課題 1

リスト 1 に原画像に$\frac{1}{N}$サンプリングを行うプログラムを示す。なお、Adobe AfterEffects および Adobe Photoshop で作成した「miku.jpg」を原画像とする。この画像は縦 512 画像、横 512 画素による正方形のディジタルカラー画像である。

```
% 課題１　標本化間隔と空間解像度
% 画像をダウンサンプリングして（標本化間隔を大きくして）
% 表示せよ.
% 下記はサンプルプログラムである.
% 課題作成にあたっては「Lenna」以外の画像を用いよ.

clear all; close all% 変数のオールクリア

figure(1)
ORG=imread('miku.jpg'); % 原画像の入力
imagesc(ORG); axis image; % 画像の表示
title('原画像')

figure(2)
IMG = imresize(ORG,0.5); % 画像の縮小
IMG2 = imresize(IMG,2,'box'); % 画像の拡大
imagesc(IMG2); axis image; % 画像の表示
title('1/2 サンプリング画像')

figure(3)
IMG = imresize(IMG,0.5); % 画像の縮小
IMG2 = imresize(IMG,4,'box'); % 画像の拡大
imagesc(IMG2); axis image; % 画像の表示
title('1/4 サンプリング画像')

figure(4)
IMG = imresize(IMG,0.5); % 画像の縮小
IMG2 = imresize(IMG,8,'box'); % 画像の拡大
imagesc(IMG2); axis image; % 画像の表示
title('1/8 サンプリング画像')
```

```
figure(5)
IMG = imresize(IMG,0.5); % 画像の縮小
IMG2 = imresize(IMG,16,'box'); % 画像の拡大
imagesc(IMG2); axis image; % 画像の表示
title('1/16 サンプリング画像')

figure(6)
IMG = imresize(IMG,0.5); % 画像の縮小
IMG2 = imresize(IMG,32,'box'); % 画像の拡大
imagesc(IMG2); axis image; % 画像の表示
title('1/32 サンプリング画像')

return
```

リスト 1 原画像に$\frac{1}{N}$サンプリングを行うプログラム

図 1 に原画像、図 2 から図 6 にそれぞれ$\frac{1}{2}$サンプリング、$\frac{1}{4}$サンプリング、$\frac{1}{8}$サンプリング、$\frac{1}{16}$サンプリング、$\frac{1}{32}$サンプリングを行った結果を示す。
例えば、$\frac{1}{2}$サンプリングを行う場合、リスト 2 の処理を行うことで実現できる。

```
IMG = imresize(ORG,0.5); % 画像の縮小
IMG2 = imresize(IMG,2,'box'); % 画像の拡大
```

リスト 2 $\frac{1}{2}$サンプリングを行うコード

このように、$\frac{1}{N}$サンプリングを行う場合は画像を N 倍に拡大すればよい。
また、図 1 から図 6 を比較すると、サンプリングの幅が大きくなるにつれてモザイク状の歪みが大きくなっていることが分かる。

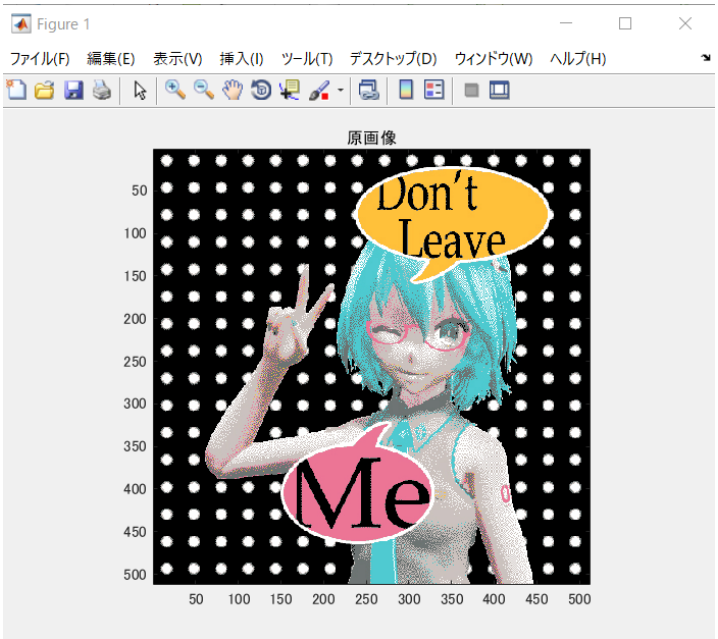

図 1 原画像

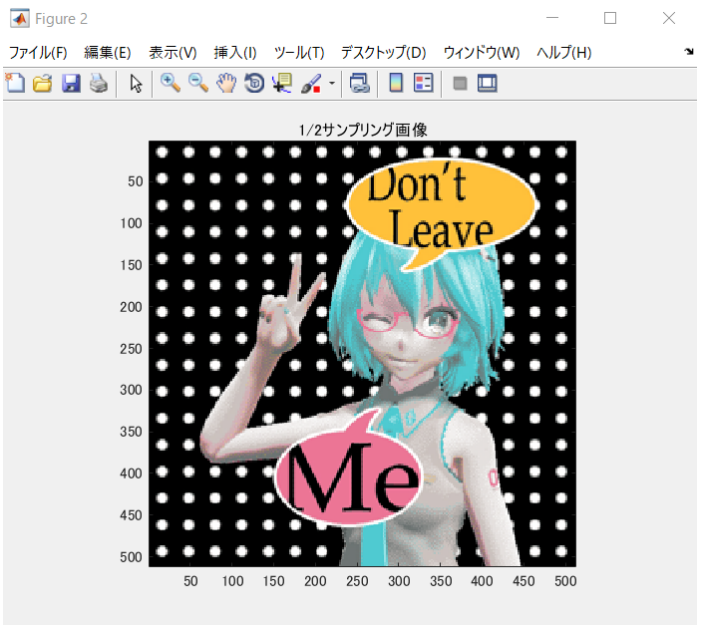

図 2 $\frac{1}{2}$サンプリング画像

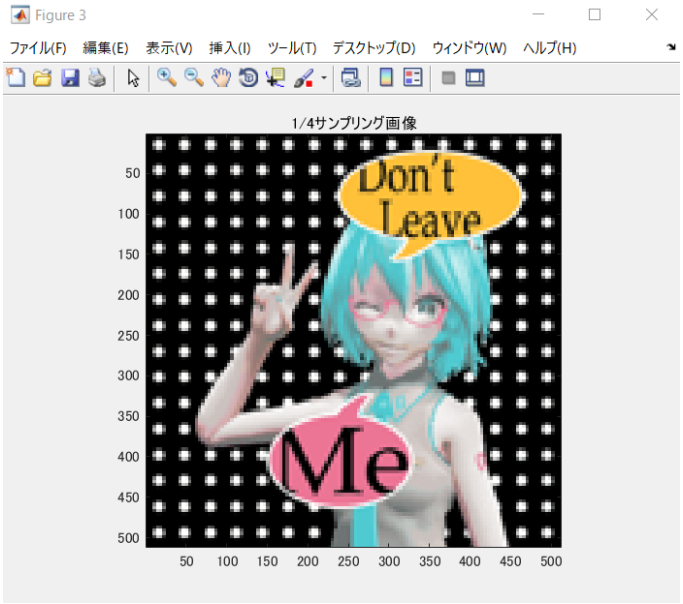

図 3 $\frac{1}{4}$サンプリング画像

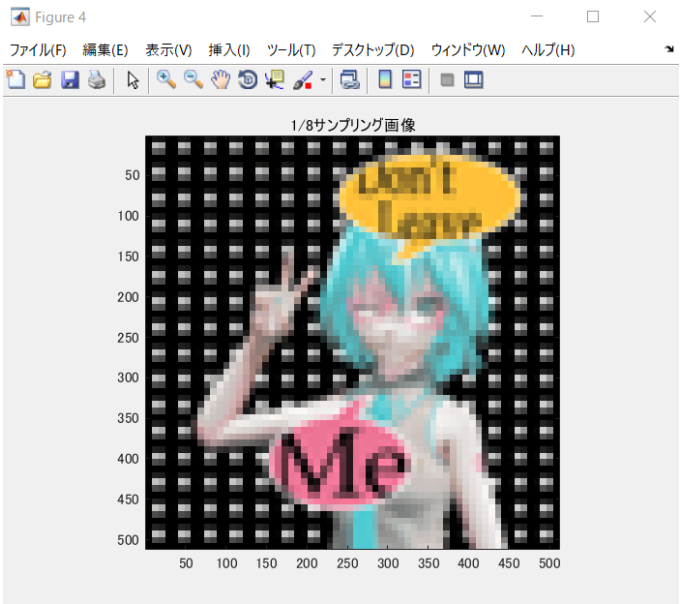

図 4 $\frac{1}{8}$サンプリング画像

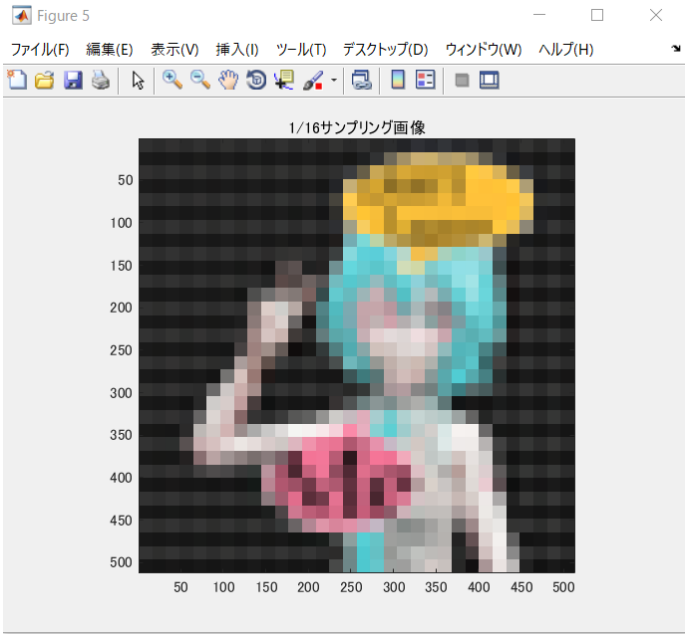

図 5 $\frac{1}{16}$サンプリング画像

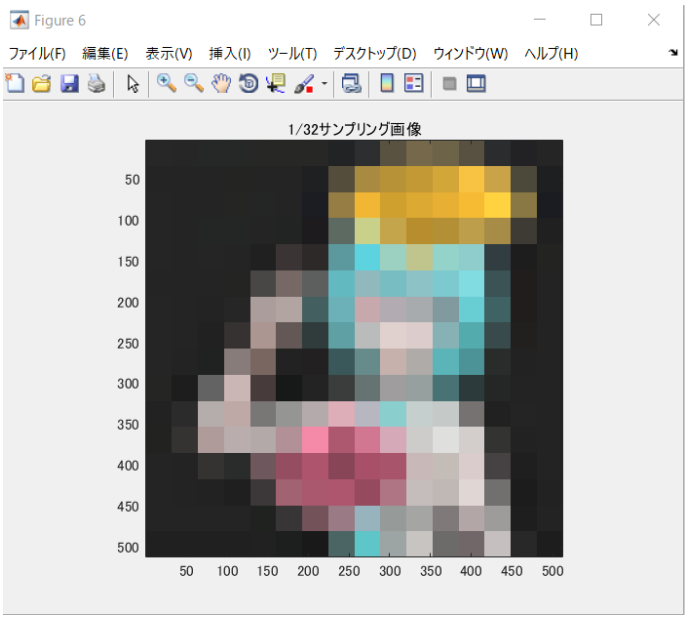

図 6 $\frac{1}{32}$サンプリング画像

課題 2

リスト 3 に 2 階調、4 階調、8 階調画像生成プログラムを示す。なお、Adobe AfterEffects および Adobe Photoshop で作成した「miku.jpg」を原画像とする。この画像は縦 512 画像、横 512 画素による正方形のディジタルカラー画像である。

```
% 課題２　階調数と疑似輪郭
% ２階調，４階調，８階調の画像を生成せよ．
% 下記はサンプルプログラムである．
% 課題作成にあたっては「Lenna」以外の画像を用いよ．

clear all; close all % 変数のオールクリア

ORG=imread('miku.jpg'); % 原画像の入力
ORG = rgb2gray(ORG); colormap(gray); colorbar;
figure(1)
imagesc(ORG); axis image; % 画像の表示
title('原画像')

% ２階調画像の生成
IMG = ORG>128;
figure(2)
imagesc(IMG); colormap(gray); colorbar;　axis image;
title('2 階調画像')

% ４階調画像の生成

IMG0 = ORG>64;
IMG1 = ORG>128;
IMG2 = ORG>192;
IMG = IMG0 + IMG1 + IMG2;
figure(3)
imagesc(IMG); colormap(gray); colorbar;　axis image;
title('4 階調画像')

% ８階調については，各自検討してください．
```

```
IMG=0;
for i=1:7
    xx=ORG>32*i;
    IMG=IMG+xx;
end
figure(4)
imagesc(IMG); colormap(gray); colorbar;  axis image;
title('8 階調画像')

return
```

リスト 3 2 階調、4 階調、8 階調画像生成プログラム

リスト 3 では、原画像を一定の閾値の大小によって出力して、それぞれを足し合わせることで 2 階調、4 階調、8 階調の画像を生成している。8 階調については、リスト 4 のように閾値を設定する際に for 文を用いて簡略化して画像を生成した。

```
for i=1:7
    xx=ORG>32*i;
    IMG=IMG+xx;
end
```

リスト 4 8 階調画像生成コード

また、図 7 から図 10 を比較すると、階調の数が増えるにつれて濃淡の表現が原画像に近づいていることがわかる。

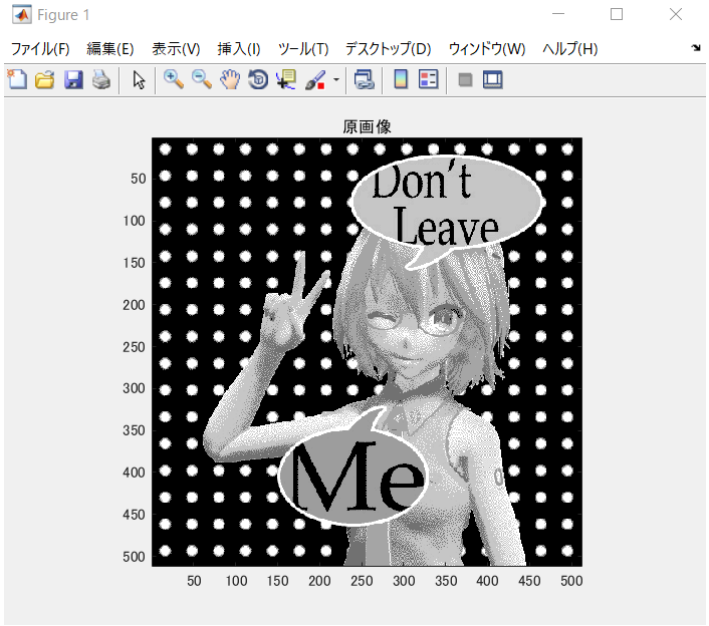

図 7 原画像

図 8 2 階調画像

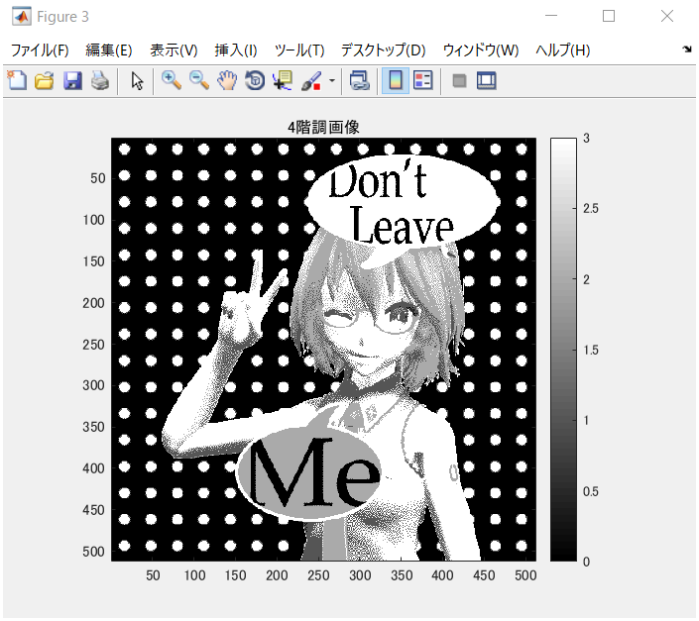

図 9 4 階調画像

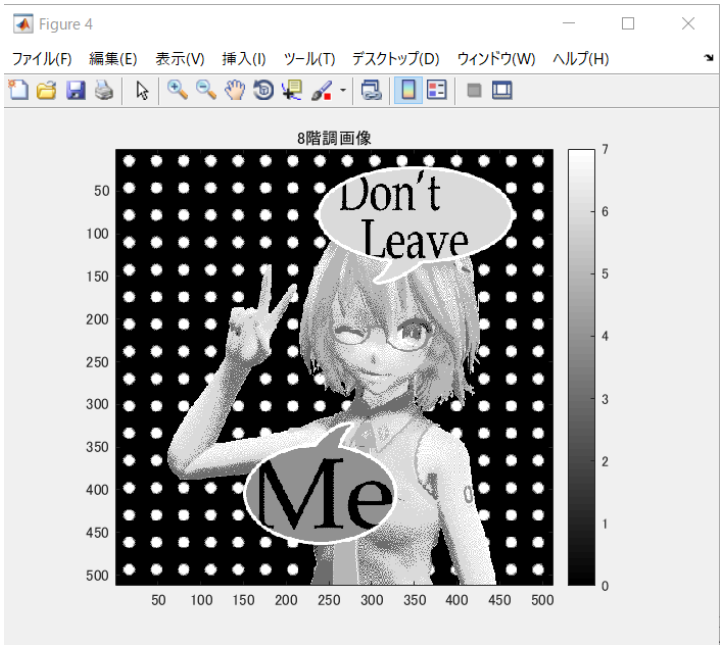

図 10 8 階調画像

課題 3

リスト 5 に閾値処理プログラムを示す。なお、Adobe AfterEffects および Adobe Photoshop で作成した「miku.jpg」を原画像とする。この画像は縦 512 画像、横 512 画素による正方形のディジタルカラー画像である。

```
% 課題３　閾値処理
% 閾値を 4 パターン設定し,閾値処理た画像を示せ.
% 下記はサンプルプログラムである.
% 課題作成にあたっては「Lenna」以外の画像を用いよ.

clear all; close all % 変数のオールクリア

ORG=imread('miku.jpg'); % 原画像の入力
figure(1)
ORG= rgb2gray(ORG); % カラー画像を白黒濃淡画像へ変換
imagesc(ORG); colormap(gray); colorbar; % 画像の表示
title('モノクロ原画像')

IMG = ORG > 64; % 輝度値が 64 以上の画素を 1，その他を 0 に変換
figure(2)
imagesc(IMG); colormap(gray); colorbar;
title('輝度値が 64 以上の画素を 1 にした画像')

IMG = ORG > 96;
figure(3)
imagesc(IMG); colormap(gray); colorbar;
title('輝度値が 96 以上の画素を 1 にした画像')

IMG = ORG > 128;
figure(4)
imagesc(IMG); colormap(gray); colorbar;
title('輝度値が 128 以上の画素を 1 にした画像')

IMG = ORG > 192;
figure(5)
```

```
imagesc(IMG); colormap(gray); colorbar;
title('輝度値が 192 以上の画素を 1 にした画像')


return
```

リスト 5 閾値処理プログラム

リスト 5 では、ORG のそれぞれのピクセルにおいて閾値以上の輝度を持つ要素を 1 としている。図 11 から 15 にそれぞれの実行結果を示す。図 11 から 15 より、閾値が大きい画像ほど黒くなっている範囲大きいことが確認できる。

図 11 原画像

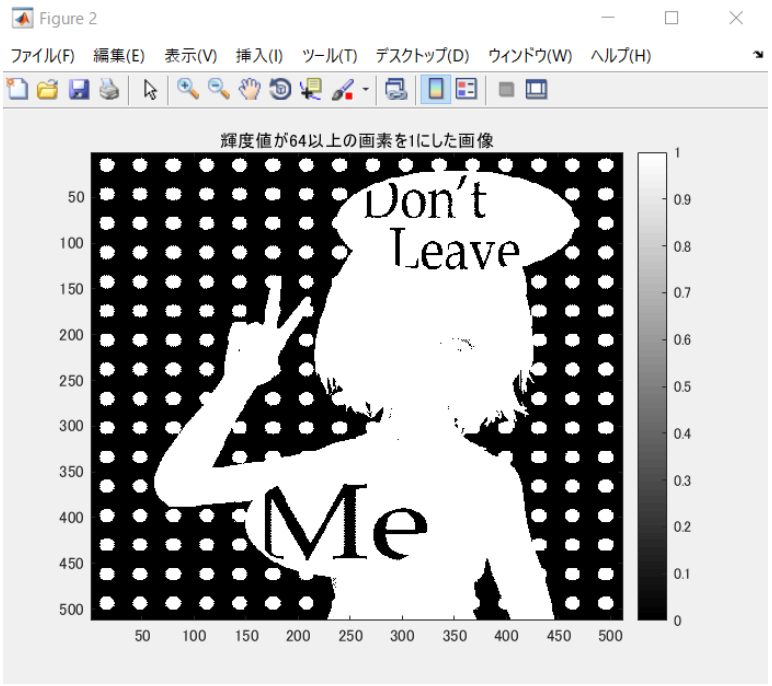

図 12 閾値が 32 のときの画像

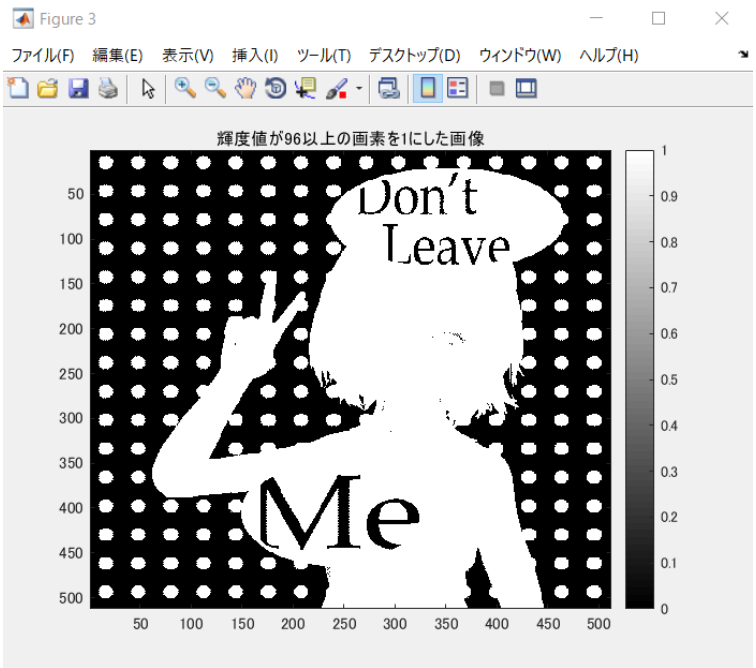

図 13 閾値が 64 のときの画像

図 14 閾値が 128 のときの画像

図 15 閾値が 192 のときの画像

課題 4

リスト 6 に画素の濃淡ヒストグラム表示プログラムを示す。なお、Adobe AfterEffects および Adobe Photoshop で作成した「miku.jpg」を原画像とする。この画像は縦 512 画像、横 512 画素による正方形のディジタルカラー画像である。

```
% 課題４　画像のヒストグラム
% 画素の濃度ヒストグラムを生成せよ.
% 下記はサンプルプログラムである.
% 課題作成にあたっては「Lenna」以外の画像を用いよ.

clear; % 変数のオールクリア

ORG=imread('miku.jpg'); % 原画像の入力
ORG=rgb2gray(ORG); % カラー画像を白黒濃淡画像へ変換
imagesc(ORG); colormap(gray); colorbar;
pause;

imhist(ORG); % ヒストグラムの表示
```

リスト 6 画素の濃度ヒストグラム表示プログラム

画素の濃淡はリスト 7 のコードで実現できる。

```
imhist(ORG);
```

リスト 7 画素の濃度ヒストグラム表示コード

実行結果を図 16、図 17 に示す。図 17 より、正しくヒストグラムが表示されていることがわかる。

図 16 原画像

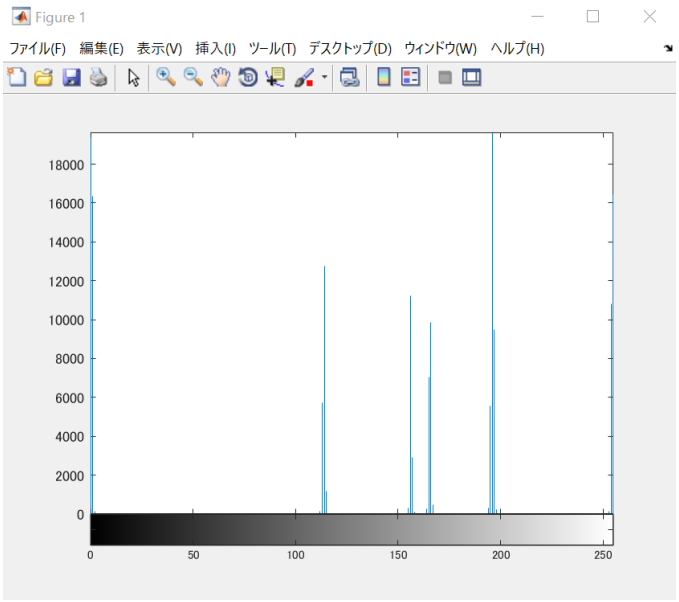

図 17 原画像の画素の濃度ヒストグラム

課題 5

リスト 8 に判別分析法を用いて画像を二値化するプログラムを示す。なお、Adobe AfterEffects および Adobe Photoshop で作成した「miku.jpg」を原画像とする。この画像は縦 512 画像、横 512 画素による正方形のディジタルカラー画像である。

```
% 課題５　判別分析法
% 判別分析法を用いて画像二値化せよ.
% 下記はサンプルプログラムである.
% 課題作成にあたっては「Lenna」以外の画像を用いよ.

ORG=imread('miku.jpg'); % 原画像の入力
ORG = rgb2gray(ORG); % カラー画像を白黒濃淡画像へ変換
imagesc(ORG); colormap(gray); colorbar;
pause;

H = imhist(ORG); %ヒストグラムのデータを列ベクトル E に格納
myu_T = mean(H);
max_val = 0;
max_thres = 1;
for i=1:255
C1 = H(1:i); %ヒストグラムを 2 つのクラスに分ける
C2 = H(i+1:256);
n1 = sum(C1); %画素数の算出
n2 = sum(C2);
myu1 = mean(C1); %平均値の算出
myu2 = mean(C2);
sigma1 = var(C1); %分散の算出
sigma2 = var(C2);
sigma_w = (n1 *sigma1+n2*sigma2)/(n1+n2); %クラス内分散の算出
sigma_B = (n1 *(myu1-myu_T)^2+n2*(myu2-myu_T)^2)/(n1+n2); %クラス間分散
の算出
if max_val<sigma_B/sigma_w
max_val = sigma_B/sigma_w;
max_thres =i;
end;
```

```
end;

IMG = ORG > max_thres;
imagesc(IMG); colormap(gray); colorbar;
pause;
```

リスト 8 判別分析法を用いて画像を二値化するプログラム

判別分析法は、対象物の濃度と背景の濃度がそれぞれ最もよくまとまり、かつ対象物と背景とに違いが際立つように閾値を定める方法である。画像をある閾値で 2 つのクラスに分けた時のクラス内分散とクラス間分散を求め、クラス内分散とクラス間分散の比が最も大きくなる閾値を計算している。リスト 8 では、for 文内でこれらの計算を行い、8 ビットの濃度のなかでクラス内分散とクラス間分散の比が最大となるとき変数 max_val に代入している。また、その閾値を変数 max_thres に代入している。その後 IMG = ORG > max_thres とすることで、判別分析法を用いた時の IMG を表示している。リスト 8 の実行結果を図 18、図 19 に示す。

図 18 原画像

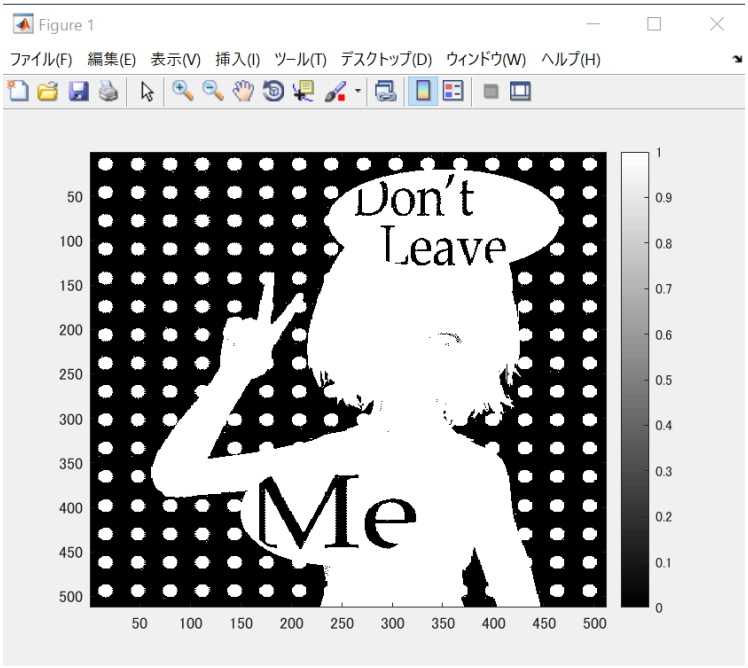

図 19 判別分析法を用いた 2 値化画像

課題 6

リスト 9 にディザ法を用いて画像を二値化するプログラムを示す。なお、Adobe AfterEffects および Adobe Photoshop で作成した「miku.jpg」を原画像とする。この画像は縦 512 画像、横 512 画素による正方形のディジタルカラー画像である。

```
% 課題６　画像の二値化
% 下記のプログラムを参考にして画像を二値化せよ.
% 下記はサンプルプログラムである.
% 課題作成にあたっては「Lenna」以外の画像を用いよ.

clear; % 変数のオールクリア
ORG=imread('miku.jpg'); % 原画像の入力
ORG = rgb2gray(ORG);
imagesc(ORG); colormap(gray); colorbar; % 画像の表示
pause; % 一時停止



IMG = ORG>128; % 128 による二値化
imagesc(IMG); colormap(gray); colorbar; % 画像の表示
pause;

IMG = dither(ORG); % ディザ法による二値化
imagesc(IMG); colormap(gray); colorbar; % 画像の表示
```

リスト 9 ディザ法を用いて画像を二値化するプログラム

ディザ法は、原画像の各画素の濃度値をあらかじめ定められたディザマトリクスの値(しきい値)と比較し、その大小で出力画素の濃度値を決定する方法である。しきい値の決定法には、平均値制限法、ランダムディザ法、組織的ディザ法がある。ディザ法は解像度成分を保存したまま階調成分が表示できる方法である。図 20 から 22 にプログラムの実行結果を示す。図 22 と図 19 を比較すると、ディザ法を用いた 2 値化画像の方が塗られている面積が大きいことが分かる。

図 20 原画像

図 21 閾値を 128 に設定した 2 値化画像

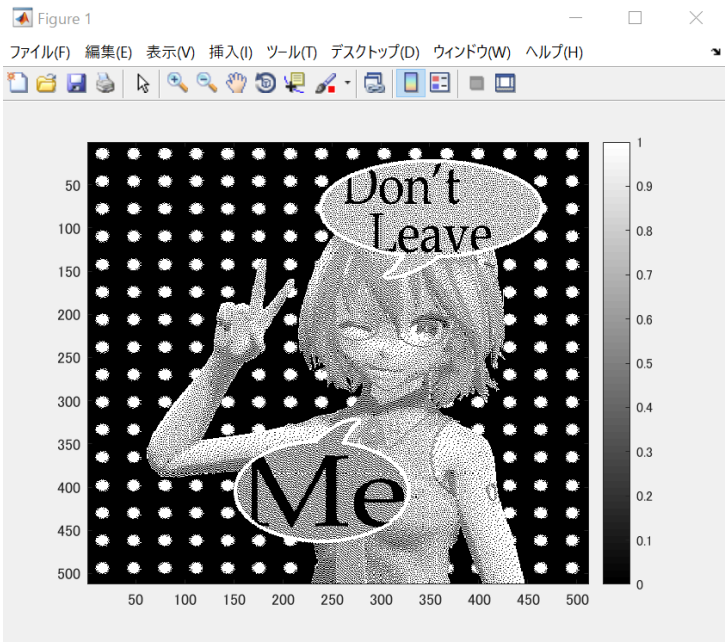

図 22 ディザ法を用いた 2 値化画像

課題 7

リスト 10 にダイナミックレンジを拡大するプログラムを示す。なお、Adobe AfterEffects および Adobe Photoshop で作成した「miku.jpg」を原画像とする。この画像は縦 512 画像、横 512 画素による正方形のディジタルカラー画像である。

```
% 課題 7　ダイナミックレンジの拡大
% 画素のダイナミックレンジを０から２５５にせよ.
% 下記はサンプルプログラムである.
% 課題作成にあたっては「Lenna」以外の画像を用いよ.
% 例

ORG = imread('miku.jpg'); % 画像の読み込み
ORG = rgb2gray(ORG); % 白黒濃淡画像に変換
imagesc(ORG); colormap(gray); colorbar; % 画像の表示
pause;
imhist(ORG); % 濃度ヒストグラムを生成、表示
pause;
ORG = double(ORG);
mn = min(ORG(:)); % 濃度値の最小値を算出
mx = max(ORG(:)); % 濃度値の最大値を算出
ORG = (ORG-mn)/(mx-mn)*255;
imagesc(ORG); colormap(gray); colorbar; % 画像の表示
pause;
ORG = uint8(ORG); % この行について考察せよ
imhist(ORG); % 濃度ヒストグラムを生成、表示
```

リスト 10 ダイナミックレンジの拡大プログラム

それぞれの実行結果を図 23 から図 26 に示す。リスト 10 のプログラムでは、リスト 11 部の ORG に格納された原画像の濃度値の最小値と最大値を用いてダイナミックレンジを拡大している。

```
ORG = double(ORG);
mn = min(ORG(:)); % 濃度値の最小値を算出
mx = max(ORG(:)); % 濃度値の最大値を算出
```

```
ORG = (ORG-mn)/(mx-mn)*255;
```

リスト 11 ダイナミックレンジの拡大コード

また、リスト 12 部について、uint8(ORG)は ORG を 8 ビットの符号なし整数値に変換したものが ORG に格納されている。よって、$2^8 = 256$であるから、ORG のダイナミックレンジが 0 から 255 までに変換されている。

```
ORG = uint8(ORG); %  この行について考察せよ
```

リスト 12 ダイナミックレンジを 0 から 255 に変更

図 23 原画像

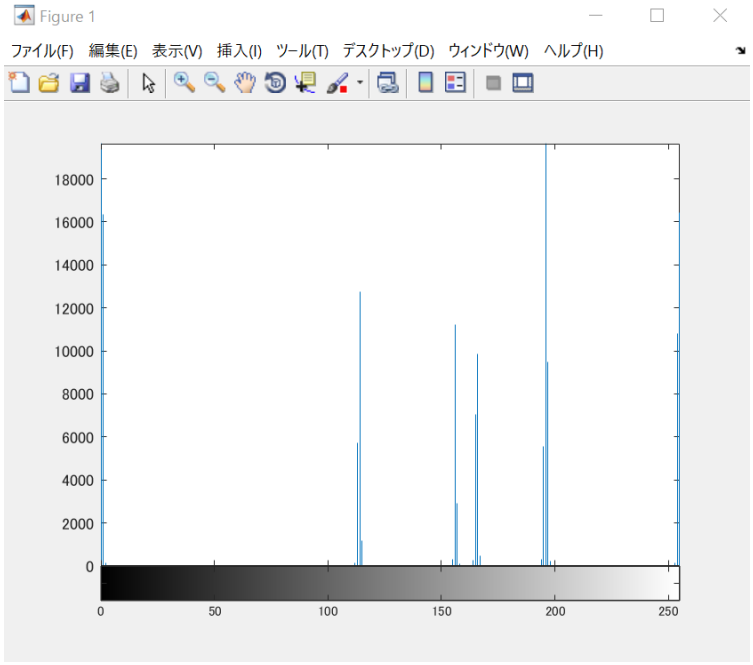

図 24 原画像のヒストグラム

図 25 ダイナミックレンジを 0 から 255 に変更した画像

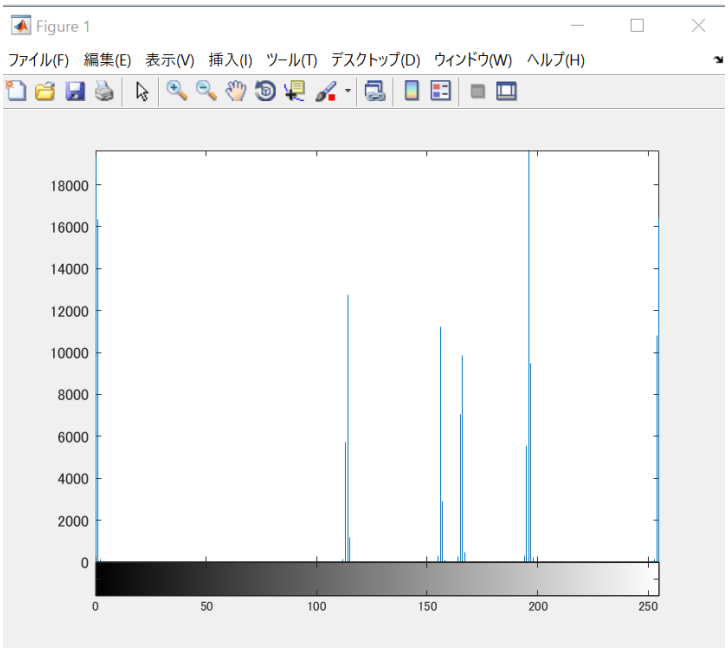

図 26 ダイナミックレンジを 0 から 255 に変更した画像のヒストグラム

課題 8

リスト 13 に二値化された画像の連結成分にラベルをつけるプログラムを示す。なお、Adobe AfterEffects および Adobe Photoshop で作成した「miku.jpg」を原画像とする。この画像は縦 512 画像、横 512 画素による正方形のディジタルカラー画像である。

```
% 課題８ ラベリング
% 二値化された画像の連結成分にラベルをつけよ.
% 下記はサンプルプログラムである.
% 課題作成にあたっては「Lenna」以外の画像を用いよ.
% 例

ORG = imread('miku.jpg'); % 画像の読み込み
ORG = rgb2gray(ORG); % 白黒濃淡画像に変換
imagesc(ORG); colormap(gray); colorbar; % 画像の表示
pause;
IMG = ORG > 128; % 閾値 128 で二値化
imagesc(IMG); colormap(gray); colorbar; % 画像の表示
pause;
IMG = bwlabeln(IMG);
imagesc(IMG); colormap(jet); colorbar; % 画像の表示
pause;
```

リスト 13 二値化された画像の連結成分にラベルをつけるプログラム

図 27 から図 29 に実行結果を示す。リスト 14 部について、bwlabeln(IMG)は連結要素に対するラベルを含むラベル行列を返す関数である。図 27 に着目すると、背景の水玉模様が赤く色づいていることがわかる。

```
IMG = bwlabeln(IMG);
```

リスト 14 二値化された画像の連結成分にラベルをつけるコード

図 27 原画像

図 28 白黒濃淡画像に変換した画像

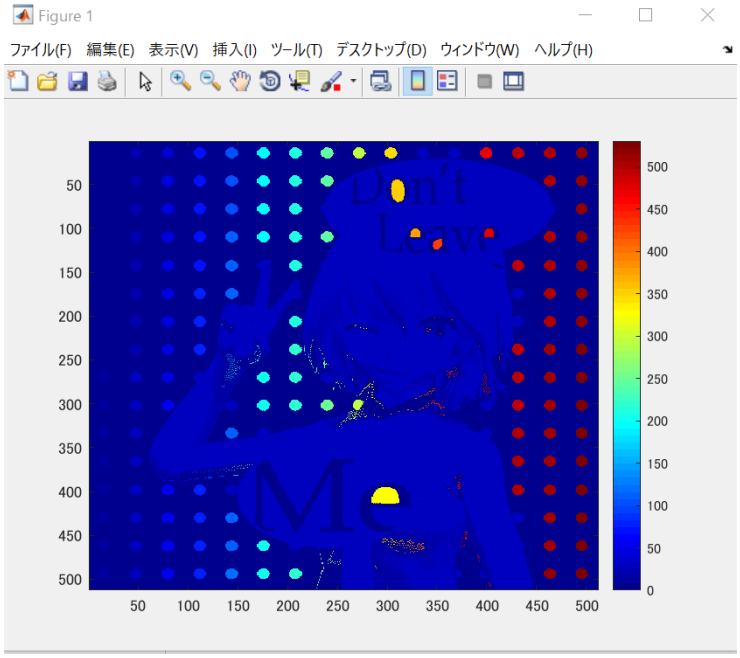

図 29 二値化された画像の連結成分の表示画像

課題 9

リスト 15 にノイズ除去プログラムを示す。なお、Adobe AfterEffects および Adobe Photoshop で作成した「miku.jpg」を原画像とする。この画像は縦 512 画像、横 512 画素による正方形のディジタルカラー画像である。

```
% 課題９ メディアンフィルタと先鋭化
% メディアンフィルターを適用し，ノイズ除去を体験せよ．
% 各自，Lenna 以外の画像を用いよ．
% 例

ORG = imread('miku.jpg'); % 画像の読み込み
ORG = rgb2gray(ORG); % 白黒濃淡画像に変換
imagesc(ORG); colormap(gray); colorbar; % 画像の表示
pause;
ORG = imnoise(ORG,'salt & pepper',0.02); % ノイズ添付
imagesc(ORG); colormap(gray); colorbar; % 画像の表示
pause;
IMG = filter2(fspecial('average',3),ORG); % 平滑化フィルタで雑音除去
imagesc(IMG); colormap(gray); colorbar; % 画像の表示
pause;
IMG = medfilt2(ORG,[3 3]); % メディアンフィルタで雑音除去
imagesc(IMG); colormap(gray); colorbar; % 画像の表示
pause;
f=[0,-1,0;-1,5,-1;0,-1,0]; % フィルタの設計
IMG = filter2(f,IMG,'same'); % フィルタの適用
imagesc(IMG); colormap(gray); colorbar; % 画像の表示
pause;
```

リスト 15 ノイズ除去プログラム

それぞれの実行結果を図 30 から 34 に示す。リスト 16 部について、メディアンフィルタは変換後の濃度値を着目画素の近傍画素の濃度値を平均値とするのではなく、それらの画素濃度の中央値とする方法である。メディアンフィルタを用いることで、エッジをぼかすことなくノイズの除去を行うことができる。

```
IMG = medfilt2(ORG,[3 3]); %  メディアンフィルタで雑音除去
```

リスト 16 メディアンフィルタでノイズを除去するコード

リスト 15 のプログラムでは、平滑化フィルタ、メディアンフィルタ、設計したフィルタの 3 通りの方法でノイズを除去した。図 32 の平滑化フィルタでは、ノイズの軽減は行えているが若干残っている。図 33 のメディアンフィルタではきれいにノイズ除去が行われているが、原画像と比較するとのっぺりした画像となっている。図 34 の設計したフィルタでは、概ねきれいにノイズ除去が行えているが、原画像と比較すると画像全体の濃度が変化している。このことから、それぞれのフィルタの特徴を把握したうえでノイズ除去を行う必要があるとわかる。また、ノイズ除去を行っても原画像そのものに戻るわけではないため、使用する際には注意が必要である。

図 30 原画像

図 30 ノイズを付加した画像

図 31 平滑化フィルタを用いてノイズを除去した画像

図 33 メディアンフィルタを用いてノイズを除去した画像

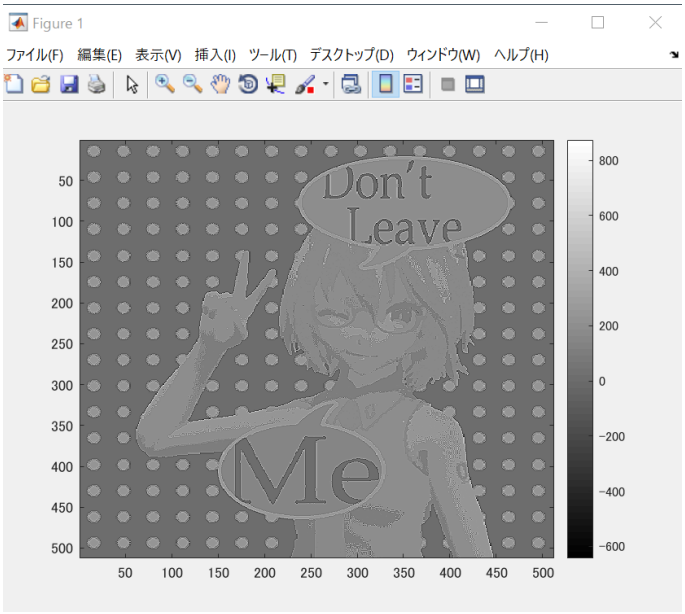

図 34 設計したフィルタを用いてノイズを除去した画像

課題 10

リスト 17 に画像のエッジ抽出プログラムを示す。なお、Adobe AfterEffects および Adobe Photoshop で作成した「miku.jpg」を原画像とする。この画像は縦 512 画像、横 512 画素による正方形のディジタルカラー画像である。

```
% 課題 10 画像のエッジ抽出
% 次のプログラムを参考にして，エッジ抽出を体験せよ．
% 各自，Lenna 以外の画像を用いよ．
% 例

ORG = imread('miku.jpg'); % 原画像の入力
ORG = rgb2gray(ORG); %カラーからグレイへの変換
imagesc(ORG); colormap('gray'); colorbar;% 画像表示
pause; % 一時停止

IMG = edge(ORG,'prewitt'); % エッジ抽出（プレウィット法）
imagesc(IMG); colormap('gray'); colorbar;% 画像表示
pause; % 一時停止

IMG = edge(ORG,'sobel'); % エッジ抽出（ソベル法）
imagesc(IMG); colormap('gray'); colorbar;% 画像表示
pause; % 一時停止

IMG = edge(ORG,'canny'); % エッジ抽出（キャニー法）
imagesc(IMG); colormap('gray'); colorbar;% 画像表示
pause; % 一時停止
```

リスト 17 画像のエッジ抽出プログラム

それぞれの実行結果を図 35 から図 38 に示す。画像から対象物の輪郭部の画素を取り出すことをエッジ検出という。リスト 17 のプログラムでは、プレウィット法、ソベル法、キャニー法の 3 種類の方法でエッジ抽出を行っている。

図 35 原画像

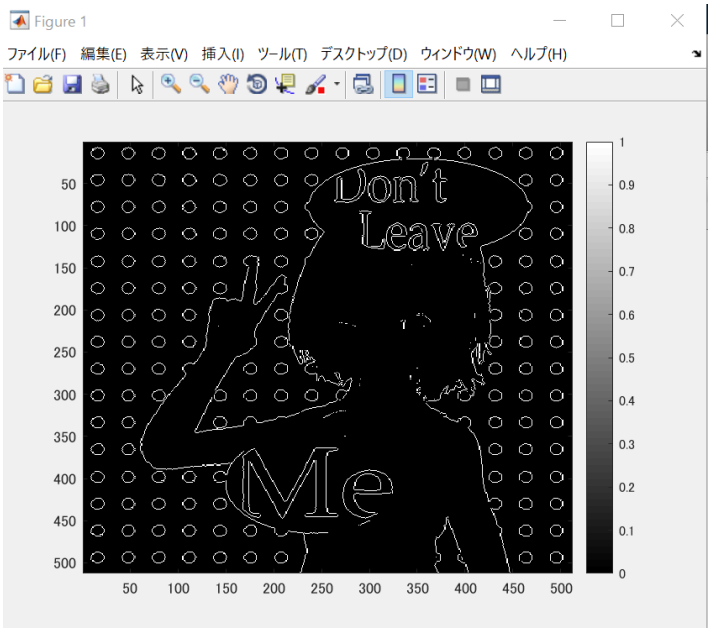

図 36 プレウィット法を用いたエッジ抽出画像

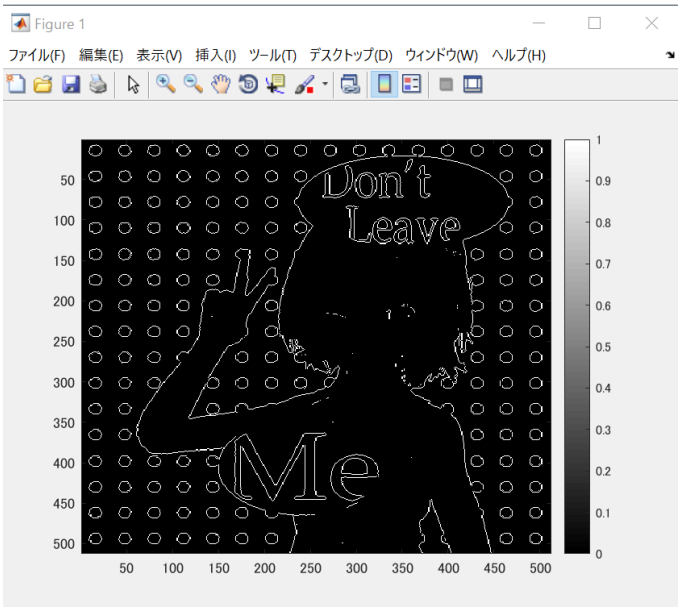

図 37 ソベル法を用いたエッジ抽出画像

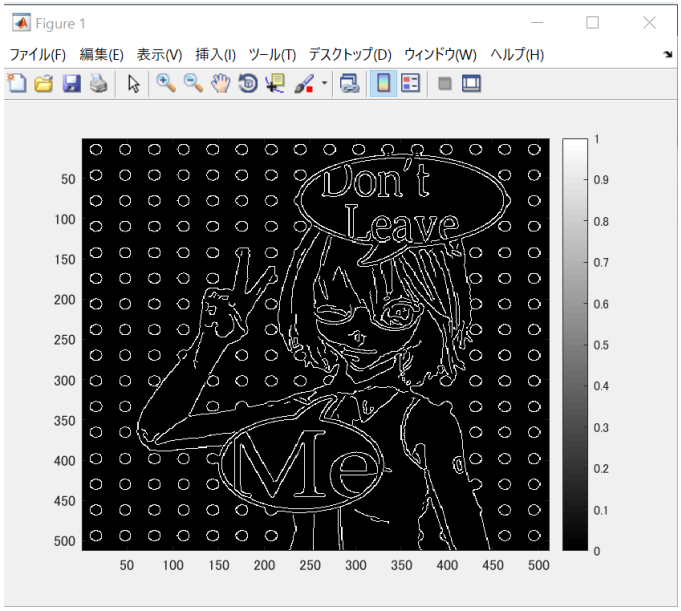

図 38 キャニー法を用いたエッジ抽出画像