INTEGRATION AND IMPLEMENTATION OF HARDWARE AND SOFTWARE OF PCG
INTEGRATION AND IMPLEMENTATION OF
HARDWARE AND SOFTWARE OF PCG

The integration and implementation of hardware and software the system[1] involve the steps of bringing together the component subsystems into one system and it is a method of linking together different computing systems and software applications physically or functionally, together to work as one system. In this work, PCG hardware[2] is used to get the sound signal obtained from the heart and waveform of the sound signal is to be used in the analysis of murmur classification. Computer with Matlab software is integrated in the whole system to classify murmurs using hamming distance technique.

Then comes smart phone to capture image form of the sound signal, and then to classify various types of heart sound and murmurs. This part of the work gives the result of heart murmur classification with the help of augmented reality (AR)[3] technique. Augmented reality (AR) is the integration of digital information with live video or the user's environment in real time. Some of the hardware components used in augmented reality are processor, display, sensors and input devices. Mobile computing devices like handy smart phones and tablet computers are containing a camera and sensors, and GPS are making them very suitable augmented reality platforms.

5.1 Hamming distance

The heart murmur signal is an acoustic signal and presents artifacts when its path is tracked in a non-optimal manner. The sound signal is very similar to the light wave and takes the shortest path when travelling from one region to another. The heart murmur occurs inside
the heart chamber (mechanical model of the heart) but is sensed external to it using a microphone or stethoscope. This difference in the distance between the occurrence point and the sensing point itself is an artifact at the sensing point. Thus, further transforms acting on the sensed signal to extract the features out of it is like an invasive mathematical operation on the data and increases the artifacts. Here, non-invasive mathematical approach for analyzing the data from the sensing point using image based approach is presented. Our approach of content extraction in this research work searches for the content in the input rather following traditional text mining techniques and translational issues.

First heart sound inputs from clinically important locations [from patients] on the chest are acquired using an electronic stethoscope and transferred to a database and murmur classifications [using Hamming distance] is done. The classifier output can be used to check the abnormality of the heart valve. In this method, with the help of a reference input and from the acquired murmur input time and spatial domain information is extracted (refer figure 5.1).
5.2 Conventional Approach to heart murmur signal based analysis

5.2.1 Heart murmur signal behavioral model

The velocity of motion[7] of heart murmur signal (in any direction) in the upper half-plane \(y > 0\) (internal to heart chamber) (of the plane with Cartesian coordinates \((x,y)\)) is equal to \(v\). In the lower half-plane (in the path takes from heart chamber to microphone sensor (externally located)) \(y < 0\) the velocity of motion is \(w = nv\) (if ‘\(v\)’=1 and \(w = ‘n’\) for heart murmur propagation in \(y>0\) and \(y<0\), respectively). There exists a least-time path DEF from a point D in the upper half-plane to a point F in the lower half-plane and this is DEF that breaks at the point E on the heart to sensor.
The angle $\theta$ of a heart murmur beam incident on a spherical surface of radius $r$ at a distance $x$ from the ray OD passing through the center of the surface and parallel to the heart murmur beam deviate from returning along the incidence direction (refer figure 5.3).

The angle XOZ is equal to $\theta_2-(\theta_1-\theta_2) = 2\theta_2-\theta_1$

The deviation angle $\theta$ is twice as large (due to the symmetry in the OX axis):

$\theta = 4\theta_2-2\theta_1$

According to the law of refraction, we have $\sin \theta_1 = n \sin \theta_2$, and by the definition of the incident ray, $r \sin \theta_1 = x$. Therefore, $\theta_1 = \arcsin(x/r)$, $\theta_2 = \arcsin(x/(nr))$, and

$\theta (x) = 4\arcsin(3x/4r) - 2 \arcsin(x/r)$

This expression can be used to plot the graph of the calculated function $\theta$. 

**Figure 5.2: Behavioral model of Heart murmur**
5.2.2 Maximal Deviation Angle of a heart murmur beam

The heart murmur beams[8] incident on a spherical surface deviates from returning along itself by a maximal angle $\theta_{\text{max}}$.

Assume $3x/(4r) = u$, then $x/r = nu$ and $\theta/2 = 2\arcsin u - \arcsin nu$.

The derivative of $\theta/2$ with respect to $u$ must vanish for a beam with maximal deviation angle $\theta_{\text{max}}$: for this beam, we have

$$2 \frac{2}{\sqrt{1-u^2}} = \frac{n}{\sqrt{1-n^2u^2}}, \quad 4 \frac{4}{1-u^2} = \frac{n^2}{1-n^2u^2},$$
\[\text{--------------------------- (5.1)}\]

So that

$$u_{\text{max}}^2 = \frac{4-n^2}{3n^2}, \quad \frac{\theta_{\text{max}}}{2} = 2\arcsin u_{\text{max}} - \arcsin nu_{\text{max}}$$
\[\text{--------------------------- (5.2)}\]

For $n = 4/3$, we find

$$u_{\text{max}}^2 = \frac{5}{12}, \quad u_{\text{max}} = \frac{\sqrt{5/3}}{2}, \quad nu_{\text{max}} = \frac{\sqrt{5/3}}{2}, \quad nu_{\text{max}} = \sqrt{5/3} \cdot \frac{2}{3}.$$  
\[\text{--------------------------- (5.3)}\]

Since $5/3 = 1.666$, we readily calculate...
\[
\sqrt{\frac{5}{3}} = \sqrt{\frac{166.6}{10}} = 1.29
\]  

---

Thus,

\[ u_{\text{max}} = 0.645, \quad n_{u_{\text{max}}} = 0.86 \]

Since

\[ \sin\left(\frac{\pi}{6}\right) = 0.5, \quad \sin\left(\frac{\pi}{4}\right) = 0.707, \quad \sin\left(\frac{\pi}{3}\right) = 0.86, \]

It follows that

\[ \arcsin u_{\text{max}} = \frac{\pi}{3}, \quad \arcsin u_{\text{max}} = \frac{\pi}{4} - \frac{\pi}{40}, \]

then \[ \theta_{\text{max}}/2 = \frac{\pi}{2} - \frac{\pi}{20} - \frac{\pi}{3}, \quad \theta_{\text{max}} = \frac{\pi}{3} - \frac{\pi}{10}, \]

which is about \(42^\circ\) \[ \]

---

5.2.3 Ideal case of heart murmur signal

Ideally, a heart murmur ray starting[9] at W (figure 5.4) and striking a surface g, is reflected towards X in such a way that the angle of incidence and the angle of reflection are equal. Under ideal conditions the heart murmur signal travels from the heart chamber to the sensor along the path WZX, such that the chosen path is the shortest of all possible paths from W to X that touch the heart chamber wall to the sensor interface ‘g’.
The heart murmur treated as a sound signal seldom takes the optimal path and once acquired, subsequent operations in the form of feature extraction, mathematical transforms etc. is equivalent to introducing artefacts and has in effect a non-optimal path from the heart chamber external wall to the sensing node. This creates an inaccuracy in the classification. To overcome the limitation of this approach, in this work, an augmented reality based detection combined with image processing is used. This is a mathematically non-invasive approach in analyzing and classifying the heart murmur signal.

5.3 Content extraction studies

5.3.1 Pattern Matching

Pattern matching[4] is used to compute similarities between images and prototypes, which can be treated with transformations like translation and rotation. Pattern matching[5][6] techniques are vital in analyzing document images and web documents. Documents in general are scanned into a binary image, and then any operations are performed, like page segmentation, character identification, erosion, dilation of binary image morphology. They can be used for extracting pixel aggregations. We can use them in image processing and for subsequent analysis.
These are in general translational invariant. We can use them for treating both foreground and background simultaneously irrespective of connected component analysis. Further, there exist a variety of methods for controlling the noise immunity of these operations. One of the most important uses of pattern matching is in the analysis of character shapes. For binary input, the result of image processing can be either binary or gray scale images. Binary results are much faster to compute, but they contain less information. Even for binary output, the internal operations can be integer or Boolean. For integer operations, such as convolution and threshold convolution, noise immunity in general is attained in performing costly arithmetic operations on each of the pixel.

5.3.2 Distance Measures and Alignments

The Hamming distance calculation rigidly assumes that the \(i^{\text{th}}\) symbol of one sequence is already aligned against the \(i^{\text{th}}\) symbol of the other. However, it is often the case that the \(i^{\text{th}}\) symbol in one sequence corresponds to a symbol at a different-and unknown-position in the other. To differentiate the falling and sitting scenarios from the captured video frame, the concept of total distance is used. It is appropriate to use total distance and make use of the three operations delete, insertion and substitution[10] to map the received frame with the reference frame.

5.3.3 Methodology:

1. Measure the total distance of the received frame with respect to the reference frame. This is defined as the number of operations required to match the pixels of the received frame with the reference frame within a threshold value.

2. Compute the minimum of the total distance \(ed_1\) & \(ed_2\) and perform decision.

\[
\therefore ed_1 = \sum_{i=1}^{3} \sum_{j=1}^{3} [P_{l/p}(i,j) - P_{ref}(i,j)]
\]

-------------------------- (5.8)
and

\[ e_{d_2} = \sum_{i=1}^{3} \sum_{j=1}^{3} [p_{i/p}(i,j) - P_{ref_{2}}(i,j)] \]

...(5.9)

& \min(e_{d_1}, e_{d_2}) \text{ sub-blocks decide the closest match of the captured frame with the individual reference frames.}

Illustrations:

Let the (i,j) sub blocks of k\textsuperscript{th} frame is given below

\[
\begin{array}{ccc}
128 & 127 & 132 \\
119 & 118 & 128 \\
121 & 122 & 126 \\
\end{array}
\]

3*3 sub block of k\textsuperscript{th} frame

Let the sub block of k\textsuperscript{th} reference frame for scenario-I is given below:

\[
\begin{array}{ccc}
128 & 127 & 131 \\
119 & 118 & 127 \\
121 & 122 & 12 \\
\end{array}
\]

Similarly, for reference frame-II, the pixel values for the sub block is given below:

\[
\begin{array}{ccc}
112 & 125 & 119 \\
122 & 127 & 117 \\
121 & 118 & 126 \\
\end{array}
\]

5.3.4 Algorithm

For k\textsuperscript{th} scanned image

1. Read i\textsuperscript{th} block values

2. Compute Hamming distance for median string of length ‘l’
3. Compute the minimum Hamming distance of (2)
4. Repeat (1) to (3) for (i+1)th block to cover all sub-blocks of the scanned image.
5. Obtain the vector representing the minimum Hamming distance of all the NxN sub-blocks of the kth scanned image.
6. Repeat steps (1) to (5) for other images representing the same context.

5.3.5 Decision algorithm

From the individual vectors for the scanned images obtained using the Total distance finding algorithm, a decision threshold is chosen to determine a correlated consensus value. As mentioned in Table-2 (Refer Annexure), Poor correlation indicates identical number of zeroes in the sub blocks between two pixel-maps. For example, using this different classes of PCG signals are packed in separate bins.

Table-2 Comparison of $\Delta x$ with x

From Table-2 (Refer Annexure), the complementary total distance $\overrightarrow{d}$ is computed and the image with $\overrightarrow{d}$ above a set threshold (in this work 5) is grouped into the same bin. The sub-block-I ‘$x$’ values of all three images are shown below:

Image$_1$Subblock – ‘$x$’ value = \{0, 0, –3, –176, 0, 0, 0, 32\}

Image$_2$Subblock – ‘$x$’ value = \{$0, –163, 57, –141, 49, 45, 89, 89\}$

Image$_3$Subblock – ‘$x$’ value = \{0, 0, –3, –140, 0, 0, 7, 140\}
The calculated complementary edit distances are

\[ \overline{e_d} \text{ Between Image}_1 \& \text{ Image}_2 = 7 \]

\[ \overline{e_d} \text{ Between Image}_2 \& \text{ Image}_3 = 3 \]

### 5.4 Implementation of the work

Multiple scanned images of uniform size with either matching[11] or non-matching content are given as input. In the first approach, the pixel values of individual images are treated as pairwise distance between points and the positions of those points are reconstructed using the partial digest problem solution approach. From the knowledge of set of distances between every pair in a sub block of kxk pixels [in this case ‘k’ is chosen as 5], the actual distance is extracted i.e., ‘x’ from \( \Delta x \) in the second approach, the matching based position vector extraction is done and using the total distance metric, the decision with respect to the content, bin packing is done. Both the approaches show striking similarity and the elements of the bin are formed with high accuracy.

#### 5.4.1 Partial Digest Algorithm

Using PDA, one set ‘x’ is derived for a’Δx ‘of length ‘L’ for each of the sub-block of size K x K, for every image. The methodology is given below.

```
Image
↓
Extract Sub-blocks
↓
For every Sub-block extract Δx
↓
```
For every \( x \) of length \( L \), the size of \( \lambda \), is determined from

\[
L = \frac{8}{\log((1-x) \cdot \text{size}(x) \cdot \text{size})}
\]

For a given \( x \) of length \( L \), \( \lambda \) is chosen as 25, so that \( \text{size}(x) \leq 8 \). In this work, \( L \) is chosen as 25, so that \( \text{size}(x) \leq 8 \).
Remove width -y from X and add length \( \Delta(width - y, X) \) from L.

Return.

**5.4.2. Median String Problem**

Given a set of sub-blocks and a search/query median string

Input: An x n matrix sub-block, and l, the length of the pattern to find.

Output: A string \( \nu \) and \( l \) that minimizes \( TotalDistance(\nu, image) \) over all strings of that length.

**5.4.3. Heart murmur Classifier Algorithm**

Let ‘L’ be the length of the Median String & ‘M’ be the size of the sub-block chosen. [More L=8, M=25] Let the positions of the elements in sub-block be \( M_0 \) to \( M_1 \)

1. Find Hamming distance between the median string size of ‘L’ and the elements of ‘M’ with starting element \( M_0 \)

2. Repeat step (1) for elements of ‘M’ but with starting element = \( M_1 \). Repeat the steps above for successive starting position of ‘M’

3. Obtain the position value corresponding to the minimum Hamming distance out of the above

For illustration:

Let Median-string values be \( X_0, X_1, \ldots, X_7 \)

Then

Step 1 Compute HD \( (X_0, M_0; \ldots, X_7, M_7) \)

Step 2 Compute HD \( (X_0, M_1; \ldots, X_7, M_8) \)

Step 3 Compute HD ( )
Compute HD (X₀,M₁₈;............X₇,M₂₅)

5.4.4 Hamming distance evaluation in Augmented reality

Reference vector1.txt
253 253 255 245 254 225 251 253 253 255 245 254 225 251 248 248254 180 25 0
0 219 Reference vector2.txt
253 253 253 245 255 253 253 255 245 254 225 251 255 249 248 248 254 227 25 0 0
219 231
Reference vector3.txt
253 253 245 255 248 253 253 255 245 255 225 251 255 241 248 248 254 227 225 0 0 219
231 237
Acquired vector.txt
253 253 255 245 254 253 251 253 253 253 253 253
Output1.txt

minimum value of non match value for input1:2

position value0:2 position value1:8 position value2:9 position value3:11 position value4:12
position value5:12 position value6:12 position value7:11 position value8:10 position value9:6
position value10:11 position value11:12 position value12:13

number of matching position for input1:1
output2.txt

minimum value of non match value for input2:8
position value0:9 position value1:9 position value2:12 position value3:12 position value4:11 position value5:8 position value6:8 position value7:12 position value8:13 position value9:13 position value10:13 position value11:12 position value12:13

number of matching position for input2: 2

output3.txt

minimum value of non match value for input3: 9


number of matching position for input3: 3

matchfile.txt

matched with input3

250 250 250 250 245 254 254 251 255 254 251 250 255 245 256 250 258 248 250 254 253 253 257 253 219

Referencevector2.txt
Referencevector3.txt

Acquired vetor.txt

Output1.txt

minimum value of non match value for input1:9

position value0:11 position value1:12 position value2:11 position value3:12 position value4:9
position value5:12 position value6:12 position value7:11 position value8:10 position value9:10
position value10:13 position value11:12 position value12:13
number of matching position for input1:1

output2.txt

minimum value of non match value for input2:11

position value0:11 position value1:11 position value2:11 position value3:11 position value4:11
position value5:11 position value6:11 position value7:12 position value8:12
position value9:11 position value10:11 position value11:11 position value12:11
number of matching position for input2:11

output3.txt
minimum value of non match value for input3: 11


number of matching position for input3: 2

matchfile.txt

matched with input2

Reference vector1.txt

Reference vector2.txt

Reference vector3.txt
253 253 245 255 248 253 253 255 245 255 254 225 251 255 241 248 248 254 227 225 0 0 219 231 237
Acquired vector.txt
253 253 253 253 253 253 253 253 253 253 253 253 253 253

Output1.txt
minimum value of non match value for input1:0

position value0:0 position value1:0 position value2:0 position value3:0 position value4:0
position value5:0 position value6:0 position value7:0 position value8:0 position value9:0
position value10:0 position value11:0 position value12:0
number of matching position for input1:13

output2.txt
minimum value of non match value for input2:7

position value0:7 position value1:8 position value2:9 position value3:10 position value4:10
position value5:10 position value6:11 position value7:12 position value8:13 position value9:13
position value10:13 position value11:13 position value12:13
number of matching position for input2:1

output3.txt
minimum value of non match value for input3:9
5.5 APP development details.

5.5.1 Android application development

5.5.1.1 XAMARIN

Xamarin studio is integrated development Environment (IDE) used to create iOS and Android applications.

mainactivity .cs file

It contains a summary sample description, sample log and a Fragment that calls call backs on application.

.axml file

It describes the layout of android application.
There are two layout are used such as linear layout and relative layout. **Linear Layout** is a view group that displays child view elements in a linear direction, either vertically or horizontally. **Relative layout** is a view group that displays child view elements in relative positions. The position of a view specified as relative to elements such as to the left-of or below a given element or in positions relative to the relative layout area such as aligned to the bottom, left of center. A relative layout is a powerful utility for designing a user interface because it can eliminate nested view groups. Several nested linear layout groups may be able to replace them with a single relative layout.

**5.5.1.2 Syntax used in development**

**Intent**

Intent is used to call another function. Some sort of operation performed in the Android OS by intent. Intent used to launch external applications such as make a phone call, display a web page, or map an address. Intent contains what the intent is (as in make a phone call), and data need (such as a phone number) to perform the intention. Start Activity (intent) syntax is used to open webpage in the browser application by intent.

**Delegate**

Delegate is a data structure that refers to a static method or to a class instance and an instance method of that class.

**Findviewbyid**

It finds a view that was identified by the id attribute from XML that was processed in activity. It is used to get id of the resources.

**FileZila**

It is used to upload files like (pdf, word) into web server.
5.5.2 Development of android application

Workflow

i. **Set up**: install android SDK, android development tools and android platforms. Create android virtual devices and connect hardware devices that will be used for testing.

ii. **Development**: create an android project with source code, resource files and android manifest file.

iii. **Debugging and testing**: build and run application in debug mode. Debug application using the android debugging and logging tools. Test application using the android testing and instrumentation framework.

iv. **Release mode**: configure, build and test application in release mode.

After launching xamarin, android application is used to create an android application with an activity. Initially mainactivity.cs created which contain source code for the android application development. In layout, axml file is created automatically. The layout of android application designed in axml file. In axml file linear layout, relative layout and buttons like getimage, submitimage, show pdf are designed. In mainactivity.cs main variables are declared within event handler. Web reference option is used to add web server with xamarin by given already created web address (URL) in it. .Net 2.0 web services is used as framework. Reference.cs file contains the webserver details. AppDemo.URLaddress.com added as header file for webservice. The operation various buttons are designed.

**Get image button code flow**
First total number of reference images is indexed. After, every image displayed on android, imagecount value is incremented by count 1. If image count is equal to total number of reference images, then image count is initiated with value of 0. After that web server is waiting for submit image button function.

**Submit image button code flow**

If user press submit button without image, the android process display “please enter the image” message displayed. User image compared with reference images. If any match occur “match” is displayed and get report button is enabled, otherwise “not match” is displayed.

**Report button code flow**

If report button is pressed by user, the corresponding report (pdf file) is displayed. The report document is get from the designed URL web service. Pdf path of the report document is mentioned.

**Deployment on Android**

After completion of coding, android phone is connect with computer. The developed android application uploaded to android device.

**App as package**

Create package is used to application as package. Minimum android version is selected as API level 14. .apk app package is developed.

**5.6 Generation of the image (conversion of document into jpeg file)**

- Extract the images from document file
- Copy the image in paint (.jpeg format)
- Saving the image in the selective path

Now with the help of movie maker various numbers of videos are clubbed together.
Figure 5.5: Extracting Image from document file

Figure 5.6: Doc file images
Now, before saving the merged videos their bit rates are changed or adjusted to get the proper size and quality of the video.
Where, Total bit rate = video bit rate + audio bit rate

The videos which are to be saved should have the average bit rate i.e., it depends on the total number of videos taken. Thus the final videos saved should have the common bit rate and it will show the estimated file size which is required in the building of the application. The video is stored in the understandable manner and is ready to play.

With the help of wildcard tool i.e. Vuforia which includes good quality image and bad quality image. In the “Vuforia” application, the file of the target image is uploaded and it is marked as the good or bad quality image according to the rating of the target image. As per the rating, if any image gets above three star it is considered as a good quality image, and if the image gets less than three star then it has the poor quality. After the image is passed through the quality checking process it is then downloaded in the Unity editor form. After downloading an icon will appear on the desktop. And the unity app is opened through the desktop.

![Figure 5.11: Downloading of Unity App](image-url)
5.7 Working on the Unity App

Click on file - new project - location

Figure 5.12: Making of new project

Figure 5.13: Target image searching
In the new project click on the assets- import package- custom package –unity package. Open the unity package and import the package. Once the importing is over, click on the “scenes” and the icon of the unity then we get two options:

- Image target chip
- Image target stone

As these two targets are the default libraries. We need to create a new library. Thus for creating a new library click on Assets- import package – custom package.

Now select the downloaded file from the Vuforia website and import the library for the function. For activating the library process click on the AR camera, then get “Inspector” window on the right side of the screen. In the “Inspector” window a default program load data set stores & chips are activated. So we have to unselect the two process and select the library what we have provided. And the AR camera app work is done here.

![Figure 5.14: importing of Video](image-url)
Now click on the image image target 1. And on the inspector window we get the “data set” where stores and chips are selected. So we have to unselect the option. Then the file which we had created is selected and with this the work of the inspector window is finished.

Figure 5.15: Working on scenes and textures

Addition of videos

Again go to the image target1 click on the video option. And in the inspector window change the path. Now clicking on the streaming assets, go to the file where video is stored and drag from the stored file to the unity app.

Addition of texture

Click on the streaming effect, the video what we have created and the splace screen for the texture purpose.
Now click on the video option click on the path and name the video what is given with the extension .mp4. A new image target is created and the properties are changed using the same procedure. This was the process for linking the target image with video.

5.8 Creating the APP

Click on file- build settings – android –player settings

In build settings a new window will pop up and it can be made for various system software. As this is the app which will work on any android so for android application it is made.

On the right side of the screen Inspector window appears where the company name and the product name is changed. Also there exists other setting where the bundle identifier is changed with a proper name and setting with the splash image.
Figure 5.17: Creating setup for android app

Figure 5.18: Saving App in APK folder
References


