Recent Innovations in Mechatronics (RIiM) Vol. 2. (2015). No. 1-2.
DOI: 10.17667/riim.2015.1-2/2.

Android based autonomous mobile robot

Gergely Nagymáté
Department of Mechatronics Optics and Engineering Informatics,
Budapest Univ. of Technology and Economics
Budapest, Hungary
nagymate.gergely@gmail.com

***Abstract*—The spreading of mobile robots is getting more significant nowadays. This is due to their ability to perform tasks that are dangerous, uncomfortable or impossible to people. The mobile robot must be endowed with a wide variety of sensors (cameras, microphones, proximity sensors, etc.) and processing units that makes them able to navigate in their environment. This generally carried out with unique, small series produced and thus expensive equipment. This paper describes the concept of a mobile robot with a control unit integrating the processing and the main sensor functionalities into one mass produced device, an Android smartphone. The robot is able to perform tasks such as tracking colored objects or human faces and orient itself. In the meantime, it avoids obstacles and keeps the distance between the target and itself. It is able to verbally communicate wit .**

Keywords— mobile robot, Android smartphone, human-machine interaction

I. INTRODUCTION

The computing capacity of today's mobile phones is many times greater than the one's was used for Moon landing [1], yet commonly used for entertainment purposes. These pocket computers are equipped with many different kind of sensors that are also used in robotics. Advanced robots often need GPS, inertial sensors and e-compass, for navigation and orientation, cameras for image processing, microphone as audio input and wireless connectivity. These are all included in smartphones. This paper not only draws attention to smartphones for their integrated sensors, but the enormous software support and open source solutions.

There have already been researches using smartphones in robotics. A common application uses smartphones as user interfaces for remote controlling [2][3]. A semi autonomous robot presented by Stansfield and Bothma is still remote controlled and has Android device on both sides, but on the robot side it is used for interfacing an external distance sensor and also implements a simple collision avoidance algorithm [4]. A mobile robot presented by Lim, Lee and Tewolde is additionally to the previous robot uses the inertial sensors of the smartphone to improve indoor navigation capabilities of the robot [5]. Another robot takes advantage of the camera integrated in the smartphone for implementing image processing for line following [6].

Smartphone based mobile robots are used for educational purposes, where smartphone based image processing is also implemented [7] [8]. There are already robot platforms for smartphones that are commercial products with easy to use developer environment designed for children educational programming [9] [10].

The present project aimed to build a mobile robot that is capable of fulfilling the following objectives while taking advantage of the advanced functionality of an everyday smartphone:

1) capable of tracking human faces and colored object
2) implements collision avoidance while following its target
3) capable of verbal communication with humans, receives orders and answers simple questions

Such robot that can understand human questions and intelligently answer them seemed very high tech a few years ago, but today this technology is in everyone's pocket [11] [12].

The robot is a car-like vehicle with ultrasonic distance sensors and a 2 DOF consol for holding the Android phone. Two servo motors can change the yaw and pitch angles of the phone in 180°. A third servo is responsible for steering the vehicle. Fig. 1 illustrates the mechanical construction of the robot.

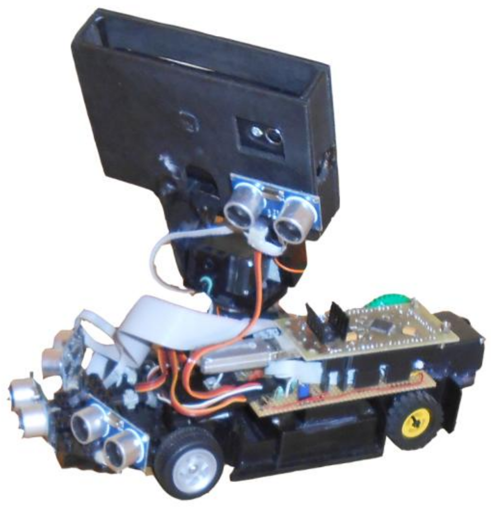


Fig. 1. Mechanical construction of the robot

In the view of organization the paper has the following sections: Section I was a short introduction. Section II describes different possibilities for connecting the smartphone to external hardware. Section III is about the software solutions of the robot. Section IV shortly describes the architecture of the software. Section V concludes the paper.

II. Connecting the smartphone to the robot

The project was carried out with a Samsung Galaxy Ace S5830 type smartphone. This was a mid-range device announced in 2011, January. It has an 800 MHz ARM11 processor and 278 Mb of RAM. The latest official software update for it was Android 2.3.3 Gingerbread. In the view of connectivity this setup can act as a USB slave device. It can also be connected through Bluetooth to an external hardware. Direct Wi-Fi connection is only possible from Android 4.0 and above [13].

Many microcontroller manufacturers offer now Android accessory frameworks to their IC-s [14]. These establish communication through USB between the Android device and the embedded system. In the USB communication there is always a host controller and a slave device [14]. The host powers the bus while usually the slave device represents simpler functionality. Devices running Android 3.1 and above can act as USB host controller. Older systems – such as in this project – can only act as slaves. Therefore, USB communication requires an external microcontroller with USB host peripheral. This solution can benefit during longer operations of the robot, because the robot can charge the Android device.

However, wireless connection is always offers wider accessibility. With the above described setup, the communication can be carried out with Bluetooth as well. This does not require Bluetooth module in the embedded system, but a USB-Bluetooth adapter can be used.

In the described system a Microchip PIC24FJ256DA microcontroller was used. The microcontroller implements an open source framework called IOIO that handles the USB peripheral and supports the Bluetooth communication [15]. The framework comes with an Android library, which offers high level functions for initializing and operating digital IO-s, PWM outputs, UART peripheral and timer functions in the microcontroller. This means that functionalities can be programmed through the Android program. Fig. 2 shows the layout plan of the system.

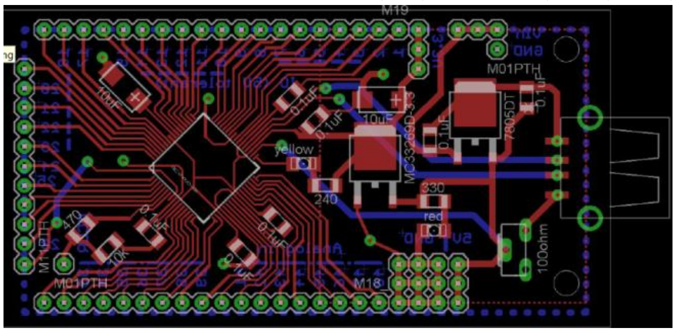


Fig. 2. Layout of the external system. *The charging current can be set with a potentiometer. Bluetooth adapter requires maximum current, while direct USB communication can be operational with minimal current, thus saves robot battery*

III. Used software solutions and other possibility

A. *Image processing*

The main functions of the robot are face tracking and color blob tracking. These tasks can be solved with external image processing software libraries such as OpenCV [16] or with native Android functions.

1) *Face tracking*

In the case of face tracking, the methods described in this section can be compared in the perspective of frame rate, and the precision of recognition. Spontaneous false positive recognitions can be filtered out with low pass filter. In case of short time false negative results during tracking the robot will look for the face in the previously identified region of interest.

2) *Face tracking with native Android functions*

Above Android 4.0 there is a Face detector function in the camera class [17]. This finds faces in the camera preview image, thus it natively serves with the highest frame rate. The function’s return array contains the position of the faces in the preview image. This certainly the most optimized solution but was not yet available due to the system setup.

Another native solution for face detection in Android is the findFaces function, which is available in every Android version [18]. This does not work on the live preview image but needs a bitmap image for processing and is relatively slow.

3) *Face tracking with OpenCV*

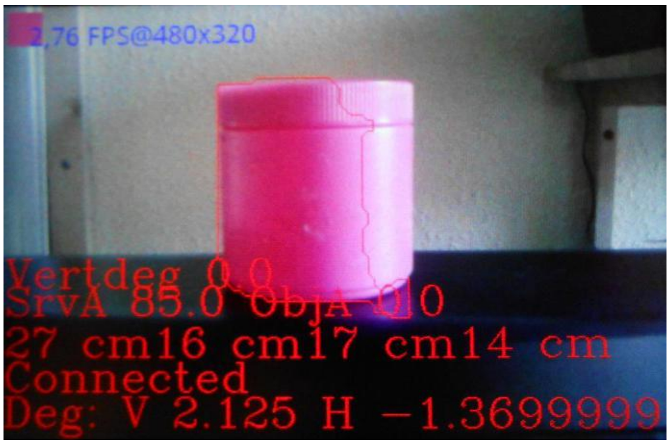
The OpenCV face detection uses Viola-Jones object detection algorithm [18], which is a Haar-feature based cascade classifier [20]. The operation and training of the detector is described in details by Viola and Jones in their work on rapid object detection [19]. OpenCV comes with many pre-trained classifier for face, eyes, smile etc. These are stored in XML files and can be simply loaded into the program. The OpenCV provides a wide scale for algorithm customization compared to the previously mentioned face detectors. The image to be processed can be downscaled, which reduces processing time per frame. Also the size of the smallest detectable face can be given. The smaller the face we want to detect the more times the algorithm has to scan the image with increasingly smaller detection window. By varying these two features an optimum can be found where the robot can sense faces in an acceptable distance, while providing sufficient frame rate.

4) *Color blob tracking*

The input image of the color blob tracker algorithm is coded in RGB color space. In the first step this is converted into HSV color space. HSV color model is a method to define colors according to the three basic features of the color: hue, saturation and luminance [21]. In the present application the robot follows a colored object. The changes of light conditions due to the environment and robot orientation affect the detected colors. These changes in RGB color space affect all the three color feature, whereas in HSV color space these mostly affect the luminance component of the color. This feature of the HSV color space allows a more useable color based segmentation. The hue of the color can be more specified, while a wider tolerance can be set to the luminance feature.

The algorithm under-samples the image i.e. reduces resolution to reduce processing time. Creates a new binary matrix where only fills those positions, where the corresponding pixel falls between the allowed thresholds in hue, saturation and luminance. This binary mask is expanded to the original size of the image. A border following algorithm developed by Suzuki and Abe [22] finds the contours around the mask and returns an array with the bounding rectangles of the individual contours. The robot wants to follow the largest object that matches the target color, therefore takes the largest contour as the representation of the followed object. It calculates the difference between the center point of the detected color blob and the center of the processed image as the regulator's error (Fig. 3). The error signal of the object follower controller is calculated similarly during face tracking with the largest, hence closest face.

Table I. summarizes the frame rate results between the experimented color blob and face detector methods in respect to their different parameter setups.



```
2,76 FPS@480x320
Vertdeg 0.0
SrvA 85.0 ObjA 0.0
27 cm16 cm17 cm14 cm
Connected
Deg: V 2.125 H −1.3699999
```

Fig. 3. Smartphone screen during color tracking

TABLE I. FPS VALUES WITH DIFFERENT TRACKING ALGORITHMS (*THE VALUES ARE IN FRAME/SECOND*)

Android native face detector					
0,75					
OpenCV face detection					
		Face size			
		40%	30%	20%	10%
Image ratio	100%	5.1	4.2	2.2	1.2
	50%	7.3	6.3	4.78	3.75
	30%	9.2	7.3	5.1	4.9
	20%	9.11	9.3	9.25	5.86
Color blob detection					
4,75					

B. *Speech recognition and verbal response*

Speech recognizer service and text-to-speech engine are basic services in Android, and can be easily implemented in any program. The speech recognizer requires internet connection, because the processing is done in the cloud [23]. It returns a set of the possibly heard word combinations. The given order to the robot is validated if the returned set contains it or something very similar. The robot answers to each instruction using the text-to-speech engine, and notifies if the instruction was misunderstood. It can also answers to some basic questions from a preprogrammed answer bank.

The verbal orders are also used for changing between the face tracking and object tracking. For a specified order the robot takes a color sample from the object held front of its camera and will follow it. The speech recognizer can be triggered with a short whistle.

IV. SOFTWARE ARCHITECTURE

The software builds on parallel running threads. The two main threads are the image processing and the microcontroller interface thread that reads the external sensors and drives the robot. These two threads communicate to each other through an intermediate controller object. There are two minor threads responsible for whistle detection.

The analog digital converter module of the microphone is capable of 44100 Hz sampling frequency on 16 bits. One thread records these samples and feeds them to the whistle detector thread in 2048 sample big packages. The detector than applies a Fast Fourier Transformation on the dataset. Spikes between 1 and 3 kHz are considered as whistles. When a whistle is detected the speech recognizer service is being launched, but first the recording has to be stopped to release the microphone as system resource.

The intermediate controller object is responsible for the eventual tracking of the detected objects. It also implements a state machine for enhanced collision avoidance algorithm.

The image processing thread does not supply the data with fixed sampling rate, thus the system clock is used to establish the elapsed time for the vertical and horizontal PID controllers.

Fig. 4 summarizes the software organization.

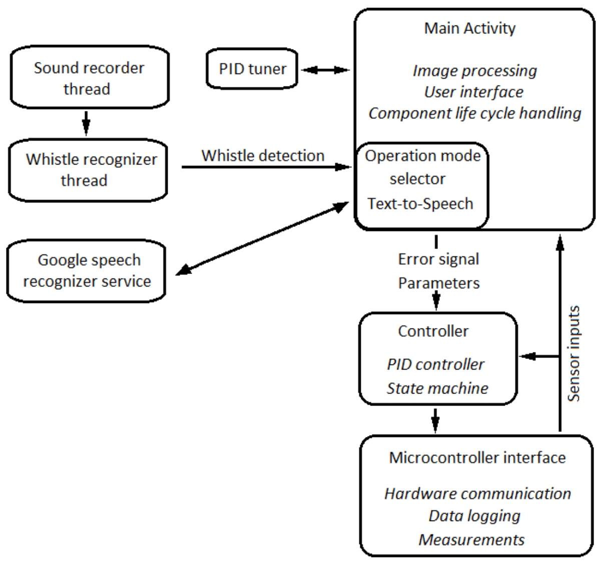


Fig. 4. Software architecture

V. Conclusion

This paper presented a possible usage of smartphones in mobile robotics as complex processing units and sensor arrays. Usage of smartphones in robotic applications can extend the capabilities of robots compared to conventional embedded systems. However, with mid-range phones the response time of the system will be too long. Simultaneous running of advanced functions might results phase shift on the real-time behavior. Higher-end smartphones can extend the embedded computers of mobile robots by taking advantages of all benefits of mass production and open source in the aspects of costs, software and hardware development.

Acknowledgment

The author would like to thank to Rita M. Kiss and Péter Tamás for their comments, which greatly improved this paper.

References

[1] Eldon C. Hall, (1996), Journey to the Moon: The History of the Apollo Guidance Computer, Reston, Virginia, USA: AIAA, p. 196, ISBN 1-56347-185-X

[2] C. Parga, Xiaoou Li, Wen Yu, "Smartphone-Based Human Machine Interface with Application to Remote Control of Robot Arm", Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on Oct. 2013, pp 2316-2321

[3] S. W. Moon, Y. J. Kim, H. J. Myeong, C. S. Kim, N. J. Cha, D. H. Kim, "Implementation of smartphone environment remote control and monitoring system for Android operating system-based robot platform", Ubiquitous Robots and Ambient Intelligence (URAI), 2011 8th International Conference on Nov 2011, pp 211-214

[4] N. Stansfield, B. Bothma, "Android based semi-autonomous collision avoidance robot", Robotics and Mechatronics Conference (RobMech), 2013 6th, pp 135-139

[5] J. Lim, Seok Ju Lee, G. Tewolde, J. Kwon, "Ultrasonic-sensor deployment strategies and use of smartphone sensors for mobile robot navigation in indoor environment", Electro/Information Technology (EIT), 2014 IEEE International Conference on June 2014, pp 593-598

[6] Nolan Hergert, William Keyes, and Chao Wang, "Smartphone-based Mobile Robot navigation" Department of Electrical and Computer Engineering, Carnegie Mellon University, May 9th, 2012

[7] C. Krofitsch, C. Hinger; M. Merdan, G. Koppensteiner, "Smartphone driven control of robots for education and research", Robotics, Biomimetics, and Intelligent Computational Systems (ROBIONETICS), 2013 IEEE International Conference on Nov. 2013, pp 148-154

[8] D. Yang, J.-K. Lim ; Y. Choi, "Early childhood education by hand gesture recognition using a smartphone based robot" Robot and Human Interactive Communication, 2014 RO-MAN: The 23rd IEEE International Symposium on Aug. 2014, pp 987-992

[9] Romo webpage: http://www.romotive.com/ Accessed in Nov 2014

[10] Smartbot webpage: http://www.overdriverobotics.com/ Accessed in Aug 2014

[11] Apple Inc. http://www.apple.com/iphone/features/siri.html. Accessed: Oct 2014

[12] Google Mobile. http://www.google.com/mobile/voice-actions Accessed: Oct 2014

[13] Android Developer Site – Wi-Fi Peer-to-Peer http://developer.android.com/guide/topics/connectivity/wifip2p.html Accessed: Nov 2014

[14] Flowers, D.: Android accessory development with the Google Open Accessory Framework. ELEKTRONET (2012, November), pp 24-25.

[15] IOIO Documentation https://github.com/ytai/ioio/wiki Accessed: Oct 2014

[16] OpenCV User Site: http://opencv.org/ Accessed: Oct 2014

[17] Android Developer Site, Camera Face Tracking http://developer.android.com/guide/topics/media/camera.html#face-detection Accessed: Oct 2014

[18] Android Developer Site, Face Detector class http://developer.android.com/reference/android/media/FaceDetector.html Accessed: Oct 2014

[19] Viola, P.; Mitsubishi Electr. Res. Labs., Cambridge, MA, USA ; Jones, M., "Rapid object detection using a boosted cascade of simple features" Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on Vol 1 pp 511-518

[20] OpenCV documentation, http://docs.opencv.org/trunk/doc/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html Accessed: Sept 2013

[21] Li Shuhua, Guo Gaizhi, "The application of improved HSV color space model in image processing" Future Computer and Communication (ICFCC), 2010 2nd International Conference on (Volume:2) 21-24 May 2010

[22] S.Suzuki, K.Abe., "Topological structural analysis of digital binary image by border following," CVGIP.30(l): 32-46, 1985.

[23] Robert McMillan: How Google Retooled Android With Help From Your Brain, http://www.wired.com/wiredenterprise/2013/02/android-neural-network/ Accessed: Sept, 2013