メロディーを楽音と歌声で合成しながら学ぶ*

○荒井隆行（上智大・理工）

1　はじめに

2009 年から上智大学理工学部情報理工学科の学生実験「情報理工学実験 I 」では，信号処理に関する実験として，音を中心とした信号処理に関する課題を取り入れてきた．2 週にわたって合計 4 コマ分（約 6 時間）の実験において，当初は次のテーマを設定し，今でも基本的枠組みは同じである：

- フーリエ級数展開とフーリエ合成
- 正弦波と純音，メロディーの実現
- DTMF 信号の合成
- サウンドスペクトログラム分析
- ディジタル・パターン・プレイバックによる音声の合成[1,2]

各テーマは Matlab を用いて実施される．最初の週は 2 コマ通しでフーリエ級数展開・フーリエ合成に関するテーマを行うが，そこではのこぎり波や方形波，三角波などを取り扱う．次の週には，再び 2 コマ通しでその後のテーマを取り扱う．メロディーに関しては，平均律による 12 音階と周波数の関係を等比数列を使って説明している．DTMF 信号では，2 つの正弦波を与えられた周波数で足し算することで合成を行う．さらに，サウンドスペクトログラム分析をすることで任意の DTMF 信号から電話番号を推定することが可能であることにも触れていた．そして実験の最後に行うディジタル・パターン・プレイバック（以後，DPP）では，音声を分析することによって得られるスペクトログラム分析の結果を画像ファイルとして与え，そのファイルからフーリエ合成の技術を使って音声信号を再合成する課題を課していた[3, 4].

上記の課題による実験はそれなりに学生の興味を引き出しながらその仕組みを理解させることに貢献をしてきた．しかし，さらなるバージョンアップのため，2013 年からは DPP による音声合成に，基本周波数を変化させて歌声を合成するように変更を加えた．本稿では，その工夫と実現方法ついて論じる．

2　新しい実験課題について

2013 年度から新しく取り組んでいる実験課題は，以下の通りである：

- フーリエ級数展開とフーリエ合成
- 正弦波と純音，メロディーの実現
- サウンドスペクトログラム分析
- 自分の音声の録音と分析
- ディジタル・パターン・プレイバックによる音声ならびに歌声の合成

各テーマが基本的に Matlab を用いて実施されるのは以前と同じである．さらに，最初の週は 2 コマ通しでフーリエ級数展開・フーリエ合成に関するテーマを行い，そして次の週の 2 コマ通しの前半にて，正弦波と純音，メロディーの実現まではほぼ以前と変わらない．ただし，メロディーを作る際に，後に歌声合成を行うためのメロディーをも兼ねるため，その波形の作り方から少し工夫を施すことにした．

2.1 メロディーを作る際の工夫

まず，平均律では等比数列によって周波数の関係が記述されることを説明する．そして，1 つの音符には周波数と長さなどが対応することを説明後，実際に簡単なメロディーを作ることになる．Fig. 1 に，改定後の新しい実験における Matlab スクリプトの例を示す．ここでは，メロディー用として，まず f という配列に C4 から C5 の 1 オクターブ分の周波数を 12 個格納している（1 オクターブ以上用いる場合は，その限りではない）．そして，f0 という配列において，どの周波数の音をどのくらい伸ばすかを格納している．例えば Fig. 1 の場合，最初の音符は C4 であり，長さは 40 という

* Learning about acoustics and signal processing by synthesizing a melody using musical tones and singing, by ARAI, Takayuki (Sophia University).

```
Fs = 16000;                % 標本化周波数
f = 440 *(2^([-9:3]/12)); % C4 から C5 の周波数
f0(    1:   40) = f(1);     % 最初の音符：C4
f0(   41:   80) = f(3);     % D4
f0(   81: 120) = f(5);     % E4
f0( 121: 160) = f(6);      % F4
f0( 161: 240) = f(8);      % G4

ns = 1;                    % 周期内の開始サンプル
frame = 1;                 % フレームインデックス
output = [ ];              % 出力信号の初期化
while frame <= 240         % 最終フレームまで
    N = floor(Fs/f0(frame)); % 1 周期のサンプル数
    ne = ns+N-1;           % 周期内の最終サンプル
    output(ns:ne) = sin(2*pi*[0:ne-ns]/N);
                           % 正弦波 1 周期の波形
    ns = ns+N;             % 開始サンプルの更新
    frame = floor((ns-1)/160)+1; % フレーム更新
end

soundsc(output, Fs)        % 音声出力
```

Fig. 1. Matlab script for melody synthesis by pure tones.

ことになる．ここで，長さの単位は任意に決めることができるか，ここでは 10 ms を最小単位として，それを frame と呼ぶことにする．Fig. 1 の場合，最終的な出力信号の長さは，240 フレーム＝ 2.4 秒となる．

Fig. 1 の例では，C4 D4 E4 F4 G4 という音列にて最後の音を引き伸ばすような単純なメロディーとなる．合成に際しては，正弦波 1 周期ごとに現在の f0 周波数を確認しつつ，対応する正弦波の波形データを 1 周期ずつ出力信号用の配列 output に格納しながら合成が進む．楽音でメロディーを奏でるようにする場合は，１つの正弦波による純音の場合を拡張して，複数の倍音を（基本音同様に）メロディーに対応する f0 周波数の整数倍に周波数を設定しながら，すべての倍音を加算していくことによって実現される．

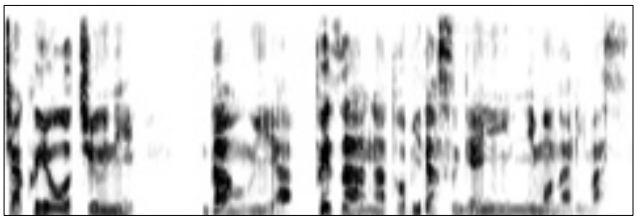

Fig. 2. Spectrogram of a sample sentence (80 ×240 pixels).

2.2 音声の録音と分析

次に，自分自身の音声を録音し，さらに分析する手段について説明する．録音に際しては，ここでは音声分析ソフトウェアの Praat [5] を使用するものとする. ここでは, 最終的に 2.1 節で作ったメロディーに乗せるための歌詞を録音する．ただし，歌声のメロディーはあくまでも合成によって作るために, Praat で録音する音声は歌声である必要はなく，むしろ抑揚はなくても構わない. ただし, 明瞭な発話のほうが好ましい. Fig. 1 で例に挙げたスクリプトでは，全体が 2.4 秒であったため, ここでも約 2.4 秒前後に収まるような発話を録音することになる．

Praat を用いて標本化周波数 16 kHz にて録音後，前後の非音声区間を削除し，さらにサウンドスペクトログラムによる分析結果を画面で確認した．さらに，その画面を Windows に付属するソフトウェアの「ペイント」にコピー&ペーストして，周囲の不要な部分をトリミング後，さらに縦ピクセル vs. 横ピクセルを 80×240 にサイズ変更し，JPEG ファイルとして保存した（Fig. 2 参照）．なお，この画像ファイルは後に，DPP の際の入力画像になるため，画像のコントラストははっきりとしていることが望まれる．音声信号が弱かったり，周囲の雑音に埋もれてしまったり，発話が不明瞭であったりなどは, 後に DPP の合成時に歌声の質の劣化につながる．そのために，録音や分析に際しては注意が必要である．

2.3 DPPによる歌声の合成

従来から行ってきた DPP による音声の合成では，基本周波数を一定にする合成が

基本であった [1, 2]．しかし，DPP にイントネーションを付加することも可能である（例えば, [3, 6]）. それを応用することで，歌声を合成することができる. そのために，従来からの DPP のアルゴリズムを 2.1 節のメロディーの合成に合わせて作り替えた．その結果，得られた Matlab スクリプトを Fig. 3 に示す.

```
Fs = 16000;                % 標本化周波数
f = 440 *(2^([-9:3]/12)); % C4 から C5 の周波数
f0(    1:   40) = f(1);       % 最初の音符：C4
f0(   41:   80) = f(3);       % D4
f0(   81: 120) = f(5);       % E4
f0( 121: 160) = f(6);        % F4
f0( 161: 240) = f(8);        % G4

f_pixel = 80               % 縦（周波数）ピクセル数
t_pixel = 240              % 横（時間）ピクセル数
S = imread('x.jpg');     % スペクトログラムの読込

ns = 1;                        % 周期内の開始サンプル
frame = 1;                     % フレームインデックス
output = [ ];                  % 出力信号の初期化
while frame <= t_pixel     % 最終フレームまで
    N = 160;                   % 1 周期のサンプル数
    ne = ns+N-1;           % 周期内の最終サンプル
    L = length(output);    % 現在の出力信号の長さ
    output(L+1:ne) = zeros(1,ne-L); % 初期化
    for k = 1:f_pixel        % Fourier 合成用ループ
        A = 255-double(S((f_pixel-k+1),frame));
                               % Fourier 係数
        output(ns:ne) =    % cos 関数の加算
            output(ns:ne)
            +A*cos(2*pi*k*[-N/2:N/2-1]/N);
    end
    ns = ns+floor(Fs/f0(frame)); % ns の更新
    frame = floor((ns-1)/N)+1;     % フレーム更新
end

soundsc(output, Fs)        % 音声出力
```

Fig. 3.　Matlab script for singing synthesis by the DPP.

Fig. 3 を見ると，まず最初のメロディーを定義しているところは，Fig. 1 とまったく同じであることが分かる．つまり，2.1 節にてメロディーを作っておけば，その結果をそのまま歌声の合成の際に使用可能となる.

次に, メロディーに歌詞を乗せるために，2.2 節で録音し分析結果として保存した画像ファイルを使うことになる．この画像ファイルは 縦 80×横 240 ピクセルの JPEG ファイルである．それを Fig. 3 の Matlab スクリプトでは imread 関数を使って読み込んでいる（ファイル名は暫定的に"x.jpg"としている）．読み込まれた画像は S という行列に格納される．そこで,

>>　image(S)

というように，image 関数を用いてスペクトログラムによる分析結果を，Matlab 上でも確認できる.

読み込まれた画像ファイルが格納されている S は，80 行×240 列という行列で，さらに光の 3 原色である RGB 値を保有している．しかし，もともとが白黒画像であるので，今回は色情報は使用しない．Fig. 3 のスクリプトは，基本的にその 80×240 の画素値を 1 つ 1 つ参照しながら合成を行っている．そのために，2 つのループが「入れ子」になっているが，外側のループが横（時間）軸に対する while ループであり，内側のループが縦（周波数）軸に対する for ループになっている.

核となるフーリエ合成に際しては，音声の基本周期ごとに波形を合成している．例えばある frame に対して，対応する f0 周波数はいったん無視し，基本周波数 100 Hz と見なして該当 frame に対するスペクトル情報を得ることとする．この場合，行列 S の第 frame 列目に注目し, その（80 行ある）列ベクトルの各要素が，基本周波数 100 Hz の各倍音に対する振幅であると見なす.（ここで，厳密には画素値をそのまま振幅と見なすのではなく，何らかの変換を施すほうが望ましいが，Fig. 3 ではそのような変換は行っていない.）そして，100 Hz の整数倍の倍音成分に対する cos 関数に画素値から得られた振幅を乗算し,（全倍音成分に相当する）80 個の cos 関数をすべて加算して

Table 1　Percentages of students who got interested in the speech/singing synthesis.

	興味を持てましたか？		
	持てず	まあ	とても
2011 年度	6.3	45.9	47.7
2013 年度	7.1	45.5	46.5

フーリエ合成を施す．その後，現在注目している frame に対する本来の f0 周波数に対応して基本周期を求め，その基本周期だけ離しつつフーリエ合成波形を overlap-add しながら合成する．

3　考察およびまとめ

2011 年度に行われた実験に対するアンケート結果と，新しく改定された実験課題施行後の 2013 年度に実施したアンケート結果の一部を Table 1 に示す．この表では，音声や歌声の合成に興味を持てたかどうかを問う 3 択の設問の結果を特にあげている．2011 年度は 111 名に対して，2013 年度は 99 名（全受講者数はそれ以上）に対して，結果をパーセント表示した．結果としてこの表を見てわかるように，学生の興味を持ってもらうことについて，年度を問わず成功している様子がわかる．しかし，改定後の実験に対して，特に興味を示す学生の割合が増えたという結果は得られなかった．その原因としては，アンケートの聞き方に対して，結果が天井効果を示している可能性も考えられる．また，2009 年度から開始した実験が 2011 年度には安定してきていた状況だったのに対し，2013 年度は改定後の新しい実験課題を始めた初年度であったため，まだ実験者側の不慣れな点も重なり，十分に魅力を伝え切れなかった可能性も否定できない．ただし，アンケートの別の設問において，スペクトログラムにどれだけ知識があるかを問う設問に関しては，2013 年度のほうがその割合が多かった．そのことが，どのように影響しているかなどを含め，今後も調査を引き続き進めていきたい．

謝辞

内容の一部は日本学術振興会の科学研究費補助金（21500841）の助成を得た．また，本調査の実施に際し，上智大学荒井研究室のメンバー，特に安啓一氏に感謝申し上げます．

参考文献

[1] 荒井隆行, 安啓一, 後藤崇公, “ディジタル・パターン・プレイバック,” 音講論, 429-430, 2005.9.

[2] T. Arai, K. Yasu and T. Goto, “Digital pattern playback: Converting spectrograms to sound for educational purposes,” *Acoust. Sci. Tech.*, 27(6), pp. 393-395, 2006.

[3] 荒井隆行, “ディジタル・パターン・プレイバックを用いた音響学ならびに信号処理工学への教育的応用,” 音講論, 1467-1470, 2012.3.

[4] T. Arai, “Digital pattern playback for education in digital signal processing and speech science,” *Proc. of the IEEE International Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 2769-2772, Kyoto, 2012.

[5] P. Boersma, “Praat, a system for doing phonetics by computer,” *Glot International*, 5(9/10), 341-345, 2001.

[6] T. Arai, K. Amino, M. Sonu, K. Yasu, T. Igeta, K. Tomaru and M. Kasuya, “Hands-on speech science exhibition for children at a science museum,” *Proc. of Workshop on Child, Computer and Interaction*, Portland, 2012.