

MIDDLESEX PLATFORM MEASURING SYSTEM

CCE 4999 PROJECT REPORT

MENG PROJECT 2018

SHIRWA HAJI M00441398

ANCILA ULUNGI M00517972

31th May 2018

LIST OF CONTENTS

Introduction ………………………………………………………………………………………………3

Background ………………………………………………………………………………………………3

Fig 1 ……………………………………………………………………………..3

Purpose of the project …………………………………………………………………………………..4

Fig 2 ……………………………………………………………………………..4

Functional overview ……………………………………………………………………………………..5

Table 1 …………………………………………………………………………..5

Fig 3 ……………………………………………………………………………...6

Fig 4 ………………………………………………………………………………6

Design Overview …………………………………………………………………………………………7

Implementation Overview ……………………………………………………………………………..10

Fig 5 ……………………………………………………………………………..11

Fig 6 ……………………………………………………………………………..11

Fig 7 ……………………………………………………………………………..12

Fig 8 ……………………………………………………………………………..12

Fig 9 ………………………………………………………………………………13

Testing …………………………………………………………………………………………………...13

Fig 10 ……………………………………………………………………………14

Results …………………………………………………………………………………………………..14

Fig 11 ……………………………………………………………………………15

Table 2 …………………………………………………………………………..16

Fig 12 ……………………………………………………………………………17

Fig 13 ……………………………………………………………………………18

Fig 14 ……………………………………………………………………………19

Key Findings …………………………………………………………………………………………....19

Lessons learned ………………………………………………………………………………………..23

Conclusion ………………………………………………………………………………………………25

Introduction

This is a report which discusses a group project that was conducted for the past 24 weeks in this academic year, which was to create a system that measures the accuracy of a 6DOF platform. The report will discuss the main purpose of the project, how the project was implemented, the outcome of the project and the testing of the system as well as what would have been done differently for a better delivery.

Background

The project will use a 6DOF (Six degree of freedom) motion platform which is built using the concept of a robotic configuration called Stewart platform which was first used by Eric Gough in 1954 and later published by D Stewart in 1965. A Stewart platform consists of six independently controlled actuators, all mounted to a fixed platform and movable platform as shown in the figure below.

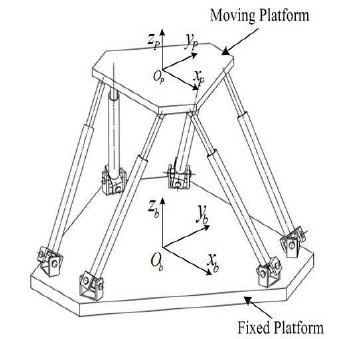
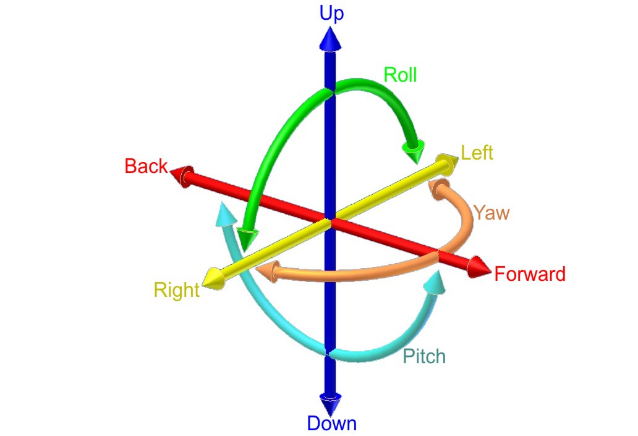


Fig 1: Steward Platform

The 6DOF platform that is to be used in this project is made up of an inverted Steward platform where the actuators are flexible and the fixed position of the actuators is at the top and then they get attached to the movable platform at the bottom.

Although the 6DOF that will be used for this project is slight different from the normal Steward platform it uses the same concept of the six degrees of freedom.

The six degrees include three linear movements x, y, z and three rotations yaw, pitch, roll.



Purpose of the project

The main purpose of this project was to create a system that measures the accuracy of the 6DOF platform which is a chair that is used for virtualization of different simulations like, roller coaster or plane flight assimilations. This chair is made up of six actuators which move up and down and can be measured to see if they move accordingly to the commanded data. The chair has also a platform which moves in three angular degrees and so the system will measure the accuracy of the angles moved. The following is a picture of the 6DOF chair that the system will measure focusing on its accuracy.

A close up of a bicycle

Description generated with high confidence

Fig 2: 6DOF Platform Chair

As we can see from the picture there are six actuators which hold the platform that the chair is mounted on, at the end of each actuator distance sensors are attached to it so that they can measure the change in length of the actuators as the chair moves. The project intends to create a system called the Middlesex Platform Measuring System which will measure system errors of the 6DOF platform chair.

Functional Overview

The system created in this project has specific functions that it needs to perform. These functions in detail can be found in the functional specification document for the Middlesex Platform Measuring System but an overview of the function is divided into two different parts the Sensor Server and the Monitor functionalities.

These two different parts are involved with reading actual distances of the actuators on the chair and angles that the chair move with using sensors and using scripts to direct the movement of the chair in what direction using commanded values. Afterwards we take the commanded data which directed the chair where to move and the actual readings of the distances and the angles that the chair moved and compare them to see what was the difference between them to identify the error of the platform chair or maybe there is no difference at all meaning that there is no error at all.

These two parts are divided in a way that the Sensor Server is involved with the reading of the actual distance and angle readings from the chair using sensors and the Monitor is involved with setting commanded data to tell the chair in what position to move and then use these commanded data values to compare it with the actual movement of the chair readings from the Sensor Server. The commanded data is in a form of sequential positions that the chair moves for sensors to get readings at certain positions and each time the chair moves to the maximum position which is +100 mm and -100 mm for the x, y and z directions and +10 degrees and -10 degrees for the roll, pitch and yaw directions it returns to the neutral position. The sequential position number will start with 1 being neutral then 2 is X+20 and so forth as follows, the numbers represent the position of the chair;

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| X | Y | Z | Roll | Pitch | Yaw |
| +20 (2) | +20 (14) | +20 (26) | +2 (38) | +2 (50) | +2 (62) |
| +40 (3) | +40 (15) | +40 (27) | +4 (39) | +4 (51) | +4 (63) |
| +60 (4) | +60 (16) | +60 (28) | +6 (40) | +6 (52) | +6 (64) |
| +80 (5) | +80 (17) | +80 (29) | +8 (41) | +8 (53) | +8 (65) |
| +100 (6) | +100 (18) | +100 (30) | +10 (42) | +10 (54) | +10 (66) |
| Neutral (7) | Neutral (19) | Neutral (31) | Neutral (43) | Neutral (55) | Neutral (67) |
| -20 (8) | -20 (20) | -20 (32) | -2 (44) | -2 (56) | -2 (68) |
| -40 (9) | -40 (21) | -40 (33) | -4 (45) | -4 (57) | -4 (69) |
| -60 (10) | -60 (22) | -60 (34) | -6 (46) | -6 (58) | -6 (70) |
| -80 (11) | -80 (23) | -80 (35) | -8 (47) | -8 (59) | -8 (71) |
| -100 (12) | -100 (24) | -100 (36) | -10 (48) | -10 (60) | -10 (72) |
| Neutral (13) | Neutral (25) | Neutral (37) | Neutral (49) | Neutral (61) | Neutral (73) |

Table 1: Sequential positions for the chair movement

The Sensor Server:

This is involved in capturing the actual motion of the platform which will read the actual positioning and movement of the platform in real-time using distance and angular sensors. These distance and angular sensors measure the lengths of the actuators and roll, pitch and yaw values for the angles. Once these values have been read they need to be sent to the monitor for comparison with the commanded values. The format used for the sending the data consists of a header followed by six actuator lengths represented in Asci digits and three angular measurements represented in floating point numbers.

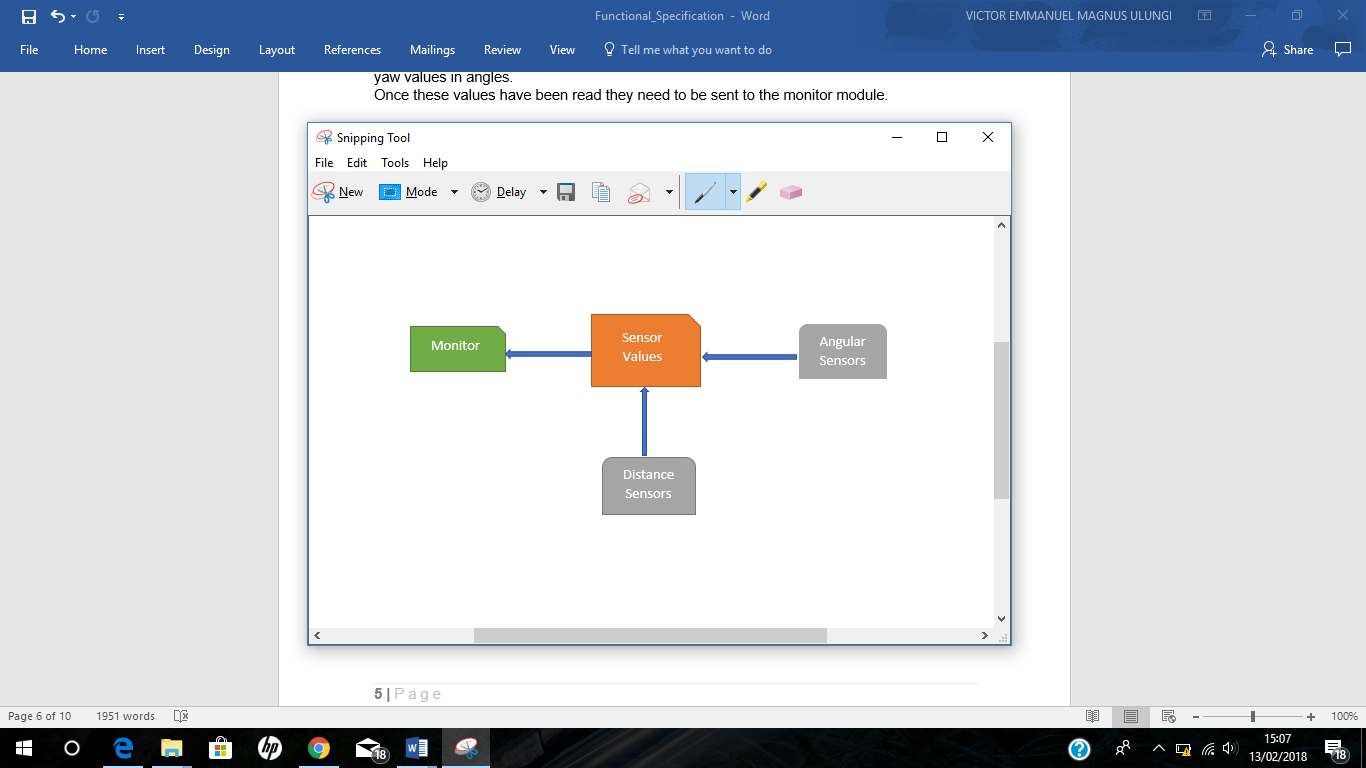


Fig 3: Sensor Server Block Diagram

For system overview see functional specification for details.

The Monitor:

This part is involved with sending commanded data to the middleware, receiving actuator lengths from the middleware, receiving sensor data from the Sensor Server, comparing the two sets of data, storing and displaying the error. The saved data will be saved in a csv extension file. The Monitor expects to receive the data from the Sensor Server in the same format that the Sensor Server sends the data. The commanded data consists of the number of steps that the chair needs to move with from neutral position to the maximum amount of length or degree.

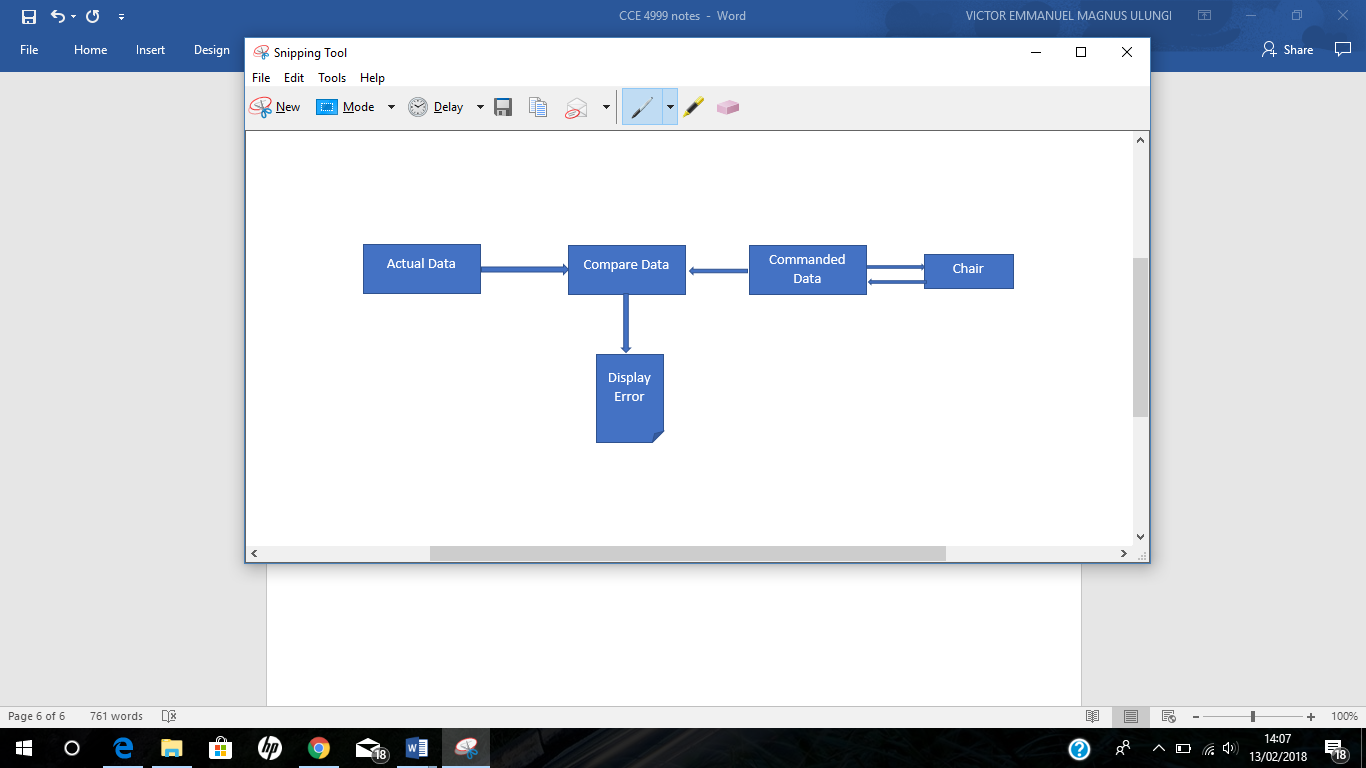


Fig 4: Monitor Block Diagram

To move the chair, the monitor set specific steps that the chair needs to move with in which the chair will always start from a neutral position and move five steps to the maximum angle or length and go back to the neutral position to move five steps again in another direction. The maximum angle that the chair can move for roll, pitch and yaw is 10 degrees. At first, we had decided to use 20 degrees as the maximum angle but unfortunately 20 degrees was too big of an angle for the chair to move and so 10 degrees was set to be the maximum angle. For more information see the functional specification document.

To compare the data, get the actual sensor values and the values from the middleware and subtract the two values to get the error results.

Design Overview

The system design consists of two parts which is the software design and the hardware design to make up the complete system design.

The Hardware design:

The hardware used in this project consists of distance sensors, Arduino Mega, Arduino Uno and an angle monitor module called jy901. These hardware devices are used for reading the actual lengths of the actuators and the angles of the chair, also these devices are used to send these readings to the specific area that is used for comparing these data with commanded data.

The distance sensors used to measure the lengths of 6 actuators are Maxbotic sonar sensor MB1043 HRLV-MaxSonarEZ4. These sensors are attached at the end of each actuator and are capable of reading 30cm to 5000cm with accurate precision up to the nearest 1mm on their serial pin. They were chosen because of its accurate precision and because they have a narrow beam which can heat the target at a specific position and measure the distance accurately.

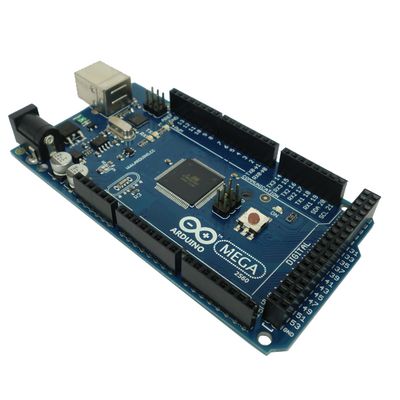
The Arduino Mega2560 is a microcontroller board based on ATmega2560, it was chosen because it has many number of pins and it can handle several sensors at the same time. To be able to connect all the sensors to the board a distance sensor adapter board is used. This board was created to connect all the six sensors to the SoftwareSerial ports in an easier manner.

The Arduino Uno is a microcontroller board based on the ATmega328p. Within the Uno there is an angle monitor hardware module called jy901 which is used to measure the angles, it was chosen because of its reliability.

To be able to send data from the Arduino Mega to the Uno we connect the two boards using jumper wires.

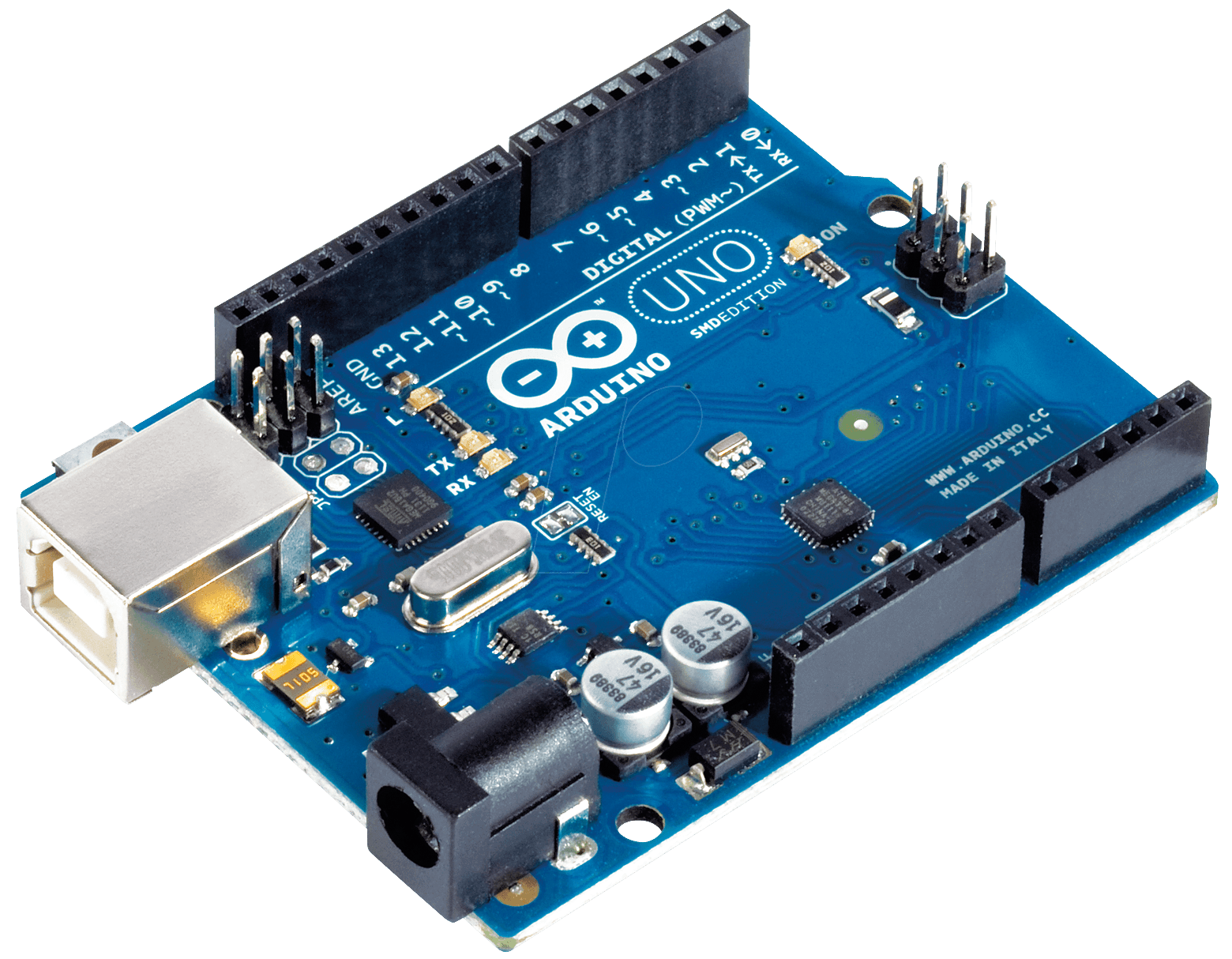
Below is a description of the hardware used

*Arduino Mega2560*

The Mega 2560 - a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button.

See design document for further details.

*Arduino Uno*



Arduino/Genuino Uno - a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button.

See design document for further details.

*MB1043 Sonar Sensors*



The HRLV-MaxSonar-EZ4 is the narrowest beam width sensor that is also the least sensitive to side objects offered in the HRLV-MaxSonar-EZsensor line. The HRLV-MaxSonar-EZ4 is an excellent choice when only larger objects need to be detected.

For more details see the design document.

*Technical Specs*

|  |  |
| --- | --- |
| Reading Rate | 10 Hz |
| Current Consumption | 3.1 mA |
| Operating Voltage | 2.5V - 5.5V |
| Maximum Range | 5000 mm (195 “) |
| Resolution | 1 mm |
| Operating Temperature | -15ºC to +65ºC |
| Sensor | 42 kHz |

*Target*

A picture containing bicycle

Description generated with high confidence

This target is made up of hard cardboard that was 3D printed in that circular shape for better capturing of the narrow laser beam from the distance sensor. The targets are attached to the actuators opposite the distance sensors.

The Software design:

Two systems are involved within the software design which are Sensor Server and Monitor module. The sensor server involves two modules which are distance sensors and angle server.

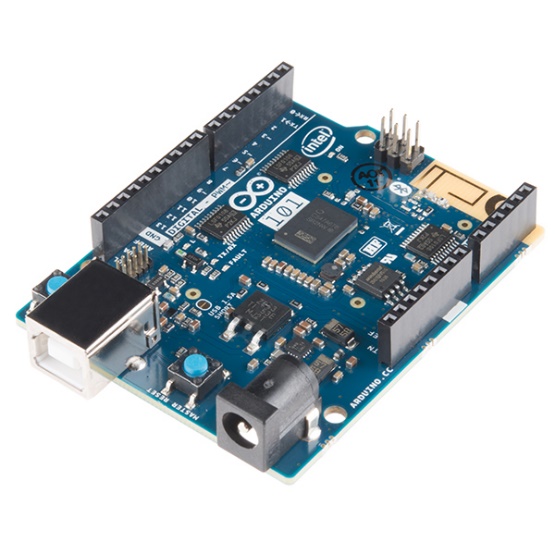
The distance sensor involves reading six distance sensor values using SoftwareSerial. After the distance sensor reads the actuator lengths it sends the data to the Angle server but if the distance sensor does not read any data then it sends a zero-value indicating that there is no data coming from the distance sensor.

The Angle server reads the 3 angular values using jy901 library which provides a method that reads the sensor angle data. It also receives the distance sensor data and creates a message format including both data, the distance data and the angle data and sends it to the Monitor in the format as mentioned in the functional specification and will send the data as soon as it is available.

Implementation Overview

At the beginning of the project each member of the group was assigned a role, one would be involved with project management, another will be involved in creating the sensor server system and the other will be involved with creating the monitor system. Each week certain tasks were assigned to everyone to complete in a specific amount of time and the progress of each week was recorded and if the task took longer than expected we recorded the reason why that task was delayed and when it was completed. As the weeks progressed and each person was working on their assigned tasks, it came a time when one of the group members dropped out and the project was left to be completed by just two members instead, since each person had already specific tasks assigned we had to invite a third party who had to do the tasks that were assigned by the member that had dropped out for the project to be completed on time.

When we began working on the project there were a few things that we had to change as the project progressed in terms of hardware. First, we used three Arduino boards in which two Mega boards were used to connect distance sensors and retrieve data from them using hardware serial. Each Mega board handled three sonar sensors and send the values to the central board. The other board was the Arduino 101 which was used as the central board that would measure the angular values and combine the data from the mega board and the angular data and send them to the monitor.



Arduino 101

This picture is an Arduino 101 that we used at the beginning of the project but later we decided to not use it and use Arduino Uno instead. We decided to not use it because it was unreliable as it was giving us garbage data on certain occasions and sometimes it would not work at all.

As we progressed we noticed that by using the two mega boards we were not able to send the sensor data simultaneously to the Arduino Uno because the i2c was not functioning properly since there were too many sketches involved keeping a track of the sketches was very difficult and so SoftwareSerial came into use which allowed us to use just one Mega board and less sketches.

To implement the project the hardware and software had to be prepared as described in the design overview. To start with Arduino sketches had to be created and uploaded on to the boards which are used to retrieve data from the sensors. Two sketches were involved where one read only actuator lengths via the mega board and the other sketch measured the angles and at the same time retrieved the distance sensor data from the mega board to the Uno and sent the combined data to the monitor.

**Arduino Uno**

1. Receives Six sensor values
2. Get the angular measurements
3. Arranging the data in seeding format
4. Sending data to monitor

**Arduino Mega**

1. Receives the six sensor values
2. Sending sensor values to Uno

Monitor

Fig 5: Sensor Server Block Diagram

The above block diagram shows the interconnectivity between the two Arduino boards and the Monitor.

The other part was to create a python script which sent commanded data to the middleware and received the actual sensor data from the sensor server and compared the sensor data and the commanded data then stores and displays the difference (error). The picture below shows the Monitor script running and displaying the actual sensor data from the sensor server named as sensor data average and the data from the middleware named as commanded data.

A close up of text on a black background

Description generated with high confidence

Fig 6: Monitor Display

Positioning of the hardware;

The Uno and the Arduino Mega board are placed on one board and that board will be placed on the chair and the sensors will be connected to the distance adaptor board using jumper wires. The following is a picture of the distance adapter board that is used to make it easier to connect all the six distance sensors to the mega board. The adaptor also provides resistors used by the software to determine if the sensors are connected. Within the adapter as seen below there are six connectors one for each sensor. The connectors consist of four pins one provides power, ground, receive serial data and the fourth pin is to test whether the sensor is connected.

A picture containing person

Description generated with high confidence

Fig 7: Distance sensor adaptor

The six sonar sensors will be mounted on the fix ends of the actuators. An actuator is like an arm with one fix end and the other one moving. The sensors will be mounted on one end, in a way that no distortion or error should occur in the measurement of distance due to the signals sent by any other sonar sensor and at the same time target boards will be used at the opposite ends of each distance sensor for accurate reading.

A picture containing bicycle

Description generated with high confidence

Fig 8: Sensor and Target mounting

Sonar sensors emit sound waves and there is a likely chance that misreading’s could occur. However, if we carefully plan on where to place the sonar sensors and perfectly direct the target opposite the distance sensor, then that chance can be minimized significantly.

The Monitor module communicates with the middleware through UDP, it has a Python script running on it which commands the chair to move in specified positions.

A circuit board

Description generated with very high confidence

Fig 9: Sensor Server Connection

Fig 4 shows how the sensors are connected to the mega board using the adaptor board and the USB cable which is used to connect the boards to the PC to upload Arduino sketches and afterwards connected to a PC where the Monitor script is running to receive the sensor data.

The implementation of this project also included how we managed and planned for this project as mentioned at the beginning. As we met up each week we planned out what needed to be done for that specific day and what is expected for the next coming days and that way we could control the tasks for each week. For more information of project management see the deliverables document.

Testing

To do a testing of the system created the following steps had to be done

Step 1:

First, we had to do a testing for the sensors where we had to test whether the sensors were working according to the data sheets and giving the correct output. to do that we connected the six distance sensors to the adapter board and ran the Arduino sketch to observe the output data. If the sensors were not connected properly a zero will be displayed for that sensor to specify that no data is coming from the sensor.

Step 2:

The next thing was to test whether we were receiving the correct sensor data which was done by running an Arduino sketch on the Uno which displays both the distance and angular data on the console to observe the sensor data read.

Step 3:

To test whether the Monitor was receiving data from the middleware and from the Sensor Server, a python script created for the monitor was ran and the output results were observed where the difference between the two data values were to be displayed in a file with a .csv extension.

Step 4:

To test the final functionality of the whole system, this was done by doing all the connections needed and running the system scripts and sketches as described earlier and observing the output results.

For further details on testing see the test plan document.

The following figure shows an example of the sensor server output when the testing took place.

A screenshot of text

Description generated with high confidence

Fig 10: Sensor Server output

Results

To get the final results for the system we ran the system three times so that we could use the average of those three output values since the sensors kept giving us different readings. These readings were very close to each other within 4mm difference for the actuators and 0.5 degrees difference for the angles. The following is a terminal display showing the monitor script running which takes three sensor readings and uses the average of those three readings as the sensor reading which is to be compared with the commanded data.

A picture containing text

Description generated with very high confidence

Fig 11: Monitor displaying the Sensor data average and commanded data

Once the system finished running and all the commanded steps were done, we had to run the system three times for a better results outcome which was done by calculating the average output from those three results. The error results were calculated by subtracting the actual sensor data values with commanded values from the middleware.

The following is an excel spreadsheet showing the error results that were taken as an average of the three runs. We also had to compensate certain values since the positions of the sensors with the target were quite different from others and so to level the results we had to compensate some of these values by either adding or subtracting a certain amount. L1 and L6 had the same measurements and so we did not do any compensation for them and the following are the compensation values used for each measurement; L2 (+14mm), L3 (-26mm), L4 (-33mm), L5 (-19mm). We did not do any compensation for the angle values because the there was only one roll reading and one pitch reading and so no need of any compensations as was done for the actuators.

We used a tape measure to measure the distance between the distance sensors and the targets that were mounted on to the actuators so that we could make sure that the distance would be same for all the actuators but due to the make of the chair and how the actuators were fitted on to the chair for some actuators we had to position the distance sensors on to the bottom part of the actuator and that making the distance between the targets and the sensors were different for some actuators. The following is a table showing the distance between the sensors and the targets measured using a tape measure and what the actual sensor reading is to observe the distance sensor performance. The measuring was done when there was no air in actuator muscles.

|  |  |  |
| --- | --- | --- |
|  | Tape measure reading | Sensor reading |
| Actuator 1 | 601 mm | 582 mm |
| Actuator 2 | 587 mm | 426 mm |
| Actuator 3 | 627 mm | 590 mm |
| Actuator 4 | 634 mm | 586 mm |
| Actuator 5 | 620 mm | 678 mm |
| Actuator 6 | 601 mm | 593 mm |

Table 2: tape measure and sensor readings

As we can see from table 2 that the difference between the sensor reading and tape measure reading is quite different due to the position where the laser beam hits the target to measure the distance and where positioned the tape measure from the sensor to the target to measure the distance. These readings can help us understand why the errors that we get are larger than expected. The values for the roll and pitch angles were the same when we compared them using the inclinometer and the jy901 sensor reading.

To measure the roll and pitch values we used an electronic inclinometer in which its accuracy was to 0.3 degrees and as we measured what the inclinometer read and what the jy901 sensor in the Arduino Uno read they were quite the same and so no compensation was needed for that.

The expected accuracy of the actuators is 1mm and for the angular measurement is ½ a degree, meaning that these values are the errors expected from the actuators and the angle motion.

The following is an excel spread sheet that shows how the results are displayed, The L’s represents the error values of the six actuators while the roll, pitch and yaw values represents the error values of the roll, yaw and pitch measurements.

At the right end side of the excel sheet the steps that the chair has moved is displayed starting from neutral position all the way to the last step of the chair moving yaw-10 meaning ten degrees in the yaw direction. The following is just a screenshot that shows a part of the error results.

A close up of text on a white background

Description generated with very high confidence

A picture containing text, newspaper, crossword puzzle

Description generated with high confidence

Fig 12: Results

According to the output results that we got 45mm was the accuracy in practice as this was the smallest error that we got from the actuators and for the angles the error was 0.3 degrees which was the smallest error.

From the excel spreadsheet above the yaw values have been highlighted red because we were not able to measure it since the angle sensor was not reading valid data for the yaw values.

With these results they tend to show us how accurate the chair motion is in regard to the commanded lengths and angles given to the chair and the actual lengths and angles that the chair actual moves with. Since the system created, measures the accuracy of the system what is displayed on the results table is the error value between the commanded measurements and the actual sensor data read from the actuators and the angle motion of the chair.

From the results table it shows that Actuator 5 has less errors compared to the other actuators and this might be due to the positioning of the target which made the sensor reading slight accurate compared to the other actuators. According to the results table above the most accurate movement for L1 is z+20 movement where the chair moves 20 cm in the z-axis. For L2 the most accurate movement is x-80 where the chair moves 80 cm in the x-axis. For L3 the most accurate movement is x+20 where the chair moves 20 cm in the x-axis. The most accurate movement for L4 is x-20, for L5 its y-40 and for L6 is y+20. All these movements are the most accurate for each of the actuator because those movements had the least error compared to the other movements.

Using Fig 12 we can see that the Roll movements were almost accurate compared to the Pitch since the errors were quite small in comparison to the Pitch values. For the Roll the most accurate movement was when the chair was in neutral position and for Pitch the most accurate movements was Pitch-4 meaning that when the chair was moving four steps in the pitch direction.

A close up of a map

Description generated with high confidence

Fig 13: Roll and Pitch errors: x-axis is the sequence number of commanded positions (see table 1) and y-axis is degrees errors

The above graph interprets the output results of the roll and pitch values where series 1 represents the roll error results and series 2 represents the pitch error results. According to the graph we can see that the error for the angular values are quite small since the values are very close to the zero value when the chair was moving in the x, y and z direction. Once the chair starts moving in the roll and pitch directions we can see that for most parts the error was closing in to the zero value but only when at certain steps that the error increased drastically. The peaks in the graphs represent the increase in error.

This increase of error was due to the problem that we had where the actual sensor reading was negative instead of positive and so when calculating the error, it tends to add the values together and so a high positive number or high negative number is resulted. This error was rectified by changing the sign of the calculation which looked at the difference between the two values within the monitor. The following is a chart that shows the output results after fixing the addition issue

A screenshot of a social media post

Description generated with very high confidence

Fig 14: Roll and Pitch errors: x-axis is the sequence number of commanded positions (see table 1) and y-axis is degrees errors

With the Yaw values we were not able to get valid data and even when we tested it out of the system to see the display results, it kept on giving us invalid data. We did some research on it and for most people who used the same sensor that we used to measure yaw data also had a similar problem but they were not able to fix it and even one person had written a comment to the manufactures asking for some assistance. But it seemed that once most of the people changed and used a new sensor they got valid yaw data and so we think that the reason might be that the yaw sensing technology on the sensor might be at faulty.

Key Findings

As the purpose of the project is stated above, we ran the system and recorded the output results as will be seen in the results section. With these results we were able to interpret what they meant in terms of how accurate the commanded values were to the actual movement of the chair. The system created and the outcome of it, did achieve the stated goals for this project as we were able to monitor and see how accurate the 6DOF platform chair is.

The accuracy between the commanded values and the actual movement of the chair was quite okay, meaning that it was not completely inaccurate only that the difference between the commanded data and the sensor data that measured the actual movement of the chair was not as huge. Some measurements were slight accurate than others, this was due to the positioning of the distance sensors in which some of them were positioned in a way that they added extra mm to the readings or they needed some extra mm for compensation.

To analyze and interpret the result reading and to see whether there was a pattern in the results we used only the error values at the neutral position of the chair to see whether these values were the same in each run that we did or they were different and if so what does it mean. First using the error results for the actuators, the following were the patterns for the first run, third and the average results of the three runs.

According to the graphs showing the error results for the actuators, the x-axis represents the neutral position steps and the y-axis represents the errors in mm. And the graphs showing the error results for the roll and pitch values the x-axis represents the neutral position steps and the y-axis represents degrees errors.

A screenshot of a map

Description generated with very high confidence

A screenshot of a map

Description generated with very high confidence

A screenshot of a map

Description generated with very high confidence

From these graphs we can observe that positioning of the actuators when the chair is in a neutral position is the approximately the same for each run but we can see that the more you run the system the better and stable the results are and the errors tend to decrease.

From these graphs we can also see the difference that multiple runs can have to the system. Also, the most accurate actuator is actuator five which is L5. One thing that can be improved for better results would be to increase the delay time between one movement of the chair to the next since with these runs we used 3 seconds as our delay time if we were to increase the delay time we would get better and stable results. The delay time helps in allowing the sensors to get stable readings since the chair is made up of flexible actuators and so during movement they tend to bounce for some time before they become stagnant so putting a delay will help for a stable sensor reading.

With these graphs it verifies that the readings are repeatable meaning that there isn’t any issue with the system created and that just different factors like positioning of the sensor and the instability of the chair that causes the errors to be bigger than expected.

We also used the roll and pitch values when the chair was at the neutral position to observe what is the pattern between different runs and the average of those runs.

A close up of a map

Description generated with high confidence

A close up of a map

Description generated with high confidence

A close up of a map

Description generated with high confidence

As explained earlier the peaks of these graphs are due to the addition of the values together instead of subtraction of the commanded data and the actual sensor data which was rectified. With the solved data the difference between the sensor reading and commanded data should be around 0.2 to 0.3 degrees for the roll values and 0.9 and 1.3 for pitch values in which it would make the lines on the graph quite closer to the zero value.

According to these graphs we can observe that the result values for the different runs and the average are the same meaning that for each run the readings done by the sensor are the same.

With the output readings we believe that the slight inaccuracies are caused by different things like, the positioning of the distance sensors on the actuators, although some compensations were done still the readings were not 100%. As seen from the monitor display that the sensors were giving multiple readings and we had to take an average of three readings as our sensor data also might have been another cause of inaccuracies. Another reason of the inaccuracies would be movement of the chair while the sensors are reading the data, when the chair moves each step as commanded we did give it a delay of 3 sec before moving to the next step so that to give time for the sensors to record accurate reading but since the chair is still in motion the readings will not be 100% accurate.

Lessons Learned

There is always a lesson to learn when one is doing a project, the lesson can be learned throughout the project period or after one is finished with the project, in our case the following are the lessons that we learned during this time when we were doing the project.

There were many requirements that had to be made in order to complete this project. Although most of the requirements were met, there were a few things that we thought needed to be improved such as time management, organizational skills, communication skills and technical skills.

Time management – at the beginning of this project we had a plan of what the project required us to do but the concept of it at the beginning was not clear enough and so it took us quite a few weeks to get a hold of what needed to be done and in doing so the time went quick and we were not able to manage our time properly thus wasting a lot of time.

Certain tasks were not completed on time because they were interdependent and some of the group members were not able to complete the tasks on time.

Another thing that made us waste a lot of time was that we did not anticipate that one of the group members was not going to fully commit to the project till the end and in doing so it came to a point where he was not able to continue working on the project and thus some tasks were left incomplete and we had to use more time to complete those tasks. Thus, we should always plan a head and anticipate for such occurrences.

Organizational skills - when doing this project, we came to realize that organization was a key to a successful project. In the beginning of the project we did not have proper organization of our work and in doing so some documents got lost and thus cost us a lot of time which we could have used for other important tasks. As things got more complex and we had a lot of documentations and coding to do a better way of organizing ourselves had to be implemented and thus GitHub came in to use. We also used Google docs at the beginning but we opted to use GitHub which made it easier for us to share and work simultaneously on the documents and coding.

Communication skills – at the beginning of the project we had a lot of issues with communication among the group members. Some of our team members were unable to participate in meetings which were held outside the class time. Also, some team members were not responding to emails and text messages which needed them to complete certain tasks, and in doing so it made us fall behind. For better communication we had to encourage each other to respond to emails, attend meetings and take responsibility for the tasks they had been assigned to do.

Technical skills – this project required a lot of technical skills which some of our team members lacked initially. To improve these skills, we had to learn as we progressed with the project and in doing so we were able to develop new skills and enhance the skills we already had.

Apart from the challenges mentioned above there were some obstacles that arose as we were testing some parts of the system in both hardware and software as described below

At the beginning of the project the Sensor Server was using three microcontroller boards which were Arduino 101 and two mega boards. Each mega board handled 3 sonar sensors and sent values to the Arduino 101 which was functioning as the central board.

With using the three boards some issues arose such that we were not able to send the sensor data to the central board simultaneously. The reason was because the i2c was not functioning properly since there were too many sketches involved and so it was very difficult to keep track and so SoftwareSerial came into use which allowed us to use less boards and fewer sketches.

The other issue that arose was with the use of the Arduino 101 which was unreliable because one minute it would work and the next it was not able to receive the sensor data at all and at the same time uploading the sketch on to the board was an issue due to Timed out error. To solve these issues, we decided to use Arduino Uno as the central board with a module called jy901 (see design document for further details).

When it came to a point where we had to run a final test of the system some issues arose where when we were running the Arduino sketches to read sensor data we were getting extra garbage messages coming together with the sensor data. To find out what and where the issue was, we had to run a few tests to ensure that all the hardware was not faulty. Also, we ran some tests to see if the problem was with the SoftwareSerial or the Arduino sketches. The issue apparently was a result of the buffer size that was set for the SoftwareSerial and so we increased it from 64Kb to 256Kb and that sorted out the problem.

Also, while doing the testing we noticed that using small targets was not giving us good reading and so we decided to use bigger targets for all the sonar sensors.

One of the other lesson we learned was that we had to calibrate the system while doing the testing and that is because we needed to make sure that the reading that sensors were giving were correct and so we had to measure the lengths and angles and compare them with the sensor readings and see if the readings are the same if not then we should take that difference and use it as a compensation value for the final output values.

During the final testing of the system some of the readings for the angles were not right, we noticed that we were getting a very big error of above 14 degrees for the roll and pitch when the chair was in neutral position in which it is impossible since even the maximum angle that the chair can move was up to +10 or -10. When we investigated what the problem would be, we found that within the monitor code when the commanded data and sensor data were being compared the commanded data was a positive value and the sensor data was a negative value and so when we subtract the two values it adds the two values together and so the output reading becomes more then 10 degrees the reason being that the sensor was returning negative values when it was positive values and vice versa. To solve this problem, we had to change the sign of the calculation which looked at the difference between the two values within the monitor.

With all these challenges we came across we have managed to complete the project and learn from those challenges. Unfortunately, it is not always possible to complete all the tasks that you have set out at the beginning on time, as there are factors you don’t take into consideration at the beginning that pops up later.

Conclusion

This project took us about 24 learning weeks to complete and within those weeks we were able to design and build a system that measures the accuracy of the 6DOF platform by comparing commanded data with actual sensor data and displaying the results. Throughout this project we were able to learn a lot and develop our skills in terms of technicality and theoretically. This report covers the process that was taken for a successful project and how it was implemented. The report also involves an overview of some of the documents that will be included within this document. During the final testing of the system a lot of issues came up that were not anticipated at the beginning and so to solve these issues we had to take a step back and look through every detail of the system and analyze it to see if the problem was either a software one or a hardware one and finally solve the issues. The system created succeeded in meeting the goals and purpose of this project.