April 11, 2017

Mohamed Al-Janabi

Shirwa Haji

Functional Specifications

**Real Time Performance and Error Correction**

**of Middlesex 6DOF Platform (The Chair)**

**CCE4999**

[**1. Abstract**](#_sa94r7skwnb0) **2**

[**2. Background**](#_lcgv66tcu6v2) **2**

[**3. Introduction**](#_6uhgdl9dbydx) **3**

[**4. Functionalities / Usage**](#_xyvvnfrzd7tw) **4**

[4.1. Capturing Real Time Data](#_19dvqs4fk4bl) 4

[4.1.1. Initializing the Sensor Server](#_hh6x6eqyi0xo) 5

[4.1.2. Troubleshooting](#_c30t9e8nf6rb) 5

[4.2. Capturing Commanded Data](#_v2x0jbvrlvh) 6

[4.2.1. Initializing4.2.1. Initializing](#_jdhn8dtrjp21) 6

[4.2.2. Troubleshooting4.2.2. Troubleshooting](#_geq4byk086r6) 7

[4.3. Viewing and Storing Error](#_q2kkd1q3prb6) 7

[4.3.1. Initializing the Monitor](#_cwad5sytdbcq) 8

[4.3.2. Troubleshooting](#_jd47hog3uss2) 8

[4.4. Viewing Stored Data](#_64o071as73mz) 8

[4.5. Unit Testing](#_if0op3nyw1fy) 8

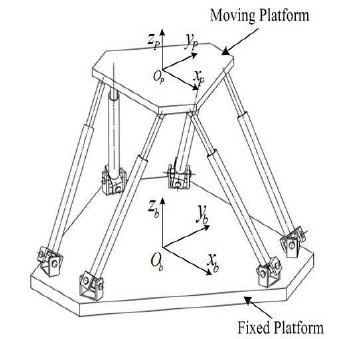
[**5. References**](#_wu2hkxv8nmnr) **9**

# 1. Abstract

This document describe the functional specifications for the hardware and software system that monitors the real time performance of the Middlesex 6DOF platform (The Chair) and also provides the capability to enhance the accuracy by error correction of existing software that controls the platform. The specifications define what the system does, the user-system relationships and how the end user will interact and use the system. This document acts as guide for system architecture and development. It also provides troubleshooting details including possible cause of errors and their solutions for various functionalities of the system.

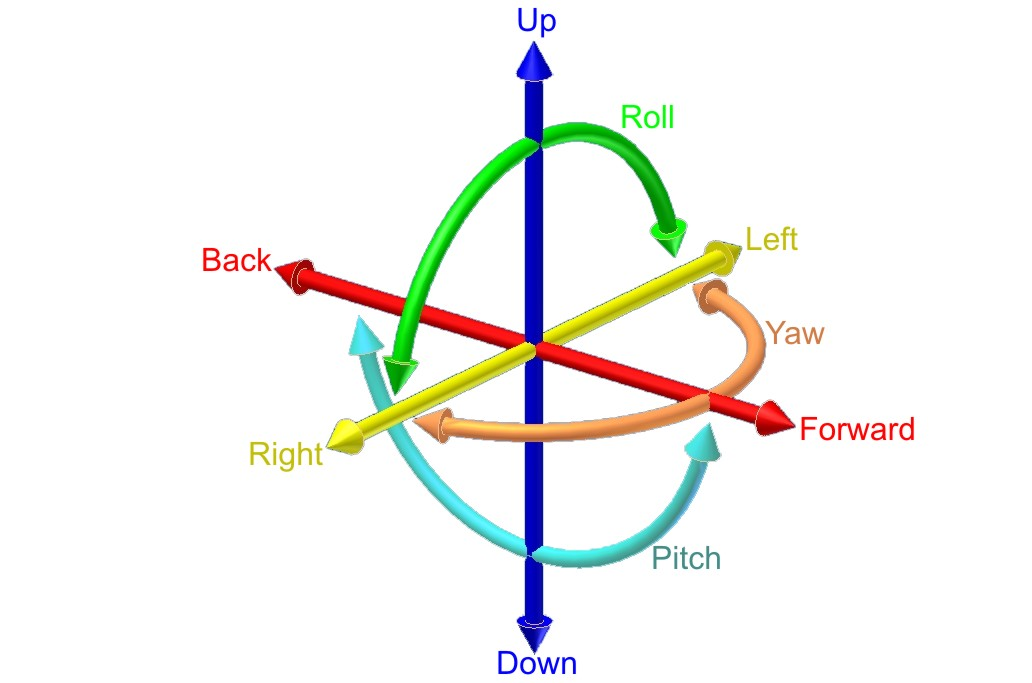
# 2. Background

The Chair is basically a 6DOF motion platform built using a robotic configuration called Stewart platform. It has two platforms, Fixed Platform and a Moving Platform which are attached together with six prismatic actuators, commonly hydraulic jacks or electric actuators. The moving platform can move in six degrees of freedom using six prismatic actuators as shown in the picture below.



**Figure 1. Stewart Platform [1]**

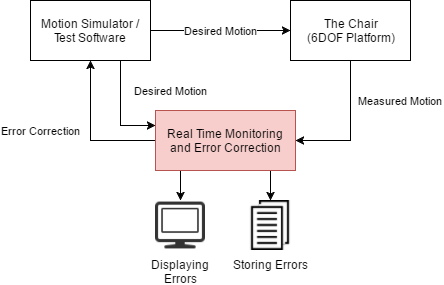
The six degrees include three linear movements x, y, z and three rotations yaw, pitch, roll.



**Figure 2. Six Degrees of Freedom [2]**

# 3. Introduction

The purpose of this project is to design and develop a complete system including hardware and software that monitors the real time performance of the Middlesex 6DOF Platform and provides error correction to the existing motion simulating software that controls the platform. The system will measure the actual movements of the platform by using a set of sensors and will compare the measured movements with the desired movements that are generated by the motion simulators.



**Figure 3. Block Diagram**

The system will measure the motion with accuracy up to 1 mm for linear motion and 0.5 deg for rotational motion. The Test Software that generates the dummy commands is also the part of this project.

# 4. Functionalities / Usage

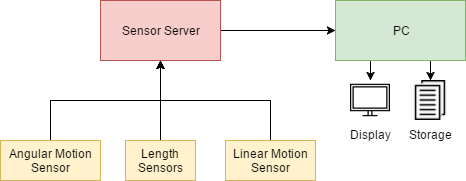
The system can be categorised in five functionalities that it provides:

1. Capturing Real Time Data
2. Capturing Commanded Data
3. Viewing Error
4. Viewing Stored Data
5. Unit Testing and Verification

The details of the above mentioned functions of the system are described below:

## 4.1. Capturing Real Time Data

The capturing of the actual motion of the platform is handled by a module called Sensor Server. The Sensor Server will read the actual position and movement of the platform in real time, at least every 30 msec, using a controller and a set of motion sensors. It will also display the real time data and store it as comma separated values.



**Figure 4. Block Diagram of Sensor Server**

There are a total of 13 parameters that are be measured by the Sensor Server i.e.

1x Unix timestamp (t)

6x Length of individual prismatic arms in mm (L1-L6)

3x Linear Motion of three axis in m/s^2 (ax, ay, az)

3x Angular Motion of three axis in rad/s^2 (gx, gy, gz)

The format for displaying and storing the actual data is illustrated below.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **t** | **L1** | **L2** | **L3** | **L4** | **L5** | **L6** | **ax** | **ay** | **az** | **gx** | **gy** | **gz** |
| 0 | 10 | 12 | 9 | 13 | 10 | 13 | 12 | 12 | 34 | 50 | 23 | 45 |
| **...** |  |  |  |  |  |  |  |  |  |  |  |  |

### 4.1.1. Initializing the Sensor Server

To start the Sensor Server, make sure that all sensors are connected to it and it is powered ON. Connect the Sensor Server with the PC and run the program SensorServer.

Once the program is initialized successfully, it will immediately start displaying the sensor data according to the format described above. Once the program is stopped, it will dump all the data in a new file created in the same folder where the program is located. Make the platform move using Motion Simulator or Test Command Server and see the actual data change in real time.

### 4.1.2. Troubleshooting

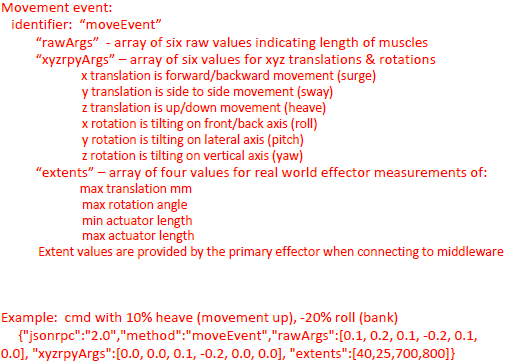
In case there is a problem in capturing or storing the real time data, an error message will be generated on the display.

Common error messages or problems and their solution are listed below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Code** | **Error Message / Problem** | **Possible Cause** | **Solution** |
| 100 | No such port as xx | The PC is not able to detect the Sensor Server | Unplug and reconnect the Sensor Server with the PC |
| 101 | Could not connect with port | The Sensor Server is used another application in the PC | Close all other programs and try restarting the PC |
| 102 | Cannot write file | The storage file is being used by another program | Close all other programs that might be using the output file |
| - | Some of the actual values are not changing or stuck at zero | The corresponding sensor is disconnected from the Sensor Server | Verify all the sensors are connected to the Sensor Server and restart the program |
| - | Inconsistent sensor values or garbage values | The sensor is loosely connected or not configured properly | Replug the sensor and reconfigure its settings for e.g. baud rate. |

## 4.2. Capturing Commanded Data

The commanded data is served by the Middleware or by the Test Command Server. The module Monitor connects to either the Middleware or the Test Command Server to get the commanded data. The commanded data is received in a json format as described below.



### 4.2.1. Initializing

To start capturing the commanded data, start the program called CommandedData. Once the program is started successfully, it will connect with the Middleware or Test Command Server and display the commanded data on screen.

The data will be displayed as it is received i.e. in a json format as shown below.

{

“jsonrpc”: “20”,

“Method”: “moveEvent”,

“rawArgs”: [0.1, 0.2, 0.1, -0.2, 0.2, 0.3],

“xyzrpyArgs”: [0, 0, 0, 0, 0, 0],

“Extents”: [40, 20, 700, 800]

}

### 4.2.2. Troubleshooting

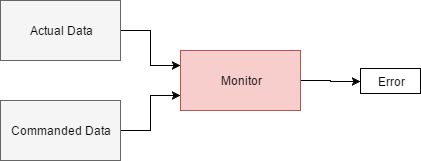
In case there is a problem in viewing the commanded data, an error message will be generated on the display.

Common error messages or problems and their solution are listed below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Code** | **Error Message / Problem** | **Possible Cause** | **Solution** |
| 104 | Could not open port xx | The PC is not able to connect with the Middleware or Test Command Server | Make sure that the Middleware or the Test Command Server is running and the port number is configured properly |

## 4.3. Viewing and Storing Error

This main functionality of the system is to compare the actual data with the commanded data and calculate the error. This is handled by the module called Monitor. It connects with the Middleware and the Sensor Server, gets both the actual data and the commanded data and calculates the difference between them.



The format for displaying and storing the error data is same as the actual data except that it is a difference from the commanded data.

### 4.3.1. Initializing the Monitor

To start calculating and viewing the error data, start the program called Monitor. Once it has initialized successfully, it will start capturing the actual data and the commanded data and then calculates the error in real time.

### 4.3.2. Troubleshooting

In case there is a problem in calculating the difference between the commanded data and the actual data, an error message will be generated on the display.

Common error messages or problems and their solution are listed below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Code** | **Error Message / Problem** | **Possible Cause** | **Solution** |
|  |  |  |  |

## 4.4. Viewing Stored Data

The data stored by Monitor is in simple comma separated values format. This file can be easily opened with MS Excel or even in text editor. The format for storing the data is the same as described previously.

## 4.5. Unit Testing

## 

# 5. References

|  |  |
| --- | --- |
| [1] | <https://www.researchgate.net/figure/264725216_fig1_Fig-1-Generalised-Stewart-platform-manipulator-motion-consists-of-longitudinal> |
| [2] | <https://en.wikipedia.org/wiki/Six_degrees_of_freedom> |
|  |  |