New York University

Tandon School of Engineering
Department of Aerospace and Mechanical Engineering

ME-GY 6933

Advanced Mechatronics

Final Project

Design and Development of World Wide Controlled Pet Feeding Robot

Submitted to
Prof. Vikram Kapila

Aswath Suresh
Bhavik Dilip Doshi
Shreyance Singhvi

Design and Development of World Wide Controlled Pet Feeding Robot

Aswath Suresh, Bhavik Doshi and Shreyance Singhvi
as10616@nyu.edu,bdd271@nyu.edu
Department of Aerospace and Mechanical Engineering, New York University, USA

***Abstract*— This paper describes a pet feeding robot which can be controlled from anywhere on the world. The robot features a differential drive mechanism, collision free distance sensing, wireless NRF SPI control, User-End GUI and Camera Vision. The robot hardware includes Arduino Microcontroller, Raspberry Pi 3, HD Camera, Sabretooth Motor Driver and 4s Lipo Battery. Inspired from nature, a reflex mechanism has also been integrated into the rover design to minimize damage, by automated safety reflexes using ultrasonic distance sensor. The four wheeled mechanism ensures that it can traverse stairs easily. The mechanism provides traction due to its body weight. The robot finds applications in feeding any kinds of pet from anywhere, remote explore the house for any potential theft and fake robbers to believe someone is at home. Also includes personalize daily meal portions, stay connected with real time alerts, know your pet's ok when you're away, stores up to 7lbs, keeps food fresh and keep your pet healthy. The robot also enables a user to prevent a pet from eating a specific food while still allowing access to that food by other pets. Thus, making it user friendly for users having more than one pet.**

Keywords: Arduino, Camera, Drive Mechanism, Raspberry Pi, SPI

I. INTRODUCTION

We choose this project because pet keeping is a time-consuming responsibility and we want to provide convenience to owners by helping them feed their pets easily and smartly from anywhere on the world.

Keeping pets takes many commitments. This includes keeping them company, showing your concerns and of course, feeding them on time and in the correct way. However, not everyone is a pet expert, taking care of your pet's diet can be hard and time consuming. One of the top health concerns of pets are overeating and obesity. Especially at younger age, they are usually satisfied with however much is given to them. Many adult pets are fed unscientifically that later may cause short lifespan. Another problem of feeding pets is that owners might not always be home regularly as they have to travel to another country as a vacation or for a business trip. Being occupied by personal plans knowing that they still have a starving little fellow at home to be taken care of is always a concern that bothers owners. The third concern that we want to deal with is the fact that there hasn't been any product on the market right now that is able to dispense food for pets monitored by its owner real-time. However, pets themselves might not necessarily recognize the potential health problems of eating the wrong food. There are products like Petnet, AutoPetFeeder, Automation Pet Feeder [1-4] which can be scheduled to dispense food at certain interval of time but it lacks real time monitoring and mobility. Therefore, we want to take care of owners' concern of feeding by building a phone/laptop controlled real-time semi-automatic pet feeder that can dispense the desired food as per the user by live camera feedback.

II.MECHANICAL DESIGN

The four-wheel mechanism ensure that it can traverse over a considerable height greater than the chassis height which could be as much as twice the diameter of the wheels. The advantage of this design is that it can still run even if it gets toppled allowing the robot to overcome obstacles and traverse over a highly-rugged terrain like stairs. The robot gets traction only due to its body weight without having to compromise the strength of the chassis. The Fig.1 shows the prototype of the four wheeled mechanism.

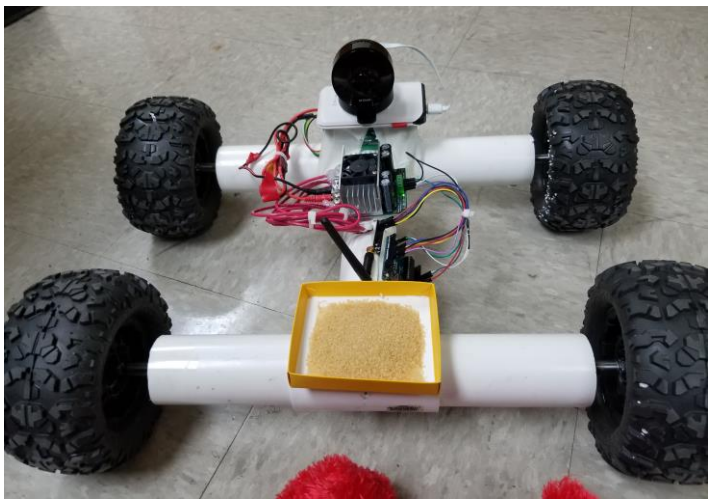


Fig.1 Prototype

III.PET FEEDING ROBOT SYSTEM

The Fig.2 shows the working principle of the complete system. At the user end there will be a GUI developed using the tkinter GUI python toolkit. The GUI helps in the movements of the pet feeding robot. The user-end device will be connected to the internet i.e. mobile or laptop. The Real VNC application is used to access the Raspberry Pi 3 at home which is connected to internet all the time. The Raspberry Pi 3 is connected to Arduino Uno which is on the robot using Serial Peripheral Interface(SPI). The SPI connection between Arduino Uno and Raspberry Pi 3 is made wireless using NRF24L01 with a range of 1 mile. The NRF24L01 is a highly integrated, ultra-low power (ULP) 2Mbps RF transceiver IC for the 2.4GHz ISM

(Industrial, Scientific and Medical) band. With peak RX/TX currents lower than 14mA, a sub µA power down mode, advanced power management, and a 1.9 to 3.6V supply range, the NRF24L01 provides a true ULP solution enabling months to years of battery lifetime when running on coin cells or AA/AAA batteries.

The Arduino gives commands to the motor using a motor driver called Sabretooth for required robot movement. The Arduino cannot handle the high voltage and high current requirement of the motors. So we used the high current (upto 60A) and high voltage (upto 32V) motor driver. So when the button is pressed in the GUI at the User End a specific function is called in the python program in the raspberry pi 3 based on which button is pressed. The function sends set of string value (Speed, Direction) to the Arduino through SPI Communication. The Arduino decodes the string value and based on the received value controls the motor driver for required robotic movements. The camera feed from the HD IP Camera is always available at the user end via internet. This system makes it really easy to find the pet and feed it on time.

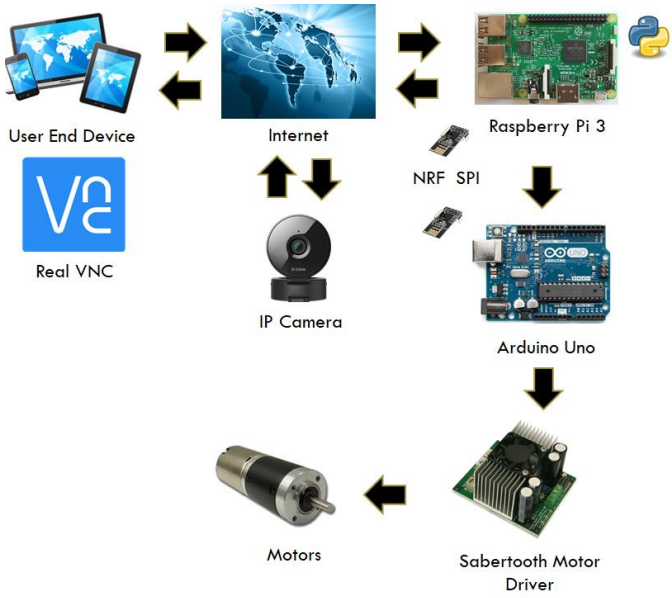


Fig.2 System Representation

A. GRAPHICAL USER INTERFACE

The Tkinter module is used in python to develop the GUI as shown in Fig.3. Tkinter is Python's de-facto standard GUI (Graphical User Interface) package. It is a thin object-oriented layer on top of Tcl/Tk. Tkinter is not the only GUI Programming toolkit for Python. It is however the most commonly used one.

A window named Control Panel is created with four buttons namely Forward, Backwards, Right and Left. If the button forward is pressed the robot will move forward, if right is pressed it moves towards right and so on. The GUI developed in python can be accessed using real VNC from anywhere on the world where internet is available.

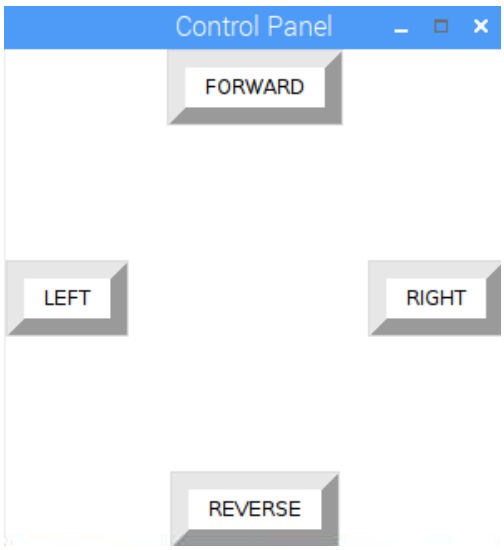
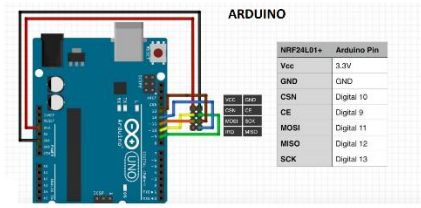


Fig.3 GUI

B. RASPBERRY PI –ARDUINO SPI COMMUNICATION



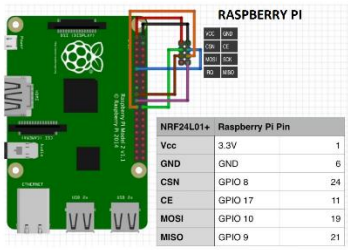


Fig.4 SPI Communication

The Serial Peripheral Interface (SPI) bus was developed by Motorola to provide full-duplex synchronous serial communication between master and slave devices. The SPI bus is commonly used for communication with flash memory, sensors, real-time clocks (RTCs), analog-to-digital converters, and more. Standard SPI masters communicate with slaves using the serial clock (SCK), Master Out Slave In (MOSI), Master In Slave Out (MISO), and Slave Select (SS) lines. The wiring for

SPI communication between Arduino and Raspberry Pi along with NRF24L01 is done as shown in the Fig.4.

C. CAMERA FEEDBACK

Fig.5 Camera Setup

The IP Camera on the pet feeding robot is connected to a wireless router using Wi-Fi Protected Setup(WPS). The router is always connected to internet and the user can access the HD camera using the camera IP address over the internet cloud. The user will be receiving the live feedback of 1080p live footage of what the pet feeding robot see. This helps the user to navigate the robot to the required destination. The Fig.5 shows the camera setup to get live feedback at the user end.

D. MECHANICAL LAYER AND POWER MANAGEMENT

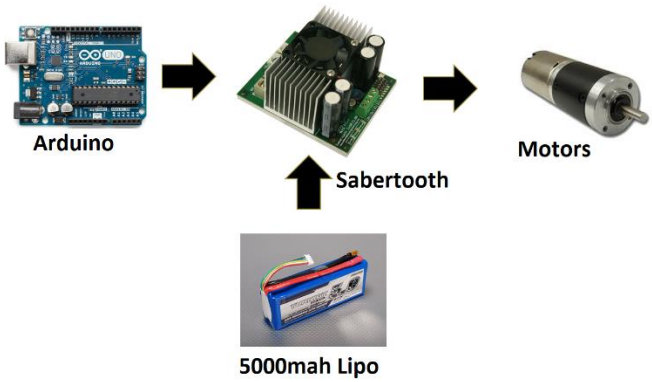


Fig.6 Mechanical and Power Management

The pet robot uses four high torque 7A motor for travelling in any sort of rugged terrain with ease. The four-wheel mechanism is developed in such a way that the robot can move even if it gets toppled. This advantage allows it to climb up and down the stairs without any issue. The signal received from Arduino to the Sabertooth motor driver drives the 28A 12V motor system in the required direction. The robot uses a 5000mah 12V high power lithium polymer batter. The Arduino and HD camera is powered from the 5v regulated supply available from the sabretooth motor driver. Back current and short is taken care using diode and protection circuit which makes the system totally safe. The Fig.6 shows the power management and mechanical layer of the system.

IV.RESULT AND DISCUSSION

The result was tested as a scenario of a person sitting at office and feeding his/her pet at home. The different scenario till the result is achieved is as shown in Fig.7.1-7.3.

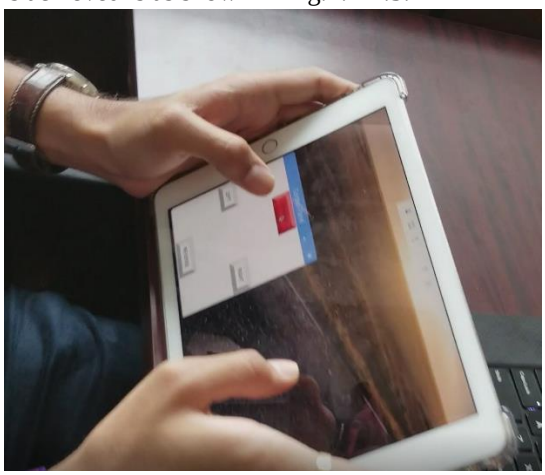


Fig.7.1 User Sitting in the Office

The Fig.7.1 shows a person sitting in the office with the GUI of the pet feeding robot trying to feed his/her pet at home.

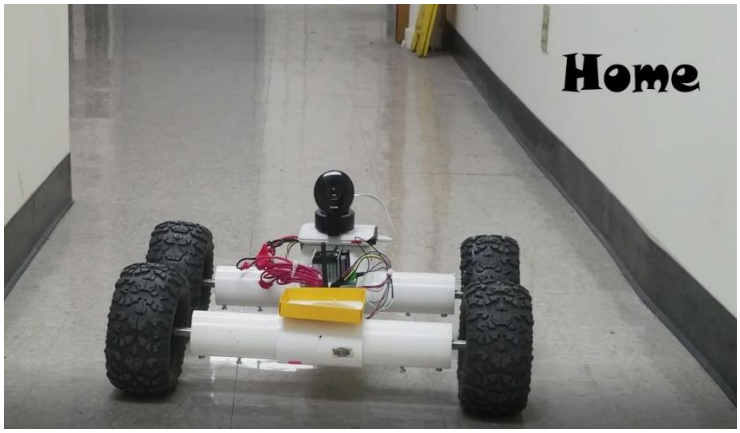


Fig.7.2 Pet Feeding Robot at Home

The Fig 7.2 shows the pet feeding robot loaded with food at home controlled from the office. Fig.7.3 shows the robot on its way to the target and also the top left shows the camera feedback which the user sees while controlling the robot.

Fig.7.3 Pet Feeding Robot reaching target

Fig.8 Pet Feeding Robot Stair Case Test

It is observed that the pet feeding robot at home is capable of doing the real time pet feeding task without any difficulty from the office which is far away and the performance is good. The four wheeled mechanism for climbing up and down was tested and led to satisfactory result. The Fig.8 shows the stairs case test of the pet feeding robot.

V.CONCLUSION AND FUTURE DISCUSSION

Over the last semester, design team have accomplished a massive amount of research, learning and coding, we feel that we have produced a great success. About what worked correctly, most of our design goals were met. We received a live feed from camera mounted on robot, over internet. We were also able to control the pet-feeder precisely through GUI for smart devices. Also, the all-terrain vehicle used for prototype, what set our product apart from any other on the market, works flawlessly. The only goals that were not fulfilled were to provide an easy user interface and to provide future meals at a predictable time. This is because now, there is no timekeeping and multi-feeder system. If we were to take this design further, we could change few things. First, we would change the material of feeder body. Next, we would implement it as a part of smart home system using Raspberry Pi and Arduino. These changes would help us to better meet the goals that we had originally laid out for our design.

ACKNOWLEDGMENT

The authors would like to thank Makerspace and NYU Tandon School of Engineering for providing support to carry out the research and experiments.

REFERENCES

1. Rachel Heil, Kristine McCarthy, Filip Rege, Alexis Rodriguez-Carlson, “The Smart Pet Feeder: A Proposal to Design and Build an Automated Pet Feeder Capable of Preventing One Pet from Eating Another Pet’s Food”, January 30,2008
2. Zhuokai Zhao, Ziyun He, Fan Ling, “Automatic Pet Feeder Project”, February 10, 2016
3. Online Available: http://petnet.io/smartfeeder
4. Online Available: http://www.autopetfeeder.com/

APPENDIX

RASPBERRY PI 3 TRANSMITTER PYTHON CODE

```
import RPi.GPIO as GPIO
from lib_nrf24 import NRF24
import time
import spidev
import tkinter as tk

master = tk()
GPIO.setmode(GPIO.BCM)
pipes = [[0xE8, 0xE8, 0xF0, 0xF0, 0xE1], [0xF0, 0xF0, 0xF0, 0xF0, 0xE1]]
radio = NRF24(GPIO, spidev.SpiDev())
radio.begin(0,17)
radio.setPayloadSize(6)
radio.setChannel(0x76)
radio.setDataRate(NRF24.BR_1MBPS)
radio.setPALevel(NRF24.PA_MIN)
radio.setAutoAck(True)
radio.enableDynamicPayloads()
radio.enableAckPayload()
radio.openWritingPipe(pipes[0])
radio.printDetails()

 def Forward:
    message = list(“1”)
     start = time.time()
    radio.write(message)
    print(format(message))

def Reverse:
    message = list(“2”)
    start = time.time()
    radio.write(message)
    print(format(message))

def Left:
    message = list(“3”)
    start = time.time()
    radio.write(message)
    print(format(message))

def Right:
    message = list(“4”)
    start = time.time()
    radio.write(message)
    print(format(message))

B = Button(master, text = “FORWARD”, command = Forward, bg = ‘white’, fg = ‘black’ bd = 10, activebackground = ‘red’)
B.pack()
C = Button(master, text = “REVERSE”, command = Reverse, bg = ‘white’, fg = ‘black’ bd = 10, activebackground = ‘red’)
C.pack()
D = Button(master, text = “LEFT”, command = Left, bg = ‘white’, fg = ‘black’ bd = 10, activebackground = ‘red’)
D.pack()
E = Button(master, text = “RIGHT”, command = Right, bg = ‘white’, fg = ‘black’ bd = 10, activebackground = ‘red’)
      E.pack()
```

ARDUINO RECEIVER CODE

```
#include<Wire.h>
#include<SPI.h>
#include<RF24.h>
#include <Servo.h>
Servo myservo1;
Servo myservo2;
int k=1000;
RF24 radio(9,10);

void setup()
{
 myservo1.attach(5);
 myservo2.attach(6);
  while(!Serial);
 Serial.begin(9600);
 radio.begin();

radio.setPALevel(RF
24_PA_MAX);

radio.setChannel(0x7
6);
 const uint64_t pipe =
0xE8E8F0F0E1LL;
radio.openReadingPi
pe(1,pipe);
radio.enableDynamic
Payloads();
 radio.powerUp();
 delay(100);
 }

void loop()
{

radio.startListening();
char
receivedMessage[10]
= {0};
  if(radio.available())
  {
radio.read(receivedM
essage,
sizeof(receivedMessa
ge));
  int
msg=(int)(receivedM
essage[0]-'0');
  Serial.println(msg);

radio.stopListening();
       delay(15);
    //1-F
    if(msg==1)
    {

myservo1.write(110);
  myservo2.write(70);
  delay(k);
    }
    //2-B
    else if(msg==4)
    {

myservo1.write(70);

myservo2.write(70);
      delay(k);
    }
    //3-L
    else if(msg==3)
    {
myservo1.write(110);

myservo2.write(110);
      delay(k);
    }
    //4-R
    else if(msg==2)
    {

myservo1.write(70);

myservo2.write(110);
      delay(k);
    }
myservo1.write(90);
myservo2.write(90);
           delay(k);
  Serial.flush();
}}
```