Indoor Navigation for Blind using Android Smartphone

CS676A
Introduction to Computer Vision & Image Processing
Prof. Vinay P. Namboodiri,
Department of Computer Science & Engineering, IIT Kanpur

Done by -

<table>
<thead>
<tr>
<th>Group 27</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pankaj Gupta (11483), 3rd year, Dept. Of Mathematics &amp; Statistics</td>
</tr>
<tr>
<td>Satyam Kumar Shivam (11658), 3rd year, Dept. Of Mathematics &amp; Statistics</td>
</tr>
</tbody>
</table>
Abstract

This project has also been submitted as an entry for a competition organised by Goldman Sachs ABLE Solutions. The competition is currently going on and the problem statement reads as:

“Participating teams are to develop a working prototype of a mobile app (on Android Platform) that will help an individual having visual impairment navigate an indoor environment (e.g. office building)”

In this report, we propose a navigation system for smartphones capable of guiding users accurately to their destinations in an unfamiliar indoor environment, without requiring any expensive alterations to the infrastructure or any prior knowledge of the site’s layout.

We have used OpenCV libraries for Android and inbuilt accelerometer & magnetometer in our application. We have done the testing of our prototype using Google Nexus-4 in the CSE building, IIT Kanpur and it gave fairly good results. All the programming have been done in Android. The mobile has to be held by user at 30 degree approx from the horizontal so that it may tackle the obstacle which come in its path as well as recognize the position markers (which we have proposed in our solution).

Motivation

Navigation for a visually-impaired person in an unknown indoor environment is near to impossible. Even in a known-indoor system, it is very difficult for him to navigate. The aim is to make office/workplaces more open and inclusive to people with disabilities and helping attract and retain differently-abled talent into the workforce.

We talked to a blind person before starting our app-development process. The following points came up as a summary after the talk.

1 - App should be able to detect doors, stairs, wall and any other obstacle.
2 - Recognize which locations are allocated to what purpose
3 - Must work for different buildings
4 - Obstacle should be detected at an appropriate distance
5 - Sound feedback should be clear

## Salient Features of our Approach

- **High accuracy**: The application should consistently guide users to their destinations and also detect obstacle from a reasonable distance.

- **Low-cost**: No external sensors were used. Only mobile back-camera was used. Even though using external sensors would increase our accuracy but they would also add discomfort for the user. External sensors would have their further maintenance cost. Hence, we decided on not using any external sensors.

- **No pre-loaded indoor maps**: Preloaded map of the entire building would diminish the flexibility of a solution.

- **Intuitive user interface and feedback**: The app should be easy to use and also it should give proper feedback to user so that user may act accordingly. We are using sound feedback in our app.

There are three major parts of our app:

1 - **Direction (Indicators)/Markers**: These markers encode information about directions and destination and are easy to detect.
2 - **Obstacle Detection**: Computer Vision libraries are used to detect obstacle from an appropriate distance.
3 - **Dead Reckoning**: If users lose his way, using Inertial Navigation system user may return to the previous marker.
Previous Work

In the past few years, a great amount of interest has been shown to develop indoor navigation systems for the common user. Researchers have explored possibilities of indoor positioning systems that use Wi-Fi signal intensities to determine the subject's position[1][2]. Other wireless technologies, such as Bluetooth[1], ultra-wideband (UWB)[4] and radio-frequency identification (RFID)[3], have also been proposed. Although some of these techniques have achieved fairly accurate results, they are either highly dependent on fixed-position beacons or have been unsuccessful in porting the implementation to a ubiquitous hand-held device.

Many have approached the problem of indoor localisation by means of inertial sensors. A foot-mounted unit has recently been developed to track the movement of a pedestrian[5]. Some have also exploited the smartphone accelerometer and gyroscope to build a reliable indoor positioning system. A system developed by Microsoft[6] relies upon a pre-loaded indoor floor map and does not yet support any navigation.

An altogether different approach applies vision. In robotics, simultaneous localisation and mapping (SLAM) is used by robots to navigate in unknown environments[7]. In 2011, a thesis considered the SLAM problem using inertial sensors and a monocular camera[8]. It also looked at calibrating an optical see-through head mounted display with augmented reality to overlay visual information. Localisation using markers have also been proposed. One such technique uses QR codes to determine the current location of the user[9]. There is also a smartphone solution, which scans square fiducial markers in real time to establish the user’s position and orientation for indoor positioning[10].

Direction Indicators/Markers

Markers should be such which stand out from the surrounding environment, We had many options for markers such as Barcode Scanning, Location fingerprinting, Triangulation, QR-codes. But we decided upon using QR-codes.

**Cicle with a letter embed**: We firstly tried a very simple marker. It looks like a circle with a letter (for e.g. L or R) written inside it. Due to orientation of mobile, it's detection was very problematic using Hough Circle Transform.
Barcode Scanning: Barcodes could be placed in various locations across the building, encoded with their respective grid coordinates. The smartphone camera could then be used to take a picture of the barcode and decode the encoded data. It had many parallel lines and hence may cause trouble in properly detecting stairs and few obstacle. Hence we rejected it. It's detection is orientation dependent.

Location fingerprinting: Location fingerprinting is a technique that compares the received signal strength (RSS) from each wireless access point in the area with a set of pre-recorded values taken from several locations. The location with the closest match is used to calculate the position of the mobile unit. With a great deal of calibration, this solution can yield very accurate results. However, this process is time-consuming and has to be repeated at every new site.

QR-codes: They are cheap to produce. They have an advantage over barcode by being orientation independent. They are detection less dependent on light-conditions. Hence we decided to use it. Probably integrating this technique with technologies like RFID may give better results but we haven't tried it.
Obstacle Detection:

We have explored many methods for obstacle detection:

1 - Canny Edge Detection and Hough Line: After applying Canny Edge detection on the image received from camera, we applied hough line function on it. These functions are available in ImageProc package of OpenCV library.

```cpp
void Canny (InputArray image, OutputArray edges, double threshold1, double threshold2 )

void HoughLinesP(InputArray image, OutputArray lines, double rho, double theta, int threshold, double minLineLength, double maxLineGap)
```

For detecting the obstacle -

```cpp
if(lines.cols() > lineThreshold) {
    playSound("wait")
}
```

For detection of QRcode, we divide the screen into 9 parts (bins). The place where QRcode appears, lines intensity becomes very high. Hence, for detection of QRcode, we compare intensity in each bin with a threshold.

```cpp
for(int i=0; i<3; i++) {
    for(int j=0; j<3; j++) {
        if(bins[i][j] > 110)
            playSound("Possibility of QR-code")
    }
}
```

2 - Homography Estimation:

Two image frames are compared for homography to match corresponding feature points. It helps to track the obstacle. Obstacle tracking is required in order to give directions to user to avoid it.

```cpp
Mat findHomography(InputArray srcPoints, InputArray dstPoints, int method=0, double ransacReprojThreshold=3, OutputArray mask=noArray() )
```
3 - Optical Flow :
The optical flow information is extracted from the image sequence in order to be used in the navigation algorithm. The optical flow provides very important information about the environment, like: the obstacles disposition, the agent heading, the time to collision and the depth. The strategy consists in balancing the amount of left and right side flow to avoid obstacles, this technique allows agent navigation without any collision with obstacles.

We tested with two optical flow algorithms. These algorithms are present in Video package of OpenCV library for android:

1 - calcOpticalFlowSF : Calculate an optical flow using “SimpleFlow” algorithm

```c
void calcOpticalFlowSF(Mat& from, Mat& to, Mat& flow, int layers, int averaging_block_size, int max_flow)
```

2 - calcOpticalFlowPyrLK : Calculates an optical flow for a sparse feature set using the iterative Lucas-Kanade method with pyramids

```c
void calcOpticalFlowPyrLK(InputArray prevImg, InputArray nextImg, InputArray prevPts, InputOutputArray nextPts, OutputArray status, OutputArray err)
```

Dead Reckoning

1 - Direction and Orientation estimation: Accelerometer and geo-magnetic field sensors are used to provide an accurate representation of the device’s orientation. Orientation angles for each axis are calculated from rotation matrix. The Android sensor API function `getRotationMatrixFromVector()` and also `getOrientation()` are used to calculate these angles.
2 - Pedometer: Pedometer is used to count the number of footsteps the visually impaired person has taken. Pedometer algorithm is based upon detecting the peaks and troughs from the time-domain waveform. When a peak was detected, the next trough is searched. Similarly, when a trough was detected, the next peak is searched. Difference between a peak and a trough is calculated by taking first derivative of acceleration data.

The X and Y coordinates are found as follows:

\[
X_{\text{new}} = X_{\text{old}} + \cos(\text{orientation}) \times \text{stridelength}; \\
Y_{\text{new}} = Y_{\text{old}} + \sin(\text{orientation}) \times \text{stridelength};
\]
Summary

Our approach has three main parts Markers, Obstacle Avoidance and Dead Reckoning. We used QRcodes as markers. For obstacle avoidance and reading QRcodes we used OpenCV library for Computer Vision. For Dead Reckoning we used sensors of mobile. The app gave fairly good result when tested.

Future Work

1 – MonoSLAM: The core of the approach of this algorithm[7] is the online creation of a sparse but persistent map of natural landmarks within a probabilistic framework. This paper presents an active approach to mapping and measurement, the use of a general motion model for smooth camera movement, and solutions for monocular feature initialization and feature orientation estimation. We can use this for giving more integrity to our app.

2 - Background Subtraction: Subtracting background will eliminate floor pattern and hence we would have more accuracy in obstacle detection.

3 - Staircase Detection: Staircase detection would enable us to navigate to different floors which would add generality to our app. The paper[11] by Stephen Se and Michael Brady looks very promising for this.

4 - Improving Pedometer Accuracy: Our current implementation of Pedometer performs if we walk a little force. We have to improve it so as to extend it to normal step.

Bibliography


location and tracking system. In 2, pages 775 – 784, March 2000.


