Indian Institute of Technology
Kanpur

Robotics Club

Summer Project, 2016

HURO

Team Members:
Abhishek Gupta
Saurabh Ranjan
Mrinaal Dogra
B0huvi Gupta
Ujjwal Varshney
Rijak Khanuja
Shruti Joshi
Nitish Pant

Supervisor:
Ankit Kumar
Anvesh Jadon
Mayank Mittal
Hemant Kumar

Acknowledgement

We would like to express our gratitude to everyone who has been assosciated with our project and contributed to the completeion of "HURO, the wheeled Humanoid". We sincerely thank the Robotics Club, IITK for providing us the opprotunity and facilities to do the project.
We also thank all the seniors without whose constant guidance the project could not have been accomplished. We would like to take the opportunity to mention some names - Prashant Kumar, Abhishek Attal, Ankit Kumar, Mayank Mittal, Anvesh Jadon, Hemant Kumar and Jatin.
We also acknowledge each other as a team which stuck together in high and low to show a consistence performance and a fruitful result.
We are indebted to the God Almighty for always being there for us and pulling us out of our hard times.

Contents

1 Introduction **3**

2 Brief Overview of Work **3**

2.1 Mechanical 3

2.1.1 Head 5

2.1.2 Torso 6

2.1.3 Simulation 6

2.2 Speech Recognition 8

2.2.1 Speech-to-Text 8

2.2.2 Chatbot 9

2.2.3 Text-to-Speech 9

2.2.4 Integration of Speech Recognition 9

2.3 Sound Localization 9

2.4 Image Processing 10

3 Integration **11**

4 Future Plans **11**

Abstract

The project was aimed at building a wheeled humanoid which could talk, localize sound, recognize faces, detect and follow a particular object, and follow line.

1 Introduction

This project was done as a summer project under the Robotics Club, Science and Technology Council, IITK. The objective of the project was to build a wheeled humanoid and implement speech recognition, image processing and sound localization on the bot. This also served as a source of gaining experience and knowledge as we aspire to participate in the Huro Cup. We now foresee this project with a new and improved design, biped and better image processing capabilities.

2 Brief Overview of Work

2.1 Mechanical

- For designing of the bot,we have used Autodesk Inventor.The bot is 60-65 cm in length.It is designed to have 8 degree of freedoms(3 in each hand and 2 DOF in the head).Initially,hands were planned to be incorporated in the bot but the plan was dropped later on due to unavailablity of servos of sufficient torque.

- The basic ideas behind the designing of the bot are as follows :-

  - **Anthropometrically correct**
  - **Compliant joints**
  - **Torque controllability**
  - **Affordability**

- The actuator used for the head is HS-422 normal servo motors while for driving the wheeled humanoid Johnson geared DC motor is used. The specifications of the actuators used in the bot is given below in the table

Actuators Weight (in g)	**Weight (in g)**	**Torque (in kg cm)**	**Speed**	**Operating Voltage(in volts)**
HS-422	45.5	3.3 to 4.2	0.21 sec/60 ° to 0.16 sec/60 °	4.8 to 6
Johnson Geared DC Motor	68	8 to 12	100 rpm	12

- For driving the bot, 2-wheeled drive ,along with 2 castors, was chosen over 4 wheeled drive since it will be much easier to control two DC motor in comparison to four DC motors.The use of Johnson Geared DC motor is justified by the requirement to provide high torque at low rpm to effectively control the turning of the bot.
- The Orientation of the wheels and Castor is depicted in figure 2.Here Grey coloured portion represent the base platform of the bot while the text "WHEEL" and "CASTOR" represent the relative position of wheels and castors.

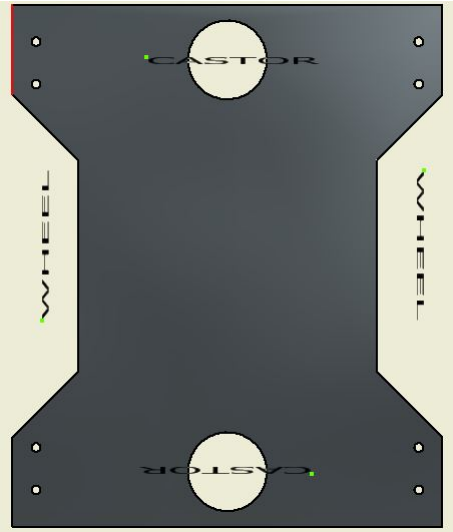

Figure 1: Relative position of Wheels and Castors

2.1.1 Head

The mechanical design of the head is made with keeping in mind to increase the view of webcam that will be mounted on it.It has two degree of freedom and can accomplish the pitch and yaw motion.The Orthographic view of the head is given below.While designing ,it is tried to maintain the ratio of head's height to shoulder's width which is approximately 1:2.

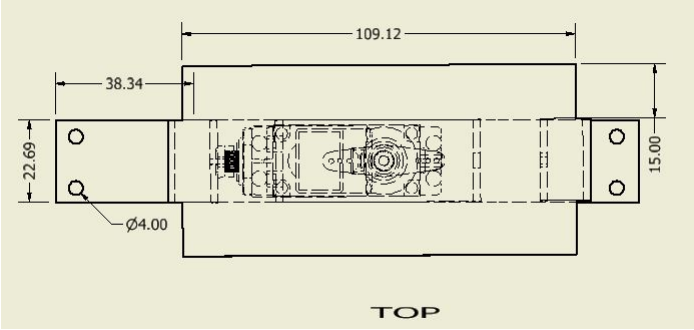

Figure 2: TOP VIEW

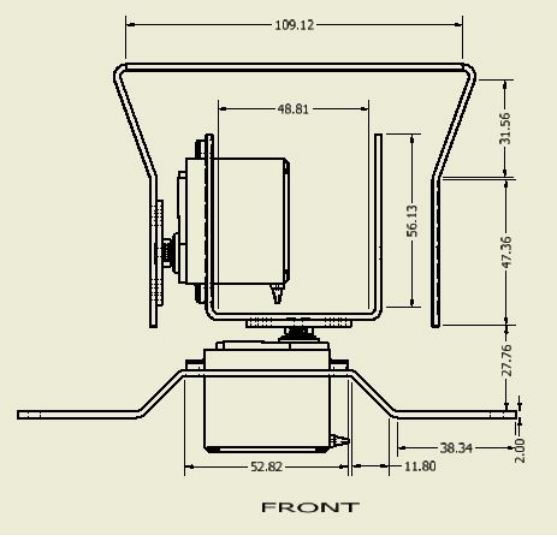

Figure 3: FRONT VIEW

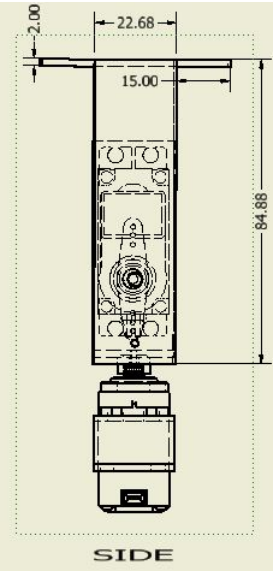

Figure 4: SIDE VIEW

2.1.2 Torso

The mechanical design of torso was made with a view to maximize the volume of torso while keeping in mind about its human like feature.It hasn't been given any degree of freedom since its sole purpose is to store equipments and devices.Orthographic View of torso is given below.

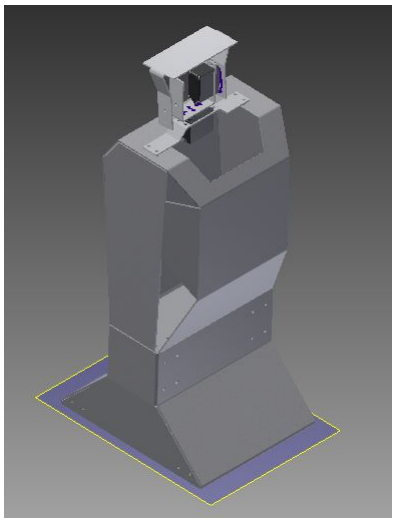

Figure 5: Huro Assembly

2.1.3 Simulation

- For simulation we have used Vrep simulating software .Firstly, we imported cad file in **"stl"** format .This format basically consist of structures of bot without any specification (such as margin of holes ,edges of plates etc).

- To make design compatible with the vrep interface , we converted the design to simple one , by merging the things in **Triangle Edit Mode**. It is important to select the significant amount of triangle (try to select less than 20,000), and then convert them into the simple 3D object such as cuboid , cylinder, sphere .This is to be done to speed up the software.

- After making body compatible in frame of physics ,now make changes

in the toolbar to convert the body dynamically active (for eg. : **static ,respondent,collidable, measurable, renderable, detectable**)

- As mention earlier body couldn't attach by its own ,infact imported motors are worthless until you make them compatible. We have insert joints in place of axles .To do so we have to set the joints at right coordinates.

- After ,doing the above task,we have to tell the interface that they are also the part of our design, to do so we have to make the distinct part of the design to be the parent of joint.

- Now to treat collection of the parts as a whole body, we have to make collection of that by using **Collection Object Dialog**.

- Now we are at the stage to analysis the different factors such as: distance from obstacle, speed analysis etc.

- To do the above step we implement a graph on our screen , which can be done by adding a graph in your hierarchy and edit it as per your requirement.

- Next step is to modulate velocity with an **open GI based custom UI**.

- Now , comes the last part that is scripting part. To do so we have to learn Lua programming language with its additional Vrep libraries. Our next step will be to include **Inverse Kinametics Module**.

2.2 Speech Recognition

A speech engine was build for the bot with the help of open source libraries in python. The engine recognized only a specific set of words and sentences. The algorithm can be broady divided into three parts-speech to text conversion, chatbot and text to speech conversion.

2.2.1 Speech-to-Text

- Read about the algorithm of speech Recognition. Decided upon the following approach:
  - Noise reduction of input signal and amplitude normalization.
  - Feature extraction using MFCC.
  - Feature comparison using vector quantization.
- Finalized python to be used as the platform of development. Found the following open source library, based on MFCC feature speech recognition: Speech Recognition
- Decided to work with the following offline speech recognition engine/API with the library: CMU Sphinx.
- Downloaded and installed the library and the speech recognition engine. Started working with the most basic functions of the library. Realized that accuracy was poor due to the huge number of words.
- Alongside, read about the language model, phoneme extraction, phonetic dictionary, and the HMM(Hidden Markov Model).
- Found a way to edit the dictionary file used by the library, resulting in better accuracy of the recognized words.
- Finally, we learnt how to make a language model of our own using the "Building Language Model" tutorial available at the official site of CMU Sphinx.

2.2.2 Chatbot

- Found a Chatbot Tutorial(in c++) at codeproject.com. It gave the basic idea of how a chatbot is implemented.
- Found the following package in python for Chatbot: ChatterBot. The chatbot is already trained in English, however it can be further trained or retrained according to our needs.
- Installed the library and worked with it. Found that due to a very large database of the package, the results were very inaccurate.
- Found the method/algorithm for training the library so that we can bypass its original database with our own database. We created our own database consisting of about 10 sentences.

2.2.3 Text-to-Speech

- Found a python library to achieve the task : pyttsx, and installed it. It includes drivers for the text-to-speech synthesizer : espeak, so installed that too.
- Configured the output rate and the speaker which best suited us.

2.2.4 Integration of Speech Recognition

- Since all of the libraries used were in python, created a python script to implement all the above mentioned parts.

2.3 Sound Localization

Sound localization included finding the angle of the sound source with respect to the humanoid.It was implemented using Python and its libraries like Numpy,Scipy and Matplotlib.The time lag between the two microphones was calculated using some mathematical tools such as Cross-correlation and FFT of a signal.

- Read live audio from microphones in small chunks and stored the data in Numpy arrays ,saving it as a "wav" file.

- Input the data from "wav" file and implementing a band pass filter to reduce noise in the audio signal
- As the data size was too large , we cross-correlated the two signals in the frequency domain and took the Inverse FFT of the product.
- Using basic trigonometry the angle between the sound source and humanoid was calculated successfully.

2.4 Image Processing

Works in the field of Image Processing taken up by team under the project were achieved using the open source library of OPENCV (version 3.1.0) implemented using C++. In order to create some sense of vision in our bot we picked up tasks like object tracking, line following and face detection and recognition.

- Casscade Classifiers were used to implement the task of face detection.
- Fisherfaces algorithm and FaceRecognizer class was used to implement the task of face recognition. A detailed study was conducted before the implementation was done and methods to enhance the accuracy of the same were looked upon and searched thoroughly.
- To increase speed and reliability of face recognition, face detection parameters were optimized as well.
- Several databases were looked upon and were studied before we attempted to create one of our owm. With the passage of time, this database was tested and ammendments were made so as to increase the speed and accuracy of the same.
- Color based contours were used to implement line following, object detection and tracking.

3 Integration

- For this purpose, Odroid XU4 and Arduino MEGA are going to be used. So, installed all the libraries, packages and softwares required by various subteams on Odroid.
- Firstly, it was decided that integration of all the codes will be implemented using ROS. But due to lack of time, we shifted to C++.
- The C++ code served the purpose well , lacking only at a point that its speed of communication between different codes was not much fast. Also, due to some unknown reasons, the python scripts of "Speech Recognition" and "Speech Localisation" were not able to run simultaneously.

4 Future Plans

- To come up with a new, lighter and better design.
- Implement integration through ROS.