Advanced Mechatronics

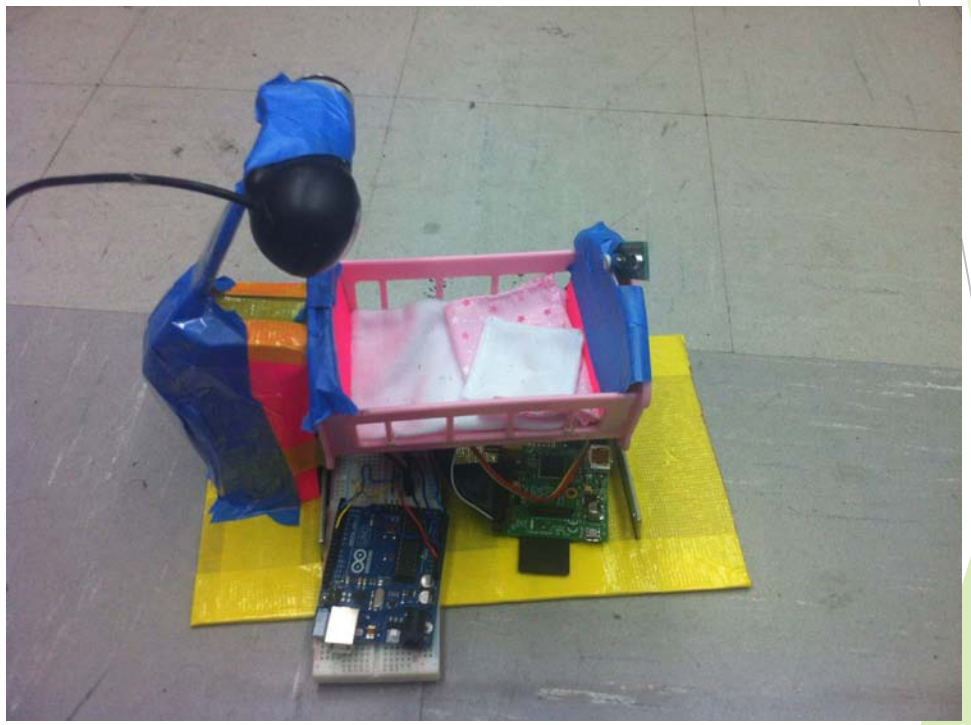
3rd Mini project: Raspberry Pi

Sweet Dreams

Jose Antonio De Gracia Gómez, Amartya Barua

Intelligent Cradle

- Detect an infant's movement and weeping
- Send an email when motion is detected
- Oscillate the cradle and play a lullaby when baby is crying
- Display baby image through internet



How Does It Work?

Raspberry Pi

Arduino UNO

parallax sound impact sensor

Speakers

Webcam

E- mail to laptop or mobile phone

parallax standard servo motor

Components

- Raspberry Pi
- Arduino UNO
- Impact Sound Detector
- Parallax Standard Servo Motor
- Speakers
- Webcam
- Laptop / Mobile phone

Raspberry Pi

- Raspberry Pi and Arduino are the body and the brain of our project.
- Low cost, Model B 39.95$
- Two USB port a USB Ethernet adapter
- SD Card
- Display: HDMI, RCA Video, Audio
- General purpose input/output (GPIO): 26Pins
- Camera (CSI connector)
- SoC—Broadcom BCM2835, located beneath a Hynix memory chip
- ARM1176JZFS, with floating point, running at 700Mhz
- BCM2835 is based on an ARM processor
- OS: GNU/Linux
- We interface Arduino with RPI using I2C, GPIO 0 (SDA), GPIO 1 (SCL) and Ground.

Arduino UNO

- Arduino is a tool for making smart devices that can sense and control
- The physical worldArduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software.
- Arduino microcontroller is programmed using:

–Arduino programming language (based on Wiring —C libraries)

–Arduino development environment (based on Processing), Arduino integrated development environment (IDE)

- We use Arduino to receive the signal form the Impact Sound detector, activate the servo and notify RPI that the baby is crying.
- Arduino is connected to RPI using I2C, Pin A4 (SDA), Pin A5 (SCL), and Ground

Impact Sound Detector

- In order to detect the cry of an infant, the Parallax 0.6 x 1.5in Sound Impact Sensor is used. It works as a device that provides an output signal voltage corresponding to sound being detected by a microphone up to 3 meters away. The microphones has the ability to detect pressure differences in the air and transform them into electrical signals.
- The sensor is connected to Arduino. When the sensor registers the cry for the infant, Arduino will send a signal to Raspberry Pi. At the same time, the Servomotor will start to move

Parallax Standard Servo Motor

- A servo motor is a rotary actuator that allows for precise control of angular position, velocity and acceleration. It consists of a suitable motor coupled to a sensor for position feedback.
- The operation of the Intelligent Cradle is simple, the servo motor is connected to Arduino, which, through a smart code, can tell the servo what we want it to do
- The way in which out servomotor works is easy. When the baby starts crying, Impact Sound detector, will perceive the baby's weeping and will send a signal to Arduino, which automatically, will send another signal to the servo motor to start to move, and to RPI for playing music.
- This Servo can oscillate from 0 to 180 degrees. And this is very important because we don't want to oscillate more than this, in fact, this is even more than the necessary but, of course we will limit our interval of oscillation from 60 to 120 degrees. This is the reason because we use this servo and not Parallax Continuous Rotation.

Webcam

- Webcam is attached to the automatic Cradle and it detects motion.
- If the baby is not crying but is moving, the camera will detect the movement, will record a 5 sec video, and will send an email notifying the parents that the baby is moving.
- This is a security method to avoid kidnapping.
- If they baby instead, is crying, the cradle will start to move, so movement will also be detected, and it will be notified to the parents via e-mail.
- The parents can see the image of the baby all the time via Internet.

Speakers

- It’s well known that a nice song helps a lot in order to get the baby sleep, and that is because we have included in our project a couple of speakers.
- Speakers are connected to the Raspberry Pi through the audio output.
- When the impact Sound Detector detects the crying of the baby Raspberry Pi will play a nice Lullaby until the baby stops crying.
- Low volume in order to not disturb the baby and in order to not confuse the microphone.

Mechanical Design

- For the mechanical design we decided to build a structure for making it, sway easier and smoother. As you can see in the pictures we used two pieces T-shaped that we plugged into the cradle. One piece is just plugged as a support and the other is plugged to the servomotor which is plugged into the cradle.
- As we explained, when the baby starts to cry the Sound Detector will send a signal to Arduino and it will activate the servomotor. As the servo is plugged into the cradle, it will start moving and swinging while the speakers play a lullaby for making the baby to fall sleep again. The cradle will keep moving until the infant sleeps .

Motion Capture

- Package: Motion
- Compares current frame with previous frames in buffer (number of changed pixels after noise filtering
- Specify the how many pre and post capture frames are stored
- Specify the length of the movie
- Webcam streaming at port 8081
- Password protection available
- The configuration file is located at /etc/motion/motion.conf

UPLOADER.PY & UPLOADER.CFG

- Python script used to upload the movies saved in /home/pi/monitor
- Utilizes google data API and SMTP library to upload videos to google drive and send alert to gmail account
- All the user data (password, account name etc.) are stored in /etc/motion/uploader.cfg
- The uploader.py located at /etc/motion/uploader.py is run automatically every 2 minutes using the ***cron*** feature in linux
- $ sudo crontab –e (note: using the root cron)
- Add */2 * * * * python /etc/motion/uploader.py

FILEDELETE.PY

- Deletes all the files that are older than the specified minutes
- Located at /etc/motion/filedelete.py
- Run automatically every 3 minutes by ***cron***
- $ sudo crontab –e
- Add */3 * * * * pyton /etc/motion/filedelete.py

MICDETECT.PY

- Utilizes i2c-tools and pygame module
- Receives impact detector values from Arduino via i2c
- Based on sensor value plays a song using pygame music library.
- Run continuously using ***cron***
- $ sudo crontab –e
- Add * * * * * python /home/pi/micdetect.py
- Remove I2C from blacklist: comment out blacklist i2c-bmc2708
- Add /etc/modules i2c-dev
- Install i2c-tools and add user ***pi*** to i2c
- Install python-smbus

Arduino Code

```
#include <Servo.h>
#include <Wire.h>
#define SLAVE_ADDRESS 0x04
int micinputpin = 2;
int volume;
Servo ServoMotor;
int i, j, k;

void setup()
{
  // initialize i2c as slave
  Wire.begin(SLAVE_ADDRESS);
  // define callbacks for i2c communication
  Wire.onRequest(sendData);

  ServoMotor.attach(6);
  ServoMotor.write(90);
}

void loop()
{

  volume = digitalRead(micinputpin);
  if (volume == 1)
  {
  volume=0;

 for (i=90;i<=110;i++)
 {
 ServoMotor.write(i);
 delay(50);
 }

 for (j=110;j>=70;j--)
 {
 ServoMotor.write(j);
 delay(50);}

  for (k=70;k<=90;k++)
 {
 ServoMotor.write(k);
 delay(50);
 }
  }
  else
  {
  }
}

 void sendData(){
  volume = digitalRead(micinputpin);
  if(volume == 1){
  Wire.write(volume);
  }
}
```

Python Code (Readmic.py and Filedelete.py)

```
import smbus
import pygame
#import time

bus = smbus.SMBus(1)
pygame.mixer.init()
pygame.mixer.music.load("/home/pi/Star3.wav")

# This is the address in the Arduino Program
address = 0x04

##def writeIns(value):
##    bus.write_byte(address, value)
##    # bus.write_byte_data(address, 0, value)
##    return -1

def readMic():
    volume = bus.read_byte(address)
    # number = bus.read_byte_data(address, 1)
    return volume

while True:
    #var = input("Enter 1 - 9: ")
    #if not var:
        #continue

    vol = readMic()
    if vol == 1:
        vol = 0
        if not pygame.mixer.music.get_busy():
            pygame.mixer.music.play()
```

```
from path import path
import time

MINUTES = 6
remved = 0
targetdir = path('/home/pi/monitor/')
time_in_s = time.time() - (MINUTES * 60)

for i in targetdir.walk():
    if i.isfile():
        if i.mtime <= time_in_s:
            i.remove()
            remved += 1
```

Python Code (Uploader.py)

```
#!/usr/bin/python2

import smtplib
from datetime import datetime

import os.path
import sys
import base64

import gdata.data
import gdata.docs.data
import gdata.docs.client
import ConfigParser

class MotionUploader:
    def __init__(self, config_file_path):
        # Load config
        config = ConfigParser.ConfigParser()
        config.read(config_file_path)

        # GMail account credentials
        self.username = config.get('gmail', 'user')
        self.password = config.get('gmail', 'password')
        self.from_name = config.get('gmail', 'name')
        self.sender = config.get('gmail', 'sender')

        # Recipient email address (could be same as from_addr)
        self.recipient = config.get('gmail', 'recipient')

        # Subject line for email
        self.subject = config.get('gmail', 'subject')

        # First line of email message
        self.message = config.get('gmail', 'message')

        # Folder (or collection) in Docs where you want the videos to go
        self.folder = config.get('docs', 'folder')

        # Options
        self.delete_after_upload = config.getboolean('options', 'delete-after-upload')
        self.send_email = config.getboolean('options', 'send-email')

        self._create_gdata_client()
```

Python Code (Uploader.py)

```
def _create_gdata_client(self):
    """Create a Documents List Client."""
    self.client = gdata.docs.client.DocsClient(source='motion_uploader')
    self.client.http_client.debug = False
    self.client.client_login(self.sender, self.password, service=self.client.auth_service, source=self.client.source)

def _get_folder_resource(self):
    """Find and return the resource whose title matches the given folder."""
    col = None
    for resource in self.client.GetAllResources(uri='/feeds/default/private/full/-/folder'):
        if resource.title.text == self.folder:
            col = resource
            break
    return col

def _send_email(self,msg,imgpath):
    '''Send an email using the GMail account.'''
    senddate=datetime.strftime(datetime.now(), '%Y-%m-%d')
    # Read a file and encode it into base64 format
    fo = open(imgpath, "rb")
    filecontent = fo.read()
    encodedcontent = base64.b64encode(filecontent)  # base64
    imgfile=os.path.basename(imgpath)
    marker = "AUNIQUEMARKER"
    p1="Date: %s\r\nFrom: %s <%s>\r\nTo: %s\r\nSubject: %s\r\nContent-Type: multipart/mixed; boundary=%s\r\n--%s\r\n" % (senddate, self.from_name, self.sender, self.recipient, self.subject, marker, marker)
    p2="Content-Type: text/plain\r\nContent-Transfer-Encoding:8bit\r\n\r\n%s\r\n--%s\r\n" % (msg, marker)
    p3="Content-Type: image/jpeg; name=""%s""\r\nContent-Transfer-Encoding:base64\r\nContent-Disposition: attachment; filename=%s\r\n\r\n%s\r\n--%s--\r\n" % (imgfile, imgfile, encodedcontent, marker)

    server = smtplib.SMTP('smtp.gmail.com:587')
    server.starttls()
    server.login(self.username, self.password)
    server.sendmail(self.sender, self.recipient, p1+p2)
    server.quit()

def _upload(self, video_file_path, folder_resource):
    '''Upload the video and return the doc'''
    doc = gdata.docs.data.Resource(type='video', title=os.path.basename(video_file_path))
    media = gdata.data.MediaSource()
    media.SetFileHandle(video_file_path, 'video/avi')
    doc = self.client.CreateResource(doc, media=media, collection=folder_resource)
    return doc

def upload_video(self, video_file_path):
    """Upload a video to the specified folder. Then optionally send an email and optionally delete the local file."""
    folder_resource = self._get_folder_resource()
    if not folder_resource:
        raise Exception('Could not find the %s folder' % self.folder)

    doc = self._upload(video_file_path, folder_resource)
```

Python Code (Uploader.py)

```
        if self.send_email:
            video_link = None
            for link in doc.link:
                if 'video.google.com' in link.href:
                    video_link = link.href
                    break
            # Send an email with the link if found
            msg = self.message
            if video_link:
                msg += '\n\n' + video_link
            imgfile = os.path.splitext(video_file_path)[0] + "-00.jpg"
            self._send_email(msg,imgfile)
            if self.delete_after_upload:
                os.remove(imgfile)

        if self.delete_after_upload:
            os.remove(video_file_path)

if __name__ == '__main__':
    try:
        if len(sys.argv) < 3:
            exit('Motion Uploader - uploads videos to Google Drive\n   by Jeremy Blythe (http://jeremyblythe.blogspot.com)\n\n   Usage: uploader.py {config-file-path} {video-file-path}')
        cfg_path = sys.argv[1]
        vid_path = sys.argv[2]
        if not os.path.exists(cfg_path):
            exit('Config file does not exist [%s]' % cfg_path)
        if not os.path.exists(vid_path):
            exit('Video file does not exist [%s]' % vid_path)
        MotionUploader(cfg_path).upload_video(vid_path)
    except gdata.client.BadAuthentication:
        exit('Invalid user credentials given.')
    except gdata.client.Error:
        exit('Login Error')
    except Exception as e:
        exit('Error: [%s]' % e)
```

Budget

Name	Costs
Raspberry Pi	$39.95
Arduino	$24.95
Parallax Standard Servo	$12.99
Sound Impact Sensor	$9.99
Speakers	10$
Webcam	$29.99
Miscellaneous	$15
Approximate Value	$142.87

Purpose

- Help new parents.
- Help in avoiding kidnappings.
- Total control of the baby while sleeping.
- Notify when the baby is moving or crying.

Questions?