Design and implementation of an information system to study the repeatability of the orientation in space for medical applications

M. CEBULA[1], P. MURAWSKI[2]

magdalena.oliwia.cebula@gmail.com

[1]Military University of Technology, Faculty of Cybernetics
Institute of Computer and Information Systems
Kaliskiego Str. 2, 00-908 Warsaw, Poland

[2]Military Institute of Medicine, Szaserów Str. 128, 04-141 Warszawa, Poland

The aim of the study was to study the repeatability of the object's orientation in space. During the study, a self-constructed system of 4 gyroscope and an available robot of the Kuka type were used. An application allowing for comparison of research results was written and presented. The article contains individual stages of work (analytical and implementation), research results, conclusions and discussion.

Keywords: orientation in space, gyroscope, medical applications.

DOI: 10.5604/01.3001.0014.4434

1. Introduction

Currently, almost every electronic device has or may have a gyroscope. It is used to determine the orientation of objects in space. It is used, among others, in aircraft, virtual reality technology or medical devices. With the use of its measurements it is possible to examine the dynamics of changes in the patient's position, position or direction of movement of the patient and, in special cases, his gaze. Another aspect is the monitoring of physical activity of rehabilitated patients – the use of information from accelerometers and gyros located throughout the body allows to track and record the parameters of movement and ultimately activity.

However, individual devices may differ and the parameters declared in the documentation do not have to be the same depending on the model, system price or manufacturer.

The natural question is, therefore, whether this availability can be translated directly into telemedicine applications, and if so, can it be translated into everyone? Thus, it is a question of indication consistency, measurement accuracy and repeatability. It was decided to verify this fact by experimentally building a system for conducting such tests.

1.1. Robot

Robots, as a machines, are designed primarily to perform manipulation and locomotion tasks characterized by high accuracy and repeatability of movements. A Kuka-type robot, used in engineering work, performs manipulative tasks (i.e. imitating some functions of a human upper limb) – so one can say that it is a manipulator. For the purposes of the work, however, the concept of the Manipulator can be extended to the form of a technical device imitating the movements of a limb, in particular the hand [1].

1.2. Orientation in space

Orientation and position are terms related to the position of one object relative to another, in this case the arms of the manipulator. Taking the system of coordinates it is possible to determine the position of any point using vectors – the so-called straight kinematics [2]. In addition to the global coordinate system, local coordinate systems are also distinguished. Therefore, vectors with which the system can be described should contain information about the coordinate system they describe. While the position is responsible for the direct position of one part of the manipulator in relation to another, the orientation in space determines the angle at which one part of the manipulator is oriented in relation to another. Ultimately, the definition of the orientation is therefore essential for a full knowledge of the robot's position, since the position of a point in space alone does not provide full information about the position of the arms. In order to describe the orientation, the robot-related coordinate system is used, whose position is described in relation to the reference system. A rotation and translation matrix is used to describe the orientation in a mathematical form. In this case, the total movement of the object (rotations and shifts) can be described as the operation of multiplying the matrices appropriate to the elements of movement. Rotation matrix – a matrix describing rotation in Euclidean space (Figure 1). Rotation in n-dimensional Euclidean space is described by a square array of dimensions n x n (1) [2].

$$\begin{gathered} n = nxi + nyi + nzk \\ o = oxi + oyj + ozk \\ a = axi + ayj + azk \end{gathered} \qquad (1)$$

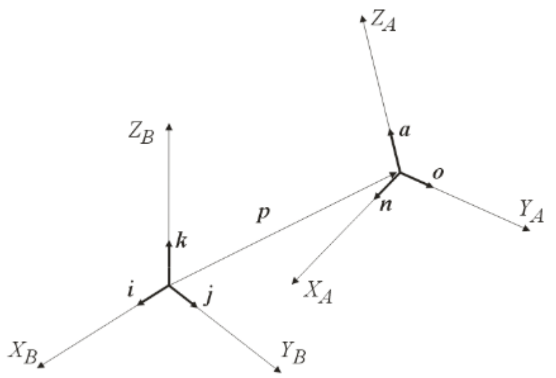

Fig. 1. Rotation matrix and orientation of system A axis in relation to system B

Orientation is usually defined in the form of Euler angles, called P – pitch, Y – deflection, R – rotation around the axes related to the object (2). Representation in the form of Euler angles is the minimum representation. However, this representation is ambiguous since there are 12 different sets of Euler angles corresponding to possible elementary rotation sequences around the axis of the current system and 12 around the axis of the established system [3].

$$R_A^B = R_{AZ_B}(\varphi) R_{AY_B}(\vartheta) R_{AX_B}(\psi) =$$

$$= \begin{bmatrix} cos\varphi & -sin\varphi & 0 \\ sin\varphi & cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\vartheta & 0 & sin\vartheta \\ 0 & 1 & 0 \\ -sin\vartheta & 0 & cos\vartheta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\psi & -sin\psi \\ 0 & sin\psi & cos\psi \end{bmatrix}$$

$$= \begin{bmatrix} cos\varphi cos\vartheta & cos\varphi sin\vartheta sin\psi - sin\varphi cos\psi & cos\varphi sin\vartheta cos\psi + sin\varphi sin\psi \\ sin\varphi cos\vartheta & sin\varphi sin\vartheta sin\psi - cos\varphi cos\psi & sin\varphi sin\vartheta cos\psi - cos\varphi sin\psi \\ -sin\vartheta & cos\vartheta sin\psi & cos\vartheta cos\psi \end{bmatrix} \qquad (2)$$

The point associated with the object to which the coordinate system is associated is selected as the beginning of the system [4].

The study of movement without taking into account the forces that cause this movement is dealt with by a science called kinematics. It examines the changes in position, speed and acceleration of each part of the robot, and at the same time it examines all points that lie on a given part. The concept of degrees of freedom is used when studying the robot/manipulator movements.

The robot used in engineering work has 6 degrees of freedom:

- three associated with displacement;
- three associated with rotation.

1.3. Methods of measuring orientation in space

The orientation in space can be measured with a device called a gyroscope. The gyroscope uses the principle of momentum conservation for its operation (the vector sum of momentum of all elements of the isolated system remains constant [5]). The gyroscope can be constructed in the form of a disc which, when introduced into a fast rotary motion, will maintain its initial axis of rotation.

Types of gyroscope:

- Directional gyroscope – keeps the direction of rotation constant so that it is possible to observe the rotation of the body to which they are attached;
- Speed gyroscope – indicates the angular speed of the object to which they are attached.

Gyroscopes belonging to the speed gyros group:

- mechanical gyroscope – has limited freedom of rotation;
- optical gyroscope (including laser gyroscope) – recognizes a change in the position of an object and uses the phenomenon of changing the frequency of light waves;
- Coriolis effect gyroscope – A gyroscope using the Coriolis effect – interacts on shifting elements;
- electronic gyroscope – has no traditional moving elements. It is built of accelerometers.

The rotational speed is converted from the indications of accelerometers.

Most often, speed gyroscope is made in MEMS technology.

1.4. MEMS technology

This technology is a technology for the design and manufacture of systems originally mechanical in the form of microstructures of which at least one dimension of the device should be on the microscale (0.1–100 μm), usually made in silicon or glass.

MEMS gyros – they are based on the Coriolis effect. A mass placed on a rotating platform is used here, which is made to vibrate. This mass is attached to a silicon frame so that it only moves in one direction. When the mass moves towards the outer edge of the disc, it is accelerated. According to the principle of action and reaction to the frame, a force is applied to the frame, which is directed in the opposite direction. When the mass moves towards the center of the circle, the force acting on the frame is directed to the right and the mass is accelerated to the left. The value of this force is proportional to the body speed. Devices built on the basis of this principle are used in the work.

2. Materials and Methods

The work began with the development of the system architecture they are part of:

- Software architecture (package diagram and class diagram);
- Communication architecture (messages sent between applications and the protocol used – UDP);
- Measurement system architecture;
- Workplace architecture.

Then developed functional requirements, use cases, state diagram and data model.

The final stage was the implementation of the application.

Individual elements are described below.

2.1. System architecture

The architecture of the system architecture used in the work includes:

- an application that images and compares the data received;
- a system designed and constructed to verify readings from gyroscopes;
- a stand consisting of a "Kuka" type robot for precise position determination.

2.1.1. Software architecture

The described task was part of the engineering work, the result was to design and ultimately produce applications using different programming environments. The application depicting and comparing the received data was written in C# language. The software for the microprocessor measurement module was made using C language, while the main control module of the robot, the communication module which was made available by the co-author was made in Embarcadero Rad Studio Architect (Delphi) environment.

2.1.1.1. Package diagram

At the development stage, the software was divided into logical packages so that they correspond to the expected functionalities (Figure 2).

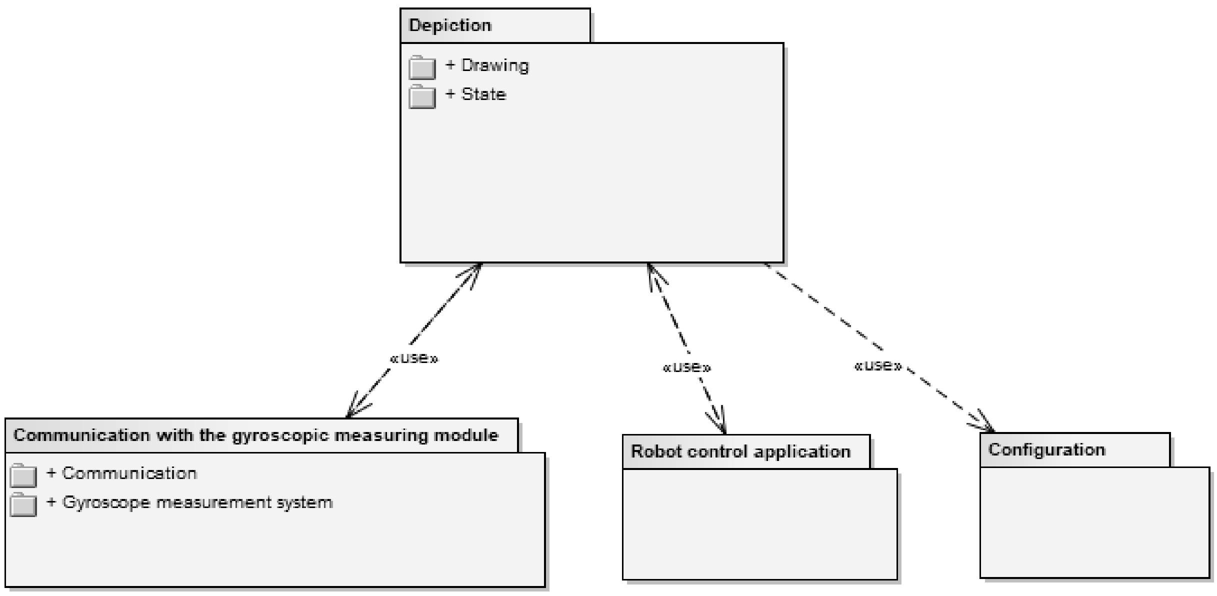

Fig. 2. Package diagram

The following packages were defined in the engineering work:

- data module – responsible for the way data is stored in the system;
- data processing module – responsible for the way data is processed in the system;
- Imaging module – responsible for the way data is imaged in the application;
- Communication module – was responsible for the manner of communication of system components.

2.1.1.2. System components

Regardless of the functionalities defined for the packets, the necessary data to be collected was analyzed in the components presented on class diagram (Figure 3).

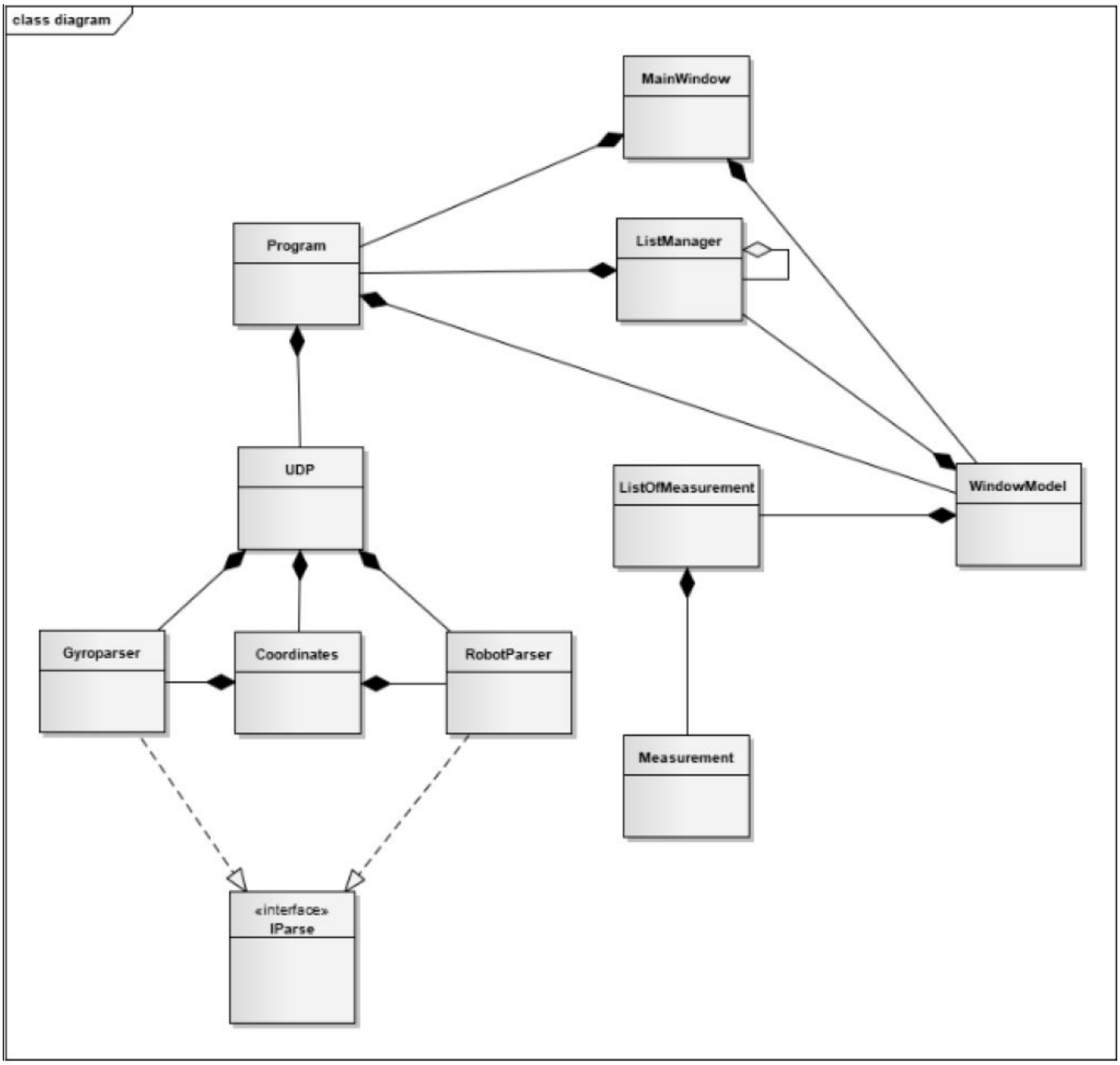

Fig. 3. Class diagram

2.1.2. Communication

The data between the robot, the program and the gyroscope system were sent as messages. This message was a sequence of established structure characters.

The structure of the message sent from the gyroscope system to the program has been defined in the form of:
{"Date":"…","GyroMeasurement":[{"x":126.56, "y":-2.0,"z":-1.81}, {"x":166.86,"y":-5.22,"z":0.56}]}

The message was sent:

- Date;
- An hour;
- X, Y, Z coordinates from two gyros.

The message was implemented using the JSON format. The structure of the message sent from the robot to the program is defined as (example): ReadSensor (X:-90 Y:90 Z:-10).
The message sent the expected X, Y, Z coordinates, with which the robot's arm should reach. These values corresponded conceptually to the robot arm's attitude.

2.1.2.1. UDP protocol

The robot's control system, the measuring system from the gyroscope system, were connected to the same computer network, understood as a fixed-address subnet. These elements used the UDP protocol to communicate with each other. This corresponded to communication in the transport layer of the OSI model. Since in the UDP protocol, the mechanism implemented by means of which endpoints can be identified (IP address and port) is used to identify the sides of communication.

2.1.3. Measurement system architecture

The architecture of the measuring system illustrates how its individual components are connected to each other. In this case, they were gyroscope, microprocessor chip, display and real-time clock. The components were communicating with each other using the I2C bus (Figure 4). When designing the measurement system, care was taken to ensure that each component had a different address on the I2C bus so that there was no address conflict.

2.1.3.1. Components

Eventually, after many attempts to select and eliminate the components, the measuring system consisted of components:

- Moduł Huzzah ESP 8266;
- Moduł Bosh BNO 055;
- Moduł LSM9DS0;
- Moduł LSM0DS1;
- SparkFun MPU-9250;
- Adafruit FeatherWing DS3231 Precision RTC;
- 182 x 32 Monochrome OLED FeatherWing.

2.1.3.2. Measuring module

The measuring module is designed on a prototype plate (Figure 5) ensuring, as far as possible, the alignment of axis directions for all gyros. Possible differences in axis returns were corrected by software.

A plate was also designed for the measuring system to attach the gyroscope system to the robot's arm (Figure 6).
The board model was made using SolidWorks software and printed with a Zortrax 3D printer at the authors' disposal.

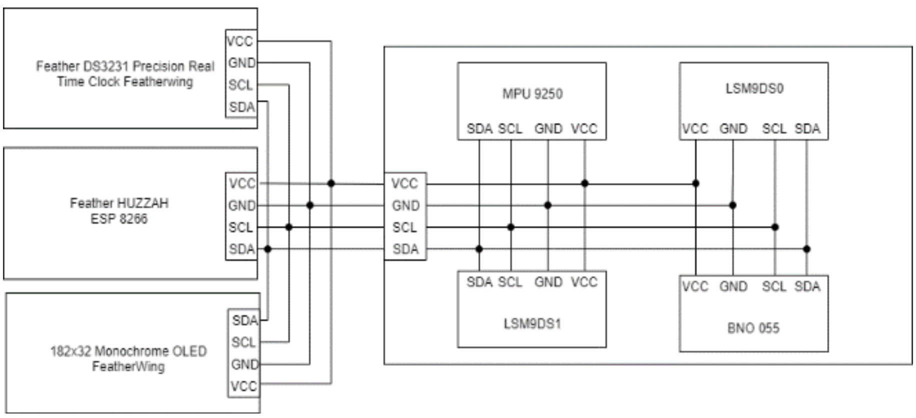

Fig. 4. Outline diagram of communication connections in the system

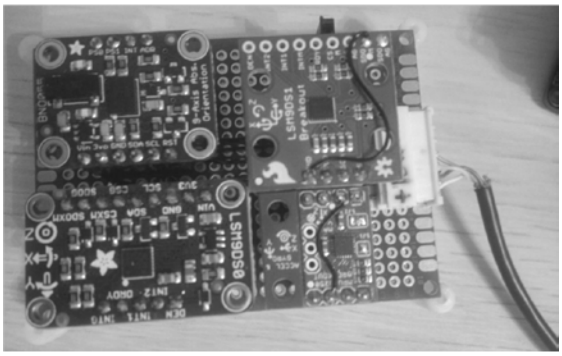

Fig. 5. Made gyroscope system with selected gyros

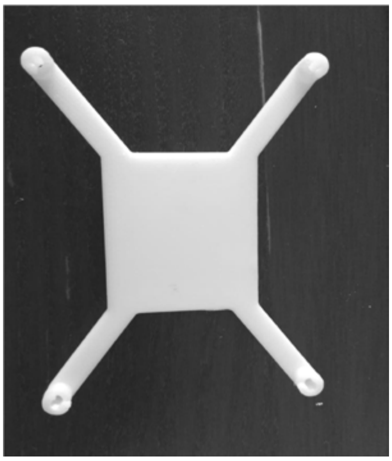

Fig. 6. Mounting plate

2.1.3.3. Magistrala I2C

It is a serial, bi-directional bus that is used to transmit data in electrical devices. [6]
In addition to power lines, it consists of:

- Serial data lines (SDA);
- The serial clock line (SCL).

They are transmitted using them:

- Data;
- Addresses;
- Control signals.

When the level on both lines is high, no transmission is carried out.

Each system using the I2C bus has its own address, usually saved by its manufacturer in the manufacturing process. During transmission, the systems can act as both master and slave. Master is a system that initiates transmissions and generates clock signal. The possibility of accepting the role is determined by the use of the device, e.g. the FRAM will not accept the role of the master regardless of the possibility [8], [9].

2.1.3.4. Component's addressing

The devices used in the measuring system communicated via the I2C bus. Figure 7 shows the effect of the auxiliary software of the so-called I2C bus scanner presenting the addresses of individual devices used in the measurement system.
To check the I2C address, the program was used: https://gist.github.com/tfeldmann/5411375.
These addresses correspond to the following devices:

- 0 x 1D – LSMDS0;
- 0 x 28 – BNO 055;
- 0 x 3C – 182x32 Monochrome OLED FeatherWing;
- 0 x 68 – Adafruit FeatherWing DS3231 Precision RTC;
- 0 x 6B – LSM9DS1.

2.1.4. Architecture of workstation

While working on the application, a Kuka-type robot (Figure 8) was made available with control software written in Embarcadero Rad Studio (Delphi) environment by the co-author. This software allows for setting the position and controlling the executive module, ultimately reaching the desired arm position by the robot. It communicates with the robot using Bluetooth technology, making the control remote.

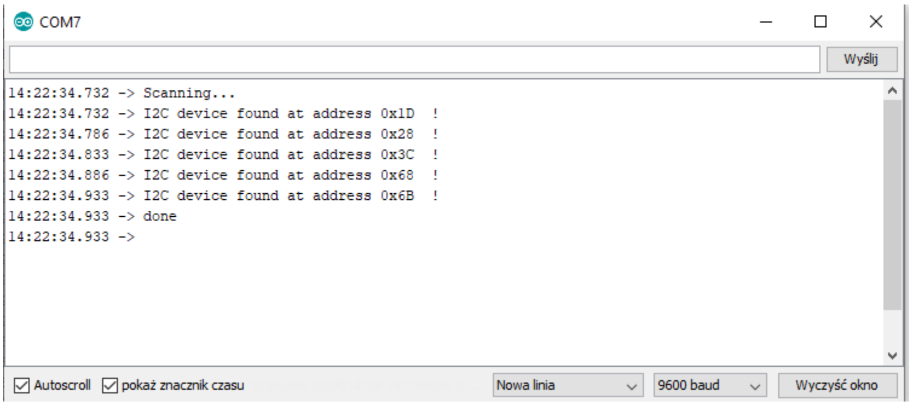

Fig. 7. I2C addresses of devices used in the measurement system

Fig. 8. Robot used in engineering work

2.2. Functional requirements

Functional requirements defined the needs expressed in a precise manner by the user, which should be met by the available system functionality. In this area they were defined as follows:

- Angle measurement – the system allows for the measurement of angles from four gyroscope, with the number of gyroscope is limited to two – measuring and reference;
- Sending data from gyros to the program – the system allows to send read data to the program. When the message “GetValue”, arrives, the gyroscope system reads the values from the gyroscope, converts them into Euler angles and sends a message in a fixed JSON format;
- Receiving data from the robot –system allows the program to receive data from the robot. Program waits for a message from the robot. If the data is received, the program sends a “Done” message to the robot;
- Collecting data from the gyro system – system allows the program to receive data from the gyro system. The program waits for the message from the robot, and as a result, it starts reading the values and sending them to the application presenting the data;
- Data comparison – system allows the program to compare the data received by the program;
- Displaying data on screen – system allows to display received data and calculation results on screen.

2.3. Use case diagram

Diagrams are a design technique that allows to present different aspects of the system. Depending on the adopted convention, they can show e.g. functionality ranges, time relations, execution sequence or data structures.

Figure 9 illustrates the cases of use supported by the system.

2.4. State diagram

A states diagram is a diagram showing the interdependence of system states. With this diagram you can:

- discover missing information;
- control whether requirements are complete;
- clarify uncertainties;
- identify conflicts of requirements and resolve them [10].

Taking into account the necessity to guarantee the correct handling of messages, a state diagram (Figure 10) was developed for the process of message acquisition and then transformed into multithreaded software.

- Received from a robot

The robot sends a message to the program using the UDP protocol;

- Received from the system

The program receives a message from the gyro system with the current position;

- Compressed

The messages received from the robot and the gyroscope system are parsed;

- Compared

The program calculates the difference between the position received from the robot and the one read from the gyros;

- Displayed on a graph

The program adds the calculated point to the graph.

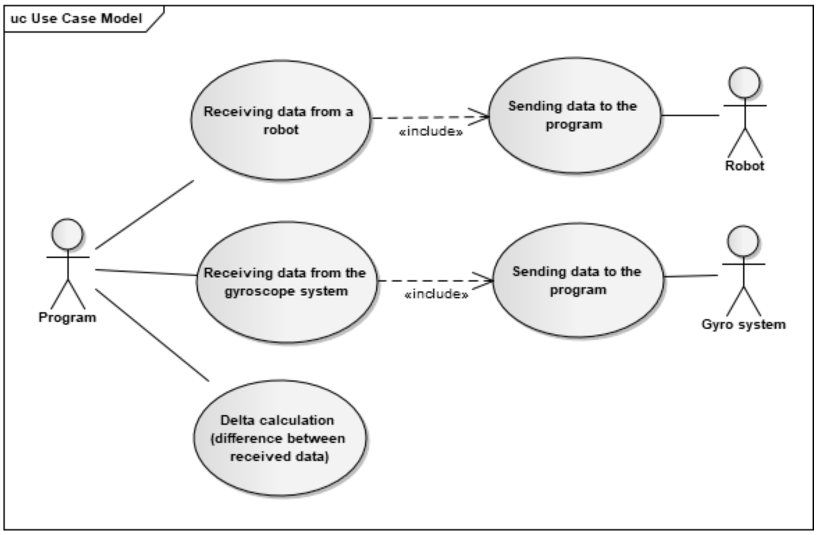

Fig. 9. Use case diagram for the program

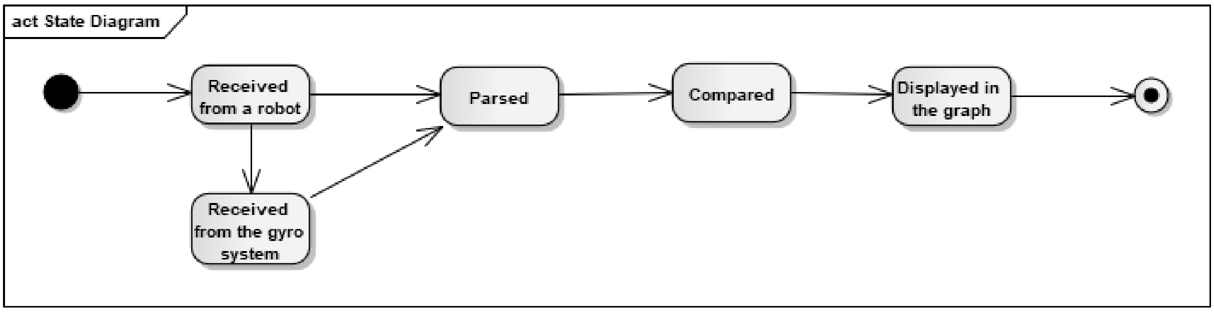

Fig. 10. State diagram for the message

2.5. Data model

The basic data structure used in the program is Coordinates class, which contains 3 properties of double X, Y, Z type. The Coordinates structure is used in the Measurement structure. It can be distinguished in it:

- list of coordinates received from gyroscopes (gyroMeasurement);
- the coordinates received from a robot (RobotMeasurement);
- a list of the coordinates obtained from the delta (DeltaMeasurement);

Measurements are aggregated in the ListOfMeasurement structure.

Figure 11 illustrates the basic data model of the program.

2.5.1. Data received from the gyroscope system

The data obtained from the gyroscope's measuring system contains information about three rotations – one for each axis. In addition, they are time-stamped, allowing for their subsequent ordering and processing in the system of dynamics of changes in time.

2.5.2. Data processing

The data obtained from the gyroscope's measuring system contains information about three rotations – one for each axis. In addition, they are time-stamped, allowing for their subsequent ordering and processing in the system of dynamics of changes in time.

2.6. Implementation

2.6.1. Transfer protocol

The mechanism using the UDP protocol for a transfer and can be divided into 3 parts:

- Client and Endpoint Initialization (so called Endpoint Initialization) UDP for communication with gyroscope system and robot in UDP class constructor;
- Data reception starts with calling the Receive() method on the corresponding Listener object with the endpoint parameter. Then, when the data are received, they are saved in a text variable (string type) and displayed on the program console. The maximum time the program waits for the data from the robot, and from the gyroscope system. If no message is

received during this time, the Receive method reports an exception, which is supported by displaying an appropriate message.

- Sending data–Before the data is sent, the text is re-coded to an array of bytes. Then the Send() method of the UDP client is called. The parameters of this method are: message, message size and endpoint.

2.6.2. Uniqueness of the objects

Object uniqueness could be realized by use of singelton. Singelton is one of many design patterns. In the program, it is used in the process of creating a ListManager class object. This ensures a single occurrence of an object of this class. It is implemented by creating a private constructor and a static method that checks if there is an object of this class. If there is no instance of an object, it is created and if it exists, a reference to this object is returned. In addition, the access method uses a mutual exclusion mechanism – because the program was executed in multithreaded technology, without this it could happen that two independent instances in two different threads of the program were created simultaneously.

These mechanisms were implemented using lock() mechanism from C# language.

2.6.3. Multithreading

Bearing in mind the need to divide the tasks and their equalised handling, the application was designed and executed as a multi-threaded one. The following threads are created in the program:

- the main thread of the program;
- the thread responsible for handling the GUI;
- the thread responsible for communication with the robot;
- the thread responsible for communicating with the gyroscope system.

The .Net Task Parallel library has been used to perform multithreaded management in the program. This library adapts to the number of available cores in the computer system. Thanks to this, it was possible to maximize code performance without focusing on work distribution between available cores.

2.6.4. Synchronisation mechanism

The program uses a lock mechanism from C#. This mechanism is the simplest mechanism to ensure proper access to resources. In case the second thread would like to perform operations in the lock area, it will be locked and will have to wait until the first thread performs all operations included in the lock clause.

In addition, the instance field declaration contains a volatile keyword, which prevents possible caching problems. This directive tells the compiler not to perform optimization on a given field. The use of lock() mechanism was presented in the work on Singleton pattern description.

2.6.5. Application model

To create an application a MVVM model has been used. In the MVVM model (Figure 12) three layers were distinguished:

- The model layer;
- The view layer;
- The model view layer;
- The layer of view:

This layer is responsible for displaying data. It should not do any business logic. For WFP technology (used in the program), the view definition is an XAML file;

- The model layer:

This layer is responsible for business logic. In the case of a program, it is responsible for receiving data, parsing it and adding it to lists;

- The ViewModel layer

This layer exposes the model for view. It contains the data and logic associated with the view as well as commands associated with the GUI. The Data binding mechanism is used here. By implementing an appropriate interface, the data in the view is refreshed automatically when it is modified on the Model side.

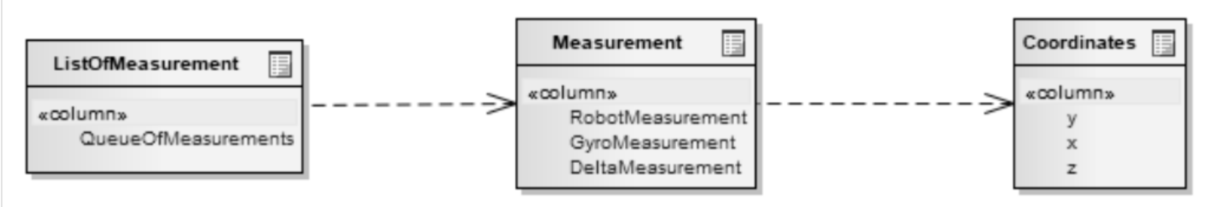

Fig. 11. Data model

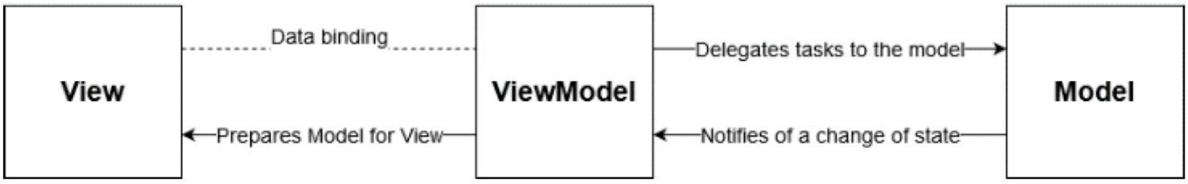

Fig. 12. Model MVVM

2.7. Software implementation for the embedded system

The sample code for the embedded system received by the co-author has been extended with elements:

- Reading data from LSM9DS0;
- Message format definition.

2.7.1. Reading data from LSM9DS0

Using the available libraries to initialize the sensor, the start() function is called, returning true if the initialization process has been successful.

Then, on the serial monitor, information about the address reported by the LSM9DS0 chip was displayed, together with the expected address.

Reading Euler angles from the BNO chip is done using the getVector() method.

In case of LSM9DS0 chip, Euler angles were calculated by defining functions (they were added to the LSM9DS0 library).

2.7.2. Message format definition

The message is send in JSON format. JSON format was a structured text. While it is possible to use high-level libraries for building and parsing a message, this may not be justified in the case of an embedded system. In such a case it should be borne in mind that the programmer does not have virtually unlimited operating memory. Therefore for building a message it was decided to use the function of concatenation of numerical values after converting them to text values, respectively for X, Y, Z while keeping the character representation in ASCII code.

3. Results

The result of the work was the creation of an electronic system allowing for the simultaneous reception of values from four gyroscope and software for the embedded system and desktop class for the imaging and preliminary analysis of the received data. Finally, the study compared the values from two gyroscope, one of which was considered as a reference, considering both its declared accuracy and measurement speed. Taking into account the fact that the directions of axis turns for some gyroses are opposite, the values obtained in the program were interpreted accordingly. Comparison of gyroscope reading results for the coordinates:

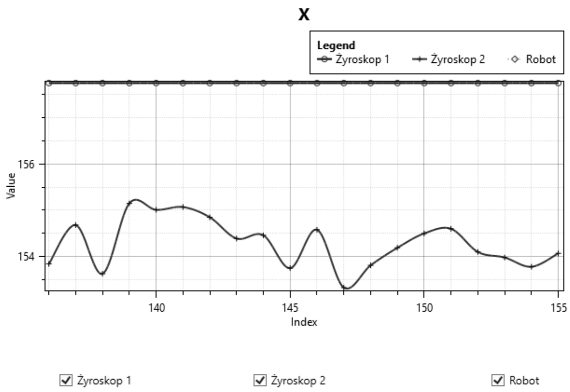

Fig. 13. Comparison of gyroscope reading results for the X coordinate

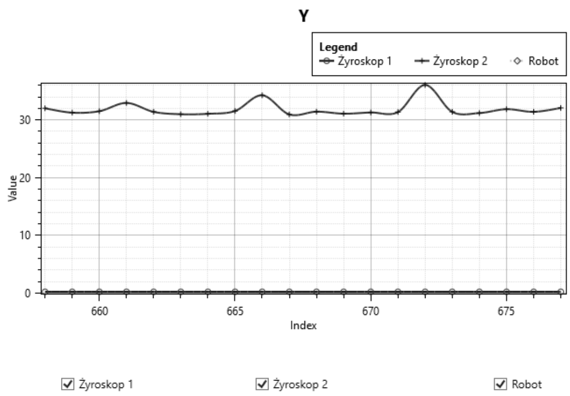

Fig. 14. Comparison of gyroscope reading results for the Y coordinate

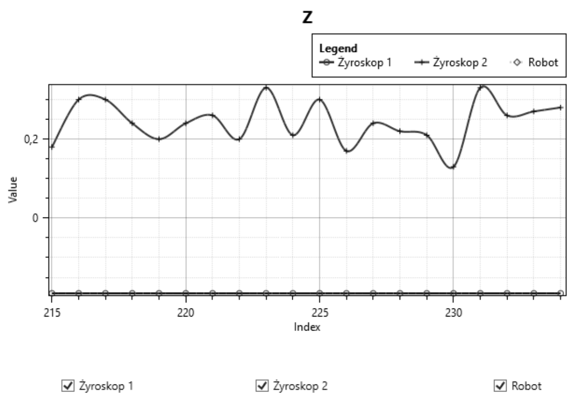

Fig. 15. Comparison of gyroscope reading results for the Z coordinate

The values obtained from the BNO gyroscope are constant. In the case of the LSM9DS0 gyroscope, the values are characterized by variability (random oscillations) which is difficult to describe and constant shifts of values, which would indicate the existence of a systematic error, e.g. caused by assembly, which could have an impact if not corrected readings in medical applications.

4. Discussion

While working on the project, there were questions and challenges to be answered.

The first challenge was to design the gyroscopic system.

During the work, it was decided to choose the system presented earlier, so that as many axes as possible would agree.

The measurements were taken for only one axis of rotation, which is directly related to the design of the robot and the cognitive character of the work. The readings that are sent to the program are burdened with error. They result from inaccuracies in the execution of the solution (e.g. the robot's stepper motor can lose its steps).

The visualised results on the diagram from the second gyroscope are not a straight line. This may be due to several reasons, including the accuracy of the system, which is significantly influenced by the fact that the system was installed “manually” and the properties of the gyroscope itself, which has worse parameters than the first gyroscope (selected as a reference).

During the work, a problem was encountered with the failure of the gyro system. After the diagnosis, it turned out that the clamps on the wires did not guarantee a certain contact and they had to be reassembled. This situation shows that the system for experimental works may have problems with the stability of solutions, resulting, among others, from imperfections in the execution of the system and the influence of the selected solutions, often resulting from limitations, including costs.

5. Conclusions

The subject of the engineering work on which this article is based is “Design and implementation of an IT system for testing the repetitiveness of determining orientation in space for medical applications”.

During the work a system consisting of 4 gyroscope was designed and built:

- BNO 055 – selected as a reference gyroscope;
- LSM9DS0 – selected as a reference gyroscope;
- LSM9DS1 – not finally used in engineering work;
- MPU 9250 – not finally used in engineering work.

The design of the system assumed the appropriate arrangement of gyros on the plate so that the gyroscope axes (X, Y, Z) would match and the risk of possible short circuits associated with soldering the system would be reduced.

Functions provided by the application:

- receiving data with UDP protocol from the gyro system (2 measurements);
- UDP data reception from the robot;
- data comparison and calculation of the difference between the gyro measurement and the data received from the robot;
- displaying the results on screen.

The analysis of the results obtained from the engineering work has shown how important the technology used in the production of the components is for the stability of their operation. The research carried out showed that choosing the BNO 055 gyroscope as a reference was a good decision. It is characterized by greater accuracy and stability as well as speed of operation.

6. Bibliography

[1] S. Wiak (red.), *Mechatronika. Tom 2*, Akademicka Oficyna Wydawnicza, Politechnika Łódzka, 2010.

[2] E. Jezierski, *Dynamika robotów*, WNT, 2006.

[3] K. Kozłowski, P. Dutkiewicz, W. Wróblewski, *Modelowanie i sterowanie robotów*, Wydawnictwo Naukowe PWN, Warszawa 2003.

[4] Encyklopedia popularna PWN, Wydawnictwo Naukowe PWN, Warszawa 2001.
[5] M. Kozielski, K. Wosińska, P. Duda, A. Staranowicz, T. Wójcik, M. Zielonka, *eFizyka – Multimedialny podręcznik do nauki fizyki dla szkół ponadgimnazjalnych*, Politechnika Warszawska.
[6] Rotation, in: *Encyclopaedia of Mathematics*, M. Hazewinkel (Ed.), Kluwer Academic Publishers, 2001.
[7] W. Mielczarek, *Szeregowe interfejsy cyfrowe*, Helion, Gliwice 1993.
[8] J. Parchański, *Miernictwo elektryczne i elektorniczne*, WSiP, 2006.
[9] *I²C-bus specification and user manual*, NXP Semiconductors N.V., 2014.
[10] *Guide to Business Analysis Body of Knowledge* (*Babok Guide*), International Institute of Business Analysis™ (IIBA®), 2015.

Projekt i implementacja systemu do badania powtarzalności określania orientacji w przestrzeni dla zastosowań medycznych

M. CEBULA, P. MURAWSKI

Celem badań wykonanych w ramach pracy było określenie powtarzalności określenia orientacji. Podczas pracy wykorzystano samodzielnie zbudowany układ żyroskopów oraz udostępniony robot typu Kuka. Napisano aplikację umożliwiającą porównanie wyników badań przez przedstawienie ich na wykresach. W artykule zawarto poszczególne etapy pracy (analityczne i implementacyjne), wyniki badań, wnioski oraz dyskusję.

Słowa kluczowe: orientacja w przestrzeni, żyroskop, aplikacje medyczne.