Technical paper

Team name

KU Robotics (Khatam University Robotics)

Author names

Elaheh Rahpeyma, Neda Aalami, Ali Daneshmand

Supervisor

Abdolah Shamisa

About us

KU Robotics is founded three years ago in Khatam University (Non-governmental and private Univ.). In the first project, after some studies and market researches, we bought a Pepper robot from Softbank Robotics Company. Some basic algorithms was performed (for about one year) for Pepper startup as a banking service robot for Pasargad Bank (the best private bank in Iran). In the second project as continue of the first project, more advanced banking scenario was developed. We worked with APIs and SDKs of Pepper and implemented different algorithms such as navigation, people detection & recognition, interaction in Persian language, Mask detection, and Object detection and so on in a bank environment. Also, some conference paper is published during these years. Our team has won the tender for the construction of a banking humanoid robot for Pasargad Bank.

Task implementation process and approach

Introduction

Robocup@home league aims to develop service and assistive robot technology with high relevance for future personal domestic applications, including different tasks to show participating team's abilities in AI and Robotics. Therefor this paper is written to determine KU Robotics team's performance and technical details. The explanation of each of three main tasks is discussed according to competition's rule book tasks.

Carry the Luggage

Object recognition

In the planned object recognition tasks, the first part is to recognize the luggage and then estimate its location and distance. This should be done in such a way that the robot can approach and then stand at an acceptable distance from the human and prepares to grab the bag from the person. In the second part, the robot has to determine the arrangement of the chairs and correctly identify the empty seat, then guide the person towards the sitting place.

In these scenarios, the Yolo3 real-time object detection and recognition algorithm; which is one of the most effective object detection algorithms, is used. A data set of paper bags was trained with TensorFlow 2.3 and Keras 2.4 and then converted to Yolo3 format for real-time object identification to achieve the highest performance on the goal object recognition.

The presence of the chairs in the environment is checked by suitable algorithms for detecting empty seats in the video stream received from Pepper's bottom camera using gstram. If there are chairs in the frame, then they are sorted by location in the video frame from left to right, and then the presence of humans is checked. If there are people in the frame, the overlap of each person and the chair is checked, and the position of an empty chair is provided to the customer depending on the results. The entire chair detection flow was applied in the approach's find my mates component. In addition, for identifying the location and distance of a paper bag from Pepper, after detecting the bag object in the video stream from Pepper's tablet camera, its location and size are identified, and the distance is computed depending on the size of the bag.

Take the bag

To take a bag from a person who is standing and take the bag in the front of its body, Pepper must identify the bag, calculate the distance from the bag, and then begin moving forward. Of course, distance estimation will occur any time Pepper reaches the last target; which is determined by the last distance information. But what is important is Pepper should stop at a specific distance from the bag. After that, it is needed to define a trajectory for its hands, and by way of is mentioned in object recognition, the information of the bag, its size, and the distance is accessible. So, for several standard bags, the size is extracted, and depend on them, four different trajectories of grasping the bag have been defined. Since a timeline is put on the Choregraphe to monitor joint movements, it was easier to use this choice, because, apart from the increased implementation speed, the moving behavior was smoother.

Mapping and navigation

In this section, a first and important part was building a map. Two ways of building a map are using ROS or using the ALNavigation API, where each one of these ways has its challenges. The most unpleasant difficulty when using the API was that Pepper didn't support radius mapping of more than three meters, and if the specified radius was more than three meters, the connection was lost and the session was disconnected. As a result, utilizing APIs for mapping was only appropriate for explorations of less than three meters. There were various packages for utilizing ROS to generate a better map all packages use “slam_gmapping” to map the environment, but because the most appropriate ROS distribution for Pepper is indigo, it was important to modify packages to make Pepper mapping. One of the issues of using ROS for mapping was that the laser range was insufficient, therefore it took a long time to generate a map. After generating a map of the environment, it was needed to localize the Pepper in the map. For moving the robot, the joy node runs to make the joy code of Pepper reachable (with moving the robot all arrows around Pepper modify its position on the map). Finally, the map construction and localizing were completed. The navigation stack was used to autonomously move the Pepper robot from one point to another.

Find My Mates

People recognition and face detection

Since the results of robot APIs for saving people's faces and recognition were unreliable, a new technique for recognizing the face of the person and communicating with the robot was needed. However, one of the challenges of using a new technique was the lack of good results, from the use of Peppers cameras to training and recognition. So, using the Pepper's tablet camera was the only way to avoid using an extra camera, while still getting good results. As a result, the tablet's camera was used for both training and detection. For the training part, the Python Face Recognition Library was used to encode the face that was found and saving it as a record with the name of the individual, who belongs to the face info. A trained KNN model is loaded for the recognition part to predict and measure the distance between recorded data and new face data. If the prediction is less than a certain threshold, the face was not recognized, and the train part should be repeated. When Pepper detects that someone is ready to communicate with it, an ID is sent to the callback function and based on that ID and face identification, the recognize part is executed. If the individual's face information is in the reported data, Pepper greets the person and tells the name of it; however, if an individual is a new person, the training flow will repeat.

Receptionist

Speech recognition and interaction

Pepper speech recognition API (ALSpeechRecognition) is appropriate for greeting, but it was inefficient in two areas: first, there was a need to recognize the individual's name, and second, it was necessary to take the order that was picked from the menu. These reasons make us to mixed Pepper speech recognition API with Google speech recognition. For recording the voice to do processing, two methods were used and tested: first, start and stop recording based on the RMS level of environmental sounds from an acceptable threshold, which was specified based on background noises, and second, using an API to connect event that fired from changes of environment sound level, the value which received was 1 or 0. In the second method of recording, each load voice and the length of speech causes the event to send value 1, but in a sentence, each pause causes the event to send value 0. But after a while, if there was a sound again, 1 was received due to this mechanism. The recording will continue until a certain time has passed, after which it will be paused. Since we give break time after every 0 value, we can ensure that we don't miss the last part of the individual's speaking when we use the second method. For example, when an individual starts to speak, the recording process will begin immediately after it, and even if the individual stops speaking, we can ensure that we didn't miss the last part of the individual's speaking because we give break time after every 0 value.