GENERATION OF INDIAN LIGHT MUSIC USING AUDIO PROGRAMMING LANGUAGE

[1]Chidambara Kalamanji
[1]Professor, Department Of Computer Science & Engineering, PES Institute Of Technology, Bangalore- India
[1]chidambara@pes.edu

Abstract : This paper is related to the application of technology for art specifically for Indian music. It is a proven fact that computers are extensively used in the life cycle of music namely music creation, generation, recording, performance and analysis. Traditional way of creation and generation of music making use of skilled artists is being replaced, in recent years, by computer assisted approach. Many software tools are available for music composition, generation. One of the latest potential technologies is to use Audio Programming languages to compose, generate and perform music. Interestingly this technology helps to achieve three things namely (EEE) Enrich the listener's experience, Empower & Enhance the creativity of the musicians. These languages are widely used in Western & European countries. But it is yet to be innovatively explored for Indian music since there are certain challenges have to be addressed. This paper shares the Engineering approach of creating Indian light music, experimented using the audio programming language namely ChucK. Three sample music compositions are presented as a proof of how an amateur musician could be empowered to create Indian music with the help of technology. Objective of presenting this work is to invite the interested researchers to join hands with the author to explore many possibilities to 'add value' to Indian music.

Index Terms—Audio programming, ChucK, Indian light music, Melody matrix, Percussion matrix

MOTIVATION:

Music is a divine art accepted by every human being in the world. I'm an Electronic Engineer, technologist and also an amateur music composer involved in Indian light music. With in me, there has been a **dialogue**, consistently from my young age, **between the Engineer and musician**. Musician is looking for solutions for certain problems and Engineer is passionate about solving musician's problems. Many ideas, rather wish list, used to spark with in me to blend technology for music art.

"Music world is giving me so much; In turn, what can I, as technologist, give back" was the urge to enter in to the exciting domain called **CompuMusic** – (Computer for Music). I was happy to know that there is a music technology community worldwide both in academics and industry contributing lot in terms of research and development. As a result, a spectrum of computer based products, tools based on variety of technologies have been developed. Musicians are utilizing these products mostly in the Western and European countries. **Motivation for me has been to apply and adapt these technologies for Indian light music. The work presented through this paper is just a humble beginning in this direction.**

INTRODUCTION

There are two aspects in the present work.

1.Music

2.Technology.

Let me give a brief overview of the musical aspects first.

MUSIC CREATION METHODS (Fig.2).

Basically, music is the combination of human voice and instruments**.** Musical instruments are of two types -melody instruments and percussion instruments. If stage performance by actual singing, playing real traditional instruments is ***one face of the coin***, recording the music in variety of storage devices and playing using computers, MP3 players and other gadgets is ***the other face.***

We may classify music creation / generation process in to Real and Virtual. **Real music** is the music created by singing and actually playing **real instruments.** Other method which is quite widely used especially in commercial entertainment music sector is to use technology. Technological approach is nothing but making use of computers and **electronically synthesized** music using **virtual instruments** which may be called as **virtual music.**

Computer based music or **CompuMusic** can be classified as :

1. Make use of software tools called **Record Edit Sequence** tools. More than 20 software tools like audacity, Protool, Cubase are used typically in recording studios by music composers and arrangers.
2. Recent technology which is the topic of discussion in this paper is to use Computer Programming Language specifically called as **Audio Programming Language.**

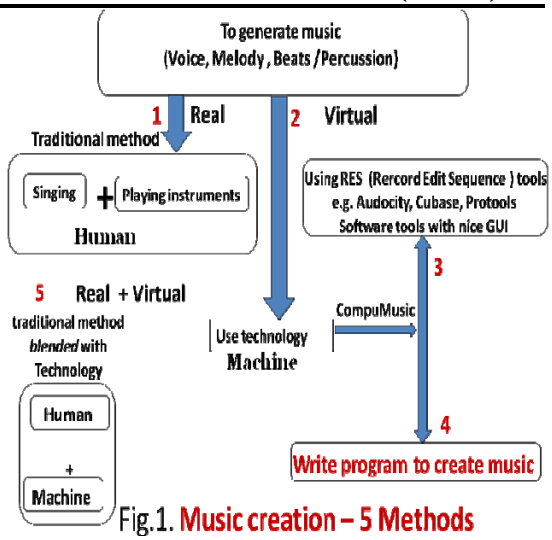

Fig.1. Music creation – 5 Methods

WHAT IS INDIAN LIGHT MUSIC?

Thousands of **genres** are used worldwide in different contexts. A **music genre** is a conventional category that identifies pieces of **music** as belonging to a shared tradition or set of conventions. In India main genres of music are **classical, semi classical, light, folk and popular(like movie music)** In classical music there two types namely **Hindustani and Carnatic.**

The basic concepts of classical music includes

- *shruti* (microtones),
- *swara* (notes),
- *alankar* (ornamentations),
- *raga* (melodies improvised from basic grammars),

and

- *tala* (rhythmic patterns used in percussion).

Classical music follows strict rules, grammar and aesthetics in terms of conforming to raga and tala.

But in light music, anything pleasant to hear and suiting the occasion is accepted. It doesn't impose strict rules to conform to a raga, tala and rendering. **However implicitly or explicitly Indian light music is a kind of derivative of**

classical music. Music composition is based on a raga, but with lots of flexibility and freedom.

Mostly in the recorded music domain, music creation cycle goes through the following steps**(Fig.2)** :

1.Write lyric.
2.Compose basic tune.
3.Arrange instrumental music.
4. Perform.
5.Record.

Ultimately it reaches the listener either directly or through recorded media.

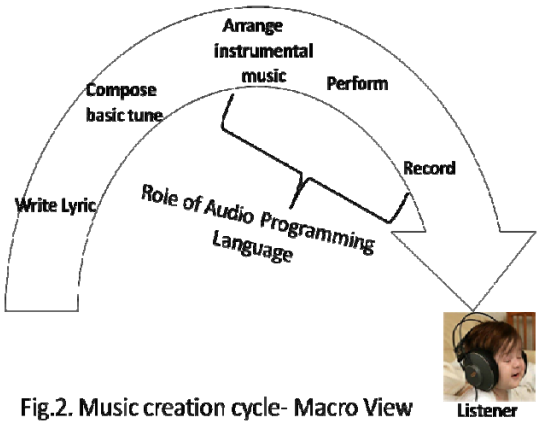

Fig.2. Music creation cycle- Macro View

Typically Audio programming languages have a role compose, arrange instrumental music, and even during performance.

Building blocks of computer generated music [Fig.3]

As shown in Fig.3, typical recorded music is a combination of multiple tracks namely:

a) Voice b) Instrument *c) Special effects*

Instrument track is comprises

a) Melody track
b) Beats/Percussion track.
c) Chord track

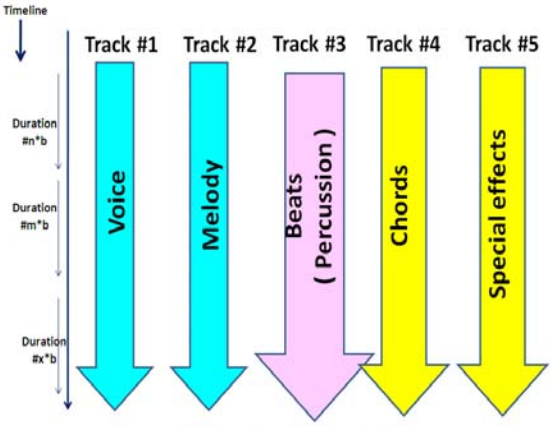

Fig.3 Music design process

Our work focuses only on creating instrument track, since voice track is created through actual singing by the artists.

Main step is to create a **'blue print'** of the final music which is called as **'Score sheet'** in western music terminology.

As shown in Fig… the score sheet represents the flow of all the three tracks on a timeline.

Melody track includes music created using one or more real or virtual instruments like Mandolin, Flute, Guitar, Violin etc.,

Beats/Percussion track includes music created using one or more real or virtual instruments like Tabala, Mridangam, Ghatam, Drums etc.,

Chord track is a special melody track that includes harmonic set of three or more notes of an instrument like Guitar, that is heard as if sounding simultaneously.

Of course, it is not mandatory to have all the tracks; depends on the context and composer's ideas.

PROCESS OF COMPUTER MUSIC GENERATION [FIG.4]

Computer music generation follows a structured way in three phases:

1. Design (Composition)
2. Engineering
3. Implementation (realization of engineered music)

It may be noted that as a novice, when I started music generation projects, I didn't

follow any structured approach. It was a random and iterative approach of course with intuition. After completing couple of projects, I realized that real good music can be generated by properly following these 3 phases.

Music design (can also be called as composition) involves basic tuning of the main melody instruments and deciding what **timbre** of instruments to be used (string, reed, blow, bowing…). Same thing applies to percussion instruments; Composer may decide whether to use Indian classical instruments like Tabla, Mridangam or folk style instruments like Dholak, khanjira or western instruments like drums, snare, hihat etc.,

Music design is a creative work manually done by the composer. As in any art form, intuition, imagination and visualization are the natural elements of design.

Music engineering is to decide the specific musical instrument to be used. **Obviously music engineering has to be done based on music design.** For example, if the result of music design is **string instrument,** music engineering decides whether it should be **sitar, guitar, mandolin or veena.**

Output of the Music design and engineering is a score sheet as shown in Fig.5. Score sheet is a kind of 'blue print'. Score sheet comprises :

a) Multiple melody tracks, each based on a specific melody instrument. Each melody track is characterized by what is known as **Melody Matrix** (explained in subsequent sections).

b) Multiple beats (percussion) tracks, each based on a specific percussion instrument. Each track is characterized by what is known as **Percussion Matrix** (explained in subsequent sections)

Music composition is done manually and music generation is done by writing and executing computer program using the **object oriented programming language called ChucK.**

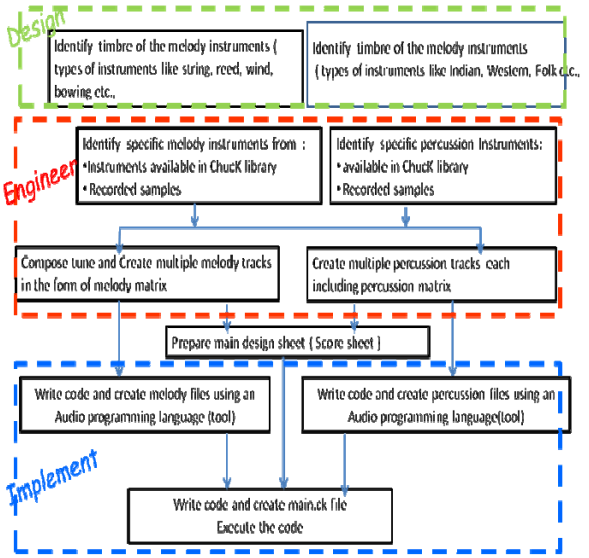

Fig.4-Music Generation phases - Design, Engineer & Implement

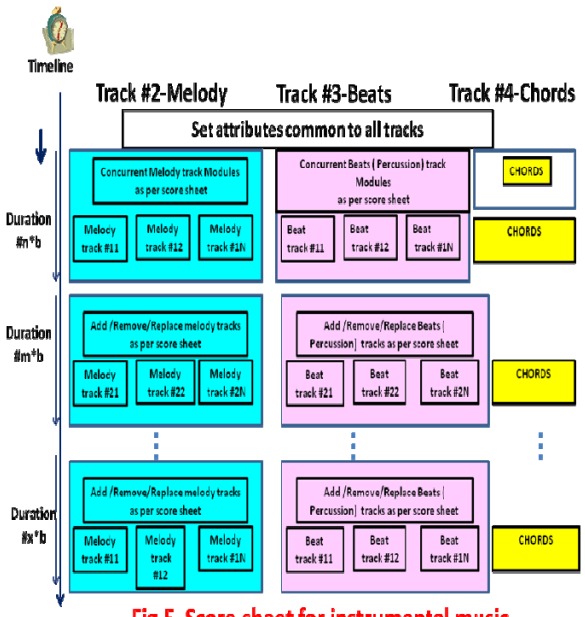

Fig.5. Score sheet for instrumental music

MELODY MATRIX [Table.1]

Melody matrix is the core of music composition that is referred by the programming language to generate music.

	Note	Duration (seconds)	Gain(Number)	Other attributes**
BMU	N1	d*	1	
	N2	2d	1	
	N3	4d	2	
BMSU	N4	3d	0.5	
	N5	2d	0.5	
	N6	8d	1	
	N7	6d	1	
	N8	d	2	

*d – duration of Basic Melody Unit (BMU)
**other attributes : Echo, Reverb, Fading
•**Basic Melody Unit (BMU)** is the Note (Swara) with a predefined frequency.
•Duration of Basic Melody Unit (BMU) – is defined as per the needs of the composition
•A melody track is made up of Multiple BMUs
•BMUs are grouped into **Basic Melody Sequence Unit (BMSU)**

Table1. Melody Matrix

A song is nothing but a series of notes (*swaras* as they are called in Indian music), each played for a specific duration. Technically each note generates a electrical / electromagnetic signal vibrates the diaphragm of the speaker to produce sound. Melody matrix gives the attributes of every note to be played. Mandatory attributes are

- Note number (indicates what frequency has to be generated)
- Duration (how long the note to be played)
- Gain (the intensity or volume of the note)

Optional attributes are to give special effect to each note like

- Reverb
- Echo
- Fading effect

Computer program picks the element of each row in the matrix and produces the melody of that instrument.

Basic Melody Unit (BMU) and **Basic Melody Sequence Unit (BMSU)** are the two elements giving a structure to the matrix.

BMU is the note (swara) with a predefined frequency . d is the duration of BMU in seconds, which decides the tempo of the track. BMUs are grouped in to Basic Melody Sequence Unit (BMSU). In other words, BMSU is a pattern of n number of notes. A melody track is made up of multiple BMSUs. **BMSU is the basis to decide the Percussion track in terms of number of beats per cycle.** BMSUs can be of different lengths as shown in Fig.5

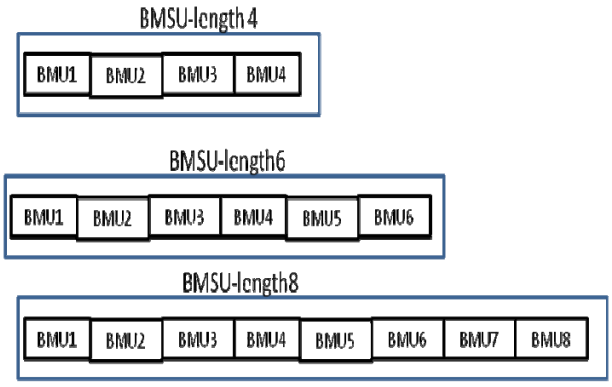

Fig.6. BMSUs of different lengths

PERCUSSION MATRIX [Table.2]

Percussion matrix also follows the same principle as in Melody matrix.

	Beat	Duration (seconds)	Gain (Number)	Other attributes**
BPU	B1	d*	1	
	B2	2d	1	
	B3	4d	2	
BPSU	B4	3d	0.5	
	B5	2d	0.5	
	B6	8d	1	
	B7	6d	1	
	B8	d	2	

*d – duration of Basic Percussion Unit (BPU)
**other attributes : Echo, Reverb, Fading
•**Basic Percussion Unit (BPU)** is one unit – (e.g. in Tabla, it is called as *theka, bol)*
•Duration of Basic Percussion Unit (BPU) – is defined as per the needs of the composition
•A percussion track is made up of Multiple BPUs
•BPUs are grouped into **Basic Percussion Sequence Unit (BPSU)**

Table2. Percussion Matrix

Percussion matrix is the basis to generate the rhythmic pattern of beats. Mandatory attributes are

- Beat number
- Duration (how long the note to be played)
- Gain (the intensity or volume of the note)

Optional attributes are to give special effect to each note like

- Reverb
- Echo
- Fading effect

Computer program picks the element of each row in the matrix and produces the percussion sound of that instrument.

Basic Percussion Unit (BPU) and Basic Percussion Sequence Unit (BPSU) are the two elements giving a structure to the matrix.

BPU is the basic unit of percussion beat. d is the duration of BMU in seconds, which decides the tempo of the track. BPUs are grouped in to Basic Percussion Sequence Unit (BPSU). In other words, BPSU is a structured sequence of beats as shown in Fig.6. A percussion track is made up of multiple BPSUs. It is extremely important that **BPSU is designed to synchronise the melody track.**

So for the discussion has been on the process of music generation, irrespective of the programming language to be used. Let me discus now the vital component – Audio programming language used in our work namely ChucK.

Chuck – AUDIO PROGRAMMING LANGUAGE-OVERVIEW

What is ChucK?

ChucK is a **"open-source" , object oriented programming language** designed specifically for **real-time sound synthesis** and **music creation.** "Real-time" means that ChucK synthesizes the sound as you are hearing it (rather than just playing back a sound file, although ChucK can also do that), and often in response to signals and gestures from the "outside world," such as you typing on the keyboard, moving the computer mouse, manipulating a joystick or other game controller, playing the keys on a musical keyboard connected to your computer, or a host of other things that are available to hook up to our computers.

Not just for sound and music, ChucK is also good for controlling and/or interacting with almost any type of real-time computer media and art, such as graphics, robots, or whatever can communicate with your computer.

ChucK is designed specifically to allow and encourage **"on-the-fly" programming,** which means you can add, remove, modify and edit, and layer segments of code at any and all times, hearing the results instantly, without interrupting other sounds being synthesized and heard.

Most other languages require you to compile, run, and debug code in a fashion that doesn't let you hear what you're doing immediately. In fact, almost all computer languages (such as C, C++, Java, etc.) are not designed specifically from the ground-up for sound, music, and other real-time tasks. **But ChucK makes immediate, real-time sound a priority.**

ChucK was initially created by Ge Wang when he was Perry's graduate student at Princeton University and now an Assistant Professor at Center for Computer Research in Music and Acoustics, Stanford University.

Why ChucK?

• **It's all about time.** Time is at the core of how ChucK works, and how one works with ChucK to make sound. Why such emphasis on time? **Sound/music is a time-based phenomenon;** without the passage of time, there would be no sound. By controlling how and when we do things through time, it's a different and powerful way to work with sound at every level – every molecule of it.

• **It's text, plain and simple.** For example, to generate **1 second music tone of the melody**

Indian instrument Sitar, following 4 lines are sufficient.

Sitar s=>dac;
400=>s.freq;
1=>s.noteOn;
1::second=>now;

• **It's fun and immediate.** It is designed to be a fun language and environment to work in, to experiment in. You can synthesize sounds, make fantastical automations, map physical gestures (e.g., with controllers) to sound, network different computers together, and even analysis to computational make sense of sound.

• **It's free.**

ChucK has been in development and non-stop use for more than 10 years, and it continues to evolve. It's open source, which really means it belongs to everyone whocare to use it, modify it, fix it, improve it, or extend it.

• ChucK has many features as in any other popularly used language like JAVA, C, C++.

EXPERIMENTS & RESULTS

More than 10 Indian music compositions were successfully tested. Out of them 3 compositions are presented in this paper. Corresponding MP3 files are attached.

Music #1

Category	**Popular Indian patriotic song**
Language original song	Indian national language- Hindi
Title	"saare jahaa se accha"
Duration	54 seconds
Voice track	Nil
Number of melody tracks	3
Melody instruments used	Mandolin, Flute, Rhodey,
Available in ChucK library (unit generator)	Mandolin, Flute, Rhodey,
Number of percussion tracks	
Percussion instruments used	ModalBar, Drums, tabla , *'Gejje'*
Available in ChucK library (unit generator)	
Sampled (Recorded) sound	Drums
	Tabla
	"Gejje"

Music #2

Category	Popular Kannada old movie song from movie 'Gandhada Gudi'
Language original song	Regional language of Karnataka - - Kannada
Title	-'Naavaaduva nudiye kannada nudi'
Duration	100 seconds
Voice track	Nil
Number of melody tracks	2
Melody instruments used	Mandolin, Clarinet
Available in ChucK library (unit generator)	Mandolin, Clarinet
Sampled (Recorded) sound	Nil
Number of percussion tracks	2
Percussion instruments used	Modalbar, *"Gejje"*
Available in ChucK library (unit generator)	
Sampled (Recorded) sound	Drums

Music #3

Category	Folk style
Language original song	Not applicable – Instrumental music

Title	'Naavaaduva nudiye kannada nudi'
Duration	4 Minutes
Composed by	Prof.Chidambara Kalamanji
Voice track	Nil
Number of melody tracks	3
Melody instruments used	Mandolin, Flute, Rhodey
Available in ChucK library (unit generator)	
	Mandolin
Sampled (Recorded) sound	Nil
Number of percussion tracks	4
Percussion instruments used	Tabla, Gejje, Dholak, Drums and some other folk instruments.
Available in ChucK library (unit generator)	Modal bar
Sampled (Recorded) sound	Drums, Tabla, "Gejje"

SAMPLE ChucK PROGRAMS

PROGRAM#1- GENERATION OF PERCUSSIN SEQUENCE - TABLA

```
//-----------------------------------------------------------
-----------------------------------------
// This module generates Beat(Percussion) sequence
// Written by Prof.Chidambara, PESIT, Bangalore, India.
// Language used is ChucK;
//Instrumental music generated : Tabla - Popular Indian instrument used both in Classical and Light music.
// Approach : Since this instrument is not avalabele as part ChucK libaray ( Unit Generator)
// Already recorded samples of Tabla ( max 1 sec)  in the form of .wav files are used
// Algorithm :
//1) Create temporary buffers; 2)Copy the .wav file from the hard disk to temporary buffer 3) Play.
// Sequence of the files played will generate Tabla pattern .
// In this program 4 beats per cycle is generated.
// Input : .wav files in a floder ( preferable to have in current folder )
// Output : Music - Tabla sound pattern
//-----------------------------------------------------------
-----------------------------------------
// Initialisation
SndBuf mySound1 => dac; // Create buffer1
SndBuf mySound2 => dac; // Create buffer2
SndBuf mySound3 => dac;//Create buffer3
SndBuf mySound4 => dac;// Create buffer4

0.25=> float tempo;// variable for the duration of each beat
// Name the path for the recorded Tabla files
me.dir()+ "/wavefiles/TB1NA.wav"=> string FileTB1NA; // Name for the path of the .wav file corresponding to beat1;
me.dir()+ "/wavefiles/TB1SUR.wav"=> string FileTB1SUR;// Name for the path of the .wav file corresponding to beat2;
me.dir()+ "/wavefiles/TB2CLSGE.wav"=> string FileTB2CLSGE;// Name for the path of the .wav file corresponding to beat3;
me.dir()+ "/wavefiles/TB1OPNGE.wav"=> string FileTB1THAP;// Name for the path of the .wav file corresponding to beat4;

fun void PlayTabla() // Function to play 4 beats in sequence
 {
  while(true)
   {
    FileTB2CLSGE => mySound1.read; // Read the .wav file to buffer1
    0.75=>mySound1.gain;// set the gain
    0=> mySound1.pos;// start from the begining of the buffer and play
    tempo::second=>now; // for the duration defined by the variable tempo.
```

```
    FileTB1SUR => mySound2.read; // Read the .wav file to buffer2
    0.75=>mySound2.gain;// set the gain
    0=> mySound2.pos;// start from the begining of the buffer and play
    tempo::second=>now;//for the duration defined by the variable tempo.

    FileTB1NA => mySound3.read;// Read the .wav file to buffer3
    0.75=>mySound3.gain;// set the gain
    0=> mySound3.pos;// start from the begining of the buffer and play
    tempo::second=>now; // for the duration defined by the variable tempo.

    FileTB1THAP => mySound4.read;//Read the .wav file to buffer4
    0.75=>mySound4.gain;// set the gain
    0=> mySound4.pos;// start from the begining of the buffer and play
    tempo::second=>now;//for the duration defined by the variable tempo.
  }
} // End of function that plays Tabla beat sequence

// Main program; calls the function to play tabla
PlayTabla();
```

PROGRAM#2-GENERATION OF PERCUSSIN SEQUENCE – MODAL BAR

```
//-----------------------------------------------------------------------------
// This module generates Beat(Percussion) sequence
// Written by Prof.Chidambara, PESIT, Bangalore, India.
// Language used is ChucK;
//Instrumental music generated : Modal Bar
// Approach : This instrument is  avalabele as part ChucK libaray ( Unit Generator)

// Algorithm :
// Modal Bar is played continuously at a set tempo.
// Input : Nothing
// Output : Music- Beats
//-----------------------------------------------------------------------------
// Initialise
 ModalBar bar => dac;
 0.5=>bar.gain; // volume set
 10=> bar.preset; //Select the sound (timbre)required. Value assigned will decide the sound generated.
 0.25=> float Tempo;// duration set; ( which also sets the tempo)
 61=> int pitch;// set the pitch of the instrument.

fun void PlayModal()// function to play continuosly beats.
{
  while(true)
    {
      Std.mtof (pitch)=>bar.freq;
       1.0 => bar.noteOn;
       Tempo*2::second=>now;
     }
}

PlayModal();
```

PROGRAM #3-GENERATION OF MELODY – INDIAN PATRIOTIC SONG “SAARE JAHAASE ACCHA”

```
//-----------------------------------------------------------------------------
// This module ( Melody Track ) plays the first line ( Pallavi) of the song  "SAARE JAHAASE ACCHA"
// Written by Prof.Chidambara, PESIT, Bangalore-30 Nov 2014
// Using Audio Programming Language : ChucK
// Input Data : Melody Matrix
// Output : Music
//-----------------------------------------------------------------------------
```

```
//Select instrument
  Mandolin m=> dac; //
  0.5=>m.gain;// Intensity of sound
  0.25 =>float KTempo;// Variable to control the duration each note and the tempo of the song.
  0.95=> float DutyCycle;// Variable to control the on duration and off duration of each note.

  KTempo*DutyCycle=>float OnDur; // set On duration of each note.
  KTempo*(1-DutyCycle)=> float OffDur; // set off duration of each note.
// Melody matrix [2X2]: Each row [ Midi Note number, Duration ]; This matrix represents the main part of this song.

SA    RE    -     JA    HA    SE
[[    [64,1], [63,1] [61,1], [63,2], [60,1],

A     CH    A
[61,2], [61,2], [999,2]  @=>  int notepallavi[][];// 64,2],

  NotePallavi.cap()=> int NoteTotal;// Total notes in the matrix

// Function to play the notes of the Melody matrix using the instrument Mandolin
fun void PlayMandolin()
{
  for(0 => int i; i<=NoteTotal-1; i++) // Loop that selects each note one by one and plays.
    {
              NotePallavi[i][0]=> int note1; // select the current note
                 if
                  (note1==999)// if it is 'silent' don't note off. 999 in the matrix represents silence.
                    {
                      <<< "silence">>>;// Dispaly that it is a silence
                        0=>m.noteOn;

  NotePallavi[i][1]*OnDur::second => now;// both On duration and off duration of the note to be silent
                        0=>m.noteOn;
NotePallavi[i][1]*OffDur::second => now;
                    }

                else // if the note is not silent, play the note;

                  {
                      <<< "Playing Mandolin Note">>>; // display that mandolin note is played
                        NotePallavi[i][0] => Std.mtof => m.freq; // pick the note number ( MIDI ) format; Convet in to frequency.
                      1.0 => m.noteOn; // Play note in this frequency

NotePallavi[i][1]*OnDur::second => now; // Play for the duration specified ( second element ) in the row;
                      0=>m.noteOn;

NotePallavi[i][1]*OffDur::second => now;// Second part of the note; silence
                    }
    }
        0=>m.noteOn;
        <<< " Mandolin Pallavi is over">>>;// when all notes are played, display;
} // End of function

//Main program

PlayMandolin();// Call the function to to play all the notes of the matrix
me.exit(); // exit
```

CONCLUSION

"Enrich the listener, Empower the musician and Enhance the creativity using audio programming language" was the statement made in the beginning of this paper. **Could this be achieved?. Answer to this is 'YES'.**

Enrich the listener: When the music created and generated through this mechanism was sent to some of the listeners, excellent feed back was received. They have expressed that

some patterns of music were unique which couldn't be possible through real music.

Empower the musician: Author himself is the proof for this. Author is just an amateur music composer, not a professional musician. Only playing keyboard was the skill set possessed. Playing percussion or playing any other instrument was not known. As can be listened from the three music clippings presented, an orchestra of melody and percussion could be created *(of course, there is a lot to improve).* Using audio programming language ChucK has empowered an amateur musician.

Enhance creativity: ***No doubt, human creativity can not be replaced by machines. How can a machine be creative when it can do only the things programmed by the humans?.*** This logic is true. But while experimenting using ChucK, many innovative melody and percussion pattern design ideas were triggered which was nothing but enhancing the authors creativity.

Quality of music generated as part of the experiment may not be as good as a professional music created in a studio. But it may be treated as a 'proof of concept' that has shown the potential of this interdisciplinary approach that can be explored by the CompuMusicians. **Through this paper, author invites the interested, passionate individuals to join in the journey of CompuMusic.**

References :

[1] Prof.H.M.Padalikar,"When Science Facilitates Art Designing A Computer Based System To Simulate Music Rendered by Tabla", International journal of Multidisciplinary Approach and Studies, Volume 01, No.5, Sep-Oct 2014, ISSN NO::2348-537X

[2] Ajay Kapur, Perry Cook, Spenser Salazar, Ge Wang, ""Programming for Musicians and Digital Artists – Creating music with ChucK", Page No 4-5, Manning publications.

[3] Ge wang and Perry Cook, " The Chuck Manual 1.2.1.3"

Online references:

[3]http://ravikumarv.wordpress.com/2010/06/13/indian-classical-vs-light-music/

[4] http://www.musicgenreslist.com/

[5] Wikipedia