## Chapter 5 Implementation Details

# Implementation

The system consists on five major modules i.e. Marker Hardware, Audio Hardware, Controller Application, Player Application and LMS Web Application. Implementation detail of each module is discussed below.

# Marker Hardware

The role of this module is to give orientation and tip pressure of the marker. The module sends data packet that contain encoded orientation of marker in 3d space in form of Euler angles.

## Requirements Addressed

|  |  |  |
| --- | --- | --- |
| **#** | **Requirement** | **Priority** |
|  |  |  |
| **1** | Determine orientation of board marker and calculate respective Euler angles | HIGH |
| **2** | Transmit the calculated Euler angles to desktop app via nrf24l01 module | HIGH |
| **3** | Transmit the calculated Euler angles to desktop app via RS232 serial connection | LOW |
| **4** | Turn on using 3.6 volts Li-po battery with Boost converter circuit | MEDIUM |
| **5** | Build battery charging circuit within board marker | LOW |
| **6** | Implement RGB Led for positioning purpose (Input for camera module) | HIGH |

Table 5.1: Board Marker Hardware Requirements

## General Flow

* Board Marker try to establish wireless connection with the receiver. RGB Led fades meanwhile.
* RGB Led turns to constant red after successful connection.
* Accelerometer unit in the Board Marker determines the orientation data.
* NRF24l01 wireless module in the Board Marker transmits the orientation data to Receiver wirelessly.
* Receiver Transfers orientation data to desktop app via serial connection

## Detailed Design

Marker hardware has two major sub-modules named as **Transmitter** (marker itself)and **Receiver**

|  |  |  |
| --- | --- | --- |
| **Component Number** | **Description** | **Diagram** |
|  |  |  |
| **1** | **Name:** Tact Tactile Switch | Figure 5.1 - Tact Tactile Switch |
| **Detail:** Toggle switch that turns on/off the system when runs on Battery. It does not have any effect when system is running via USB cable. |
| **2** | **Name:** USB DC-DC Boost Converter | Figure 5.2 - DC-DC Boost Converter |
| **Detail:** Converts the 3.7V to 5V to turn on and constantly run the Arduino nano prototype board. This sub-module has built-in charging circuit that charges the battery through USB connection. |
| **3** | **Name:** Glowing ball | Figure 5.3 - Glowing Ball |
| **Detail:** Round shaped glowing ball can glow in any combination of RGB colors. It is not for just looks but acts as an input to Stereo cameras for position tracking. |
| **4** | **Name:** Arduino nano | Figure 5.4 - Arduino nano |
| **Detail:** Arduino nano acts as main processing board to which all modules and sensors are attached. It acts just like a motherboard with central processor chip soldered on mainboard. |
| **5** | **Name:** Li-po battery | Figure 5.5 – Li-po Battery |
| **Detail:** 600mAh 3.7V Li-po battery used to run system while there is no USB connection. Voltage may be up to 4.2 volts when fully charged.  DC-DC Boost Converter is hooked up with the battery that charges the battery as well as raises its voltage to 5V to make the Board Marker Transmitter working properly. |
| **6** | **Name:** dc-le14112 RGB Led | Figure 5.6 - RGB Led |
| **Detail:** 3W RGB Led used to create custom color of choice, the corresponding color that is required for position sensing can glow in Glowing ball. It may be given external power source but, in our case, it is directly connected to Arduino nano. |
| **7** | **Name:** MPU-6050 | Figure 5.7 - MPU-6050 |
| **Detail:** MPU-6050 or GY-521 board contains accelerometer and gyroscope packed in a single chip. It senses the orientation of the object. It is connected to Arduino nano via I2C bus. |
| **8** | **Name:** nRF24L01 | Figure 5.8 - nRF24L01 |
| **Detail:** nRF24L01 is a single chip radio transceiver. It is responsible for transmitting orientation data from transmitter module to receiver. |

Table 5.2: Board Marker Components Detail

## Marker Transmitter

The objective of the transmitter module is to extract orientation of the board marker. The challenge is the marker is changing its orientation while writing on the board. Transmitter module is designed as a back cap of board marker. It is attached to the board marker to record orientation of marker. The role of this sub-module is to transmit the tip pressure and orientation of marker in space. Further details of each part constituting the transmitter are given below.

### Components Used

Detailed description of electronic components excluding discrete consumer parts, e.g. wires, is given below and described in Table 5.2.

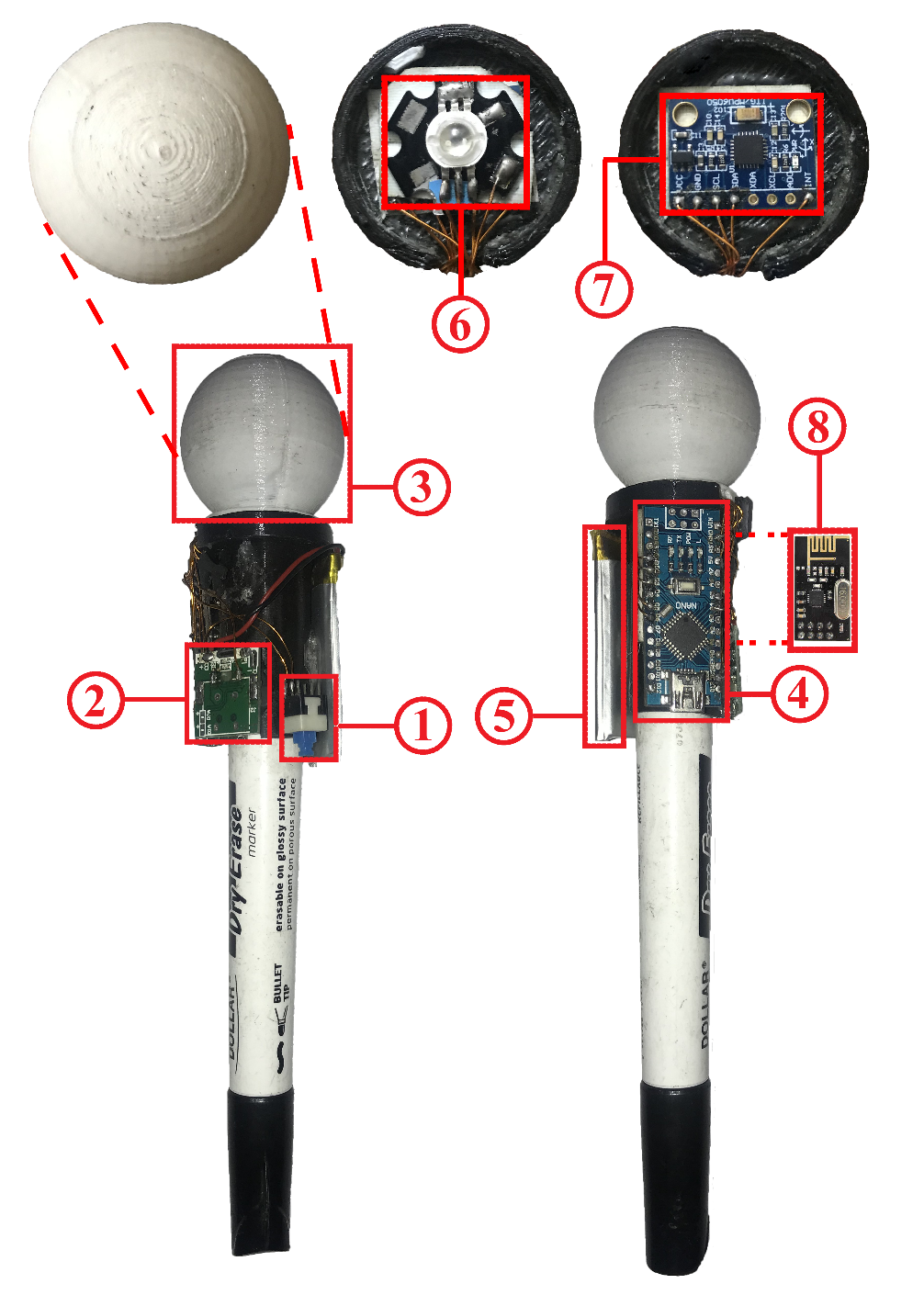


Figure 5.9: Board Marker Transmitter Component Detail

### Component Connection Diagram

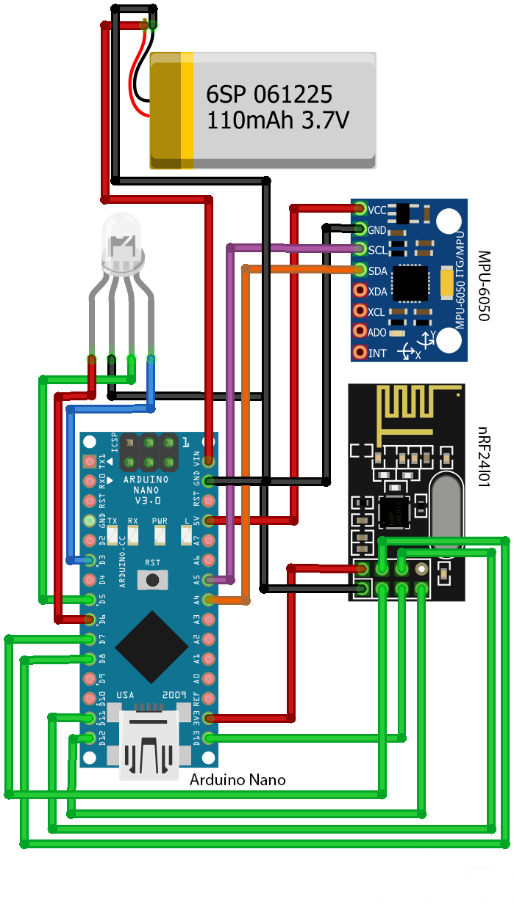
This diagram represents how sub-modules or components are connected in Transmitter module.

Figure 5.10: Component connection diagram of Board Marker Transmitter

### Schematic Diagram

Schematic diagram of Board Marker Transmitter can be seen as below

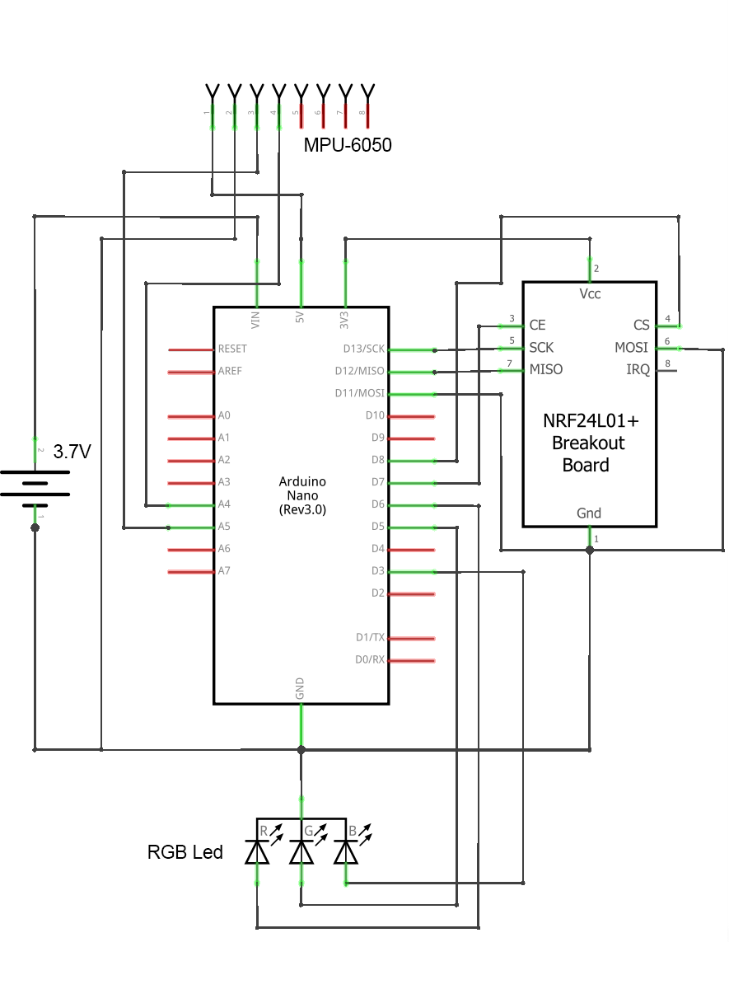


Figure 5.11: Schematic diagram of Board Marker Transmitter

### General Flow

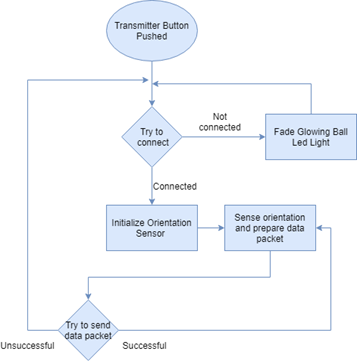


Figure 5.12: General Flow of Board Marker Transmitter

## Marker Receiver

Receiver module receives orientation data as Euler angles and transfer it to the desktop application via USB connection. As it is connected via USB so it does not need any external power source.

### Component Connection Diagram

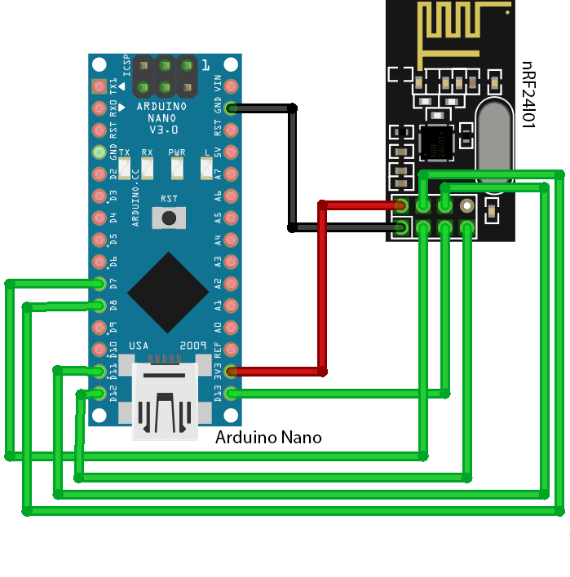


Figure 5.13: Component connection diagram of Board Marker Receiver

### Schematic Diagram

Schematic diagram that shows abstract component view of Receiver module is given below

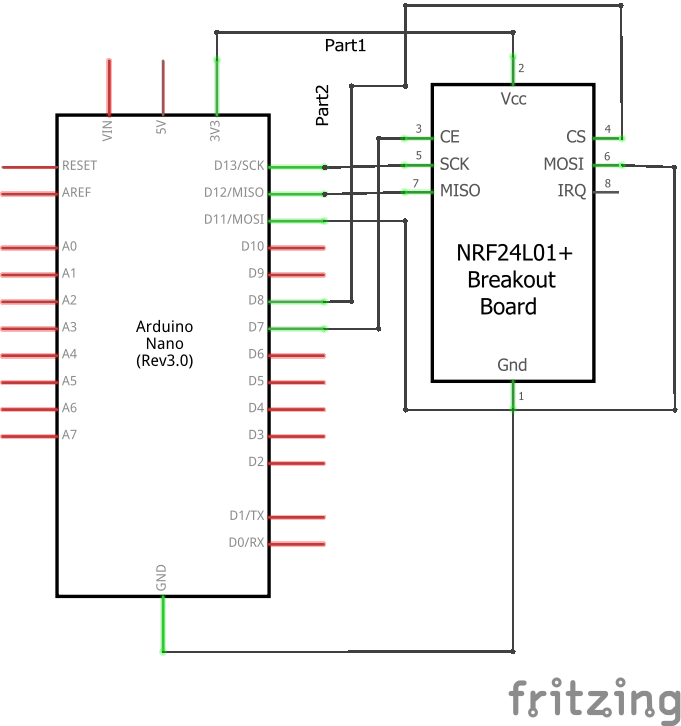


Figure 5.14: Schematic diagram of Board Marker Receiver

### General Flow



Figure 5.15: General Flow of Board Marker Receiver

## Rules and Assumptions

Following are rules and cases of assumptions that are assumed to be true while normal working

* Board Marker Transmitter and Receiver are in range of 2 meters for less noise and preventing latency issues.
* Pressure threshold of board marker tip is 5Pascals that is equivalent to pressure of lead pencil tip. Above this pressure, marker will write and otherwise not.
* Glowing ball of board marker must be at least partially visible by either of the cameras. Precision of marker position decreases from Case 1 to 6. Best Accuracy in Case 1 and no output at all in Case 6.
* User is supposed to be not touching the Marker tip while recording the lecture.
* User is supposed to be write only in the boundary of the defined platform i.e. whiteboard.

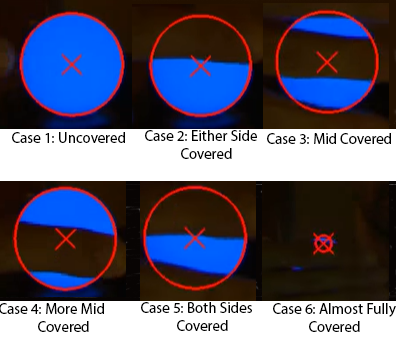


Figure 5.16: Glowing Ball Detection While Covered

## Tools and Technologies used

List all software that are used to develop and needed to operate the developed module are detailed below.

### Arduino IDE v1.8.9

Code environment in which all code for Board Maker Transmitter and Receiver is written. This IDE is numerously used as a debugging tool as well.

### Processing v3.5.3

This tool is used for debugging and visualization of Board Marker Transmitter as a teapot object. In order to view Board Marker Transmitter and verify the placement of MPU-6050 orientation sensor and latency, we visualized the teapot object moving in the window of Processing software. Following parameters and properties are visualized and debugged.

* Correct orientation data packet format of Board Marker Transmitter.
* Generation of noise with respect to obstacles and distance involved while data transmission.
* Latency in data transmission with respect to obstacles and distance involved while data transmission.

Sample image of object is given below

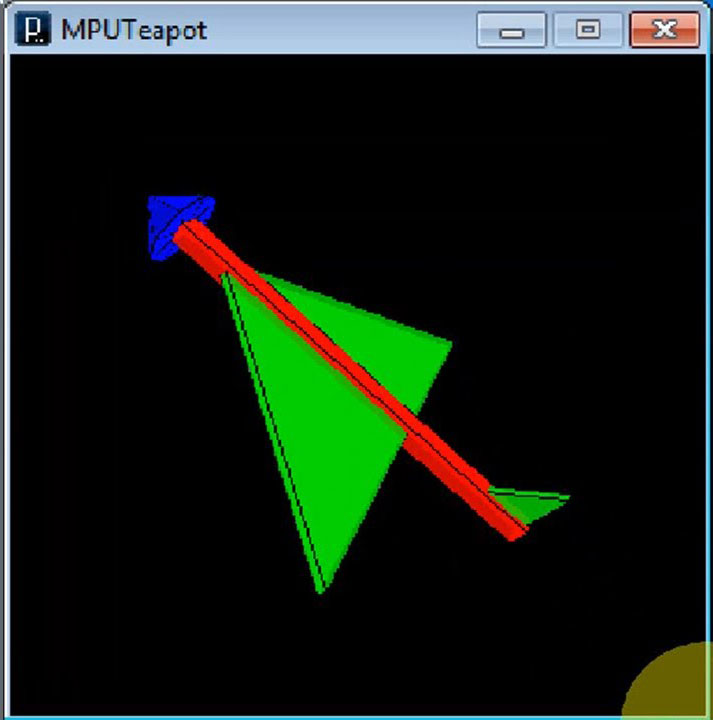


Figure 5.17: Marker Orientation Image in Processing Software

# Audio Hardware

Role of Audio Hardware is to establish a wireless voice communication between Teacher and Controller Application. The module wirelessly transmits the voice of Teacher to controller app. The module works on 2.4G frequency band approved and approved by RoHS.

## Requirements Addressed

|  |  |  |
| --- | --- | --- |
| **#** | **Requirement** | **Priority** |
|  |  |  |
| **1** | Transfer Voice data from one point to another. | HIGH |
| **2** | Transfer Voice data from transmitter to receiver wirelessly. | HIGH |
| **3** | Transmitter should be standalone in terms of power. | HIGH |
| **4** | Receiver should output voice data as analogue audio wave. | MEDIUM |
| **5** | Implement noise control knob in transmitter. | MEDIUM |

Table 5.3: Audio Hardware Requirements

## General Flow

* Transmitter try to connect to the Receiver
* After a successful connection, Transmitter reads analogue signal and converts it into digital PWM wave.
* Transmitter then starts transmitting the voice data through nRF24L01 module.
* Voice data arrives at nRF24L01 of Receiver.
* Receiver converts the incoming signal into audio wave.

## Detailed Design

Audio hardware has two major sub-modules named as Transmitter and Receiver

|  |  |  |
| --- | --- | --- |
| **Component Number** | **Description** | **Diagram** |
|  |  |  |
| **1** | **Name:** Electret Microphone | Figure 5.18: Electret Microphone |
| **Detail:** 9767 Condenser Electret Microphone used to capture voice. |
| **2** | **Name:** 100KΩ Resistor | Figure 5.19: 100K Resistor |
| **Detail:** Used to adjust input gain of microphone. It is connected with the Microphone Circuit. |
| **3** | **Name:** Microphone Circuit | No Image |
| **Detail:** An electric circuit implemented on a dotted Veroboard. It transfers the voltage change due to microphone to the Arduino nano mainboard |
| **4** | **Name:** Input Audio Socket | Figure 5.20: Audio Jack |
| **Detail:** 3.5mm Audio Socket that is used to input the audio wave. It acts as mono input audio channel. |
| **5** | **Name:** Output Audio Socket | Figure 5.20: Audio Jack |
| **Detail:** 3.5mm Audio Socket that is used to output the audio wave. It acts as mono output audio channel. |
| **6** | **Name:** Arduino nano | Figure 5.4: Arduino nano |
| **Detail:** Arduino nano acts as main processing board to which all modules and sensors are attached. It acts just like a motherboard with central processor chip soldered on mainboard. |
| **7** | **Name:** nRF24L01 Adapter | Figure 5.21: nRF24L01 Adapter |
| **Detail:** 5V to 3.3V nRF24L01 adapter gives constant 3.3V from input 5V. It prevents nRF24L01 module not to drain power from Arduino nano mainboard. |
| **8** | **Name:** Noise Reduction Circuit | No Image |
| **Detail:** The circuit is used to reduce random noise with the help of gradual grounding the input audio wave. |
| **9** | **Name:** nRF24L01 | Figure 5.22: nRF24L01 Antenna Version |
| **Detail:** nRF24L01 is a single chip radio transceiver. It is responsible for transmitting voice data from transmitter module to receiver. |

Table 5.4: Audio Hardware Components Detail

## Audio Transmitter

The objective of the Audio Transmitter is to get voice data from microphone and transmit it to the Audio Receiver. After getting the data from microphone, it converts analogue audio data into a digital Pulse Width Modulation or PWM wave.

### Components Used

Detailed description of electronic components excluding discrete consumer parts, e.g. wires, is given below and described in Table 5.4.

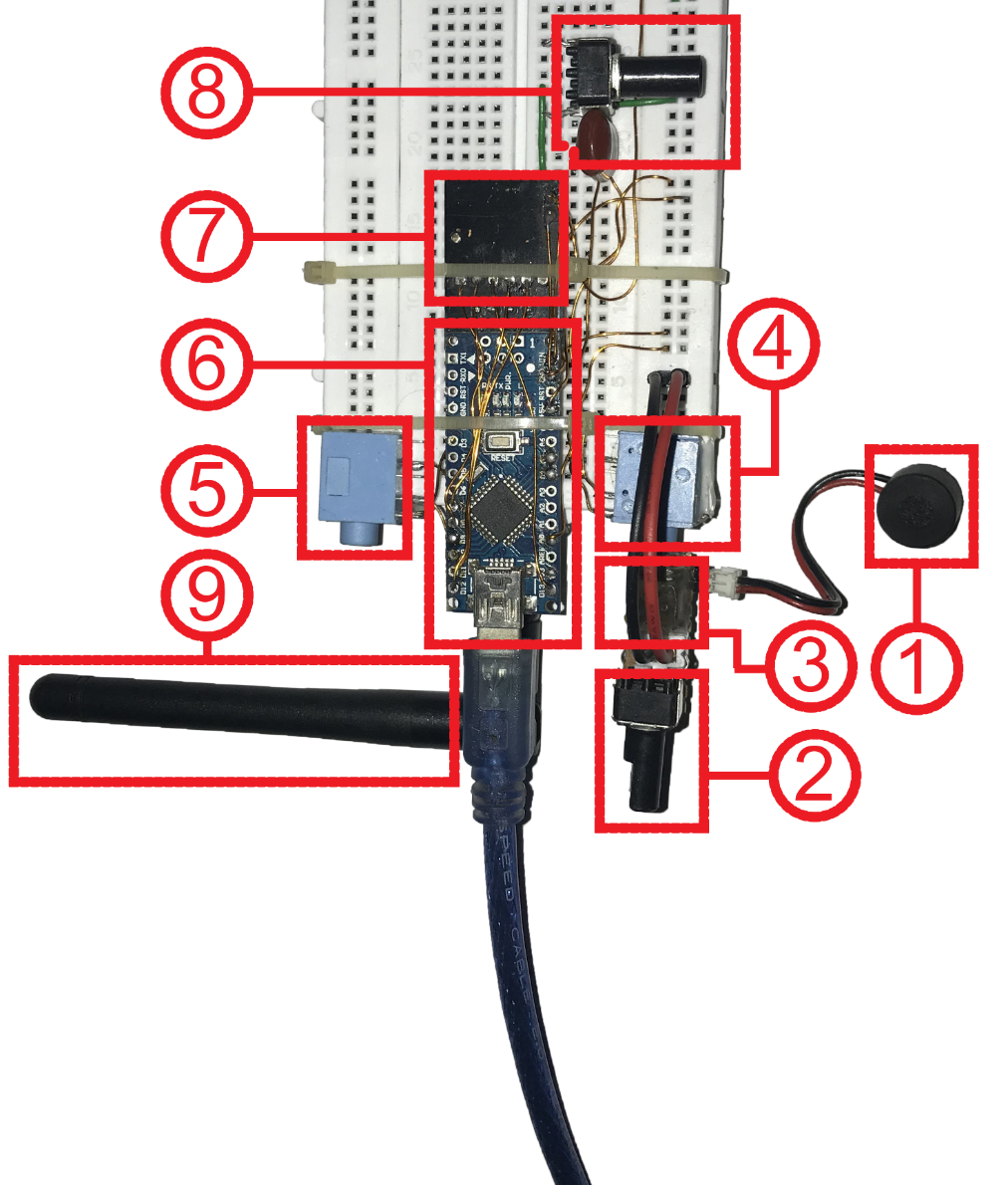


Figure 5.23: Audio Transceiver Component Detail

### Component Connection Diagram

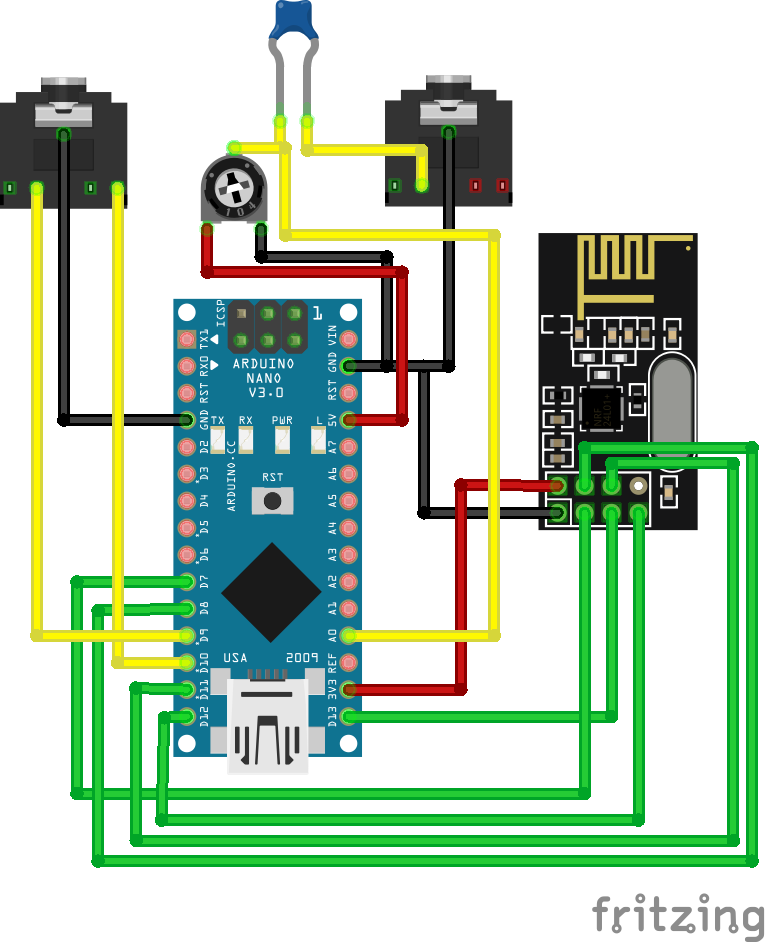
This diagram represents how sub-modules or components are connected in Transceiver module. 

Figure 5.24: Audio Transceiver Component Connection Diagram

### Schematic Diagram

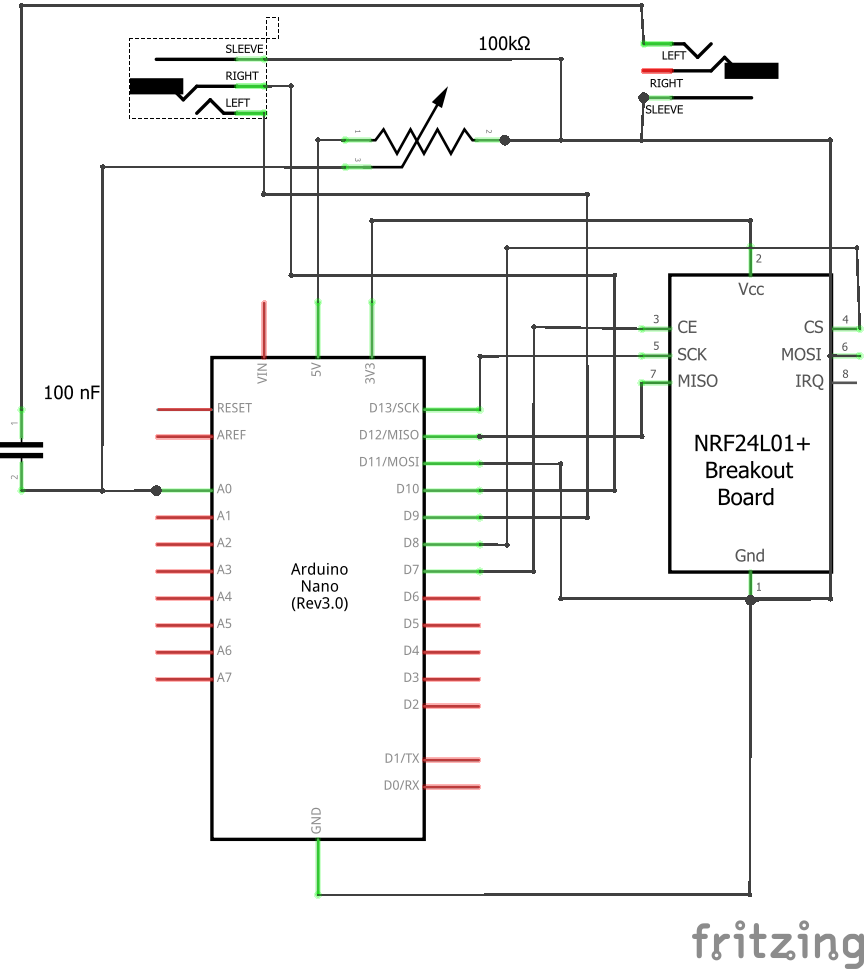


Figure 5.25: Audio Transceiver Schematic Diagram

### General Flow

Figure 5.26: General Flow of Audio Transmitter

## Audio Receiver

Receiver module receives audio data from transmitter. Although the Arduino nano mainboard is programmed differently but the circuit and composition of receiver module is identical to transmitter module.

### Components Used

Detailed description of electronic components excluding discrete consumer parts, e.g. wires, is given below and described in Table 5.4.

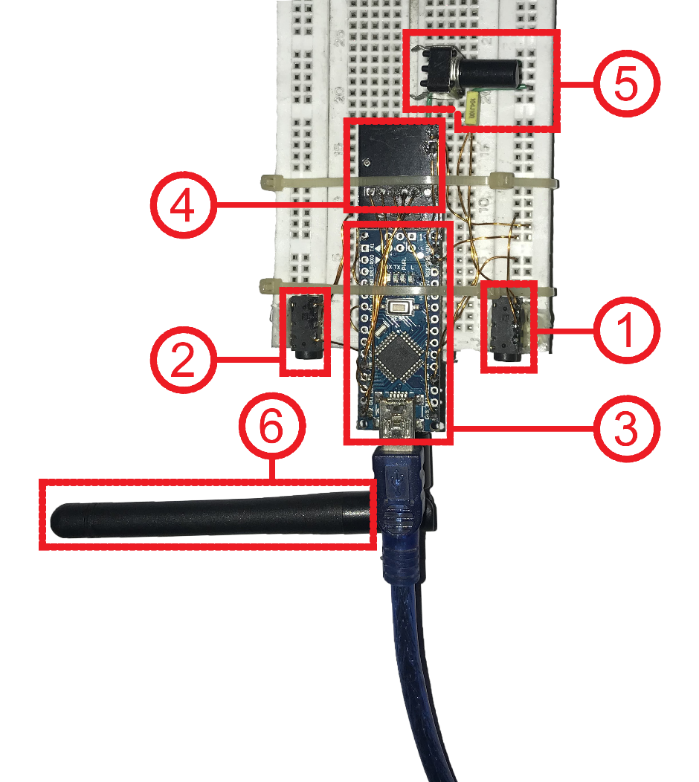


Figure 5.27: Component Diagram of Audio Receiver

### Components Connection Diagram

Same as Audio Transmitter

### Schematic Diagram

Same as Audio Receiver

### General Flow

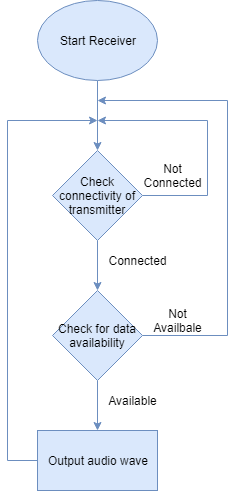


Figure 5.28: General Flow of Audio Receiver

## Rules and Assumptions

Following are rules and cases of assumptions that are assumed to be true while normal working

* Audio Transmitter and Audio Receiver must be in range of 5 meters.
* Microphone should be in range of 10cm from the audio source i.e. speaker’s mouth.

## Tools and Technologies used

List all software that are used to develop and needed to operate the developed module are detailed below.

## Arduino IDE v1.8.9

Code environment in which all code for Audio Transmitter and Audio Receiver is written. This IDE is numerously used as a debugging tool as well.

# Controller Application

This module controls the recording of lecture. It acts as a receiver end from the Marker and Audio Hardware point of view. It generates a lecture file with .dbm extension. That file is then uploaded to central server. Lecture file now can be played on the website using WebGL player or Offline player application.

Complete description along with UI screens of controller application are given below

## Home Screen

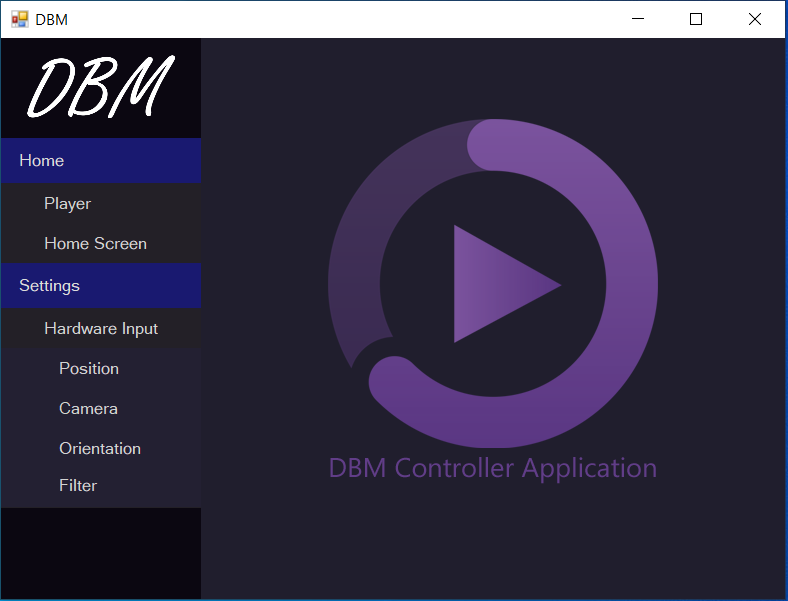


Figure 5.29: Home Screen Controller Application

Controller application has navigation bar on left side that control panel and sub-panels that contain buttons. Buttons navigate to the corresponding form. A hierarchy that describe the categorization of forms is given below.

1. Home Panel
   1. Player
   2. Home Screen
2. Settings Panel
   1. Hardware Input Sub-Panel
      1. Position
      2. Camera
      3. Orientation
      4. Filter

**Home Screen:** Home Screen button is clicked; home screen of Controller Application is shown. It acts just as splash indicating that application is running fine. Application show home screen by default on launch.

## Record Panel Screen

When player button is clicked, application opens the lecture recording control panel. This panel contain the following controls.

1. Seek bar: Indicates the current playing time of the lecture video. It also indicates how much video as passed and how much is remaining.
2. Start Recording button: Lecture recording is started after pressing this button.
3. Stop Recording button: Lecture recording is stopped after pressing this button.
4. Start Playing button: Plays the recorded lecture.
5. Duster button: Press it when you want to erase a little.
6. Clear button: Press it when whole board is to be erased.
7. Enable Device Input button: Press it to enable Marker hardware and Audio hardware input to controller application for lecture recording. When disabled, mouse input is considered i.e. you will be using mouse to write on canvas.
8. Save File button: To save the recorded lecture file. It will generate a file with .dbm extension.
9. Load File button: To load the recorded lecture file. It will open a file browser from which a file with .dbm extension is selected.
10. Thickness Trackbar: Controls the line thickness of writing being written and also duster size.
11. Tick Resolution numeric up down: Controls the refresh rate of Canvas. Lower the value, higher the resolution and hence better quality but more storage required.
12. Color picker: Change the color of writing. In case you are using multiple board markers with different colors.
13. Position textbox: Indicates the current position of pointer on the Canvas.
14. Load Saved button: Load the saved settings on local storage.
15. Save Settings button: Save the settings from local storage.
16. Canvas: Black window on which words are being written i.e. Board.

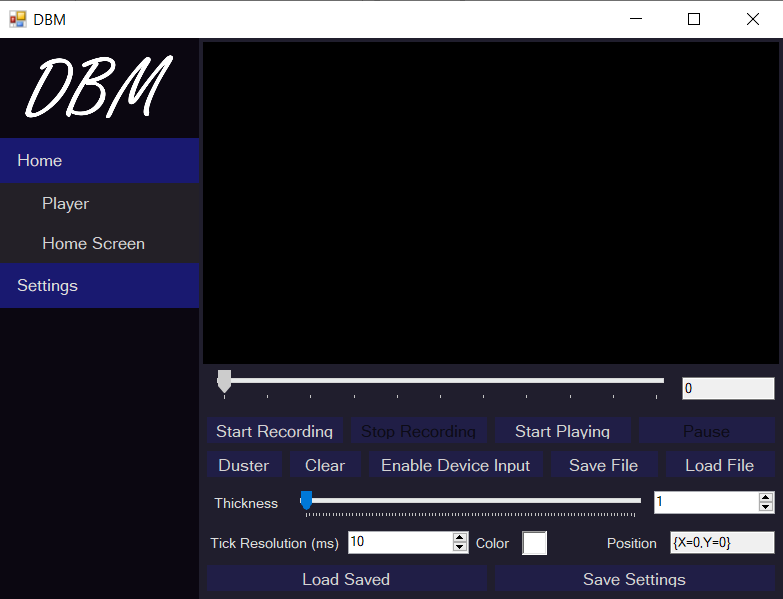


Figure 5.30: Controller Application Record Panel

## Position Calibration Screen

When Settings->Hardware Input->Position from navigation bar is clicked, Position Calibration form is displayed. It contains the following controls

1. Canvas: A container that holds visual elements of either side of board. It contains the lines of sight of each camera and red circle indicates the glowing ball position of the marker. The blue dot indicates the tip position of the marker.
2. Invert Left button: Inverts the control of left-side camera.
3. Invert Right button: Inverts the control of right-side camera.
4. Position textbox: Indicates the current tip position of the Marker Hardware.
5. Load button: Load the saved settings from local storage.
6. Save button: Save settings on local storage.

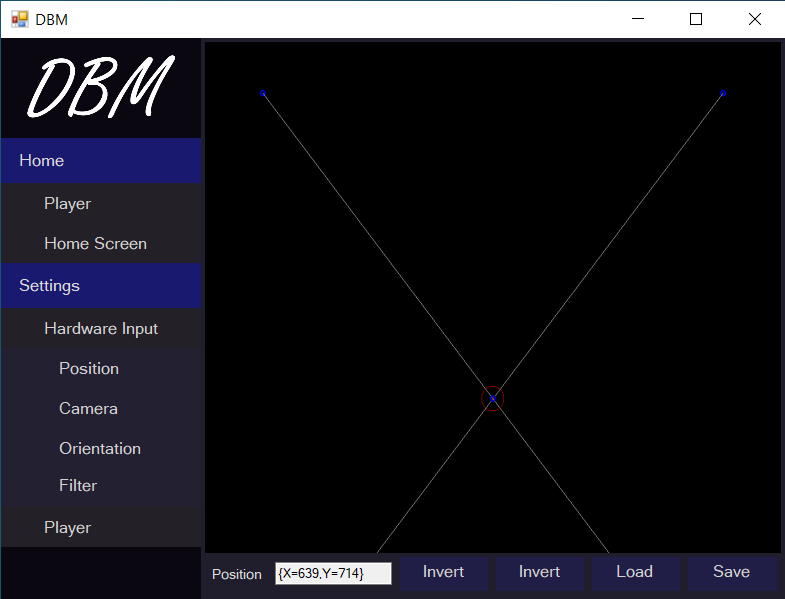


Figure 5.31: Controller App Position Settings

## Camera Control Panel Screen

When Settings->Hardware Input->Camera from navigation bar is clicked, Camera Control Panel form is displayed. Role of this form is to start camera stream and preview in the form window from connected cameras. It contains at least two sub-forms that contain settings of each camera. Each camera form contains the following controls

Select Camera dropdown: Contains list of all connected cameras from which any of the cameras can be selected. Choose carefully i.e. what camera is placed on Left or Right on the board.

FPS textbox: The framerate of camera. Higher the framerate, better the performance. Framerate should not exceed from the mentioned framerate on camera.

Capture button: Start the camera capture.

Preview button: Camera preview is displayed on the preview box below.

Load Saved button: Load saved settings from the local storage.

Save Settings button: Save the settings on local storage.

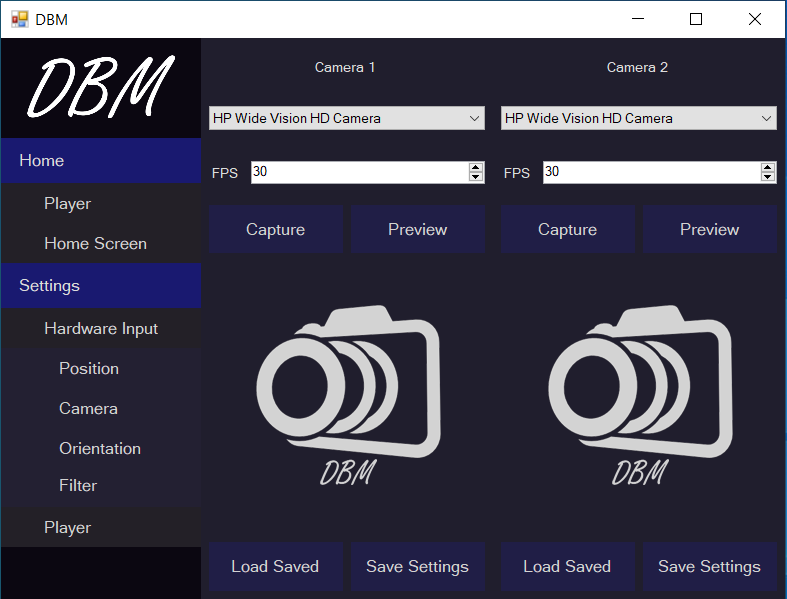


Figure 5.32: Controller App Camera Settings

## Orientation Calibration Screen

When Settings->Hardware Input->Orientation from navigation bar is clicked, Orientation Control Panel form is displayed. Role of this form is to calibrate and visualize the orientation of the board marker. It contains the following controls

Start button: Starts the orientation input from Marker Hardware.

Refresh button: Refreshes the list of available communication serial ports.

Serial Port dropdown: Dropdown list of available serial ports. Choose the one with which Marker Hardware is connected.

Current Orientation group-box: Shows X-axis, Y-axis and Z-axis rotation of Marker Hardware along its pivot point.

Current Pressure textbox: Shows the amount of pressure applied on the tip of Marker.

Tip Offset textbox: Shows the calculated tip offset from the origin point as compared to when Marker was kept at 90 degree with the Board.

Apply Offset group-box: Trackbars that give offset to calculated angles alone X-axis, Y-axis and Z-axis respectively to calibrate the orientation.

Load Saved button: Load saved settings from the local storage.

Save Settings button: Save the settings on local storage.

Show 3d Orientation button: Opens another form that show orientation of Marker in 3d space.

Show 2d Orientation button: Opens another form that show orientation of Marker in 2d plane.

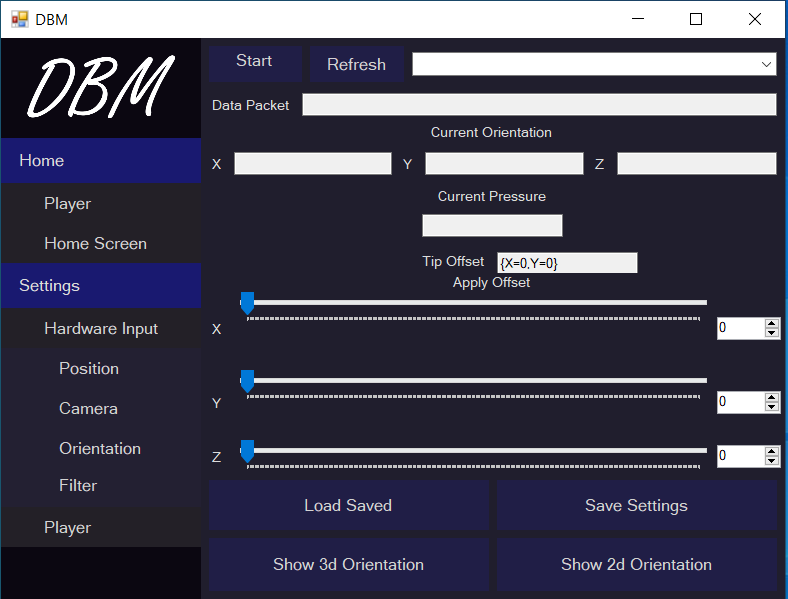


Figure 5.33: Controller App Orientation Calibration

## 3d Orientation Screen

When ‘Show 3d Orientation’ button from Orientation Calibration Form is clicked, 3d Orientation form is displayed. It shows orientation of marker in three-dimensional space. Box of 3d model show tail of the marker and conical head show tip of marker.

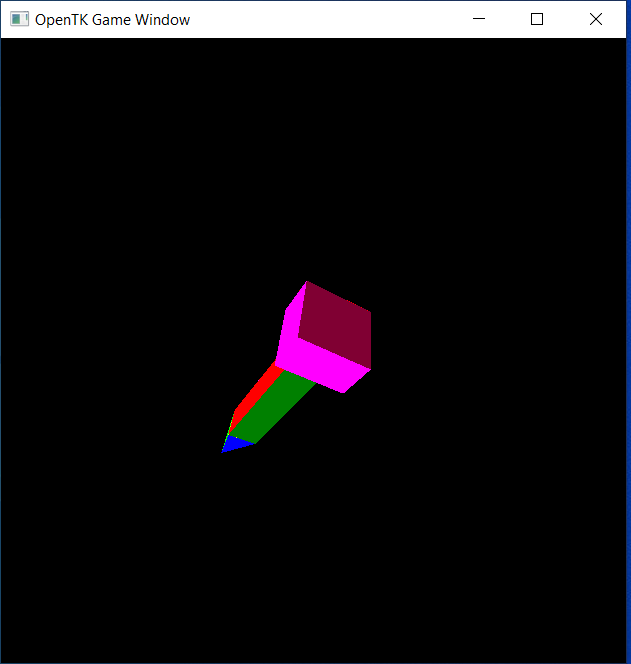


Figure 5.34: Controller App Orientation Visualization in 3d Space

## 2d Orientation Screen

When ‘Show 2d Orientation’ button from Orientation Calibration Form is clicked, 3d Orientation form is displayed. It shows orientation of marker in three-dimensional space. Box of 3d model show tail of the marker and conical head show tip of marker.

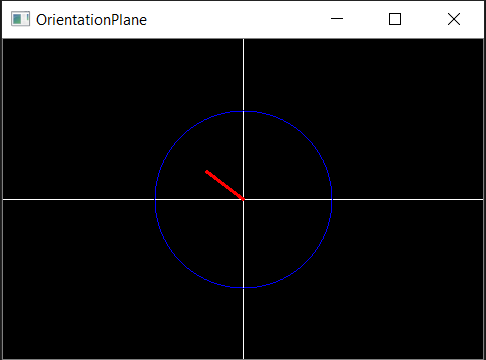


Figure 5.35: Controller App Orientation Visualization in 2d Space

## Filter Control Panel Screen

When Settings->Hardware Input->Filter from navigation bar is clicked, Filter Control Panel form is displayed. Role of this form is to perform computer vision algorithms and filters to get the position of the glowing ball of the board marker. It contains at least two sub-forms that contain settings of each camera filter. Each filter form contains the following controls

Upper Limit group-box: Contains three trackbars that is used to adjust the upper limit of color filter in HSV color space.

Lower Limit group-box: Contains three trackbars that is used to adjust the lower limit of color filter in HSV color space.

Preview button: Launches the filter preview form.

Load Saved button: Load saved settings from the local storage.

Save Settings button: Save the settings on local storage.

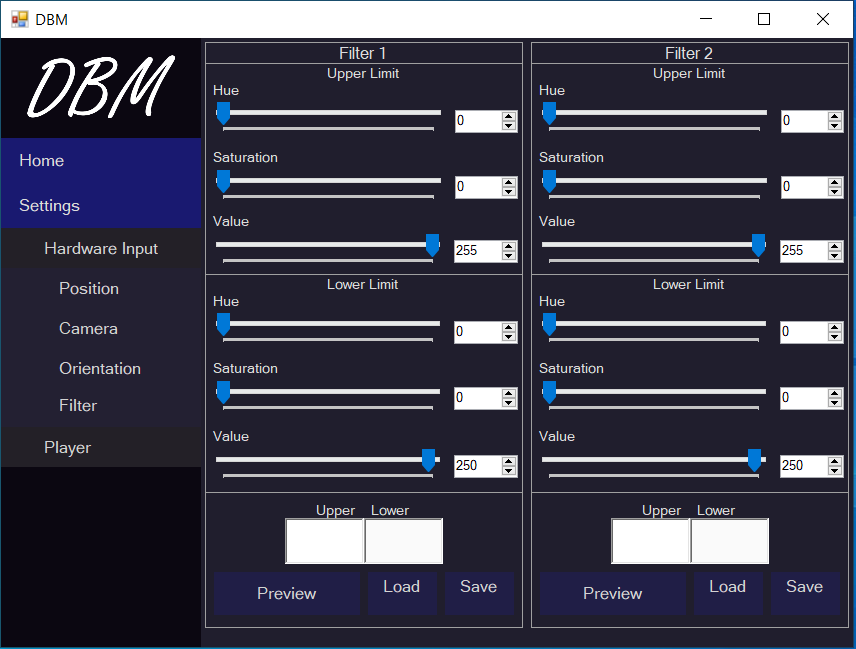


Figure 5.36: Controller App Camera Filter Control Panel

## Filter Preview Screen

When ‘Preview’ button from Filter Control Panel Form is clicked, 3d Orientation form is displayed. Role of this form is to display the mask applied to camera to extract the position of glowing ball. It contains the following controls

Offset textbox: Displays the relative position of glowing ball with respect to mid. It displays the value in percentage.

Show Mask: Toggle show filter and normal preview.

Undo button: undo last placed dot on this preview screen.

Preview Screen: Dots i.e. particularly three, can be placed. First two dots display a line that is aligned with the surface of Board. Third dot display the corner of the board just ahead of the camera.

Load Saved button: Load saved settings from the local storage.

Save Settings button: Save the settings on local storage.

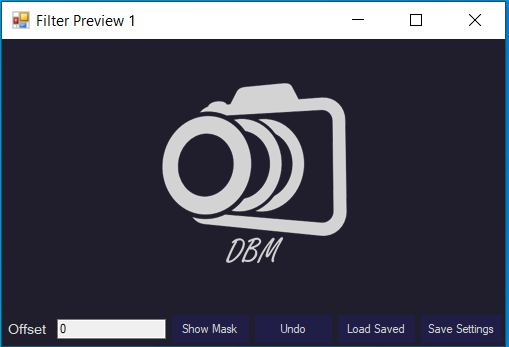


Figure 5.37: Controller App Camera Filter Preview

## Flow Diagram

The general flow diagram of controller application is given below

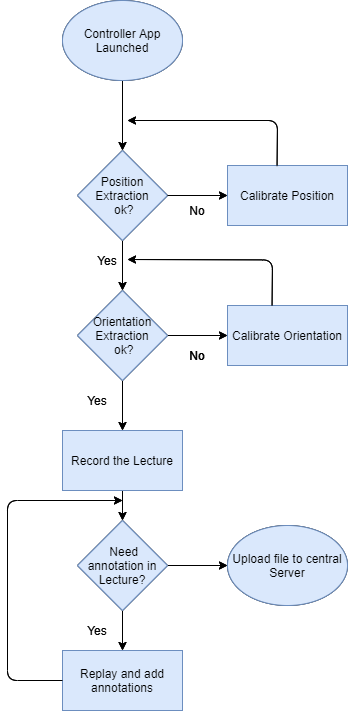


Figure 5.38: Controller App Flow Diagram

## Rules and Assumptions

Following are rules and cases of assumptions that are assumed to be true while normal working

* Application is not closed while recording otherwise all recording will be wasted.

## Tools and Technologies used

List all software that are used to develop and needed to operate the developed module are detailed below.

### Visual Studio 2019

Code environment in which all code for Controller application is written. This IDE is numerously used as a debugging tool as well.

#### Libraries Used

Following external libraries are used while development of controller application

* Newtonsoft.Json v12.0.3: To serialize and deserialize data into and from Json objects.
* ColorMine v1.1.3: For Inter-conversion of color spaces i.e. RGB and HSV
* EMGU.CV v4.1.1.3497: OpenCV image processing library used in capturing and filter formation.
* DirectShowLib v1.0.0: Used to get list of all connected cameras.
* OpenTK v3.1.0: Used to render 3d model of marker for orientation calibration.

# Player Application

This module acts just like regular media player that play video file. The difference is that it plays a lecture file only, generated by controller application. File if of .dbm extension.

Player application has navigation bar on left side that control panel and sub-panels that contain buttons. Buttons navigate to the corresponding form. A hierarchy that describe the categorization of forms is given below.

1. Home Panel
   1. Player
   2. Home Screen
2. Settings Panel
   1. Player Settings

Complete description along with UI screens of controller application are given below

## Player Screen

When ‘Player’ button from navigation bar is clicked, Player screen is displayed. It contains the following controls

1. Seek bar: Indicates the current playing time of the lecture video. It also indicates how much video as passed and how much is remaining.
2. Start Playing button: Plays the recorded lecture.
3. Load File button: To load the recorded lecture file. It will open a file browser from which a file with .dbm extension is selected.
4. Thickness Trackbar: Controls the line thickness of writing being written and also duster size.
5. Load Saved button: Load the saved settings on local storage.
6. Save Settings button: Save the settings from local storage.
7. Canvas: Black window on which words are being written i.e. Board.

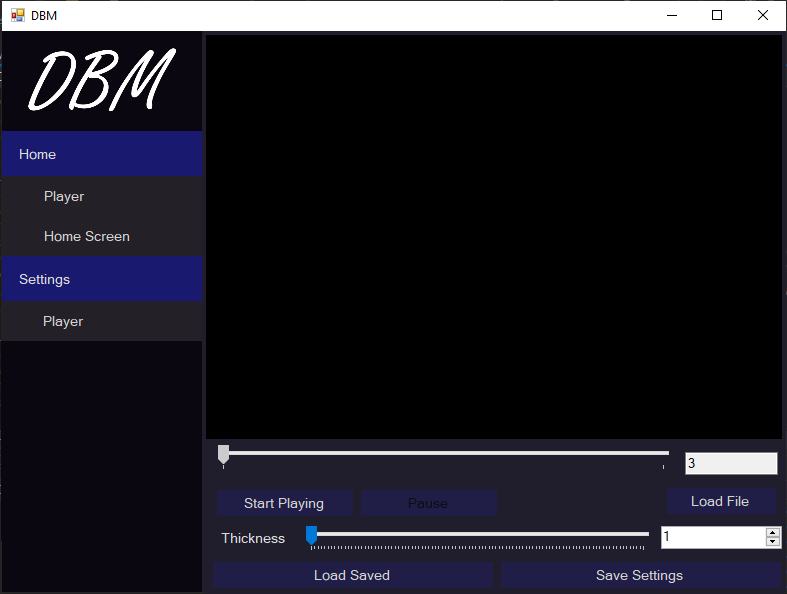


Figure 5.39: Player Application Main Screen

## Flow Diagram

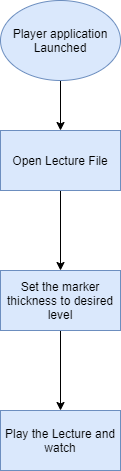


Figure 5.40: Player App Flow Diagram

## Tools and Technologies used

List all software that are used to develop and needed to operate the developed module are detailed below.

### Visual Studio 2019

Code environment in which all code for Controller application is written. This IDE is numerously used as a debugging tool as well.

#### Libraries Used

Following external libraries are used while development of controller application

* Newtonsoft.Json v12.0.3: To serialize and deserialize data into and from Json objects.
* ColorMine v1.1.3: For Inter-conversion of color spaces i.e. RGB and HSV
* EMGU.CV v4.1.1.3497: OpenCV image processing library used in capturing and filter formation.