Two Extensions of a Phase Vocoder

Peter Møller-Nielsen
Department of Computer Science
Aarhus University
petermn@cs.au.dk

15/6/22

Introduction

The basic elements of a Phase Vocoder can be combined and extended in various ways depending on the application. This document describes two such extensions. For each the complete code is provided along with a description of how to run it.

One extension allows stretching (and compression) of a sound file in real-time. The stretch factor can be changed by moving a cursor inside a window (up and down). The sound produced can be heard while the stretching is taking place and the result is saved in a file.

The other allows stretching (and compression) as well but based on a table. The table divides the input sound into time intervals. For each interval a stretch factor is given. The factors can be modified in real-time by moving the cursor in one direction (up and down). Simultaneously the dynamics can be reduced by moving the cursor towards the right. This extension was used to adjust the agogic and dynamics in music in order to study the influence of agogic and dynamics on the perception of the music.

The Basic Elements of a Phase Vocoder

Time stretching of a sound without changing the pitch has been used by composers of electroacoustic music for several years. A popular algorithm is the so-called Phase Vocoder. This section is a short description of the implementation, which is underlying the two extensions.

The modified sound is formed by adding short sections, called bursts, of the original sound with an adequate overlap. Depending on where the burst is extracted from the original sound, the created sound will appear as the original sound stretched (or compressed) in time.

In order to obtain a reasonable quality of the created sound, the added burst must be adjusted to fit the previous burst. This adjustment is done by transforming the burst to the frequency domain, and the phases of the components are adjusted to fit the corresponding components in the preceding burst. The adjusted representation in the frequency domain (the spectrum of the burst) is then transformed back to the time domain. It can now be added to the created sound.

Adjusting the phases of the spectrum to the previous burst is not trivial. Jean Laroche og Mark Dolsons [JL, MD] found a method that gives a reasonable quality to the created sound. The Phase Vocoder underlying the two extensions is based on this method.

An overview of the process is shown in fig. 1. The created sound ‘Sound out’ is shown along the horizontal axis with time increasing towards the right. The original sound ‘Sound in’ is shown along the vertical axis with time increasing upwards. Starting in origo and increasing towards the right, a curve ‘Stretch factor’ is shown. This curve determines where in the original sound ‘Burst in’ is taken. The slope of the curve thus determines the stretch factor by converting ‘T out’ to ‘T in’. The curve can be the result of a preceding analysis of the original sound. But it can also be created in other ways. The slope can be controlled by moving the cursor in a window, while the process is taking place. Before the processing starts, the position length and overlap of the bursts ‘Burst out’ is fixed. They all have the same length and overlap, e.g. 8, which means that each sample in ‘Out sound’ receives contributions from 8 bursts.
The next ‘Burst out’ is not calculated until it is needed. This is necessary in order to make the motion of the cursor influence the stretching in real-time.

The calculation of ‘Burst out’ from ‘Burst in’ is done as follows:

When ‘T in’ is determined from ‘T out’ and the curve, ‘Burst in’ is extracted from ‘Sound in’ by multiplication with a suitable window in order to mute leading and trailing samples.

The spectrum is created by means of a Fourier transform. If the pitch is unchanged ‘Burst out’ will have the same amplitude spectrum as ‘Burst in’.

The amplitude spectrum is then divided into intervals. The intervals are found by first finding the dominant peaks and then the valleys between the peaks. An interval is the section between the valleys.

In each interval the phases of the frequency components are calculated from the phase of the corresponding peak in the spectrum of the previous ‘Burst out’ and the time difference between the bursts.

When amplitudes and phases are determined for all the components of the spectrum, ‘Burst out’ is calculated by means of an inverse Fourier transform.

The ‘Burst out’ can now be added to ‘Sound out’.

Reference

[ JL, MD ] Jean Laroche and Mark Dolson,
Improved Phase Vocoder Time-Scale Modification of Audio,
IEEE Transactions on Speech and Audio Processing, vol. 7, no. 3, May 1999
1. **extension:**
   Stretching a sound.
   The stretch factor can be changed in real-time by moving a cursor in a window.
   Code and sound examples are in phaseVocoder_dyn.tar.

**Source code and sound examples:** phaseVocoder_dyn.tar

**Platform:** LINUX/C++

**External packages:**
- **SDL2:** audio out, graphics out, cursor in and keyboard in
- **fftw3:** Fourier transformation

**Compilation:** ./MAKE

**Execution:**
1. ./phaseVocoder <>.wav
2. ./phaseVocoder <>.wav < stretch factor >

   **case 1)**
   The sound in <>.wav is stretched by a factor depending on the position of the cursor in the window.

   **case 2)**
   The sound in <>.wav is stretched by the factor < stretch factor >. The position of the cursor is ignored.

```
tar -xf phaseVocoder_dyn.tar creates a directory phaseVocoder_dyn, which contains source code, aids for compilation and some sound examples.
```

When execution is started, a window (fig. 2) is created. Creation and play back of the generated sound starts (and is restarted) by typing `space` with the cursor in the window. The stretch factor is changes by moving the cursor up and down in relation to the red line.

**Keyboard:**
- **q:** execution is stopped and the created sound is written in the file pVocOut.wav.
- **d:** the stretch factor corresponding to the position of the cursor is printed.

![Fig. 2](image)
2. extension:

Stretching of a sound controlled by a table of stretch factors. The stretch factor is given for
time intervals in ‘Sound in’.
Stretch factors and dynamics can be modified in real-time by moving the cursor in the
window (fig. 3).
Code and a sound example and the associated table can be found in phaseVocoder_tab.tar.

**Source code and sound example:**  phaseVocoder_tab.tar

**Platform:** LINUX/C++

**External packages:**
- SDL2: audio out, graphics out, cursor in and keyboard in.
- fftw3: Fourier transformation

**Compilation:**  ./MAKE

**Execution:**
1.  ./phaseVocoder <>.wav <>.str
2.  ./phaseVocoder <>.wav <>.str < gammaA > < gammaD >

    tar -xf phaseVocoder_tab.tar creates a directory phaseVocoder_tab, which contains the program
    modules, some aids for compilation and an example of a sound and the associated stretch table.

    When execution is started a window (fig. 3) is created with a red and a green axis, and the table
    <>.str is loaded. By moving the cursor in the window, the stretch factors in the table <>.str is scaled
    along the green axis and the dynamics along the red axis.

    If < gammaA > og < gammaD > is given (case 2) they determine the scaling of stretch factors and
    the dynamics.

    Where the axis are crossing there will be no scaling. At the lower end of the green axis there is no
    agogic. At the upper end the agogic is amplified.

    At the left end of the red axis there will be no dynamics. It can’t be enhanced by moving the cursor
to the left of the green axis.

**Keyboard:**

    space: start (or restart) creation and play back of ‘Sound out’.

    q: execution is stopped and the created sound is written in the file pVocOut.wav.

    d: the values of < gammaA > and < gammaD > corresponding to the position of the
cursor is printed.
The file `<>.str` determines for each time interval how much the sound in the time interval must be stretched in order to remove agogic. The intervals are terminated by -1.

E.g.:

```
0.000
0.58
0.687
1.24
1.009
1.26
1.324
0.88
1.741

-1
```

In the interval from 0.000 sec. to 0.687 sec. the sound is stretched by the factor 0.58. In the interval from 0.687 sec. to 1.009 sec. it is stretched the factor 1.24 etc.