experiment with different children with different cultural background; then we should collect their impressions and those of their parents, submitting to them a suitable questionnaire. Certainly we can say that our game allows children to interact with Pepper in a very stimulating way under the aspects of motor coordination and mnemonic learning. It could be improve computational processing speed using a more powerful remote device, or executing the pose estimation on a partitioned hard disk instead of on a Virtual Machine; in this way could also be possible to perform an online evaluation.

Some interesting aspects could be improved or add in order to improve or extend our project:

- inserting level difficulty according to the user skills;
- show on the screen the user himself while performing the positions, illuminating the part of the screen in which a specific part of the body must be positioned;
- adding a *1 VS 1* playing mode;
- inserting different type of games which requires different skills.

Without any doubt many other changes and improvements can be made to our project, and with this good feeling we strongly believe that our project could hopefully encourage new and more extensive work regarding this field.

References

[1] Joao Avelino, Hugo Simao, Ricardo Ribeiro, Plinio Moreno, Rui Figueiredo, Nuno Duarte, Ricardo Nunes, Alexandre Bernardino, Martina Caic, Dominik Mahr, Gaby Odekerken-Schroder, "Experiments with Vizzy as a Coach for Elderly Exercise", *The 13th Annual ACM/IEEE International Conference on Human Robot Interaction - McCormick Place, Chicago, IL, United States*, 2018.

[2] Griffiths, Sascha & Alpay, Tayfun & Sutherland, Alexander & Kerzel, Matthias & Eppe, Manfred & Strahl, Erik & Wermter, Stefan. (2018). Exercise with Social Robots: Companion or Coach. *The 13th Annual ACM/IEEE International Conference on Human Robot Interaction - McCormick Place, Chicago, IL, United States*, 2018.