An Encoder Fault Tolerant FPGA Based Robot Control Using Bluetooth of a Smart Phone

Ameen Majeed, Salih Baris Ozturk, and Didem Kivanc Tureli

Department of Electrical and Electronics Engineering, Okan University, Istanbul, Tuzla 34959 Turkey
baris.ozturk@okan.edu.tr

Abstract

An FPGA based Bluetooth controlled robot with encoder fault tolerant algorithm is presented. The smart phone is used to obtain user intentions, such as turning, speeding or braking and this data is sent to a robot using Bluetooth. The motor control algorithms and robot communication interfaces are implemented on FPGA for parallel processing. The robot is capable of changing its position with the help of two independent PI closed loop speed controlled DC motors. The motor speed is determined by the accelerometer of the sensor data of a smart phone. The FPGA receives speed information and generates appropriate PWM signals. Motor rpm is calculated via rotary encoder. In case of an encoder failure, an on-board gyroscope helps maintaining normal operation. An Android based smart phone application has been developed. MATLAB is used for simulating an encoder failure and observation of results. The motor control algorithm has been implemented using Verilog and tested on the field.

1. Introduction

Smart phone technology has been evolving and its adoption continues to grow [1], [5]. Latest smart phone processing capabilities are comparable to a low end personal computer [2]. Presence of wireless communication capabilities and on board sensors have allowed smart phones to be used in a significant number of remote applications [3], [4], [6].

Smart phone interfaced robots have brought a new dimension to the use of smart phones as a control device for various applications such as drones, search and rescue, surveillance, educational and indoor robots [14], [15]. Although smart phone processes sent and received data, the robot must perform various tasks, such as motor control, perform predetermined operation in case of communication break without smart phone intervention. Since robot must complete parallel tasks, FPGA is a good candidate for these types of applications.

Encoders are widely used to obtain rotor speed and position information in closed loop motor control systems. A faulty or a failed encoder at the robot's wheel may lead to incorrect speed information and result in collapse of the encoder feedback based closed loop controller. Encoders are prone to failure for a number of reasons as discussed in [11].

An encoder failure, surface roughness, wheel slip, tread wear an obstacle or an external push can infer a deviation in course of mobile robots with only encoder feedback controlled motors. A need for a method for calculating angle deviation independent of encoders and correcting the direction of the robot is eminent.

A method for angle estimation in industrial manipulators using only inertial sensors is discussed in [6]. A sensor fusion approach (encoders and IMUs) for angle estimation is used to improve precision and reduce cost is presented in [7], [8].

Human body motion kinetic modeling [9], gait based human modeling, walk assist systems [12] and rehabilitation therapy systems [10] readily employ inertial sensors for angle estimations.

The contribution of this paper is FPGA implementation and testing of a gyroscope feedback based control loop on a smart phone of a Bluetooth controlled robot which resists set path deviation and retains normal operation in a scenario where either the primary encoder sensor fails or an external disturbance attempts to deviate the robot's from its set path. In this research, a 3-axis MEMS gyroscope sensor is used for angular movement estimation and MATLAB is used for capturing results.

The rest of the paper is organized as follows. In Section 2, brief information about the hardware system, FPGA circuit Android application and data logging using MATLAB code are presented. The implementation and measurement results are given in Section 3. Concluding remarks are given in Section 4.

2. Methodology

2.1. System Introduction

An FPGA based system which is capable of establishing a Bluetooth connection between an Android based smart phone or a PC and an encoder fault tolerant motor control algorithm based on a secondary gyroscope feedback based control loop sensor is realized.

The rpm speed is derived from smart phone's tilt angle which is calculated using the accelerometer sensor data of the smart phone. A colored circle to indicate the tilt of the phone is displayed on the screen as a visual aid. The smart phone transmits the desired speed in rpm over the Bluetooth link to the FPGA. The FPGA maintains two DC motors at the desired speed with the help of PI closed loop speed controllers using Hall Effect rotary encoders and a gyroscope.

In case of an encoder failure, the rpm values are not calculated correctly and the encoder feedback based control loop fails. As a result, the robot deviates from its set path. This deviation in path is detected by the on board gyroscope. The gyroscope feedback based controller kicks in and helps maintaining system operation.

For simulating an encoder failure and observing results, a MATLAB code has been developed. With the help of this code, various FPGA circuit parameters can be set and variables can be observed on a PC in real time via Bluetooth connection.

2.2. Hardware and Software Platforms

Basys2 FPGA kit from Diligent is used as the FPGA hardware platform. The kit has sufficient Hardware resources to support the system. PmodHB5 modules are used as current

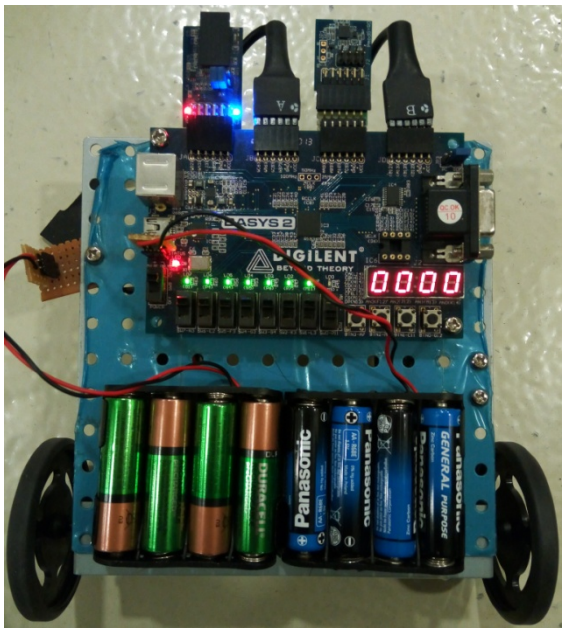

Fig. 1. The FPGA based Bluetooth controlled two wheel robot.

drivers and direction controllers. PmodBt from Diligent is used as the Bluetooth device for the system. This device houses a LMX9838 Bluetooth Integrated System IC from Texas Instruments. This Bluetooth device uses UART interface for host communication.

The PmodBT's UART baud rate is selectable via jumpers present on it. The baud rate is set at 115.2 kbps. Some basic parameters at PmodBT need to be known or are required to be reset. These include device name and device PIN. These parameters are reset by connecting the PmodBT to a PC with the help of a PmodUSBUART. A computer program from TI called "Simply Blue Commander" is used to reset these parameters. For connecting the PmodBT to the PmodUSBUART, a signal rerouting circuit is synthesized on the FPGA. The circuit reroutes UART data to a PmodUSBUART and vice versa.

The Pmodgyro houses a L3G4200D STMicroelectronics MEMS gyroscope. This device can utilize two communication interfaces I2C and SPI. In this system, only SPI is used. The L3G4200D can measure angular rate of change in three axes. Each axis value is 16 bits wide. Angular rate of change scale value is selectable from 250, 500 and 2000 dps. The designed system uses 250 dps angular rate of change which gives the highest precision which is 8.75 mdps per digit.

The DC motors run independently, hence the robot shown in Fig. 1 exhibits a tank like motion. To move forward or backward, both the motors should turn identically in forward or backward direction. A speed difference in the motors results in turning.

The rotary encoders attached to the motors have a resolution of 58 pulses per revolution (ppr). The motors can turn the wheels at a maximum speed of 320 rpm at 100% duty cycle. However, our system limits the maximum wheel speed to 160 rpm.

For Verilog code development, Xilinx ISE V 14.5 is used. All the FPGA circuit modules have been coded using Verilog HDL. Adept from Diligent is used for programming the binary file to the FPGA. The Android application has been coded using java. However, some of the user interface components are coded using XML. Android SDK tools are used for Android application development. The application is tested on a device running Android OS version 4.0, however it is compatible with devices running Android OS version 2.2 and above.

The MATLAB code is developed using version R2015a.

2.3. System Design

The android application connects to the robot via Blue-tooth and transmits motor speed in rpm and direction information to the FPGA. The raw accelerometer values are adjusted by the application, so that they represent motor speeds in rpm. An on screen moving circle is displayed which gives visual aid to phone tilt.

The FPGA circuit receives and processes Bluetooth data. Simultaneously, the circuit processing data from the encoders calculates current DC motor speed in rpm. A PI controller adjusts a 16 bit PWM signal according to the calculated rpm. Direction is set according to the received direction information. The motor direction switching has enough delay to prevent any damage to the motor driving circuits.

2.4. PmodBT (TI LMX98938) Commands via UART Interface

The TI LMX9838 has two modes of operation command mode and transparent mode. In command mode, all data sent to the device at the UART is interpreted to known commands. By default, the device is in command mode when it is not connected to any remote device.

The device is configured by sending specific commands to it via the UART interface. It is connected to the PC with the help of a PmodUSB UART and the PC based software (Simply Blue Commander) is used to send the parameters and commands as needed. A simple signal rerouting circuit is synthesized to connect the PmodBT to PmodUSB UART.

2.5. PmodBT (TI LMX98938) Transparent Mode

Once the device has established a connection with a remote device, it switches to "transparent mode" automatically. In transparent mode, all the data sent to the device at the Bluetooth link is directly routed to the UART interface and vice versa.

2.6. Mechatronics Hardware Arrangement

Four I/O ports are available on the Basys2 kit. One port is occupied by the Bluetooth device, one by Gyroscope and the other two are occupied by the two motors (one for each motor).

The Bluetooth module (PmodBT), the FPGA kit (Basys2), two current driver and direction control circuits (PmodHB5), two geared DC motors and two battery packs are mounted on the mechatronics platform, as shown in Fig. 2. These I/O ports are connected to two direction control and current driver circuitries (PmodHB5) which run the motors. The same I/O ports feed encoder data to the FPGA.

2.7. FPGA Circuit

The closed loop PI motor speed and direction controls of the two DC motors are performed by the FPGA circuit. It receives data from PmodBT connected to a remote device via PmodBT's UART interface (transparent mode). If it is connected to the smart phone, the incoming data has motor speed (rpm) and direction information. If it is connected to a PC via MATLAB, the incoming data can have commands and parameter settings for the circuit as well. The FPGA circuit receives pulses from rotary encoders mounted on the motor shaft. The circuit counts

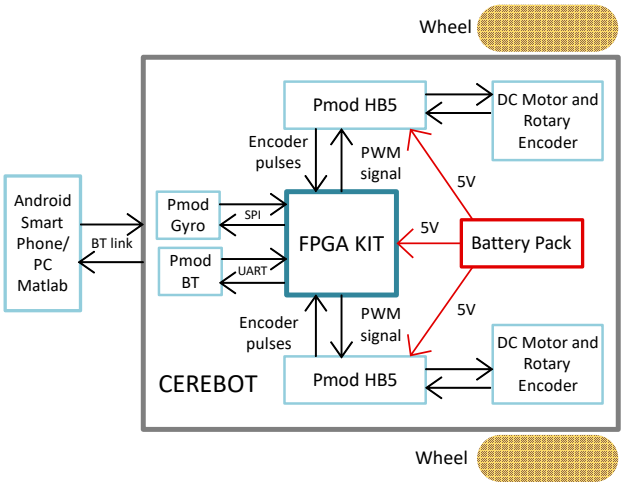

Fig. 2. System hardware arrangement.

the number of pulses in a period of time and feeds these values to a moving average filter. The filtered number of pulses in a set period of time is then used to calculate motor speed in rpm.

This speed data is fed to a PI controller. The controller compares the calculated rpm from encoder pulse with the desired rpm speed received via the Bluetooth and adjusts PWM duty cycle, accordingly.

2.8. Brief Overview of The Verilog Modules and Their Functionality

The FPGA circuit consists of instances of Verilog modules, as shown in Fig. 3. The main module is set as the top module and connects the modules among themselves and with the I/O ports at the FPGA. These modules perform communication between Pmod devices and the FPGA circuits and vice versa, Bluetooth communication data selection and processing, gyro-scope sensor data processing and angular position calculation, encoder data processing and rpm calculations, PI closed loop control via encoder and/or gyroscope feedback.

Data from motor encoders is monitored by "Rpm calculator with moving average filter" module. Encoder pulse count method is used for speed estimation. The resolution of the encoder is 58 ppr. Hence the rpm can be calculated by using the following expression:

$$Speed\ (rpm) = \frac{\frac{Enc.\ pulses}{sec.} \cdot 60}{\frac{Enc.\ pulses}{rev.} = 58}. \quad (1)$$

In order to conserve FPGA resources, mathematical calculations are minimized. Hence, the following equation provides motor rotation speed in rpm without performing any mathematical operations at all:

$$Speed\ (rpm) = \frac{Enc.\ pulses}{1.0345\ sec.}. \quad (2)$$

The module counts the number of pulses in 1.0345 seconds to obtain motor speed in rpm. The module continuously feeds number of pulses in 1.0345 seconds to a moving average filter. This filter uses a 32 byte sized memory, hence it stores 32 encoder pulse counts at a time. On every 1.0345 seconds, a fresh value (encoder pulse count) is written to the memory while discarding the oldest one, an average of these values is calculated. The average is the motor rpm speed with noise suppressed. This speed is fed to the PI motor controller module as the process variable of the controller. The PWM generator module generates DC motor PWM speed control and direction control signals.

The PmodGyro gives current angular speed in dps (degrees per second) for x, y and z axes. This angular speed is integrated to find the angular deviation. The refresh rate of the gyroscope is 100 Hz. As soon as the robot deviates from its path due to an external interference or encoder failure, the gyroscope feedback based controller kicks in and adjusts the motor speeds, accordingly.

In case of a total encoder failure, the calculated rpm be-comes zero as no pulse ticks are registered. The encoder feedback based controller progressively starts to increase PWM duty cycle which increases the motor speed. This results in a speed mismatch between the two motors and the robot starts to deviate from its path. This angular deviation is continuously calculated and monitored from the gyro-scope data and is fed to another PI controller which corrects the error in PWM duty cycle. The angular distance values range from 0 to 8129 representing 0 to 360 degrees for the counterclockwise rotation of the robot.

2.9. Android Application

An android application has been solely developed for this system. This application tailors the raw accelerometer values to meet the system requirements.

The data that is sent to the remote Bluetooth device are speed in rpm and direction information. The application displays current speed in rpm values that are being sent and a circle which moves with respect to orientation values provide a visual aid in controlling the remote device.

2.10. Data Logging Using MATLAB

MATLAB Instrument control toolbox provides support for Bluetooth communication and is used to connect, send and receive data to and from the robot. The data sent to the FPGA includes motor speed rpm set points, controller gains, angular displacement set points and control commands to request data from the FPGA circuit, enable/disable encoder's signals and controller circuits.

Upon data send request, the FPGA circuit sends a data set of 16 bytes. Variables from any Verilog module can be accommodated in this data set in the FPGA circuit and sent to MATLAB for observation. Variables that change at a frequency less than 800 Hz are observed without any information loss. Variables with higher frequency of change are observable as well but an information loss is eminent.

3. Experimental Results

Five experimental tests have been performed. All the experiments are performed on only one of the two motors with the other one running normally and hence its data is also not presented. Controller parameters for encoder feedback based controller are set as k_{ep} = 64 and k_{ei} = 8. For gyroscope feedback based controller k_{gp} = 256 and k_{gi} = 8.

In all figures, top graphs present the control variables of the controllers. These control variables are thirteen bits wide and have 0.0122% Duty Cycle value per digit. Black line plots the control variable of the encoder feedback controller. This variable is overflow protected. Gray line plot is Gyroscope

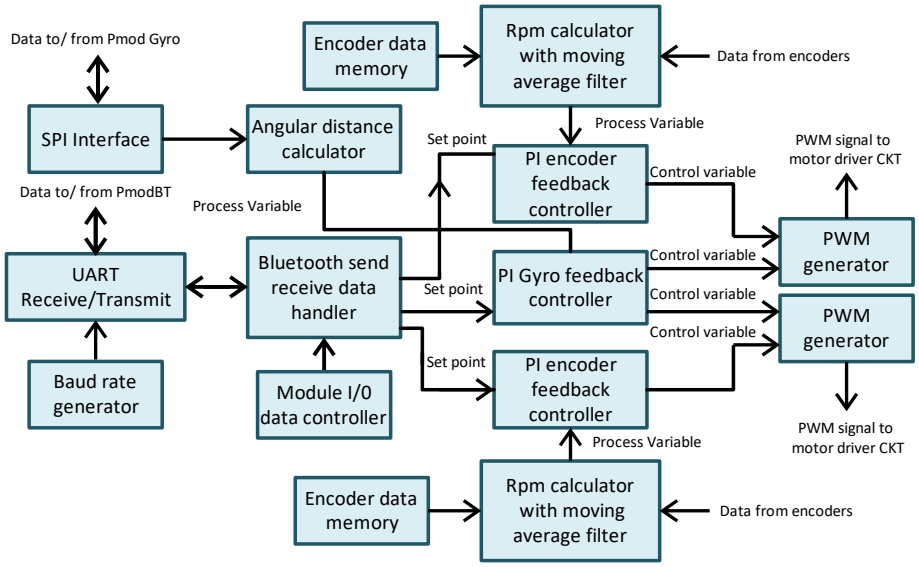

Fig. 3. Verilog circuit block diagram.

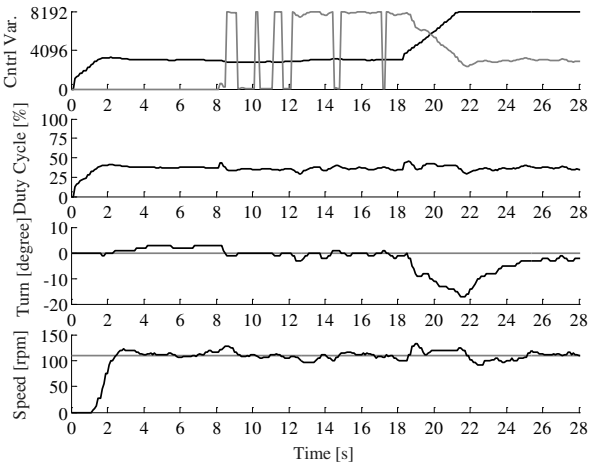

Fig. 4. Experimental test under normal and faulty single encoder case.

feedback controller control variable which is allowed to overflow by design and is the reason behind the flipping of variable values from minimum to maximum and vice versa. This overflow allows subtraction operation based on 2's complemented addition. Second graph in each figure presents the duty cycle percentage calculated from the resultant of the two control variables. Third graph presents robot's angular deviation (Yaw axis) in degrees. Fourth graph presents motor speed in rpm.

In Fig. 4, initially both encoders run as normal and the system is allowed to gain momentum. An approximately 5 degrees deviation is gained from 0 to 8 s. At this instant, the gyroscope feedback based controller starts and rectifies the deviation. At 18 s, the motor encoder is disconnected. From this instant, the encoder feedback based control variable which is output of the speed PI regulator progressively proceeds to maximum value and the gyroscope feedback based control variable which is output of the angular deviation PI regulator attempts to counter balance it. From 18 s to 22 s, a 20 degrees clockwise deviation is recorded which rectifies between 22 s to 24 s. A small raise and loss in rpm is also recorded at these instants.

In Fig. 5, both encoders run normally throughout the test. The robot is subjected to physical disturbances at 13 s and 21s. At 13 s, a wheel slip is emulated which raises the wheel speed. At 23 s, an obstacle reduces the wheel speed. As seen in Fig. 5, this disturbance results in an angular rotation of the robot which is corrected by the Gyro controller (PI regulator used for angular deviation).

In Fig. 6, both encoders run normally. The direction set point is set at 45 degrees and –60 degrees at 16 s and 36 s, respectively. Both controllers run normally. When the gyroscope feedback based controller attempts to alter wheel speed, the encoder feedback based controller recognizes it as an error and tries to correct it. Due to this, a small rise at 17 s and a decline at 36 s in its value is observed. As the PI gains for the gyroscope feedback based controller are much higher, the robot is easily directed to the set angle.

In Fig. 7, the system is initially allowed to gain momentum and the encoder is disconnected at 9 s. The direction set point is reset to 45 degrees and –60 degrees at 19 s and 35 s, respectively. An overflow in control value results in a spike at 35 s. The robot turns to the specified angle with only gyroscope feedback based closed loop control.

In Fig. 8, one of the encoders is disconnected at 8 s. The system is subjected to distortions at 13 s, 18 s and 27 s. As seen in Fig. 8, the robot returns to its set point successfully with encoder in the failed state and using only gyroscope based closed loop controller.

6. Conclusion

In this paper, a dual sensor feedback motor control algorithm for a robotic platform is proposed. The purpose of this research is to resist direction deviation and maintain functionality in case of an encoder failure in mobile robots using gyroscope. The results indicate that the system can retain normal functionality in case of an encoder failure and demonstrates ability to deviation.

Similar systems already employing encoders and gyro-scopes can use this method for increasing fault tolerance and control

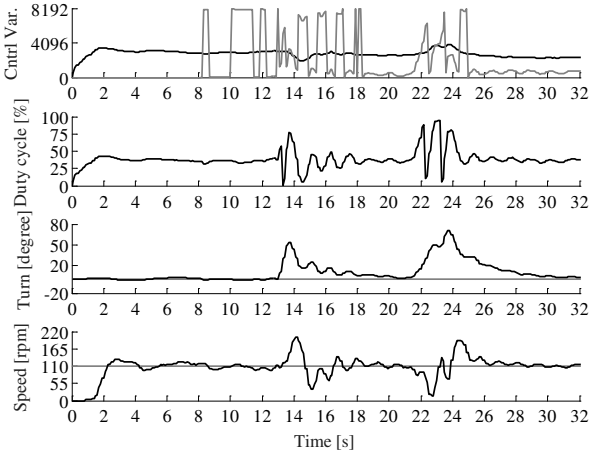

Fig. 5. Experimental test under normal encoder case with external disturbance.

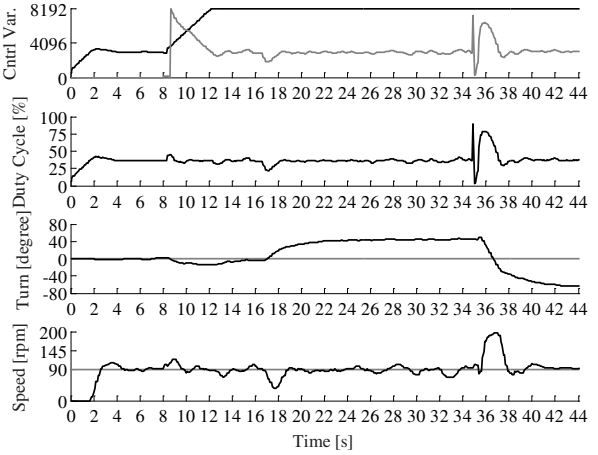

Fig. 7. Experimental test under faulty single encoder case with gyroscope feedback based control deviation under different angle setpoints.

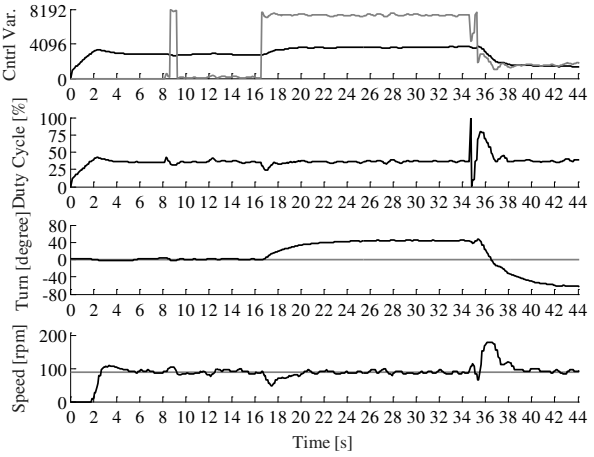

Fig. 6. Experimental test under normal encoder case with gyroscope PI regulator control deviation.

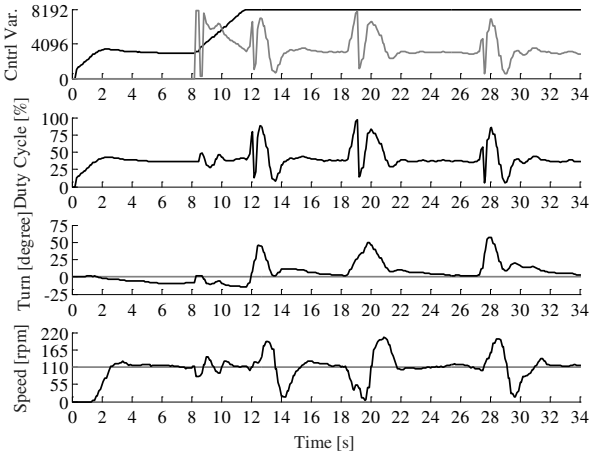

Fig. 8. Experimental test under faulty single encoder case with external disturbance.

robustness without any added hardware. Future works may include implementation of more complex filtration and control algorithms to the current system.

7. References

[1] F. Aldhaban, "Exploring the adoption of Smartphone technology: Literature review," in *Proc. IEEE PICMET*, pp. 2758–2770, 2012.

[2] X. Li, P. J. Ortiz, J. Browne, D. Franklin, J. Y. Oliver, R. Geyer, Y. Zhou, and F. T. Chong, "Smartphone Evolution and Reuse: Establishing a More Sustainable Model," in *Proc. IEEE ICPPW*, pp. 476–484, 2010, doi: 10.1109/ICPPW.2010.70.

[3] P. N. A. Fahmi, P. Budhi, J. Song, Ardiansyah, D. Choi, and Y. Kim, "3D-to-2D Projection Algorithm for Remote Control Using Smartphone: Enhancing Smartphone Capability for Costless Wireless Audio Visual Consumer Appliance Control," in *Proc. IEEE WAINA*, pp. 1044–1049, 2013.

[4] M. M. Makki, G. A. Saade, A. G. Altouma, S. Al-Terkawi, A. Baobeid, and R. Tafreshi, "Acquiring and analyzing electrocardiograms via smartphone to detect cardiovascular abnormalities," in *Proc. IEEE EMBS BHI*, pp. 277–280, 2014, doi: 10.1109/BHI.2014.6864357.

[5] V. Guimarães, D. Ribeiro, L. Rosado, I. Sousa, "A smartphone-based fall risk assessment tool: Testing Ankle Flexibility, Gait and Voluntary Stepping," in *Proc. IEEE MeMeA*, pp. 1–6, 2014, doi: 10.1109/MeMeA.2014.6860087.

[6] L. Cantelli, G. Muscato, M. Nunnari, and D. Spina, "A Joint-Angle Estimation Method for Industrial Manipulators Using Inertial Sensors," *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 5, pp. 2486–2495, Oct. 2015, doi: 10.1109/TMECH.2014.2385940.

[7] D. Kubus, C. Guarino Lo Bianco and F. M. Wahl, "A sensor fusion approach to improve joint angle and angular rate signals in articulated robots," in *Proc. IEEE/RSJ IROS*, pp. 2736–2741, 2012, doi: 10.1109/IROS.2012.6386176.

[8] D. Kubus and F. M. Wahl, “A sensor fusion approach to angle and angular rate estimation,” in *Proc. IEEE/RSJ IROS*, pp. 2481–2488, 2011

[9] K. Y. Lim, F. Y. K. Goh, W. Dong, K. D. Nguyen, I-M. Chen, S. H. Yeo, H. B. L. Duh, and C. G. Kim, “A wearable, self-calibrating, wireless sensor network for body motion processing,” in *Proc. IEEE ROBOT*, pp. 1017–1022, 2008, doi: 10.1109/ROBOT.2008.4543338.

[10] Z. Luo, C. K. Lim, W. Yang, K. Y. Tee, K. Li, C. Gu, K. D. Nguen, I-M. Chen, S. H. Yeo, “An interactive therapy system for arm and hand rehabilitation,” in *Proc. IEEE RAMECH*, pp. 9–14, 2010, doi: 10.1109/RAMECH.2010.5513222.

[11] M. Bourogaoui, I. Jlassi, S. K. E. Khil, and H. B. A. Sethom, “An effective encoder fault detection in PMSM drives at different speed ranges,” in *Proc. IEEE SDEMPED*, pp. 90–96, 2015, doi: 10.1109/DEMPED.2015.7303674.

[12] C. Yang and T. Murakami, “An approach to walking assist control by a multi-legged system in human gait motion,” in *Proc. IEEE IECON*, pp. 5236–5241, 2014, doi: 10.1109/IECON.2014.7049298.

[13] M. S. Shah and P. B. Borole, “Surveillance and rescue robot using Android smartphone and the Internet,” in *Proc. IEEE ICCSP*, pp. 1526–1530, 2016, doi: 10.1109/ICCSP.2016.7754413.

[14] C. Bodenstein, M. Tremer, J. Overhoff, and R. P. Würtz, “A smartphone-controlled autonomous robot,” in *Proc. IEEE FSKD*, pp. 2314–2321, 2015, doi: 10.1109/FSKD.2015.7382314.

[15] C. Krofitsch, C. Hinger, M. Merdan and G. Koppensteiner, “Smartphone driven control of robots for education and research,” in *Proc. IEEE ROBIONETICS*, pp. 148–154, 2013, doi: 10.1109/ROBIONETICS.2013.6743595.