# Generating Computer Music from Skeletal Notation for Carnatic Music Compositions

**M.Subramanian***
maniams@yahoo.com

## ABSTRACT

Although a high degree of improvisation is the hall mark of Carnatic music, it still revolves around compositions mostly written in the past 250 years. The music is carried down the generations by oral tradition. A composition may be preceded by or interspersed with improvisations. Carnatic music notation uses the sol-fa (sa ri ga ma pa da ni for the 7 notes) which is written on one line and the lyric on the next line. Books containing notation for Carnatic music compositions were printed in the 19th century and continue to be printed. The notation available in books is only skeletal and does not represent the music completely though many musicians can fill up the nuances intuitively. The objective of the present work is to generate acceptable music from the notation with the computer filling up for the *gamakams* and other requirements. This paper describes the work done and under development. The notation player Gaayaka uses the traditional notation transliterated into English with slight modifications and can play acceptable music if the nuances are also notated but cannot automatically add nuances for which a separate program has been written.

## 1. INTRODUCTION

Carnatic music has many types of compositions such as *kritis*, *varnams*, *svarajatis*, *padams* and *jāvalis* which are presented in the concerts. The *kritis* are the major ingredients of a concert. A *kriti* may run into many lines or rhythmic cycles, certain lines being repeated with progressive embellishments (*sangatis*). The basic music for the compositions is predefined by the composer, though there is scope for improvisation extending the composer's ideas. Thus, in a Carnatic music concert, a considerable part will be devoted to predefined music which can be written down with notation.

Carnatic music notation uses the sol-fa (sa ri ga ma pa da ni for the 7 notes) which is written on one line and the lyric on the next line. Notation for Carnatic music compositions is available in books (some more than a century old) and manuscripts. As the notation available in books is skeletal, musicians have to fill up the nuances intuitively.

Any system meant to generate music from Carnatic music notation, has to provide for continuity between notes within a phrase and control of transit duration between notes and possibly minute adjustment of the pitches of notes. Gaayaka [1] is such a program which accepts notation in the traditional format and plays the notation as entered. Since the notation available in books is skeletal the music will in most cases not be acceptable.

Crucially, generating computer music from notation in Carnatic music requires sophisticated handling of *gamakams* essential for bringing out the correct mood of the *rāgam*, and the composer's ideas. The term *gamakam* used in Carnatic music is different from the term *gamak* used in Hindustani music. In Carnatic music it covers all types of continuous movements of pitch including *jāru* (*meend* of Hindustani music). Generating computer music with appropriate *gamakams*, however, faces a formidable challenge since the notation available is tantamount to a "lossy compression" of the music as originally conceived, with many possibilities for "filling the gaps". Further the appropriate *gamakam* at a certain point may vary considerably depending upon the *rāgam*, and the context - whether the movement at that point is up or down, whether the pitch movement turns at the note, to name just a few.

This paper presents a technique for synthesizing Carnatic music from skeletal notation, complete with *gamakams*. The technique has been implemented in a separate program AddGamakam in which the user can enter skeletal notation (transcribed from texts containing notations for a *kriti*, for instance), and the program automatically adds appropriate notes (called *anusvarams*) and produces notation incorporating *gamakams*. The output of this program can then be played in Gaayaka which can be invoked from within AddGamakam. Eventually the two programs will be integrated. The acceptability of the *gamakam* rendering has been validated by informed listeners though improvements were suggested. Generation of computer music with *gamakams* from bare notation is useful for *kritis* available in books for which no renderings, either transmitted by oral tradition or as recordings are available and the user has no access to a well trained musician who can sing from bare notation.

This paper also describes issues other than *gamakam* which are required to be taken care of when transcribing music from books and the work in progress.

## 2. BACKGROUND

### 2.2 Carnatic Music Notation

The Carnatic music sol-fa (sa rig a ma pa da ni) is used both at the learning phase and in concerts (*svarakalpana)* The same sol-fa is used to write down notation. The notation system has evolved during the 19th and early 20th centuries and has adapted some symbols of the staff notation [2]. A sample of the notation (in Tamizh and English transliteration) with explanations is at [3].

### 2.2 *Gamakam*

It is an accepted fact that appropriate *gamakams* (graces, ornamentation or nuances) are essential to bring out the correct mood of a Carnatic *rāgam*. Sangīta Ratnākara, a 13th century Sanskrit work on music describes *gamakam* as the 'shaking of a note imparting pleasure to hearing and mind' [4]. Sanigīta Sampradāya Pradasrsini [5] describes 15 varieties. In current practice *gamakam* could be described as oscillations of a note or smooth transition between notes and sometimes usage of crushed notes imparting stress. Phrases of identical sets of bare notes can lead to different *rāgams* based on the *gamakams* (and a few other features). *Gamakams* are not simple periodic up and down movements of the pitch as may be seen from the pitch graphs of live music (Figures 1 to 4). The voice may remain at a lower note for considerable period and move up in spurts (Figures 1 and 2) or it may be anchored on an upper note (Figure 3) or the spacing and duration of the oscillations may change if the note is prolonged (Figure 4). There is often overshooting of the peak (with reference to theoretical values) especially in voice renderings. A more detailed analysis of the ranges and shapes of *gamakms* is given by M.Subramanian [7, 8].

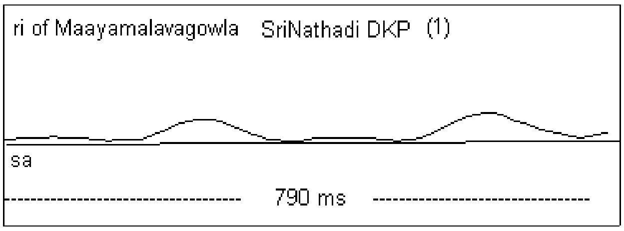


**Figure 1.** Mayamalavagowla ri (1)

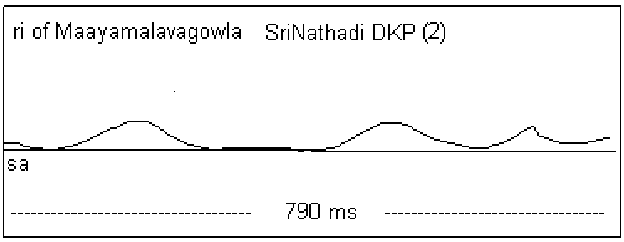


**Figure 2.** Mayamalavagowla ri (2)

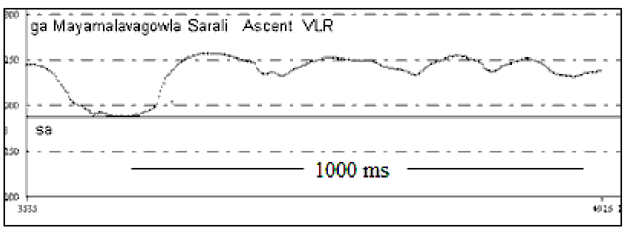


**Figure 3.** Mayamalavagowla ga

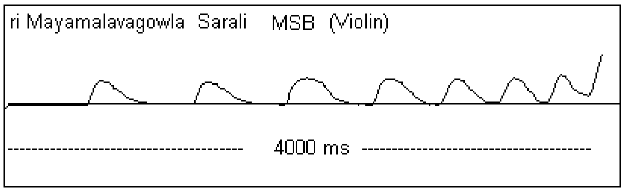


**Figure 4.** Prolonged ri of Mayamalavagowla

A.Krishnaswamy [9] has also given pitch graphs of many *gamakams*. An intuitive understanding of the required *gamakams* is presumed and usage of different types of *gamakams* is not always mentioned in description of *rāgams* and rarely while teaching.

### 2.3 Notation and *Gamakam*

The notation available in most of the books is simple and generally has no indication for the *gamakams* except an occasional wavy line over a note to indicate that it is to be shaken. Detailed symbols for the *gamakams* have been used in Sangita Sampradaya Pradarsini [5] and more recently Sangita Svararaga Sudha [10], but the practice has not caught up. The symbols are qualitative whereas quantitative parameters (such as ranges and durations) are required for accurate description.

In spite of this and other shortcomings described later, a good musician can sing or play from the notation filling up the gaps by his expertise on the *rāgam's* characteristics. Because of this no significant changes have been made in the notation format. It is however true that the same notation could lead to different renderings.

When attempting to generate computer music from notation many gaps have to be filled in. Of these, adding appropriate *gamakams* is the most challenging for the computer music programmer and is considered first. (The other gaps may be filled by suitable algorithms and in case of ambiguity applying heuristic techniques and are considered later)

## 3. CARNATIC MUSIC NOTATION PLYAER

To generate music from notation, a program is required. The program Gaayaka[1] provides for continuity between notes within a phrase and control of transit duration between notes and minute adjustment of the pitches of notes. Traditional sol-fa notation is entered as input with slight modifications and many enhancements. Lyrics and comments can be entered within square brackets which are ignored while playing. Scales, tempo and pitch of tonic can be defined. It plays the music in the tones of *Veena* (Indian Lute) or Flute. As the input is unformatted text notation available on the internet in English can be copied and pasted into Gaayaka screen after some processing. Another program for playing notation from the Carnatic sol-fa is at [11]. It plays using MIDI and does not connect the notes and cannot play *gamakams*. No further development of this program appears to have been undertaken.

Adding *gamakams* to standard notation poses considerable challenge since the notation is often more symbolic than representing the actual pitch of the note. The voice may not stop at all at the note shown in the notation. For instance the note 'ni' in Bhairavi *rāgam* is oscillated from 'da' to 'Sa' not stopping at 'ni' at all but is notated as 'ni'. The note is played by deflecting the string on the ‘da’ fret of the Veena. Figure 5 shows a vocal rendering of the note in Bhairavi *varnam*.

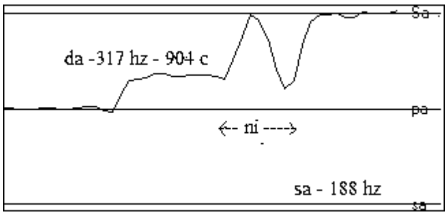


**Figure 5.** Ni of Bhairavi

The note ‘ma’ of Sankarabharanam *rāgam* (in ‘ga maa paa’) is played similarly from the ‘ga’ fret deflecting it all the way almost reaching the pitch of ‘pa’

## 4. ADDING *GAMAKAMS* AUTOMATICALLY

The AddGamakam program described in [12] generates notation replacing, where required, a simple straight note by a set of notes representing the movement of the pitch as in the required *gamakam*. Gaayaka can be invoked from within the program with the output loaded and music played with *gamakam*. The program requires *gamakam* definition files for each *rāgam*. The program is available for downloading at [13] but requires Gaayaka for playing the converted notation. The help file available at [13] describes how the *rāgam* definition files are developed so that a user can write his own file. Some audio files showing the results of conversion are available at [13]. Due to the variability in interpretation, in some cases the program gives two alternatives which can be easily exchanged in the newer version of Gaayaka. Only a limited number of *rāgam* definition files have been made available so far as the process is manual and based on the personal knowledge of the developer as a musician. (Both Gaayaka and AddGamakam programs work in Microsoft Windows[1])

### 4.1 The approach used

Briefly the approach used is based on (a) the *rāgam*, (b) the context in which the note occurs and (c) its duration. In the program 8 main types of contexts are used (in an upward movement, in downward movement, turning at the note from below, turning from above, following or preceding the same note in up or down movements). In addition 2 contexts of silence preceding or coming after the note have also been used. Though in most cases the direction of movement of the pitch is adequate to get the *gamakam* notation there may be exceptions (for instance for the note ‘da’ in the *rāgam Kāmbodi* in the phrases 'pa da Sa' and 'pa da ni' da). Where required the actual note following or preceding the note can also be used to generate a different *gamakam* notation.

The duration is very important since when the music is faster the number of oscillation of the *gamakams* or the duration of the lower steady note is reduced rather than speeding up the whole phrase (Figures 1 to 4). However there is no prescriptive correlation between the duration and the number of oscillations as seen from these figures. For the same duration Figure 1 shows two oscillations and Figure 2 three. The mean time per oscillation varies from 250 ms (Figure 3) to 500 ms (Figure 4).

In the program 6 duration ranges have been provided with facility to alter the range boundaries. The input of plain notation is read and ‘context strings’ are generated for each note. The first 4 characters of the string show the note name, duration range and the context. Other information like the actual preceding and succeeding notes, position of the note in the phrase, duration of the note etc. follow. Using the context string the program chooses the required *gamakam* replacement notation from the *rāgam's gamakam* definition file, brings it to the correct note duration as in the original file and replaces the original note. A detailed description of the context string is available in the help file of the AddGamakam program (available at [13]).

Instead of generating music keeping the conversions in the background, replacement notations were used so that any other notation playing program can also use the system (if need be converting the notations into the format required by it).

Nevertheless the system cannot be considered anywhere near perfect. Being an art form there are many imponderables which lead to the final creation. The program can to a good extent fulfill the objective mentioned at the outset.

### 4.2 Modelling *Gamakams*

*Gamakams* could be modelled in different ways. The ideal would be to analyse large number of live recordings and extract common features for each note of the *rāgam*. This implies a reliable program to identify note boundaries and transcribe live music into the current simple notation format. The transcription cannot be in great detail with detailed notation for the *gamakams* since the purpose would be to identify movements associated with a single note in the traditional notation.

The other alternative is to use the available knowledge (in writings or with the musicians). The simplest model is to consider *gamakam* as a continuous variation in pitch with some constant pitch regions. A set of $3*n - 1$ numbers can represent a *gamakam* where $n$ is the number of pitch positions touched, the first number being the starting frequency followed by its duration and duration of transit to the next frequency and so on (the last frequency not having transit) as described in [14] This method was used in Rasika program [6]. Writing these

[1] Trademark acknowledged

numbers requires musical training to interpret movements of pitch as numbers and repeated testing.

For transcription, the individual oscillations of a *gamakam* have been conceived as 'atoms' by A. Krishnaswamy [9] and it is suggested that any type of *gamakam* can be assembled from the 'atoms'. Graphic symbols used by A. Mallikarjuna Sharma [10] shows how the pitch moves. However, any modelling would eventually require knowledge of which *gamakam* (or group of entities) is to be used for a note in a particular place in a *rāgam* and how the entities are to be linked. It is for this reason that the context was considered as the starting point for the insertion of *gamakam* notations. [12].

### 4.3 Results

Being an art form providing for different styles and extemporisation, judgement of the results is difficult and is likely to be subjective.

Results have been good for *varnams* (which are composed with notation as the basis) and acceptable for krithis in most cases. In some cases the present day version of a *kriti* or the version with which the listener is familiar with is different from the notated version in old books. This is one of the reasons for some results not being acceptable. In some cases changes in the note duration before conversion improved the music generated from the converted notation.

Synthetic music lacks 'expressiveness'. In the case of *gamakam,* modulations of voice in volume and quality often add to the expression. This is also possible in the case of instruments like violin. Lack of these effects is also a reason for 'not good' quality of the output in some cases. However, for the limited objective mentioned earlier the output can be considered satisfactory.

## 5. OTHER REQUIREMENTS

### 5.1 Grouping of notes

As compositions are central to Carnatic music concerts, even instrumentalists try to play with a view creating the feeling of hearing the lyric, which requires separation of the music into phrases. In the currently used notation system, apart from the absence of indication for *gamakams*, there is no standard for marking groups of notes with reference to the lyric or points of accent. In the lyrics there is also no standard system to show the alignment of the notation with lyric when vowels are prolonged over many notes except for physical alignment on the printed page which often gets disturbed during printing.

Gaayaka allows up to 20 notes to be linked without break. In practice for singing or playing compositions the notation is intuitively grouped into phrases – for faster songs the consonants in the lyric and for slower ones at the consonants and other appropriate places. A new phrase is played with a plucking on the *Veena* or reversing the bow on the violin or momentarily stopping the blowing in the flute. It is very rare to find a phrase 20 notes long in compositions. When copying notation from books and testing their conversion, it was found that if these points are not correctly marked (Gaayaka uses a hyphen "-" for this) the song is often unrecognisable in the synthetic music. This segregation of notes based on the lyric was found to be a time consuming process when done manually. Automating this process has been attempted, taking into account the fact that writing lyrics in Indian languages using English alphabets is itself not a fully satisfactory process as no standards have been adopted. After laying down certain rules it is found that this process can be automated for simpler medium paced or fast paced songs. The algorithm breaks the lyric part into syllables and assigns them duration units and marks the notation such that the phrase durations synchronise with the syllable durations. Durations of syllables depend upon the vowel (long or short) and in the case of short vowels whether a single or multiple consonants follow the vowel. For instance, in the Sanskrit word 'putra' the vowel 'u' is 2 units while in 'pura' it is one unit. There are exceptions such as 'bh' in 'subha' which is only short and one unit. The algorithm developed so far works well for songs which do not have unduly prolonged vowels beyond the 2 units.

The real difficulty is that, unlike the notation itself, there is no standard practice for indicating prolonged vowels beyond 2 units in the lyric. Some leave spaces, others put dots or hyphens and mostly attempt is made to align vertically the notation and corresponding words of the lyric which could get disturbed in printing. A sample scanned from a 1956 publication is at Figure 6.

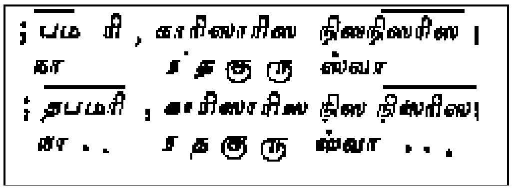

**Figure 6.** Prolonged vowels in lyric

In the second line dots are used while blanks are left in the first line.

A standard may have to be prescribed for typing the lyric when it is copied. Old publications are being studied and this part is yet to be developed.

### 5.2 Silences

There are 2 types of silences. One is due to the lyric starting after the beginning of the rhythm cycle (*āvartham*) or ending at the middle of a cycle. The first poses no problem. The second type of prolonged silence often in the middle or end of rhythm cycle can be taken care of by using some rule of thumb, such as the last note being extended to a fourth or a third of the period of the gap depending on the gap duration.

The other type of very short silence occurs when a vowel is followed by a double consonant (other than sibilants). The word 'bhakta' in the lyric is pronounced with a short gap before 'ta'. The *Veena* player damps the string for a very short moment. This has to be correctly reflected in instrumental music for proper feel of the lyric and the notation altered to show the silence. If the alignment of the lyric to the notation mentioned earlier is correctly done, then the insertion of silence can be implemented automatically for selected consonant combinations.

### 5. 3 Using notation available on the Internet

Carnatic music notation is found in many web sites in English. While transcribing from the native sol-fa (sa,ri ga ma pa da ni) which includes vowels, the practice that has come to stay uses only the letters S R G M P D N. (This is not the system in Gaayaka which uses the vowel part to indicate the octave and also to show notes of 2 units).

Two different standards are being used in writing notation in English. In one system only upper case characters are used for the notes and prolonged notes are indicated by adding commas (Figure 7).

```
P , ,- P, M R - G R   |S  N , - P N  S R N
Sa - - mi -  -   ni -   |nne - -  ko - - - - |

S , , , , , -       N  S |R G R  S R M P N||
Ri - - - - - -      yu -  |- - - -  nna - -     nu ||
```

**Figure 7.** Notation in English (1)

The other standard uses lower case for notes of 1 unit duration and upper case for 2 units. Longer notes are indicated by commas or semicolons (Figure 8).

```
M - mg R – S  G M P ;      | D ds nn-  D  P ;  ;  ; ||
Ga  jaa -- -  na na yu tham    Ga- ne -- shwa ram - - -

mpd pmgr , – S  G  M  P ;   | D ds nn- D P ; nddp    ||
Ga -- jaa -- -   na na yu tham  Ga- ne - shwa ram - --
```

**Figure 8.** Notation in English (2)

In either case there is no indication for the octaves which have to be guessed. Underlines are used for halving the note duration and it is not possible to show a quarter-note.

Such notations can be processed for automatic conversion it into Gaayaka (or other notation player) format using heuristics to guess the octave. Parts underlined in the notation (to mark half notes) have to be manually indicated by brackets in Gaayaka. A program has been written for conversion into Gaayaka format.

It would be simpler if Gaayaka type of notation is used in English, as the notation is unambiguous, covers all aspects (and more) and uses only ordinary characters (Ascii 32 to 127) and easily portable.

## 6. FROM THE BOOK TO THE SOUND

The steps in the process of generating music from the notation in books or manuscripts or available on the internet would be:

For notation in books and manuscripts type manually into a text file, marking Lyrics in square brackets.

For notation available on the internet, copy as unformatted text and use a program to convert to Gaayaka notation with manual editing for octave jumps and half note markings.

Use a program to mark phrase boundaries with hyphen in the text of Gaayaka notation based on the syllables of the lyric.

Paste this notation into AddGamakam screen. Enter *melam* (scale). Guess and enter note duration (tempo). Check tempo, correctness of note durations and bracket balance by invoking Gaayaka from within AddGamakam..

Convert into notation with *gamakams.*

Play the notation by invoking Gaayaka from within AddGamakam program.

## 10. FUTURE WORK

In the present AddGamakam program the replacement notation is based on duration ranges and when the duration of a note is not the middle of the range it has to be 'stretched' or 'shrunk' which sometimes leads to unacceptable results especially when 'stretched'. The algorithm now used can be refined to avoid this. One approach to handling the problem of durations is suggested by S. Subramanian et al in [15]. Basically the notation system has adopted progressive halving of note durations for faster phrases. Extending the same to smooth movements of *gamakam* is not the best possible way to represent *gamakams* but it has the advantage of easy readability and editing.

The alternative is to define parameters for the *gamakam* with duration as one of the parameters. The other parameters could be the context as mentioned above, the anchoring point, transit durations and range of oscillations and their shapes. The algorithm has to fit the *gamakam's* oscillations within the duration without significantly shrinking or stretching the oscillations themselves. The algorithm has also to decide the number of oscillations, constant pitch areas and their durations.

While the traditional *rāgams* require full-fledged definition files, for newer *rāgams* which came into vogue after 72 scale system was proposed in the 17th century by Venkatamakhi, it may be possible to define 'generic' *gamakam* notations for many of the notes requiring separate definition only for one or two notes.

The existence of different styles would also suggest that the system could even provide for them, inserting *gamakam* notations differing in (say) the oscillation range or oscillation durations.

These and the points mentioned in Sec. 5 would be the scope of future work.

## 9. CONCLUSIONS

Generating acceptable computer music from bare skeletal notation of Carnatic music compositions available in books requires filling up many gaps in the notation. One of them is *gamakam* (nuances). A system for automatically inserting notation containing *gamakam* into the skeletal notation based on the *rāgam* and the context in which the note occurs is described. Possible other approaches are discussed. There is scope for future work based on the results. The other aspects such as phrase segregation in the notation, alignment with lyric, marking silences are also discussed. For some of these programs have been developed or under development.

**Acknowledgements**

We would like to thank Dr. N.Rāmanāthan, former Head of the Department of Music, Madras University for his guidance during the development of the work. We would also like to thank Dr. S. Rāmanāthan of BBN Technologies, USA for studying the draft and giving suggestions,

## 10. REERENCES

[1] M. Subramanian, "Gaayaka - carnatic music notation player". [Online]: Available: http://carnatic2000.tripod.com/gaayaka6.chm

[2] S. Rāmanāthan, "THE INDIAN SARIGAMA NOTATON", in Journal of Music Academy, Madras, XXXII (1961), pp 83-90

[3] M. Subramanian, "Carnatic Music Notation System" [Online]. Available: http://carnatic2000.tripod.com/notation .htm

[4] Sangītta Ratnākara of Sārngadeva, Anandāsramam Edition (Sanskrit), Vol 1 (1985), p 253

[5] Subbarama Dikshitar, (Translated into Thamizh by S.Ramanathan and B Rajam Ayyar), "Sangīta Sampradāya Pradarsini", Music Academy, Madras, India (1961). English version.[Online]: Available http://www.ibiblio.org/guruguha/ssp.htm

[6] M. Subramanian, "Carnatic Music Software - Rasika & Gaayaka" Audio [On line]: Available http://carnatic2000.tripod.com/akshpln.mp3 http://carnatic2000.tripod.com /akshgmk.mp3

[7] M. Subramanian "An Analysis of Gamakams using the Computer", in Sangeet Natak, Vol. XXXVII, 1 (2002) pp. 26-42. [Online]: Available http://carnatic2000.tripod.com/article.htm/gamcomp .zip

[8] M.Subramanian, "Carnatic Ragam Thodi: Pitch Analysis of Notes and Gamakams" in.,Sangeet Natak, Vol. XLL, 1, (2007) pp. 2-20. [On line]: Available http://carnatic2000.tripod.com/article.htm/thodigam akam.pdf

[9] A. Krishnaswamy, "Melodic Atoms for Transcribing Carnatic Music" in Proceedings of. International Conference on Music Information Retrieval. (2004). [Online]: Available http://ismir2004.ismir.net/proceedings/p063-page-345- paper219.pdf

[10] A. Mallikārjuna Sharma, "Sangīta Svararāga Sudha", Sai Sannidhi Sangita Publications, Hyderabad, India (2001)

[11] "Future Carnatic Music" [Online] Available: http://homepages.ihug.co.nz/~ganeshan.

[12] M.Subramanian, "Carnatic Music - Automatic Computer Synthesis of Gamakams" in Sangeet Natak, Vol. XLIII, 2009 pp 28-36

[13] M. Subramanian, "Computer Synthesis of Carnatic Music Gamakam". [Online]: Available: http://gamakam.tripod.com

[14] M. Subramanian, "Synthesizing Carnatic Music with the Computer", in Sangeet Natak, 133- 134 (1999) pp 16-24

[15] S. Subramanian, L. Wyse and K. McGee, "Modeling Speed Doubling in Carnatic Music" presented at the International Computer Music Conference 2011, University of Huddersfield, UK, 2011, pp 478-485.

---

* Formerly visiting faculty, Department of Music, Madras University