TOT BOT

INTERRIM PROJECT REPORT

Submitted in Partial Fulfillment of

the Requirements for

the Degree of

MASTER OF SCIENCE (Mechatronics and Robotics)

at the

NEW YORK UNIVERSITY
TANDON SCHOOL OF ENGINEERING

by

Tanaya Bhave

May 2017

TOT BOT

INTERRIM PROJECT REPORT

Submitted in Partial Fulfillment of

the Requirements for

the Degree of

MASTER OF SCIENCE (Mechatronics and Robotics)

at the

NEW YORK UNIVERSITY
TANDON SCHOOL OF ENGINEERING

by

Tanaya Bhave

May 20*17*

Approved:

_____
Advisor Signature

_____
Date

_____
Department Chair Signature

_____
Date

University ID: **N19181894**
Net ID: **tb1761**

ABSTRACT

TOT BOT

by

Tanaya Bhave

Advisor: Prof. Vikram Kapila, Ph.D., P.E.

Submitted in Partial Fulfillment of the Requirements for

the Degree of Master of Science (Mechatronics and Robotics)

May 20*17*

Environmental Stimulation plays a big role in the intellectual development of a person, especially in the first two years of life(infancy). Studies and experiments have shown how enabling babies to explore their surroundings can create an encouraging atmosphere to grow. An assistive tool as such would particularly help infants with physical disabilities. Tot Bot is a mobility assistant for toddlers with special needs. This exhibit will demonstrate a mobility device that uses a tablet-based visual interface for smart navigation. Tot Bot enables a kid to see its surrounding on the tablet screen and then reach a selected point by a touch on the screen. The Tot Bot follows the simple principle that a kid would reach forward to grab an object that it wants, and in this case, is able to see in the tablet screen, hence touches it. Steering should be relatively simple for a child in this as he/she simply has to want something and reach for it on the screen. The Tot Bot utilizes a simple app to communicate with the microcontroller and send data of the desired position to the mobile robot. Once you tap on the screen, the position of touch is translated the object's position using the tablet camera.

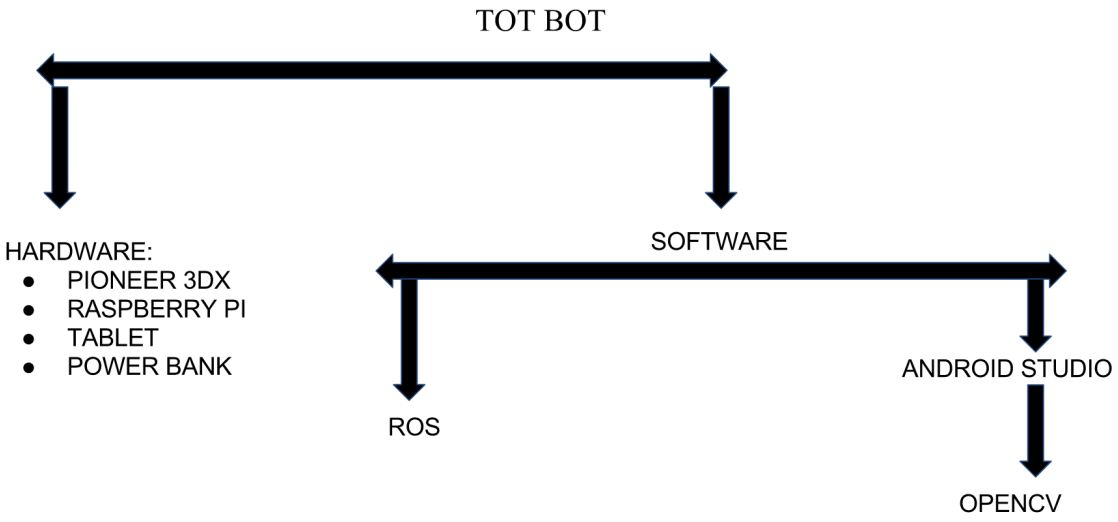
LITERATURE REVIEW

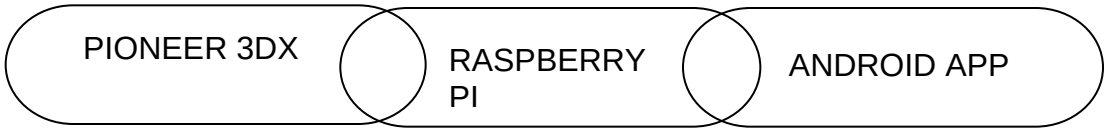
According to studies, Environmental Stimulation plays a big role in the advancement of infants ranging in the early two years of life. Infants who are not exposed to enough environmental stimulation, have shown to have lower IQs in later ages. Infants who are brought up in orphanages as compared to foster homes have shown IQ differences of up to 20 points. Socio economic factors have also proved to make a big difference as it ultimately impacts the environment of the infant. In such circumstances, an infant suffering with a disability or a handicapping condition is not able to explore its environment due to its limited mobility. at all hence receives no stimulation to develop to it full potential. It can delay or affect even the way a child plays, the kinds of activities the child engages in, and the child's ability to use objects as an avenue to learning and generalizing new skills or concepts. The child may have difficulty moving to the materials or areas available for play. He/She may have difficulty manipulating materials in a constructive or meaningful way. Hence limited mobility becomes a hindrance in the development of children.

Ithaca college, New York is involved in a project with mobile-robots for infants called WeeBots. They are using Nintendo wii balance boards on top of Pioneer mobile robots. The balance board detect the shift in weight as the child leans in a direction and then moves accordingly with sonars to the front and back of the robot to detect objects. They could train 90 percent of the infants in the range of five to nine months to maneuver the WeeBot. They sat disabled kids and had mixed results. In another trial, the researchers modified their WeeBot with a button panel to drive. A 3-year-old boy with cerebral palsy who could not sit upright was sat in the WeeBot, but he kept overshooting his target and got frustrated. Another 15-month-old baby with cerebral palsy was sat in the WeeBot and he could train himself to steer with the button panel.

In another study in Columbia University, joysticks were used to train toddlers seated on mobile robots, to drive indoors amid obstacles. Toddlers who are unable to independently move around, but can drive mobile robots, driving safely inside their homes becomes important. In their research, they ten typically developing infants within an obstacle course and a toddler with spina-bi fida who could not independently walk. The researchers have used algorithms based on artificial object field for object avoidance and created a force field on the joystick to help train the kid to drive. They successfully tested the force feedback joystick with better learning results.

STRUCTURE OF THE ROBOT



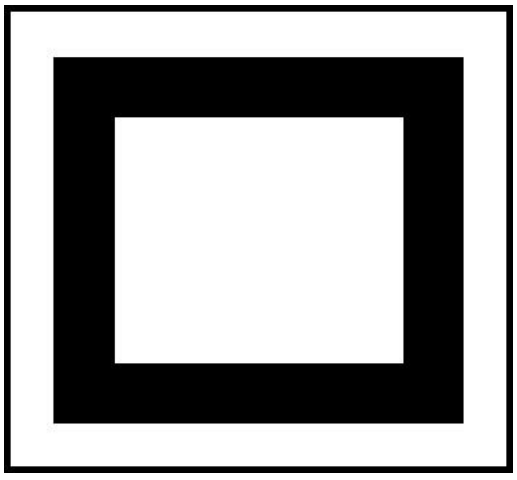


PROGRESS SO FAR

- Control of Pioneer based on ultrasonic sensors is done using ROS.
- Improved Structural Design of the child seat.

- Android application to recognize the tags.

To detect distance, Tags are being used. A HAAR cascade classifier was built to recognize the tag and estimate distance with it. The advantage of having a tag to control the movement is that the parents can deem zones safe for the child.



ANDROID APPLICATION

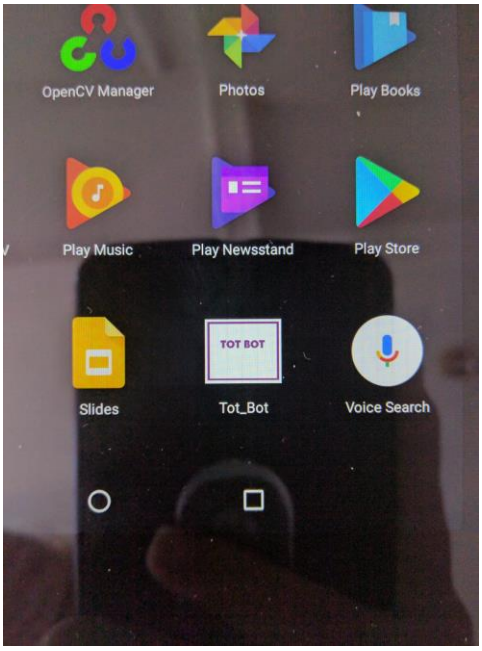
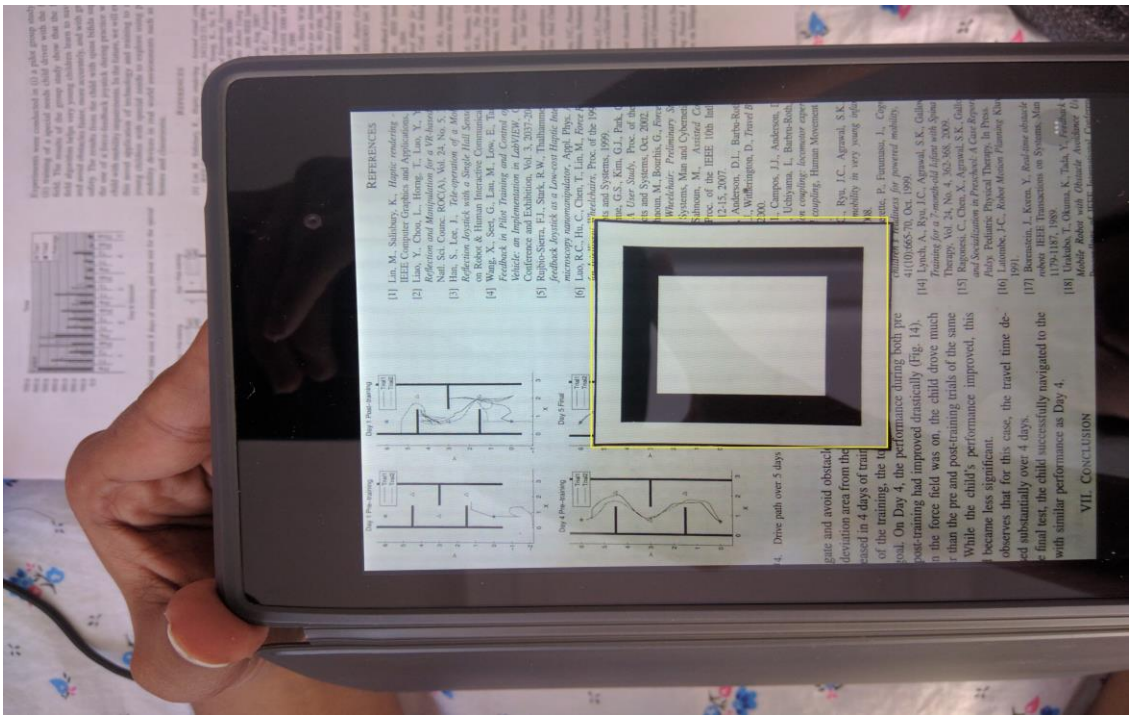


Figure: Tag detection in Application



CODE FOR THE ANDROID APP

```
package com.example.tanaya.tot_bot;

import android.app.Activity;
```

```
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;

import org.opencv.android.CameraBridgeViewBase;
import org.opencv.android.JavaCameraView;
import org.opencv.android.Utils;
import org.opencv.core.Core;
import org.opencv.core.CvType;
import org.opencv.core.MatOfRect;

import org.opencv.android.BaseLoaderCallback;
import org.opencv.android.LoaderCallbackInterface;
import org.opencv.android.OpenCVLoader;
import org.opencv.core.CvType;
import org.opencv.core.Mat;
import org.opencv.core.Scalar;
import org.opencv.core.Rect;
import org.opencv.core.Size;
import org.opencv.imgproc.Imgproc;
import org.opencv.objdetect.CascadeClassifier;
import org.opencv.objdetect.Objdetect;

import android.util.Log;
import android.view.SurfaceView;
import android.view.View;
import android.view.WindowManager;
import android.widget.Toast;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.InputStream;

public class MainActivity extends Activity implements
CameraBridgeViewBase.CvCameraViewListener {

    private static final String TAG = "OCVSample::Activity";

    public CameraBridgeViewBase mOpenCvCameraView;
    private boolean              mIsFrontCamera = false;
    private MenuItem             mItemSwitchCamera = null;
    private Mat mRgba;
    private Mat mRgbaT;
    private Mat mRgbaF;
    private CascadeClassifier  haarCascade;

    private BaseLoaderCallback mLoaderCallback = new BaseLoaderCallback(this) {
                @Override
                public void onManagerConnected(int status) {
                    switch (status) {
                        case LoaderCallbackInterface.SUCCESS:
                        {
                            Log.i(TAG, "OpenCV loaded successfully");
                            try{
                                InputStream is =
getResources().openRawResource(R.raw.cascade);
                                File cascadeDi =
getDir("cascade",Context.MODE_APPEND);
                                File mCascadeFile = new File(cascadeDi,
```

```
"cascade.xml");
                                FileOutputStream os = new
FileOutputStream(mCascadeFile);
                                byte[] buffer = new byte[4096];
                                int bytesRead;
                                while((bytesRead = is.read(buffer)) != -1)
                                {
                                    os.write(buffer, 0, bytesRead);
                                }
                                is.close();
                                os.close();
                                haarCascade = new
CascadeClassifier(mCascadeFile.getAbsolutePath());
                                if (haarCascade.empty())
                                {
                                    Log.i("Cascade Error","Failed to load
cascade classifier");
                                    haarCascade = null;
                                }
                            }
                            catch(Exception e)
                            {
                                Log.i("Cascade Error: ","Cascade not found");
                            }
                            mOpenCvCameraView.enableView();
                        } break;
                        default:
                        {
                            super.onManagerConnected(status);
                        } break;
                    }
                }
    };



    @Override
    protected void onCreate(Bundle savedInstanceState) {
        Log.i(TAG, "called onCreate");
        super.onCreate(savedInstanceState);
        getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);

        setContentView(R.layout.activity_main);

        mOpenCvCameraView = (CameraBridgeViewBase)
findViewById(R.id.java_surface_view);
        mOpenCvCameraView.setCvCameraViewListener(this);

    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.menu_main, menu);
        Log.i(TAG, "called onCreateOptionsMenu");
        mItemSwitchCamera = menu.add("Toggle Front/Back camera");
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        String toastMesage = "";

        if (item == mItemSwitchCamera) {
            mOpenCvCameraView.setVisibility(SurfaceView.GONE);
```

```
            mIsFrontCamera = !mIsFrontCamera;

            if (mIsFrontCamera) {
                mOpenCvCameraView = (CameraBridgeViewBase)
findViewById(R.id.java_surface_view);
                mOpenCvCameraView.setCameraIndex(1);
                toastMesage = "Front Camera";
            } else {
                mOpenCvCameraView = (CameraBridgeViewBase)
findViewById(R.id.java_surface_view);
                mOpenCvCameraView.setCameraIndex(-1);
                toastMesage = "Back Camera";
            }

            mOpenCvCameraView.setVisibility(SurfaceView.VISIBLE);
            mOpenCvCameraView.setCvCameraViewListener(this);
            mOpenCvCameraView.enableView();
            Toast toast = Toast.makeText(this, toastMesage, Toast.LENGTH_LONG);
            toast.show();
        }

        return true;
    }
    @Override
    public void onCameraViewStarted(int width, int height) {
        mRgba = new Mat(height, width, CvType.CV_8UC4);
    }

    @Override
    public void onCameraViewStopped() {

    }

    @Override
    public Mat onCameraFrame(Mat inputFrame) {
        CameraBridgeViewBase.CvCameraViewFrame imput2mat=
(CameraBridgeViewBase.CvCameraViewFrame) inputFrame;
        Mat mGray = (Mat) imput2mat.gray();
        mRgba = imput2mat.rgba();
        if (mIsFrontCamera)
        {
            //Core.transpose(mRgba, mRgbaT);
            //Imgproc.resize(mRgbaT, mRgbaF, mRgbaF.size());
            Core.flip(mRgba, mRgba, 1);
        }


        //Detecting tags in the frame
        MatOfRect tags = new MatOfRect();
        if(haarCascade != null)
        {
            haarCascade.detectMultiScale(mGray, tags, 1.1, 2, 2, new
Size(200,200), new Size());
        }

        Rect[] tagsArray = tags.toArray();
        for (int i = 0; i < tagsArray.length; i++)
            Core.rectangle(mRgba, tagsArray[i].tl(), tagsArray[i].br(), new
Scalar(100), 3);
        return mRgba;
    }

    @Override
    public void onPause()
```

```
    {
        super.onPause();
        if (mOpenCvCameraView != null)
            mOpenCvCameraView.disableView();
    }

    @Override
    public void onResume()
    {
        super.onResume();
        if (!OpenCVLoader.initDebug()) {
            Log.d(TAG, "Internal OpenCV library not found. Using OpenCV Manager
for initialization");
            OpenCVLoader.initAsync(OpenCVLoader.OPENCV_VERSION_2_4_9, this,
mLoaderCallback);
        } else {
            Log.d(TAG, "OpenCV library found inside package. Using it!");

mLoaderCallback.onManagerConnected(LoaderCallbackInterface.SUCCESS);
        }
//        OpenCVLoader.initAsync(OpenCVLoader.OPENCV_VERSION_2_4_9, this,
mLoaderCallback);
    }

    public void onDestroy() {
        super.onDestroy();
        if (mOpenCvCameraView != null)
            mOpenCvCameraView.disableView();
    }
}
```

FUTURE GOALS

- To integrate the application with the robot.
- Conduct trials with toddlers and understand their learning on the robot.