ANEXA 3

UNIVERSITATEA “BABEŞ-BOLYAI” CLUJ-NAPOCA

FACULTATEA DE FIZICĂ

SPECIALIZAREA FIZICĂ

LUCRARE DE LICENŢĂ

Coordonator științific

Conf. dr. Ioan Burda

Absolvent

Ecaterina Vlaico

2024

ANEXA 4

UNIVERSITATEA “BABEŞ-BOLYAI” CLUJ-NAPOCA

FACULTATEA DE FIZICĂ

SPECIALIZAREA FIZICĂ

LUCRARE DE LICENŢĂ

DEVELOPMENT OF A SMART ELECTRONIC ASSISTIVE DEVICE FOR PEOPLE WITH VISUAL IMPAIRMENT

Coordonator științific — Absolvent

Conf. dr. Ioan Burda — Ecaterina Vlaico

2024

Abstract

ETAs represent an emerging technology which aims to compensate people’s low visual acuity by converting visual information into audio, haptic or other forms of data. GP2Y0D02YK0F infrared sensors and an HC-SR04 ultrasonic sensor are used to detect and measure the distance against obstacles. Median filtering and data linearization is performed on IR GP2Y0D02YK0F output to achieve stable and valid results, which are proved experimentally in the last chapter. By using MPU6050 IMU, the distances are projected into the horizontal plane. Employed libraries exploit the MPU6050’s DMP unit, which performs onboard calculations, thus reducing overall errors. The projections are converted to haptic information by establishing a linear dependence between them and the input voltage of vibrating disk motors.

Table of Contents

List of Abbreviations

LCD	Liquid-Crystal Display
PSD	Position Sensitive Detector
IRED	Infrared Emitting Diode
PIN	p-type, intrinsic, n-type semiconductor
PN	p-type, n-type semiconductor
SPI	Serial Peripheral Interface
I2C	Inter-Integrated Circuit
UART	Universal Asynchronous Transmitter Receiver
IC	Integrated Circuit
FET	Field-Effect Transistor
IP	Intellectual Property
FPGA	Field Programmable Gate Array
PC	Personal Computer
FIFO	First In First Out
MCU	Microcontroller Unit
DLL	Divisor Latch LSB (Least Significant Bit)
DLH	Divisor Latch MSB (Most Significant Bit)
FCR	FIFO Control Register
THR	Transmitter Hold Register
TSR	Transmitter Shift Register
LCR	Line Control Register
RSR	Receiver Shift Register
RBR	Receiver Buffer Register
CCD	Charge - Coupled Device
MEMS	Microelectromechanical System

IMU	Inertial Measurement Unit
DMP	Digital Motion Processor
ADC	Analog to Digital Converter
dps	degrees per second
LSB	Least Significant Bit
IDE	Integrated Development Environment
PID	Proportional Integral Derivative
RAM	Random Access Memory
SRAM	Static Random Access Memory
EEPROM	Electrically Erasable Programable Read Only Memory
PWM	Pulse Width Modulation
ETA	Electronic Travel Aid
EOA	Electronic Orientation Aid
PLD	Position Locator Device
GSM	Global System for Mobile Communications
ROS	Robot Operating System
LiDAR	Laser imaging, Detection and Ranging

Introduction

Visual impairment is a health condition that affects sight sense and cannot be restored by any medical intervention [3]. Because of several factors, such as narrowed visual field range and low visual acuity, this condition constricts human activity and impedes proper living, resulting in major changes in one's lifestyle. Due to such restrictive factors, children are prone to emotional isolation and worse school performance, adult people manifest lower rates of work productivity, and older population are more exposed to physical injury [1].

About 29% of global population have a form of visual impairment [1] [2]. According to the World Health Organization report published in 2019, around 2.2 billion people have a form of visual impairment, out of which 11.9 million suffer from moderate or severe blindness [1]. Such concerning statistics call upon efficient and sustainable technological solutions that will integrate these individuals back in our society.

Currently, there has been developed a considerate spectrum of electronic devices that assist blind people to an extent in their daily lives. Some of the projects incorporate a white cane [5][6], other designs implement a detection and signaling system within a pair of glasses [8] or shoes [9], (for detailed description refer to the introduction of the first chapter).

The current study aims to describe the development of an electronic circuit, designed to assist people with visual impairment in space orientation, and to prove the applicability of the developed prototype.

The content of the thesis is divided into three sections: *theoretical framework* – introduces all theoretical concepts required to understand the functionalities and implementation of this project; *technical implementation* – describes the design of the electronic circuit, as well as software packages used to interface sensors with the microcontroller; *analysis of results* – presents an experimental background which proves the applicability of the project.

The system incorporates HC-SR04 and GP2Y0D02YK0F distance measuring sensors, an MPU6050 sensor which projects the distances into the horizontal plane, and an actuating system that converts distance data to haptic information. The circuit functions autonomously and offers a good sensory approximation of the user's location in space with respect to the encountered obstacles.

Chapter 1

Theoretical Framework

Generally, electronic devices that assist blind people are classified in 3 categories: EDAs – devices that enhance vision through a CCD camera output; ETAs – devices that substitute vision by converting visual information into auditory or haptic data; PLDs – devices that interconnect with the ocular nerve and transfer visual information directly to it [4].

To provide a better insight of the current technology trends for people with visual impairment, several intricating methods and devices are listed below:

Smart stick –The white cane is an essential travel aid for every blind person, that primarily assists him in obstacle detection. Throughout the decades, researchers tried to enhance the usability of the white cane by incorporating position detecting sensors and implementing complex numerical algorithms. Some researchers added ultrasonic, water level sensors, as well as a GSM module to a white cane [5], providing long-distance obstacle detection, water level detection, a vibratory signaling system and message directing toward a dedicated person in case of critical situations. Others included infrared distance detection sensors and an audio signaling system [6], that serve for the same purpose as the device described above.

IR camera – a genuinely innovative approach was taken by a group of Belgian, Polish, and Greek researchers that proved both theoretically and experimentally that temperature is distributed slightly unequal toward the corner of a wall [7]. Such information can be used to detect corners with an IR camera, without excessive power consumption.

Smart glasses – an alternative proposal implies the implementation of glasses supporting a system of ultrasonic sensors and a LiDAR sensor mounted on a hand attachment [8]. The system is implemented using ROS software and is controlled by two microcontrollers: the master responsible for data processing and decision making, and the slave that collects measurements from all sensors. The ultrasonic sensors provide information about the presence of obstacles and the LiDAR sensor analyzes the depth of the ground, assessing the safest path which is communicated though a haptic/vocal message to the user.

Tactile foot interface – this project uses smartphone GPS data and its computational resources to implement a navigation device by developing user a friendly app which interconnects with an AT-tiny2313 microcontroller that coordinates the actuating system [9]. The actuating system consists of four vibrational actuators placed on a foam insole according to the most low-frequency sensitive zones of the foot, each indicating back, forward, left, or right movement according to GPS data and the input destination.

Obviously, there are more devices that implement different methods and algorithms (see Fig. 1, article [4]), but most of them use ultrasonic, IR, lidar, IMU sensors and a form of controlling and signaling system that is designed according to real-time processing requirements, response-time, accessibility, efficiency, and cost-effectiveness.

To gain a proper comprehension of the subject, this chapter will introduce a thorough explanation of all theoretical aspects that are encountered in the project's implementation.

1.1 Introduction to Arduino

Arduino was designed as an open-source platform that provides an easy approach to interface between a computer and sensors, actuators, or other peripheral devices. Arduino comes in the form of development boards, containing an onboard microcontroller, digital pins, analog I/O pins, and serial communication interfaces, as well as an IDE, used to write and upload code to the microcontroller.

1.1.1 Arduino Board

There are several types of Arduino boards, each differing in complexity and functionality. For this project, the Arduino Nano board is used. Arduino Nano operates at 5 volts and can be powered by an USB cable or by supplying a voltage within 7-12 volts range to the Vin pin (see Arduino Nano pinout reference [10]), that is regulated down to 5 volts by an onboard voltage regulator [11].

It runs code on the ATmega328 microcontroller, at a clock speed of 16 MHz, providing 32 KB of flash memory, 2 KB of SRAM, and 1 KB of EEPROM [12]. The difference between all three types of memory is that SRAM is volatile, meaning that it is used for temporary storage during the execution of the uploaded code; flash memory is non-volatile and stores the uploaded program; EEPROM stores small amounts of data that are modified rarely [13].

Like other Arduino boards, Arduino Nano contains analog I/O pins, and digital pins, 6 of which support PWM. Pulse Width Modulation represents a method that allows obtaining analog signals by modifying the length of the duty cycle of a digital signal [14].

Additionally, Arduino Nano supports UART, SPI, and I2C communication, the latter being encountered in the implementation of this program.

1.1.2 PlatformIO IDE

Arduino software is an open-source integrated development environment, based on C/C++ programing language, with pre-defined functions that makes interfacing between MCU and peripherals easier [15].

PlatformIO is an alternative option to the Arduino IDE, encompassing additional features such as smart code completion, integrated debugger, and unit testing [16]. It provides the attributes of Arduino programming language, but extended for 33 types of microcontrollers, with a dependency management system that handles over 7000 libraries [17]. Thus, considering the advantages of PlatformIO IDE, it can be considered the best approach for this project.

1.2 Serial Communication

To receive data and send commands to peripheral devices, the ATmega328 microcontroller must interface with them. Interfacing is achieved through a communication protocol - a set of standards that specify data format and hardware characteristics.

There are two types of communication protocols: serial, that transmits one bit at time, consecutively, and parallel, which makes use of several channels to send bits simultaneously [18].

To establish serial communication between the microcontroller and a device, only a few wires are used, which means fewer pins are needed on the board, e.g. SPI uses 4 wires, I2C uses 2 wires. The wires used for digital data exchange are called data bus lines.

1.2.1 Classification of Serial Communication Protocols

Serial communication protocols are divided into two subcategories: asynchronous and synchronous. Second type is considered asynchronous in sense that character transfers do not occur at a constant speed [18].

To differentiate distinct characters, asynchronous serial interface splits incoming data into blocks, which are used to create *frames*. One frame contains data bits that range from 4-8 bits, a START bit (logic 0) at the begging of the sequence, a STOP bit (logic 1) at the end of the sequence, and optionally, a parity bit that confirms whether transmitted data corresponds to the one which was sent. The parity bit has three states: NONE, meaning the protocol doesn't check if there is any error in the transmission process, ODD when the value 1 corresponds to an odd number of bits, and EVEN when the value one is equivalent to an even number of transmitted bits [19].

The bus line is kept in an idle state between the transmission of frames, meaning it is set at a constant voltage equal to the STOP bit [18]. Because each frame is transmitted at an arbitrary time, a synchronized data sampling is required [18]. Synchronization is achieved through maintaining the same bit rate between the transmitter interface and the receiver interface, which is defined in formula (1).

$$Bit\ rate = \frac{number\ of\ data\ bits\ per\ frame}{total\ number\ of\ bits\ per\ frame} * baud\ rate \qquad (1)$$

where the baud rate is the speed of data transfer of the transmitter interface and the receiver interface, measured in [Bits/s] [19]. A typical transfer of data in an asynchronous communication interface is presented in Fig. 11.

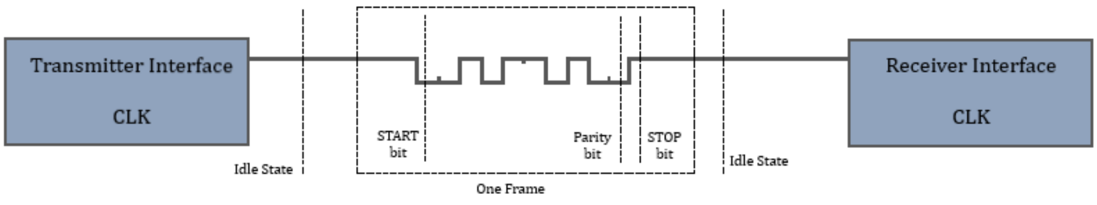


Fig. 11 Components of asynchronous communication interface

Synchronous communication interface operates by using an additional clock line to synchronize the data flow between transmitter and receiver, or the clock signal is transmitted along the data line. Fig. 12 shows that data is sampled at each rising edge of the clock, no START or STOP bits being used. The receiver interface starts data acquisition once it receives the SYN character, which is 16 in the hexadecimal system [18]. Because characters can be passed on repeatedly, a higher transmission rate is observed [20].

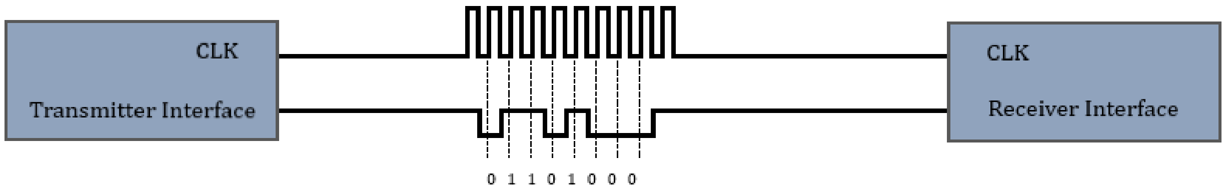


Fig. 12 Components of synchronous communication interface

Arduino board supports 3 communication protocols: I2C, SPI and UART [12]. To establish communication between Arduino Nano and the MPU6050 sensor, described in section 1.2.2, the I2C protocol is used.

1.2.2 Inter-Integrated Circuit Communication Protocol

I2C is a serial synchronous half-duplex communication protocol, that was developed by Philips. It operates based on the master/slave model, reaching a data transmission speed up to 400 KHZ [18].

Signal generation is achieved through an open drain pin – an output pin that pulls the voltage down to the ground and releases the bus data line when there is no signal from the master's/ slave's IC. Physically, the open drain pin represents the drain pin of a FET. When the IC applies a voltage to the gate, the conductivity between the source and the drain (connected to the ground) is increased, the bus line being shortened to the ground, which in turn generates a logic LOW signal. To generate a logic HIGH signal, the pull up resistor is used, pulling the line up to a known voltage, otherwise the bus would be left floating. I2C protocol manages to avoid current shortage between two devices, since a device can only pull the bus down [21].

The master and the slave share a data line SDA and clock line SCL that can be accessed successively. A master device can initiate the transfer once the SDA line is in idle state, meaning both lines output a HIGH logic value. To read data from a slave device, the master must initiate communication by sending a START condition, the slave's address (7 bits), and a WRITE command (LOW). The START condition is interpreted as the 'falling edge' of the SDA line while the clock is HIGH. Once the byte is recorded to the slave's buffer, it outputs and acknowledges bit ACK, pulling down the SDA line during the ninth clock period's low phase, signaling success of transmission. The master device sends the register's address to be read and READ command (HIGH). After the master unhands the SDA line, the slave will output bytes of data, followed by the master's ACK bit in between. To terminate data transfer, the master will send a NACK bit, outputting a HIGH value during the ninth clock period's low phase, followed by a STOP condition, which is interpreted as the 'rising edge' of the SDA line while the clock is HIGH [21].

Consequently, the bit's state must remain stable during a clock transition, otherwise this change will be interpreted as a command toward the addressed slave device.

1.2.3 Universal Asynchronous Receiver-Transmitter

UART is a serial asynchronous hardware communication protocol, used by the Arduino board to interface between the PC and the microcontroller [22]. Usually, UART is found as a hard-core block within the physical circuitry of a device with preprogrammed characteristics that define the data frame, but it also can be implemented using FPGA technology [23]. When using a UART device, familiarization with the manufacturer's datasheet is advised, because supported features differ on each device, and being unaware may lead to erroneous data reception.

The advantage of using UART is its flexibility of frame configuration, data speed control, simultaneous data transmission and reception, and interrupt request generation for data overflow prevention.

UART's integrated circuit includes registers that track whether data is transmitted / received successfully by checking the number of stored bits in the buffer registers. Such precocious techniques, assist in communication management and efficient data transfer

[24]. The speed of data transfer is an adjustable parameter, that can be changed according to the manufacturer's requirements. Typical baud rate values employed in Arduino projects are 9600, 19200, 38400 or 57600 Bits/s. Unlike I2C protocol, UART does not use a shared CLK signal between the communicating devices, thus the frequency of data sample must be specified at both ends [25].

The latest versions of UART devices incorporate FIFO buffers. Initially, UART was able to store temporarily 2 bytes, one in the shift-register and one in the data-register. Later, FIFO buffers were introduced. FIFO can store a queue of bytes and the first byte in a queue received by FIFO is the first to be outputted [26]. 16550 UART is the most popular UART used presently, released by National Semiconductor, an American semiconductor manufacturer, which today is a part of Texas Instruments [18]. It supports both FIFO and non-FIFO operation and extends buffering capability to 16 bytes [27].

In Fig. 13, the FIFO buffer is represented as a separate data buffer that can be accessed through the FIFO control register. This approach ensures that if the speed of incoming data is greater than the speed at which the data is read, the next incoming byte won't overwrite the current byte, which in non-FIFO mode would lead to data loss [27].

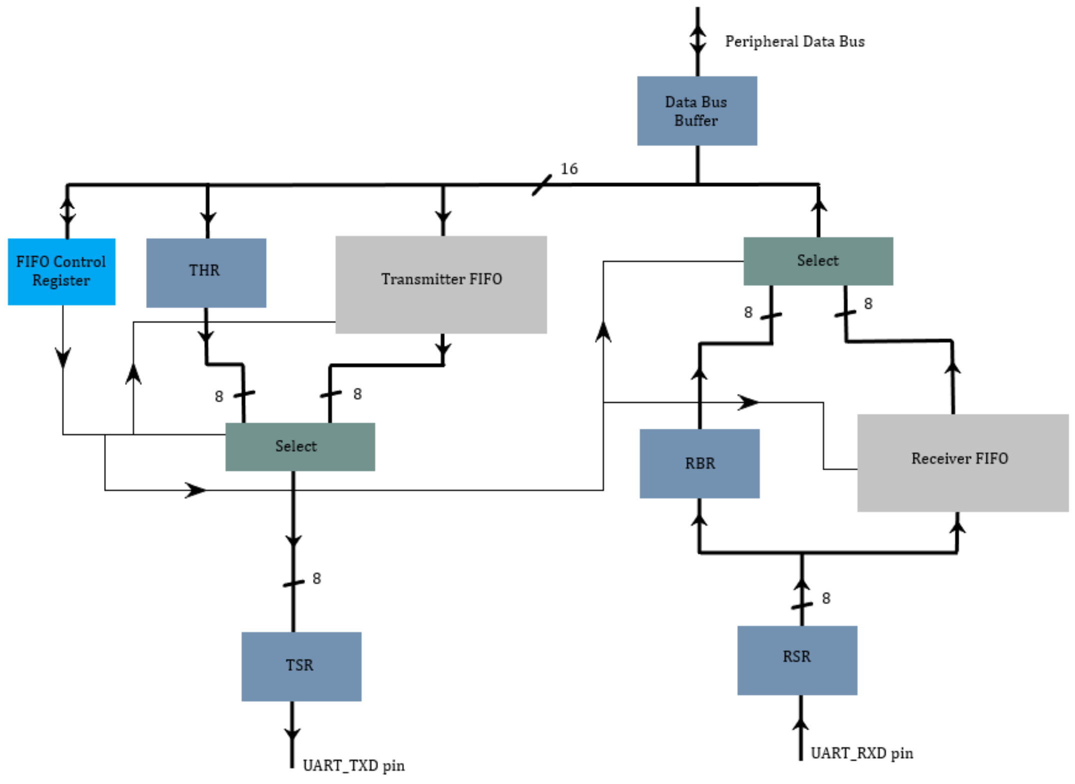


Fig. 13 Data flow diagram of an UART device

UART uses a transmitter TX, a receiver RX, and a ground GND pin [25]. The transmitter sends data, and the receiver samples it via a baud clock signal BCLK. The BCLK signal is generated by a programable baud generator that uses the MCU clock generator's outputting signal and divides it by a value to produce the desired baud rate for UART communication. This value is calculated using formula (2) [24]. For example, to achieve a baud rate of 9600 Bits/s, using a 150 MHz MCU clock signaling, the divisor will be 977.

$$Divisor = \frac{MCU\ clock\ frequency}{desired\ baud\ rate * 16} \quad (2)$$

Each bit lasts 16 BCLK cycles, and the frequency of the baud clock is 16 * the baud rate. For an UART to function at the desired baud rate, the divisor must be written in the in the DLL and DLH registers. The DLH stores the most significant bits of the divisor, and the DLL stores the least significant bits of it [24]. Correlating with the example above, 977 is 0011 1101 0001 in binary, which means that DHL will store the byte 0000 0011 and DLL will store the byte 1101 0001.

Like every asynchronous communication device, UART's data frame consists of command bits, a parity bit if requested, and between 5-8 data bits. As stated before, UART provides flexibility in data format, which is achieved by programming the line control register LCR. It stores the data that defines parity enable status, the number of data and stop bits and character length.

For UART to understand the information sent via a USB cable, a USB to UART bridge chip is used. The bridge chip used on the Arduino boards is the CH340C model, developed by Nanjing Qinheng Microelectronics [28].

UART operates in two modes: FIFO and non-FIFO. In FIFO mode the data register buffers 16 bytes simultaneously, and in non-FIFO mode it stores 1 byte [27]. In idle state, data transmission line outputs a logic HIGH value [24]. As seen in Fig. 13, to start data transmission in non-FIFO mode, parallel data is fetched into the transmitter holding register THR. Then FIFO control register enables data transmission from THR to transmitter shift register TSR, that converts one data frame into a serial queue of bits, outputting data on the UART_TXD pin [24].

Data reception starts when the START bit – a HIGH to LOW logic transition is detected, and the bit is sampled on the 8th clock cycle [18]. This means that UART samples a bit at its midpoint to ensure that the signal is valid data, not a noise spike. Received data on the

UART_RXD pin is fetched into receiver shift register RSR, where the command and parity bits are discarded, serial data is converted to parallel, and sent to the receiver buffer register BRB or receiver FIFO, depending on the chosen operation mode in the FIFO control register FCR [24].

Besides data transfer UART provides error detection capability. When UART detects an error, for example empty THR while data transmission has started, filled to the trigger level receiver FIFO, or the UART_RXD line held LOW for longer than the data frame transmission time, an interrupt request is enabled in the Interrupt Enable Register IER [24]. Such a request is forwarded to the MCU to stop data transmission. On the other hand, the Interrupt Identification Register encodes that an interrupt request is pending and the type of occurred error [24].

1.3 Sensing Devices

Sensing devices are the pivotal electronic components of this project and understanding their operation as well as possible error sources will facilitate their effective implementation.

1.3.1 Sharp IR GP2Y0D02YK0F sensor operation

Sharp Corporation is a Japanese company that was founded in 1912 by the businessman Tokuji Hayakawa [29]. Starting with belt buckles and adjustable flow faucets, Sharp company expanded to radio and television production [29], which laid the path for further inventions such as LCD modules, image sensors, optoelectronic and laser sensors [30].

There are two main types of position sensors developed by Sharp: proximity sensors that output two values of voltages, based on the presence of an object at the distance specified in the corresponding datasheets, and PSD-s that output a voltage which corresponds nonlinearly to a measured distance [31].

For this project, the Sharp Infrared Sensor GP2Y0A21YK0F will be used. This model consists of a PSD unit, an IRED and a signal processing circuit. It has a working upper limit of 80 cm, requires a supply voltage of 4.5-5.5V, and outputs a signal with a maximum delay of 5.0ms after the detection of radiation [31].

Its main component, the PSD unit, is designed to detect incident radiation and measure the intensity with the use of a photodiode. The photodiode functioning is based on the photoelectric effect. For the conversion of light signal into the electrical one, PIN or PN diode is required [32]. A PIN diode is made up of three layers, a p-type semiconducting material, a n-type one, and an intrinsic semiconductor, surrounded by the previous two [33]. The intrinsic layer acts as a sensitive area to the incident radiation. When radiation penetrates the crystal lattice, an electron – hole pair is formed. The electric field draws away the electron and the hole in opposite directions, thus generating an electric current. The width of the intrinsic material determines the quantum efficiency value, meaning a higher chance to form an electron-hole pair [34]. This type of diode is used under reverse-biased mode; consequently, a small amount of current will be present even when there is no detection of light.

The PSD receives a modulated infrared signal of a wavelength about 880nm, emitted by the IRED and reflected by the obstacle [35]. To compute the distance from the sensor to the object, the triangulation method is used [36]. Referring to Fig. 14, a flux of radiation hits an obstacle and bounces off at a specific angle a’. Some of the emergent radiation is

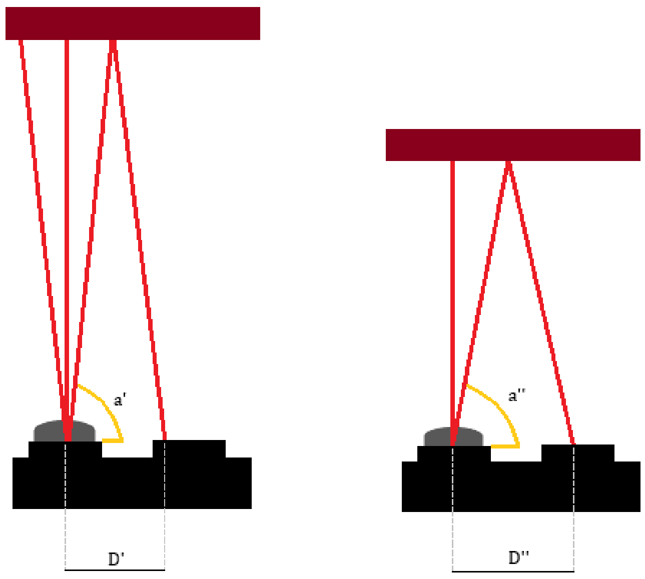


Fig. 14 Triangulation method schematic of IR Sharp distance detection

detected by the CCD array (the PSD unit) and the signal processing circuit computes the distance D'. The distance between the sensor and the obstacle is determined using trigonometric formulas.

The GP2Y0A21YK0F model application note specifies that the sensor is little influenced by the surface color or by the surface reflectance [31]. Because IR sensor's output may vary slightly depending on the particularities of internal structure, it is a good practice to output the experimental data of the sensor and calculate the transfer function, instead of using the graph provided in datasheet [37].

1.3.2 HC-SR04 Sensor Operation

HC-SR04 is an integrated circuit that detects and measures the distance between an obstacle and the module itself. It sends a burst of sound waves at a frequency of 40 kHz, which is above the human ability to hear. The burst consists of 8 cycles of air contraction and rarefaction [38]. HC-SR04 detects distances between 2 and 400 cm, outputting stable and accurate signals. As seen in Fig. 15 a, in order to integrate this sensor into an electronic circuit, one must wire 4 pins. VCC pin corresponds to "Voltage at the Common Collector" and represents the power supply pin, which along with the GND pin, generates current flow. TRIG pin is the sensor control pin, that signals to the sensor when to take a measurement. ECHO pin outputs a HIGH signal with the duration proportional to the distance between the sensor and the obstacle. According to Fig. 15 b, HC-SR04 is controlled by sending a 10 μs pulse or longer to the TRIG pin, which triggers the internal circuitry to send an 8-cycle ultrasonic burst and measure the duration until the burst is detected by the module. Then, it sets the ECHO pin HIGH for the detected direction, or for 38 ms if the burst is not detected [38].

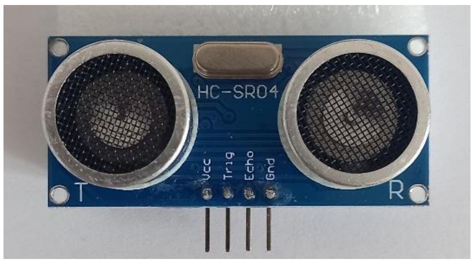


Fig. 15 a) Picture of the HC-SR04 sensor

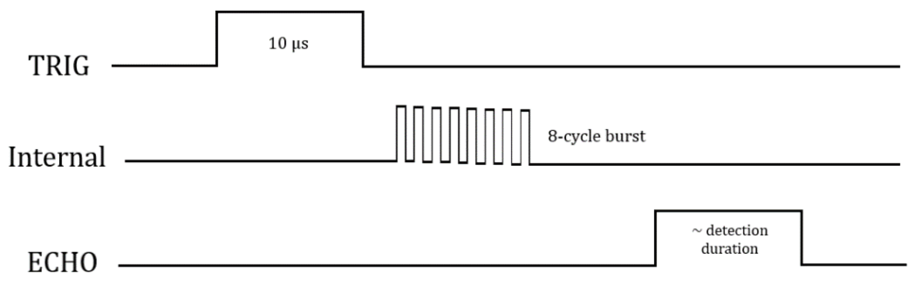


Fig. 15 b) Timing diagram of the HD-SR04 operation

Because the sensor operates with sound waves, its output depends on different parameters, such as temperature, pressure, and humidity [39]. For standard parameters the speed of the sound is 344 m/s [40]. Considering that the wave travels back and forth, the actual duration of distance traveling is half of the measured time. Thus, to calculate the distance between the object and the sensor, the following formula is used:

$$distance = 0.5 * sound\ speed * measured\ time$$

Of course, there are some drawbacks and sources of errors introduced into the system because of using HC-SR04. For example, cloth is an acoustically soft material that is hard to detect using sonar waves. If the object is oriented at a big enough angle with respect to the surface of the sensor, the wave will scatter into a direction out of the detection range. If the sensor will receive the signal due to echo while it has sent a burst of waves, it will consider that the object is closer than it is [39].

1.3.3 MPU6050 sensor operation

To output proper data to the user, this project requires both data about the distance relative to an object and the inclination angle of the PSD sensor about two axes of an inertial frame of reference. The inclination angle can be determined using an inertial sensor. Such sensors can measure angular velocity and acceleration relative to an inertial frame of reference and convert data into electrical signals that can be digitally processed for further use [41].

Nowadays, accelerometers are manufactured using micromachining technology, a methodology used to create microstructures, that are either craved in a silicon wafer (bulk micromachining) or are developed through thin film deposition (surface micromachining) [42]. Because of the implied production technology and the operation principle, that will be explained below, current accelerometers are considered microelectromechanical systems, or MEMS [43]. Such systems incorporate several microsensors that interact with the medium and a central unit that processes input data to make it readable for the user [44].

The microsensor inside a MEMS accelerometer consists of a proof mass that moves relative to a fixed frame of reference due to an inertial force, according to the measured parameter [43]. There are several types of accelerometers based on the type of sensing element being used, piezoresistive, capacitive, piezoelectric, and tunnelling accelerometers being some of them [41].

MPU6050 is an integrated circuit, which incorporates an on-chip DMP and calibration firmware, mounted on the GY-521 breakout board (see Fig. 16). It was developed by InvenSense Incorporation, an American consumer electronics manufacturer of inertial sensors for consumer electronic devices, such as smartphones and gaming consoles [45]. MPU6050 IMU measures and outputs acceleration and angular velocity along each of the inertial reference frame's axes.

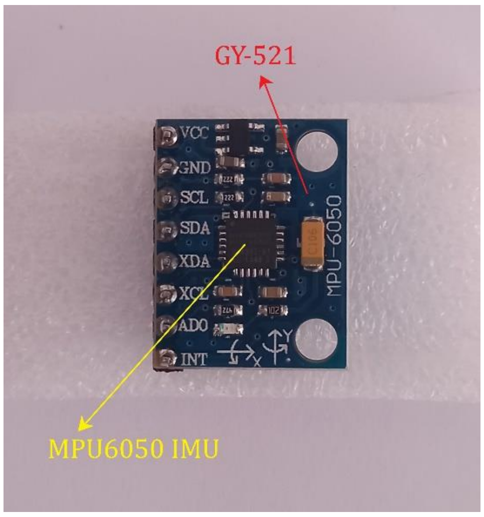


Fig. 16 MPU6050 IMU sensor, mounted on the GY-521 breakout board

MPU6050 incorporates 108 registers with different functionalities, that store information about the configuration parameters or measured data. The sensor supports I2C communication with a transfer rate of 400 kHz in Fast Mode operation [46]. Thus, the MPU6050 registers can be accessed by wiring the sensor to a host processor and requesting data according to the I2C protocol. It requires an operating voltage in the range of 2.375 – 3.46 V and a current of 3.9 mA when the al the 6 MEMS and the DMP are enabled, thus the Vcc pin on the GY-521 board will be connected to a 5V supply voltage to cover the power consumption requirements for the peripheral electronic components mounted on the breakout board for the MPU6050 proper operation.

The advantage of MPU6050 IMU implementation is the incorporation of a Digital Motion Processor (DMP) that collects data from the 6 MEMS sensors and implements firmware algorithms to compute quaternions [46].

MPU6050 is a capacitive sensor, meaning that the existence of acceleration along one of the axes is detected by measuring some change in capacitance. Fig. 17 introduces the simplified version of a capacitive sensing element.

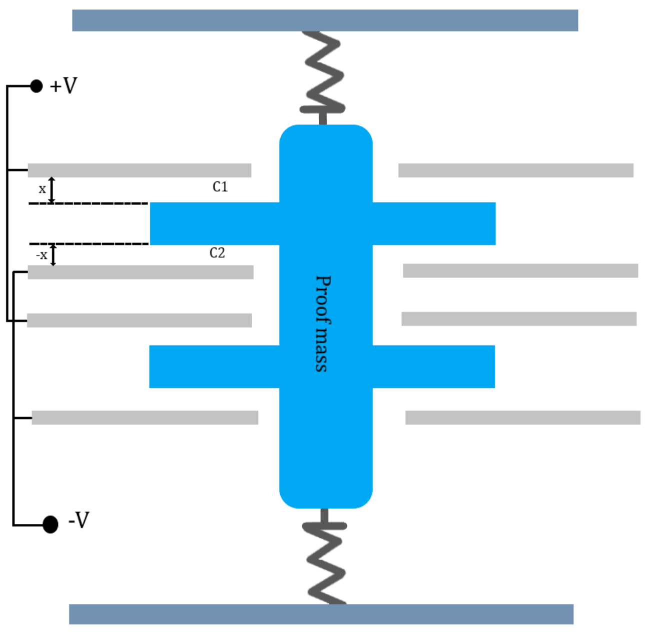


Fig. 17 Diagram of a capacitive sensing element in an accelerometer sensor

When the proof mass moves away from its equilibrium state a change in capacitances C_1 and C_2 between the legs of the proof mass and the fixed electrodes occurs. The part of the system consisting of one proof mass leg and 2 consecutive electrodes can be considered as two condensers wired in parallel, thus the total change in capacitance can be calculated as follows [47]:

$$C_1 = \varepsilon * \frac{S}{d + x} \quad C_2 = \varepsilon * \frac{S}{d - x} \quad \Delta C = C_2 - C_1$$

where ε is the electric permittivity of the medium, d is the distance between the plates when the system is at rest, and S is the area of the plate.

Considering the limit of small displacements of x, which is applicable for dimensions of 10^{-6} order, and that x is proportional to the acceleration exerted on the system:

$$x = 2 * \Delta C * \frac{d^2}{\varepsilon * S} \qquad a = \frac{k}{m} * 2 * \Delta C * \frac{d^2}{\varepsilon * S}$$

where m is the proof mass, k is the elastic constants of the springs attached to the proof mass and a is the acceleration exerted on the system.

The change in capacitance corresponds to a potential difference that can be measured, processed, and output as readable data [41].

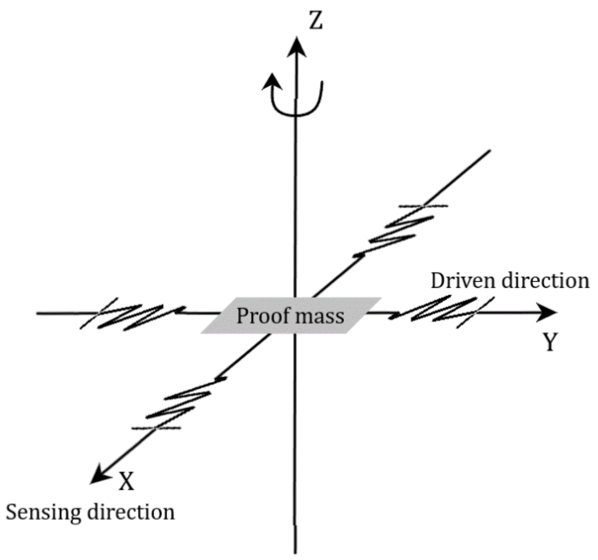


Fig. 18 Diagram of a surface sensing element in a gyroscope sensor

To measure angular velocity, MPU6050 uses a mechanical structure similar to one used for measuring linear acceleration, but the sensing element employs the effect of Coriolis force for its operation. According to Fig. 18, the microsystem is driven into an oscillating mode with constant amplitude and frequency along the Y axis, called driven axis. When an angular velocity is applied perpendicular to the plane of oscillation, around the Z axis, due the existence of a change in velocity, the Coriolis force acts on the proof mass in the direction of this change (X axis), called the sensing direction, which consequently excites the mass into a oscillating mode [48].

Because the oscillation amplitude in the sensing direction is proportional to the angular velocity applied along the Z axis [43], such relation can be employed to measure the angular rate.

Fig. 19 depicts a surface micromachined Z-axis vibratory rate gyroscope, which uses a capacitive sensing element (comb finger deflection sense capacitors) to transform physical displacement of the proof mass in the sensing direction into a measurable electrical signal, similarly to the accelerometer's capacitive sensing element.

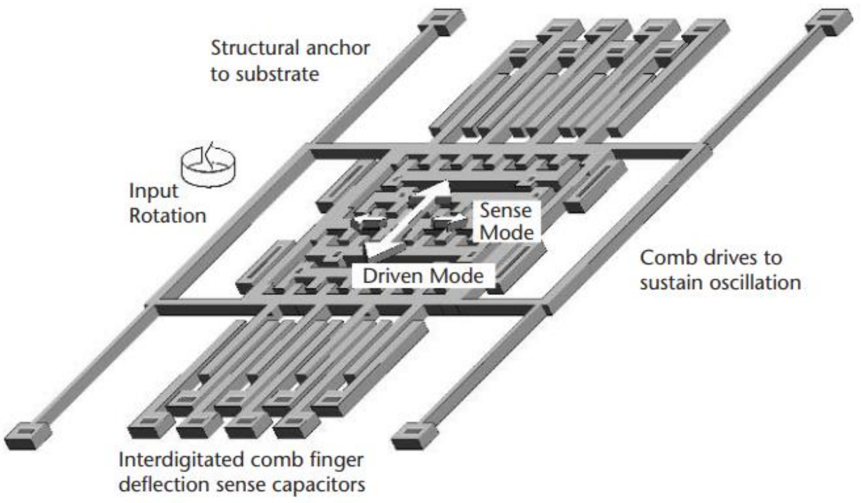

Structural anchor to substrate
Input Rotation
Sense Mode
Driven Mode
Comb drives to sustain oscillation
Interdigitated comb finger deflection sense capacitors

Fig. 19 Surface micromachined Z-axis vibratory rate gyroscope (Source [49]*)*

According to Fig. 110, each angular rate about one of the 3 axes of rotation is measured with a separate vibratory MEMS gyroscope, employing the effect of the Coriolis force, as stated above. The output signal, sampled at a rate of 8 kHz in Fast Mode operation, is amplified, demodulated, filtered [46] and fetched into the 16-bit ADC, which converts the voltage signal into a numeric value, stored in one of the corresponding registers. The acceleration data from the capacitive MEMS accelerometers is processed in a similar approach and stored in the intended registers.

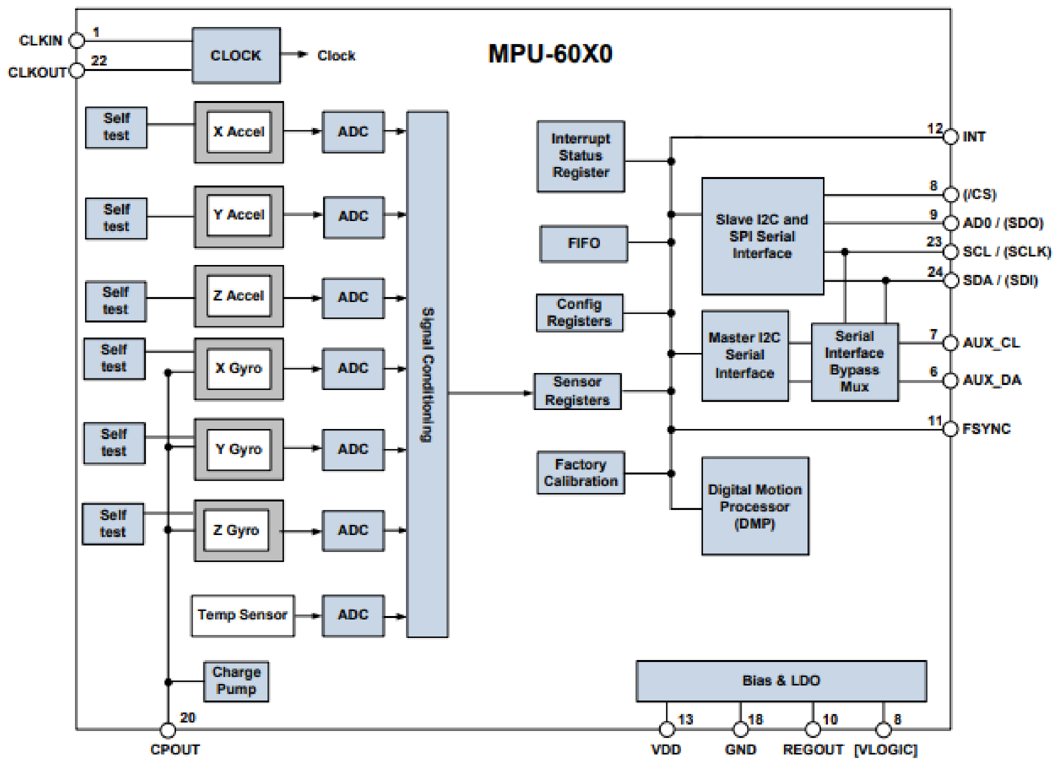


Fig. 110 MPU6050 IMU block diagram (Source [46]*)*

MPU6050 gyro supports a configurable output range of ∓250, ∓500, ∓1000 or ∓2000 dps and a configurable sensitivity scale of 131, 65.5, 32.8, or 16.4 LSB/dps [46], that can be set depending on the accuracy requirements of the user.

When MPU6050 is placed on a flat surface, the accelerometer will measure 1g on the Z axis and 0 on the X and Y axis. Its output range supports a scale of ∓2, ∓4, ∓8, and ∓16 g, with a sensitivity scale factor configurable to 16384, 8192, 4096, and 2048 LBS/g [46].

Chapter 2

Technical Implementation

This chapter will focus on sensors' implementation and guidance for proper data acquisition, as well as description of the implied communication interface for the MPU6050 module and utilized software packages.

2.1 Concept of the project

As stated before, the purpose of this project is to assist a person with visual deficiency in space orientation. For achieving this task, distance information is converted into haptic data. Four distance measuring sensors are mounted on the frontal (HC-SR04), dorsal (IR) and lateral (IR) sides of the head. Each sensor is configured to output a distance up to 80 cm. When the head is inclined, the measuring distance does not correspond to the actual sensor-obstacle span. To eliminate this error, an MPU6050 sensor is used. It measures linear acceleration and angular speed, as well as performs on-board calculations and outputs inclination angles along the three axes of orientation. Projected distances are used to calculate a proportional voltage that is input to the vibrational disk motor. Thus, each sensor has a corresponding motor which vibrates with a frequency proportional to the sensor-obstacle projected distance.

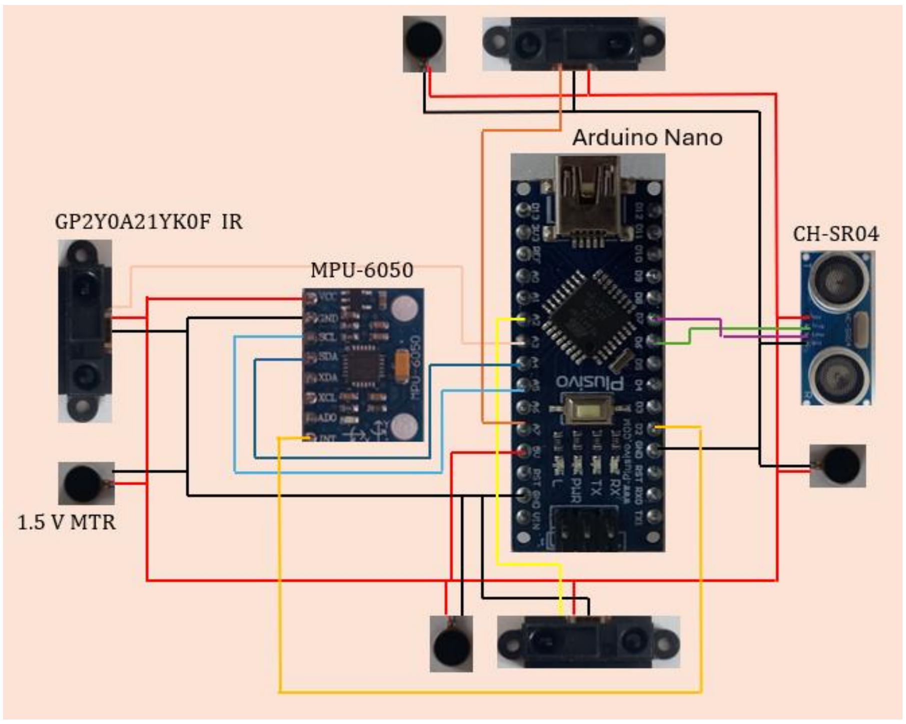


Fig. 21 The wiring diagram of all electronic components involved in this project

2.2 IR GP2Y0A21YK0F Sensor Implementation

Adding the GP2Y0A21YK0F sensor to the project and making it output proper data is quite a straightforward task. According to Fig. 21 and GP2Y0A21YK0F datasheet, the sensor has three pins, two of which are connected to the supply voltage 5V (red) and ground GND (black). The third one outputs a voltage nonlinearly proportional to the distance between the sensor and the object, which is converted to a numeric value by the 10-bit ADC incorporated on the Arduino Nano board [50].

An Arduino board can detect 1024 (2^10) discrete voltage levels, or a change no less than 0.0048 V of the output voltage, considering a power supply of 5V. The value of the voltage level was calculated using the formula:

$$detectable\ voltage\ level = \frac{power\ supply\ voltage}{2^n - 1}$$

where n is the number of ADC's bits.

To find the distance according to the output voltage, one must calculate the corresponding transfer function for the implied sensor.

2.2.1 Data Linearization

As stated before, the IR sensor's voltage output is a nonlinear function of distance, which can also be observed in Fig. 32. There are several methods to predict an unknown distance with respect to the measured voltage, such as using a look-up table, linearizing the data on smaller ranges within the input range of the sensor, modeling the sensor's behavior with an exponential or power transfer function [51]. In this project, the latter method will be applied.

The graph in Fig 32. was plotted using experimental data from one of the GP2Y0A21YK0F sensors. Let's assume that this plot can be fitted with a power function of the form:

$$y = a * x^b$$

To linearize the function above, one can apply the logarithm operation:

$$\ln(y) = \ln(a) + b * \ln(x)$$

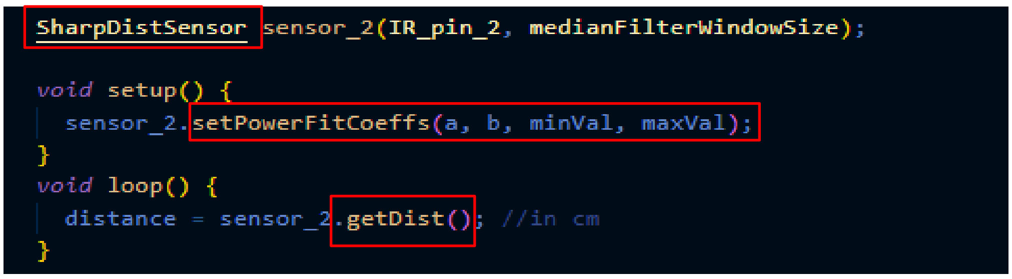
Thus, the coefficients a and b can be calculated by plotting the logarithmic values of distance as a function of voltage.

2.2.2 SharpDistSensor Routine

The SharpDistSensor routine is an open-source library which can interface with the IR Sharp sensors [52]. It provides calculated power and polynomial coefficients for the transfer function of IR sensors with different output ranges, as well as routines for custom power and polynomial functions. For this project the custom power function approach will be used.

Another useful feature of this library is real-time median filtering. This non-linear filtering technique replaces each sensor output with the median of last n entries, where n is the dimension of the shifting window [53]. For example, considering the set 4 5 4 9 15 4 3 5 and a window size of 3, the output set is replaced by 0 4 4 5 9 9 4 4 5. One can observe that value 15 was removed and output changes in a gradual manner.

To interface an IR sensor with the PlatformIO IDE, the SharpDistSensor class is used. In Fig. 22, the *setPowerCoeffs()* constructor assigns the coefficients a and b, calculated using the method described in 3.2.1, to the object *sensor_2*, as well as sets the analog input range within *minVal* and *maxVal*. The *getDist()* constructor calculates the distance corresponding to the output voltage, fitting type, and filters data using real-time median filtering, if specified.



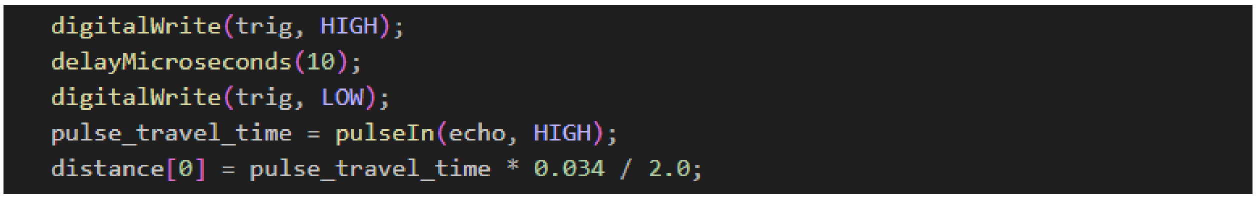
```
SharpDistSensor sensor_2(IR_pin_2, medianFilterWindowSize);

void setup() {
  sensor_2.setPowerFitCoeffs(a, b, minVal, maxVal);
}
void loop() {
  distance = sensor_2.getDist(); //in cm
}
```

Fig. 22 Code example for IR Sharp sensor data sampling

2.3 HC-SR04 Sensor Implementation

As specified in subchapter 1.3.2, HC-SR04 outputs stable signals and there is no need to implement any filtering on the measurements of this sensor, as long as the readings are accurate enough for the purpose of this project.



```
  digitalWrite(trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig, LOW);
  pulse_travel_time = pulseIn(echo, HIGH);
  distance[0] = pulse_travel_time * 0.034 / 2.0;
```

Fig. 23 Code example for HC-SR04 data sampling and transfer function implementation

To make HC-SR04 output data, the user sets a voltage of HIGH = 5 V on the TRIG pin using bult-in *digitalWrite()* function (see Fig. 23) and delays the implementation of the next function for 10 μs. After that, the code sets LOW = 0 V on the TRIG pin and measures the duration of the signal on the ECHO pin using *pulseIn()* function. Considering that time is returned in μs, the transfer function is written according to the stated formula in subchapter 1.3.2.

2.4 MPU6050 IMU Implementation

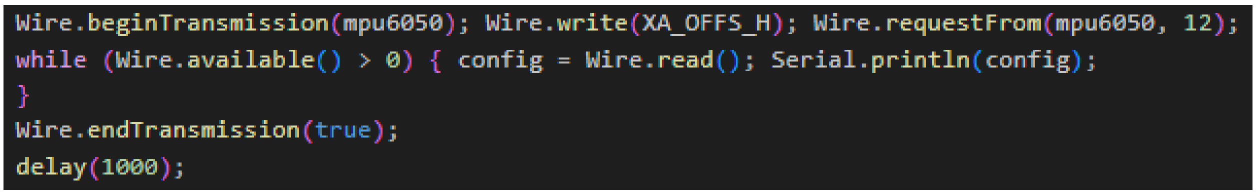
MPU6050 IMU is a complex sensor with a variety of functionalities, whose implementation requires following a sequence of predefined steps to calibrate, transfer data and retrieve measurements.

According to Fig 16. , MPU6050 has 8 pins, 4 of which must be wired to establish communication between the sensor and the Arduino board. Voltage common collector pin Vcc is connected to the 5V power supply line and GND pin to ground. Pins SDA and

SCL are used to establish communication between the sensor and the Arduino board through the I2C protocol (see subchapter 1.2.2) and receive data stored in registers.

2.4.1 Establishing I2C Communication using the I2Cdev.h library

As explained in subchapter 1.2.2, there are several steps that must be taken to establish I2C communication between the MPU6050 and the Arduino board. Fig. 24 represents an example of code that initiates I2C communication and requests 12 bytes of information from the MPU6050's XA_OFFS_H = 6 register. It uses the Wire.h library which comes in the Arduino IDE package [54]. This routine abstracts low-level bit manipulation and provides a more friendly approach to use I2C.

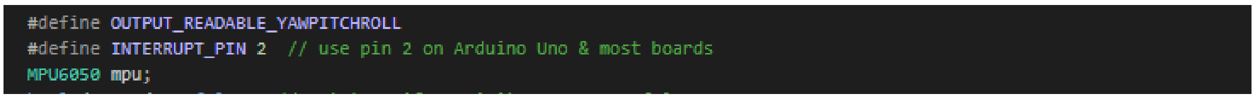


```
Wire.beginTransmission(mpu6050); Wire.write(XA_OFFS_H); Wire.requestFrom(mpu6050, 12);
while (Wire.available() > 0) { config = Wire.read(); Serial.println(config);
}
Wire.endTransmission(true);
delay(1000);
```

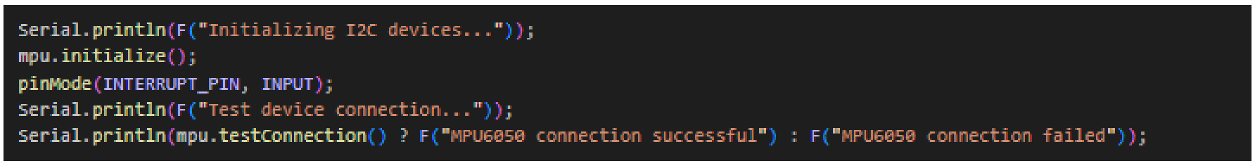
Fig 24. Code example for I2C communication initialization and data request from the slave device using Wire.h library

Wire.h library is pretty much enough to implement everything that has been employed in this project. Because there are a considerable number of steps to implement, such as initiating I2C, testing communication, setting offsets, calibrating MPU6050, enabling DMP, reading from DMP, converting quaternions in rotation angles, enabling interrupts, it would be tedious to work with the Wire.h library. For this reason, the I2Cdev.h routine will be used.

I2Cdev.h library is implemented for a variety of devices and development boards that use I2C communication, including MPU6050 [55]. It provides a range of classes corresponding to different I2C devices and an intuitive approach to manipulate with them, abstracting the necessity to access registers within the sensor or to implement sophisticated algorithms.



```
#define OUTPUT_READABLE_YAWPITCHROLL
#define INTERRUPT_PIN 2  // use pin 2 on Arduino Uno & most boards
MPU6050 mpu;
```



```
Serial.println(F("Initializing I2C devices..."));
mpu.initialize();
pinMode(INTERRUPT_PIN, INPUT);
Serial.println(F("Test device connection..."));
Serial.println(mpu.testConnection() ? F("MPU6050 connection successful") : F("MPU6050 connection failed"));
```

Fig. 25 Code example for I2C communication initialization between MPU6050 and an Arduino board using I2Cdev.h library

Fig. 25 shows how to establish connection between the Arduino board and MPU6050. The class corresponding to the MPU6050 sensor is named after it, thus mpu represents an object that inherits all member functions of the class. As specified in subchapter 1.2.2, each slave device on the I2C line has a unique standardized address, that must be sent to the master device. By creating the mpu object, a constructor within the class will send the default MPU6050 address, which is 0x68 [56]. Calling the member function initialize() will turn off the sleep mode and will set the Full Scale Range to ∓2g for the accelerometer, and to ∓250 dps for the gyroscope, on the MPU6050 IMU. The function testConnection() will return True if the slave device responds properly to the master, thus the user will know that lack of data transmission can be due to faulty connection.

2.4.2 IMU_Zero Calibration Routine

When MPU6050 IMU lays on a flat horizontal surface, all values for angular speed must be zero, as well as the measurements of linear acceleration, except for the Z axis, which must output 1g. Practically, the user will observe small fluctuating non-zero values, that will inevitably alter the output signal. To compensate for such errors, one must determine the optimal offsets or voltages, which applied to the signal, will cause the sensor output zero values and 1g for the Z axis ideally [57].

Registers 0x06 to 0x18 store preprogrammed offset values for acceleration and angular speed along all three axes which will be applied to the sensor's output [58], unless custom offset values are sent to the registers.

IMU_Zero.ino is a routine developed by Robert R. Fenichel that implements PI tuning to find optimal offsets for the MPU6050 IMU [59]. PI control is a type of feedback control, that applies proportional and integral action over the control error to minimize the influence on internal variations of the sensor [60]. P control acts on the control variable (in this case on the offset) proportionally to the difference between the reference point (0 for all values, except 1g for acceleration along the Z axis) and actual measurements. I control acts on the control variable proportionally to how control error evolves over time, meaning that it considers the effect of past errors over the control variable as well. PI is implemented using the formula:

$$u = K_p * e + K_I \int e * dt$$

where u is the control variable, e is the control error, K_P and K_I are independent user-defined parameters.

The way IMU_Zero.ino implements PI control is by considering a LOW and a HIGH limit of offset values for MPU6050 in steady state. Given the current LOW offset values, if the measurements are above the setpoint, the routine will subtract a value from the LOW offset. In case of the HIGH offset values, if the measurements are below the setpoint, the routine will add a value to the HIGH offset. This way a rough range of allowed values for the offset will be created.

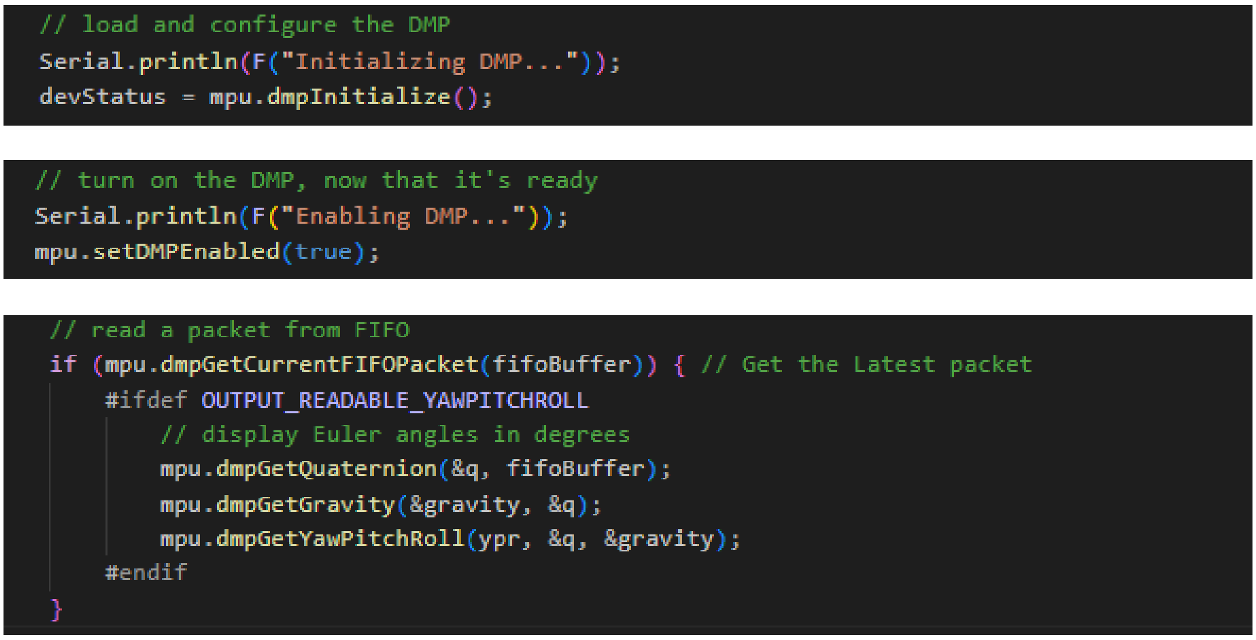
To achieve precise values, the calculated range will undergo a finer tuning method and will be narrowed to small a value, such that the measurements will be close enough to the setpoint. The results of this calibration method will be presented in subchapter 3.2.1.

2.4.3 MPU6050_DMP6 Routine

There are a few ways to calculate rotation angles, and the most straightforward is to integrate angular speed over time. The drawback of this method is that integration adds all past errors to the current value, affecting actual measurements. A better approach is to calculate rotation angles using acceleration [61]. Yet, the accelerometer's values fluctuate around (0,0,1g) and do not output stable measurements.

To compensate for internal noise and gyroscope's drift [62], modern IMUs incorporate DMP units with firmware that combine multiple sensor data and perform calculations to determine the state of the system. The advantage of using an onboard DMP is that it runs complex fusion algorithms such as Kalman filter [63], that can barely run on an Arduino microcontroller because of the RAM limitations.

The DMP unit on the MPU6050 IMU combines gyroscope and accelerometer data to calculate the state of the system, and outputs it as quaternions. Further calculations lead to extracting either Euler angles or the coordinates of aircraft principal axes [64] [65].



```
// load and configure the DMP
Serial.println(F("Initializing DMP..."));
devStatus = mpu.dmpInitialize();
```

```
// turn on the DMP, now that it's ready
Serial.println(F("Enabling DMP..."));
mpu.setDMPEnabled(true);
```

```
// read a packet from FIFO
if (mpu.dmpGetCurrentFIFOPacket(fifoBuffer)) { // Get the Latest packet
    #ifdef OUTPUT_READABLE_YAWPITCHROLL
        // display Euler angles in degrees
        mpu.dmpGetQuaternion(&q, fifoBuffer);
        mpu.dmpGetGravity(&gravity, &q);
        mpu.dmpGetYawPitchRoll(ypr, &q, &gravity);
    #endif
}
```

Fig. 26 Code example of MPU6050 DMP initialization and acquisition of data about the state of the system in the form of quaternions and rotational angles about aircraft axes.

To obtain information about rotation angles of the system, this project exploits the MPU6050_DMP6 routine developed by Jeff Rowberg [66]. According to Fig. 26 , the routine initializes, enables, and reads from the DMP, using a set of MPU6050 class member functions. *dmpInitialize()* function resets the sensor, disables sleep mode, revises hardware configurations, sets DMP interrupts, sample rate, and gyro sensitivity. The *if* statement checks whether the FIFO packet doesn't overflow the allocated *fifoBuffer* buffer, and retrieves the quaternion data, as well as performs calculations by executing *dmpGetYawPitchRoll()*.

2.5 MPU6050, GP2Y0D02YK0F, and HC-SR04 Sensor Output Dara Combination and Vibrating Disk Motor Incorporation

Both GP2Y0D02YK0F and HC-SR04 output the same information, but the advantage of using HC-SR04 sensor is larger output range and stability. For this reason, it was chosen as the main sensor that coordinates forward flexion [67]. For backward flexion and side bending GP2Y0D02YK0F sensors will be used.

As seen in Fig. 27 , to obtain the distance between the sensor and the obstacle one must compute the projection of the vector pointing from the sensor to the object in the horizontal plane YOX corresponding to a fixed reference system represented in black. For example, considering the distance along the OY axis, the projection d1 is calculated using the formula:

$$d_1 = Distance_{OY} * \cos\,(Angle_{Roll})$$

Similarly, the projection along the X axis is computed by substituting roll angle with the pitch angle.

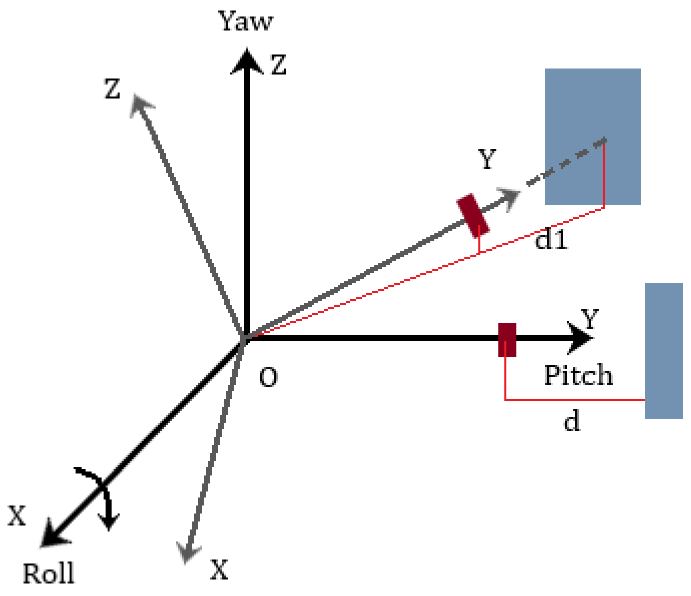


Fig. 27 Representation of distance projection along one axis of the reference system attached to the sensor

The following step is to transform distance into information that a person with visual impairment can understand. To achieve this task, a vibrating disk motor will be used (see Fig. 28).

Fig. 28 Picture of a vibrating disk motor

There are several ways information can be represented through vibrations. One can modulate the amplitude of the vibration of a single actuator or transmit pulses of voltage at a specific frequency. Because the second method introduces a delay into the system, it will output erroneous information during real-time performance, thus the first approach will be considered.

Assume that the output range of a distance sensor is [m,k], m corresponding to minimum value, and k to the maximum value. Also, suppose that the working voltage range of the disk motor is [0,z] volts. To compute the voltage with respect to an intermediate distance, consider a linear function y = a * x + b, where y is the input voltage and x is the distance projection. If z volts correspond to m centimeters and 0 volts correspond to k centimeters, the following function arises:

$$y = \frac{z}{m-k} * x + \frac{z}{1-\frac{m}{k}}$$

Implementing this function on the projected output of each sensor, specifying m, k, and z parameters, and sending a proportional voltage to the disk motor, will transform distance data into sensory information.

Chapter 3

Analysis of results

Lastly, this chapter focuses on the experimental proof and validation of the projects utility by displaying measurements from the sensors and compare it with real data.

3.1 HC-SR04 and GP2Y0D02YK0F sensor output interpretation

As stated previously, the circuit incorporates two types of position sensors: HC-SR04 module which operates with ultrasonic waves, and GP2Y0D02YK0F sensor that uses the effect of electromagnetic radiation incident on a CCD array.

The output measurements of the HC-SR04 sensor are in accordance with the real distance values, exhibiting an average relative error of 3 %. Because noise spikes were not observed during data acquisition, the application of filtering is not necessary for this sensor. According to Fig. 31, sensor output fluctuates little around the line of equality and provides sufficiently accurate measurements.

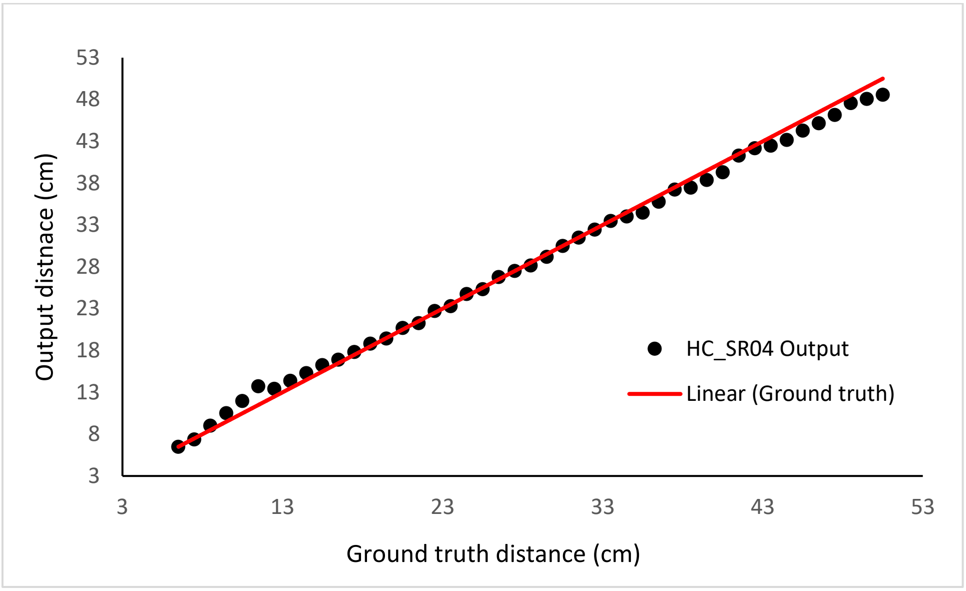


Fig. 31 The graph displays the HC-SR04 sensor output data with respect to the actual distance represented by the line of equality (red line)

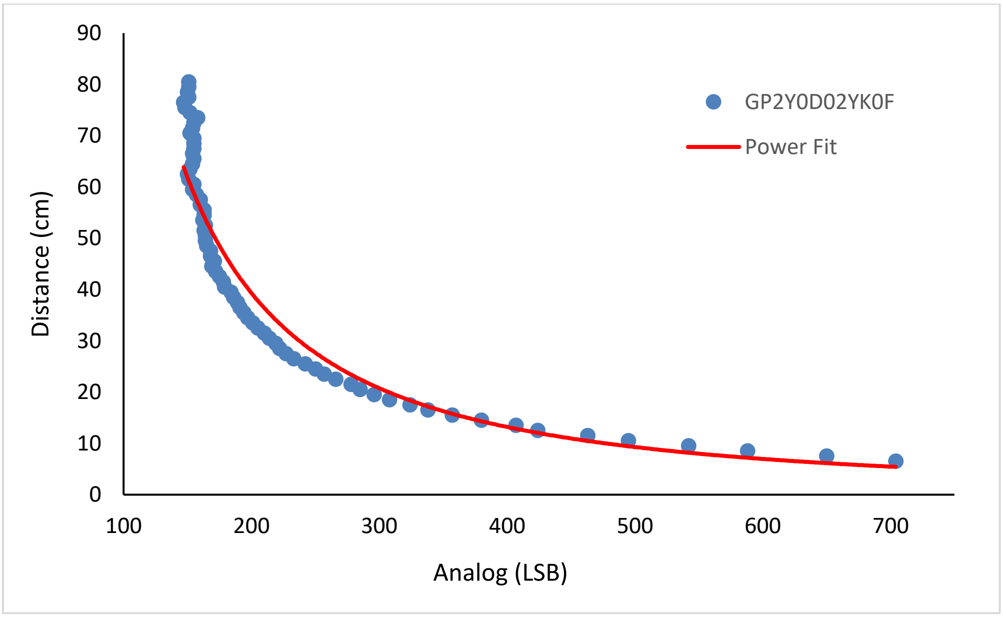


Fig. 32 The graph displays the reciprocal of GP2Y0D02YK0F sensor voltage output with respect to measured distance

Compared to the sensor described above, GP2Y0D02YK0F is less efficient in outputting accurate data, due to its non-linear characteristics. According to Fig. 32, the sensor is most vulnerable in the vicinity of the output range's upper limit, where sensitivity has the lowest value (sensitivity is proportional with the slope; also, pay attention that Fig. 32 is the reciprocal of the sensor characteristics, thus, to analyze qualitatively the slope, refer to attachment 9 in the application note [31]). The effect of the latter deduction is reflected in Fig. 33 , where the output close to the upper limit of the range deviates drastically from the line of equality.

To avoid further misinterpretations, the unit of measurement of the horizontal line in Fig 32. is LSB and not volts, because Arduino board converts inputted voltage into its numeric representation according to the ADC bit-number (10-bit for Arduino Nano) [68].

The next step after data acquisition is either linearization and linear fitting or power fitting. The red line in Fig 42. represents a power fit, plotted accordingly to the equation:

$$y = 1651148 * x^{-1.575}$$

where x is the numeric representation of voltage and y is the distance.

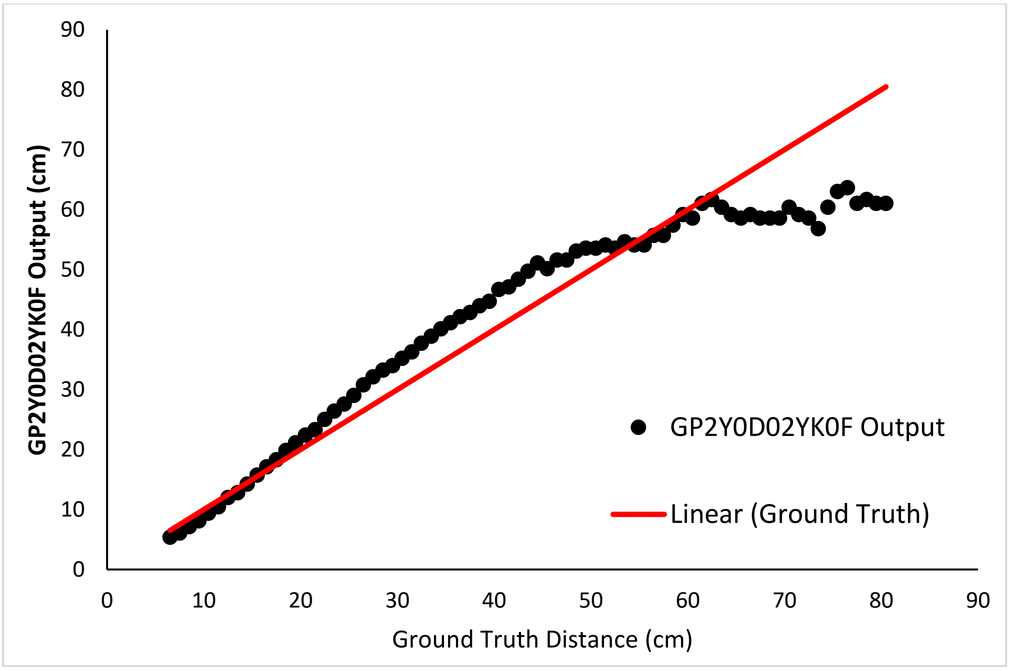


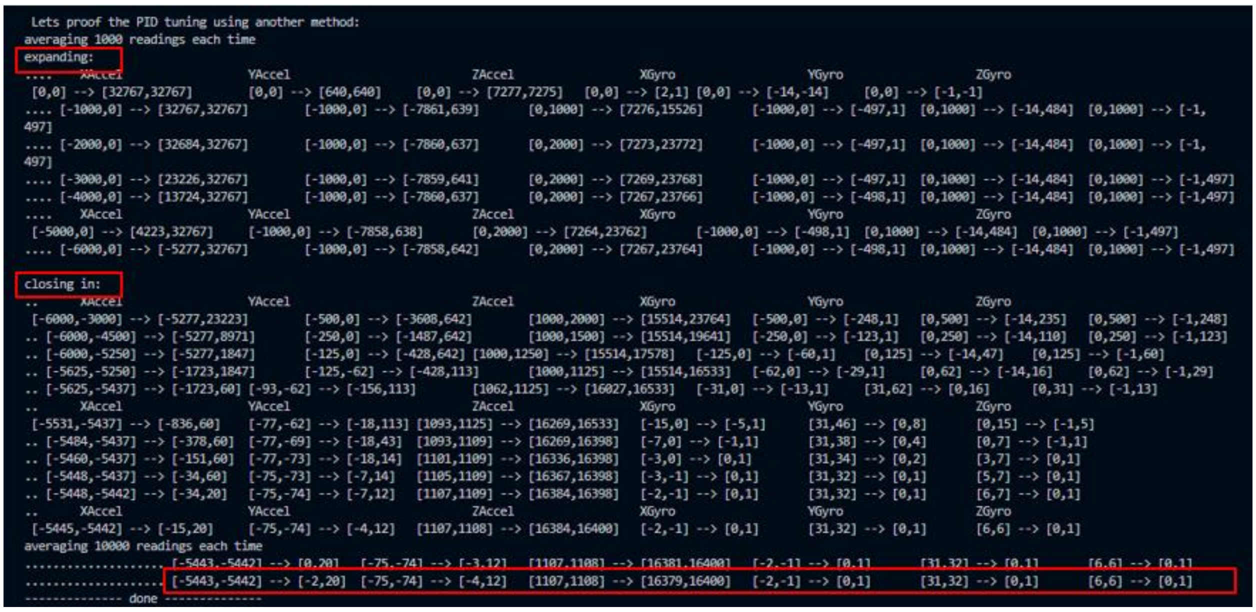
Fig. 33 The graph displays the GP2Y0D02YK0F sensor output data computed with a power function fit, with respect to the actual distance represented by the line of equality (red line)

3.2 MPU6050 IMU output data

Except the applications described in subchapter 2.4.3, stated routines perform real-time recalibration, using offsets provided by the user as a starting point. This approach, as well as onboard motion processing increases the accuracy of the output, providing stable and valid measurements.

3.2.1 Calibration results

Fig. 34 presents calibration results for MPU6050 module, that will be used for real-time calibration in the MPU6050_DMP6 routine. The Full-Scale Range is set to ∓2g for the accelerometer and ∓2000dps for the gyroscope, which corresponds to a 16.4 LSB/dps sensitivity for the gyroscope and a 16384 LSB/g sensitivity for the accelerometer [69]. Considering this range and the results given in the Fig., the offsets for gyroscope's (x, y, z) axes are (0.12, 1.89, 0.36) dps and for the acceleration's z axis is 0.06g.



```
 Lets proof the PID tuning using another method:
averaging 1000 readings each time
expanding:
....     XAccel                YAccel                    ZAccel               XGyro                  YGyro                ZGyro
 [0,0] --> [32767,32767]       [0,0] --> [640,640]     [0,0] --> [7277,7275]   [0,0] --> [2,1] [0,0] --> [-14,-14]     [0,0] --> [-1,-1]
.... [-1000,0] --> [32767,32767]       [-1000,0] --> [-7861,639]       [0,1000] --> [7276,15526]       [-1000,0] --> [-497,1]  [0,1000] --> [-14,484]  [0,1000] --> [-1,
497]
.... [-2000,0] --> [32684,32767]       [-1000,0] --> [-7860,637]       [0,2000] --> [7273,23772]       [-1000,0] --> [-497,1]  [0,1000] --> [-14,484]  [0,1000] --> [-1,
497]
.... [-3000,0] --> [23226,32767]       [-1000,0] --> [-7859,641]       [0,2000] --> [7269,23768]       [-1000,0] --> [-497,1]  [0,1000] --> [-14,484]  [0,1000] --> [-1,497]
.... [-4000,0] --> [13724,32767]       [-1000,0] --> [-7860,637]       [0,2000] --> [7267,23766]       [-1000,0] --> [-498,1]  [0,1000] --> [-14,484]  [0,1000] --> [-1,497]
....     XAccel                YAccel                    ZAccel               XGyro                  YGyro                ZGyro
 [-5000,0] --> [4223,32767]     [-1000,0] --> [-7858,638]       [0,2000] --> [7264,23762]       [-1000,0] --> [-498,1]  [0,1000] --> [-14,484]  [0,1000] --> [-1,497]
.... [-6000,0] --> [-5277,32767]       [-1000,0] --> [-7858,642]       [0,2000] --> [7267,23764]       [-1000,0] --> [-498,1]  [0,1000] --> [-14,484]  [0,1000] --> [-1,497]

closing in:
..       XAccel                YAccel                    ZAccel               XGyro                  YGyro                ZGyro
 [-6000,-3000] --> [-5277,23223]       [-500,0] --> [-3608,642]        [1000,2000] --> [15514,23764]   [-500,0] --> [-248,1]   [0,500] --> [-14,235]   [0,500] --> [-1,248]
.. [-6000,-4500] --> [-5277,8971]      [-250,0] --> [-1487,642]        [1000,1500] --> [15514,19641]   [-250,0] --> [-123,1]   [0,250] --> [-14,110]   [0,250] --> [-1,123]
.. [-6000,-5250] --> [-5277,1847]      [-125,0] --> [-428,642] [1000,1250] --> [15514,17578]   [-125,0] --> [-60,1]    [0,125] --> [-14,47]    [0,125] --> [-1,60]
.. [-5625,-5250] --> [-1723,1847]      [-125,-62] --> [-428,113]       [1000,1125] --> [15514,16533]   [-62,0] --> [-29,1]     [0,62] --> [-14,16]     [0,62] --> [-1,29]
.. [-5625,-5437] --> [-1723,60] [-93,-62] --> [-156,113]       [1062,1125] --> [16027,16533]   [-31,0] --> [-13,1]     [31,62] --> [0,16]      [0,31] --> [-1,13]
..       XAccel                YAccel                    ZAccel               XGyro                  YGyro                ZGyro
 [-5531,-5437] --> [-836,60]    [-77,-62] --> [-18,113] [1093,1125] --> [16269,16533]  [-15,0] --> [-5,1]      [31,46] --> [0,8]       [0,15] --> [-1,5]
.. [-5484,-5437] --> [-378,60]  [-77,-69] --> [-18,43]  [1093,1109] --> [16269,16398]  [-7,0] --> [-1,1]       [31,38] --> [0,4]       [0,7] --> [-1,1]
.. [-5460,-5437] --> [-151,60]  [-77,-73] --> [-18,14]  [1101,1109] --> [16336,16398]  [-3,0] --> [0,1]        [31,34] --> [0,2]       [3,7] --> [0,1]
.. [-5448,-5437] --> [-34,60]   [-75,-73] --> [-7,14]   [1105,1109] --> [16367,16398]  [-3,-1] --> [0,1]       [31,32] --> [0,1]       [5,7] --> [0,1]
.. [-5448,-5442] --> [-34,20]   [-75,-74] --> [-7,12]   [1107,1109] --> [16384,16398]  [-2,-1] --> [0,1]       [31,32] --> [0,1]       [6,7] --> [0,1]
..       XAccel                YAccel                    ZAccel               XGyro                  YGyro                ZGyro
 [-5445,-5442] --> [-15,20]     [-75,-74] --> [-4,12]   [1107,1108] --> [16384,16400]  [-2,-1] --> [0,1]       [31,32] --> [0,1]       [6,6] --> [0,1]
averaging 10000 readings each time
.................... [-5443,-5442] --> [0,20]   [-75,-74] --> [-3,12]   [1107,1108] --> [16381,16400]   [-2,-1] --> [0,1]       [31,32] --> [0,1]       [6,6] --> [0,1]
.................... [-5443,-5442] --> [-2,20]  [-75,-74] --> [-4,12]   [1107,1108] --> [16379,16400]   [-2,-1] --> [0,1]       [31,32] --> [0,1]       [6,6] --> [0,1]
-------------- done --------------
```

Fig. 34 Calibration results (rectangle at the bottom of the picture) obtained for MPU6050 module after running the IMU_Zero routine on the Arduino Nano board

According to the results above, the uncalibrated gyro has an error less than 0.1% relative to the output range. At first thought, this sensor can be used without calibration, but the 0.1% error is a gain type error, which means that after each iteration the sensor will add, for example 0.12 dps to the actual value x axis values, thus increasing the error in time.

3.2.2 MPU6050 IMU and GP2Y0D02YK0F IR combination results

According to Fig. 35, one can observe that the deviation of the calculated projection from the actual distance is relatively small. The accurateness of calculations is determined by the MPU6050 and the PSD sensors output, which motivates the elaborate description and thorough approach to their proper function.

Calculated values fluctuate around the true value, but the relative error does not exceed 5%. Such measurements are sufficiently accurate, because changes lower than 2.067 cm won't be reflected in the actuator's input voltage.

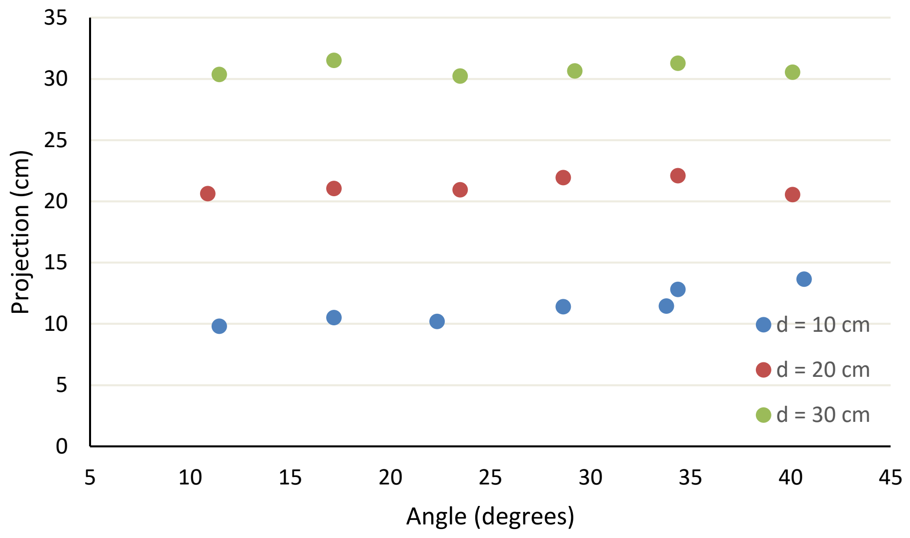


Fig. 35 The graph depicts the projection values relative to the horizontal plane for a set of angles and a fixed sensor – obstacle distance. The measurements were performed over 3 distance values.

The value 2.067 cm is determined by the PSD sensor output range [6 , 80] cm and the actuator input range [1, 4] volts. The onboard ADC detects a change of approximately 0.0196 V, which corresponds to 153 representable units. Considering the given distance output range, 1 unit corresponds to a change of 2.067 cm. When the sensor will output $y(n) \geq y(n-1) + 2.067\ cm$, the output voltage will be $v(n) \geq y(n-1) + 0.0196\ V$, where n is the number of iterations.

Conclusion

The current study aims to prove the applicability of the created prototype as an aid for people with visual impairment. The device does not require any user intervention, except for power control. The circuit works autonomously, providing a good sensory approximation of the user's location against obstacles.

As seen in the previous chapter, MPU6050 outputs stable and reliable results that provide an accurate description of the system's orientation. IR GP2Y0D02YK0F performs well over shorter distances but has faulty results close to the upper limit of its working range, due to low sensibility, which makes it unable to distinguish among a range of distances. Because HC-SR04 outputs accurate measurements, this sensor is mounted on the frontal side of the head to offer precise information. Angles measured with MPU6050 are combined with position sensor data and converted to haptic information, accessible to a blind person.

The only challenge that a blind person can encounter is understanding sensory information. But by gradual exercise, one will master the particularities of the technology and adapt to its operation.

Bibliography

[1] W. H. Organization, "World Report on Vision," 2019.

[2] United Nations, Department of Economic and Social Affairs, Population Devision, "World Population Prospects 2022," Online Edition, 2022.

[3] Shivani Naipal, Nishanee Rampersad, "A review of visual impairment," *African Vision and Eye Health,* vol. 77, no. 1, 2018.

[4] W. Elmannai, K. Elleithy, "Sensor-Based Assistive Devices for Visually-Impaired People: Current Status, Challenges, and Future Directions," *Sensors,* vol. 17, no. 3, p. 565, 2017.

[5] Keswani, Krunal Thakre, Mayank Sharma, Mansing Dhirbassi, Nayan Mahilawar, Pratik Satpute, Praveg Meshram, "Microcontroller Based Smart Stick for Blind Person," *International Journal of Innovations in Engineering and Science,* vol. 4, no. 7, 2019.

[6] A. A. Nada, M. A. Fakhr and A. F. Seddik, "Assistive infrared sensor based smart stick for blind people," in *Science and Information Conference*, London, 2015.

[7] "I. Papagianopoulos, G. De Mey, A. Kos, B. Wiecek, V. Chatziathasiou," *Sensors,* vol. 23, no. 16, p. 7198, 2023.

[8] Y. Bouteraa, "Design and Development of a Wearable Assistive Device Integrating a Fuzzy Decision Support System for Blind and Visually Impaired People," *Micromachines,* vol. 12, no. 9, p. 1082, 2021.

[9] Tachiquin R, Velázquez R, Del-Valle-Soto C, Gutiérrez CA, Carrasco M, De Fazio R, Trujillo-León A, Visconti P, Vidal-Verdú F, "Wearable Urban Mobility Assistive Device for Visually Impaired Pedestrians Using a Smartphone and a Tactile-Foot Interface," *Sensors,* vol. 21, no. 16, p. 5274, 2021.

[10] Arduino, "Arduino Nano Pinout Reference," [Online]. Available: https://docs.arduino.cc/resources/pinouts/A000005-full-pinout.pdf. [Accessed 11/5/2024].

[11] "What is Arduino," [Online]. Available: https://learn.sparkfun.com/tutorials/what-is-an-arduino/all. [Accessed 11/5/2024].

[12] Arduino, "Arduino Nano Product Reference Manual, A000005 datasheet," 06/10/2023.

[13] "An intermediate guide for low-level concepts of AVR Microcontroller," [Online]. Available: https://technicodes.weebly.com/avrarch.html#:~:text=AVR%20is%20a%208%2Dbit,has%20on%2Dchip%20flash%20storage.. [Accessed 11 5 2024].

[14] P. Horowitz, 'The Art of Electronics', United Kigdom, Cambridge University Press, 2021.

[15] S. Chatterjee, "What are the Key Advantages and Disadvantages of the Arduino Programming Language?," 26/02/2024. [Online]. Available: https://emeritus.org/blog/coding-arduino-programming-language/. [Accessed 12/5/2024].

[16] PlatformIO, "Your Gateway to Embedded Software Devlopment Excellence," [Online]. Available: https://platformio.org/. [Accessed 12/5/2024].

[17] "Why use PlatformIO instead of Arduino IDE," [Online]. Available: https://simplyexplained.com/courses/programming-esp32-with-arduino/platformio-vs-arduino-ide/#:~:text=PlatformIO%20is%20an%20alternative%20to,Nordic%2C%20STM32%20...)https://www.arduino.cc/reference/en/. [Accessed 12/5/2024].

[18] Dawoud, D. S., Dawoud, P., Serial Communication Protocols and Standards, Germany: River Publishers, 2022.

[19] Noergaard, T., Embedded Systems Architecture: A Comprehensive Guide for Engineers and Programmers, Netherlands: Elsevier Science, 2012.

[20] Barrett, S. F., Pack, D. J., Microcontrollers Fundamentals for Engineers and Scientists, Poland: Springer International Publishing, 2022.

[21] Valdez J., Becker J., "Texas Instruments, 'Understanding the I2C Bus', SLVA704 application note," June 2015.

[22] C. Tan, "UART and Arduino," [Online]. Available: https://littlebirdelectronics.com.au/guides/147/uart-and-arduino. [Accessed 24/11/2023].

[23] Subir Maity, Sohini Chatterjee, Sona Gava, Shreya Sharan,Sonam Sinha, "Synthesis & FPGA Implementation of UART IP Soft Core.," *International Journal for Scientific Research and Development ,* vol. 1, no. 9, pp. 1992-1994, 2014).

[24] Texas Instruments, "KeyStone Architecture Universal Asynchronous Receiver/Transmitter (UART)," November 2010.

[25] E. Pena, M. G. Legaspi, "UART: A Hardware Communication Protocol Understanding Universal Asynchronous Receiver/Transmitter," *AnalogDialogue,* vol. 54, no. 4, 2020.

[26] P. Forstner, "Texas Instruments, 'FIFO Architecture, Functions, and Applications', SCAA042A application report," November 1999.

[27] C. Yang, "Moxa Incorporation, 'The Secrets of UART FIFO', technical note," 01/10/2009.

[28] Nanjing Qinheng Microelectronics, "Nanjing Qinheng Microelectronics, 'USB to Serial Port Chip CH340', datasheet".

[29] Sharp Corporation, "Tokuji Hayakawa, Sharp Founder," [Online]. Available: https://global.sharp/corporate/info/history/voice/. [Accessed 07/11/2023].

[30] Sharp Corporation, "Products Lineup," [Online]. Available: https://global.sharp/products/device/lineup/selection/index.html. [Accessed 07/11/2023].

[31] Sharp Corporation, "Sharp, 'GP2Y0A Series / GP2Y0D Series Application Note', OP14030EN application note".

[32] "Photodiode," [Online]. Available: https://en.wikipedia.org/wiki/Photodiode, [Accessed 07/11/2023].

[33] "PIN diode," [Online]. Available: https://en.wikipedia.org/wiki/PIN_diode, [Accessed 07/11/2023].

[34] R. Minasian, in *Encyclopedia of Modern Optics*, Sydney, Elsevier, 2005.

[35] T. Bräunl, Embedded Robotics, Germany: Springer, 2002.

[36] "Sensors - Sharp IR Range Finder," [Online]. Available: https://www.societyofrobots.com/sensors_sharpirrange.shtml#:~:text=Additional%20to%20this%2C%20the%20Sharp,of%20the%20object%20being%20detected.. [Accessed 5/12/2023].

[37] Sharp, "'GP2Y0A21YK0F Distance Measuring Sensor Unit', E4-A00201EN datasheet".

[38] Cytron Technologies, "'HC-SR04 Ultrasonic Sensor' - Product User's Manual".

[39] E. J. Morgan, "HC-SR Ultrasonic Sensor," 16.11.2024.

[40] "Speed of sound," [Online]. Available: https://www.sfu.ca/sonic-studio-webdav/handbook/Speed__Of_Sound.html#:~:text=At%200%C2%B0%20Centigrade%20and,sec%20or%20344%20m%2Fsec.. [Accessed 2/5/2024].

[41] N. Agnihotri, "Engeineers Garage: An EE World Online Resource," [Online]. Available: https://www.engineersgarage.com/what-are-inertial-sensors/. [Accessed 12/04/2024].

[42] "Bulk micromachining," [Online]. Available: https://en.wikipedia.org/wiki/Bulk_micromachining. [Accessed 13/4/2024].

[43] S. Beeby, G. Ensell, M. Kraft and N. White, MEMS Mechanical Sensors, Artech, 2004.

[44] "MEMS," [Online]. Available: https://en.wikipedia.org/wiki/MEMS. [Accessed 13/4/2024].

[45] "Invensense," [Online]. Available: https://en.wikipedia.org/wiki/InvenSense. [Accessed 13/4/2024].

[46] InvenSense Incorporation, "'MPU-6000 and MPU-6050 Product Specification Revision 3.4', PS-MPU-6000A-00 datasheet," 19/08/2013.

[47] P. Septimiu, Instrumentation and Sensor Systems Courses, Cluj-Nnapoca.

[48] "What is MEMS? Accelerometer, Gyroscope, and Mangetometer with Arduino," [Online]. Available: https://howtomechatronics.com/how-it-works/electrical-engineering/mems-accelerometer-gyrocope-magnetometer-arduino/. [Accessed 17/4/2024].

[49] Clark, W. A., R. T. Howe, and R. Horowitz, "“Surface Micromachined Z-Axis Vibratory," *Digest of Solid-State Sensors and Actuator Workshop,* p. 283–287, 1996.

[50] SparkFun, "Analog to Digital Conversion," [Online]. Available: https://www.sparkfun.com/?_ga=2.46594849.741691677.1713687452-105205211.1710742986. [Accessed 21/4/2024].

[51] U. Papa, Embedded Platforms for UAS Landing Path and Obstacle Detection, Germany: Springer International Publishing, 2018.

[52] "SharpDistSensor," [Online]. Available: https://github.com/DrGFreeman/SharpDistSensor. [Accessed 21/4/2024].

[53] Z. Yaniv, "'Median Filtering' summary," [Online]. Available: https://yanivresearch.info/writtenMaterial/median.pdf. [Accessed 2024/4/22].

[54] Arduino, "Wire," 05/12/2023. [Online]. Available: https://www.arduino.cc/reference/en/language/functions/communication/wire/. [Accessed 4/5/2024].

[55] "I2Cdevlib," [Online]. Available: https://www.i2cdevlib.com/. [Accessed 4/5/2024].

[56] "I2C Address List," [Online]. Available: https://i2cdevices.org/addresses. [Accessed 4/5/2024].

[57] N. Kularatna, Modern Component Families and Circuit Block Design, Netherlands: Elsevier Science, 2000.

[58] "Register Map," [Online]. Available: https://www.i2cdevlib.com/devices/mpu6050#registers. [Accessed 6/5/2024].

[59] "IMU_Zero.ino," [Online]. Available: https://github.com/jrowberg/i2cdevlib/blob/master/Arduino/MPU6050/examples/IMU_Zero/IMU_Zero.ino. [Accessed 6/5/2024].

[60] A. Visioli, Practical PID control, London: Springer, 2006.

[61] "Carbon Aeronautics Quadcopter Manual," 22/4/2023. [Online]. Available: https://github.com/CarbonAeronautics/Manual-Quadcopter-Drone/blob/main/Carbon_Aeronautics_Quadcopter_Manual.pdf. [Accessed 7 5 2024].

[62] I. Beavers, "The case of the Misguided Gyro," Analog Devices, 3 2017. [Online]. Available: https://www.analog.com/en/resources/analog-dialogue/raqs/raq-issue-139.html. [Accessed 7/5/2024].

[63] Oğuz, A. E., & Aydemir, M. E., "A Practical Implementation of a Low-Cost 6-DOF IMU by Kalman Algorithm," *European Journal of Science and Technology,* no. 32, pp. 167-170, 2021.

[64] "Aircraft Principal Axes," [Online]. Available: https://en.wikipedia.org/wiki/Aircraft_principal_axes. [Accessed 7/5/2024].

[65] J. Vince, Mathematics for Computer Graphics, London: Springer, 2010.

[66] MPU6050_DMP6. [Online]. Available: https://github.com/ElectronicCats/mpu6050/blob/master/examples/MPU6050_DMP6/MPU6050_DMP6.ino. [Accessed 7/5/2024].

[67] B. Jung, A. C. Black and B. S. Bhutta., "Anatomy, Head and Neck, Neck Movements," 9/11/2023. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK557555/#:~:text=Cervical%20flexion%3A%20bending%20the%20head,ear%20to%20the%20ipsilateral%20shoulder.. [Accessed 8 5 2024].

[68] S. Monk, Programming Arduino: Getting Started with Sketches, New York: McGraw-Hill Education, 2012.

[69] InvenSense, "'MPU-6000 and MPU-6050 Register Map and Descriptions Revision 4.2', RM-MPU-6000A-00 register map," 2013.

[70] "Serial," [Online]. Available: https://docs.particle.io/reference/device-os/api/rgb/mirrordisable/. [Accessed 5/12/2023].

Anexa 2

DECLARAȚIE PE PROPRIE RĂSPUNDERE

Subsemnatul, VLAICO ECATERINA, declar că Lucrarea de licență pe care o voi prezenta în cadrul examenului de finalizare a studiilor la Facultatea de FIZICĂ, din cadrul Universității Babeș-Bolyai, în sesiunea 2024 , sub îndrumarea Conf. dr. IOAN BURDA, reprezintă o operă personală. Menționez că nu am plagiat o altă lucrare publicată, prezentată public sau un fișier postat pe Internet. Pentru realizarea lucrării am folosit exclusiv bibliografia prezentată și nu am ascuns nici o altă sursă bibliografică sau fișier electronic pe care să le fi folosit la redactarea lucrării.

Prezenta declarație este parte a lucrării și se anexează la aceasta.

Data VLAICO ECATERINA

17.06.2024s [illegible]