1 Matlab correction

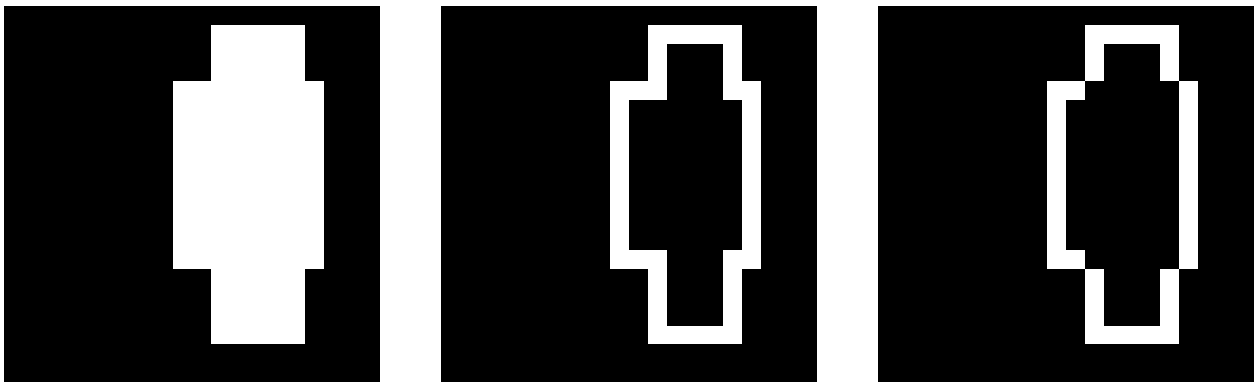
1.1 Shape contours

To generate a simple object, here is an example:

```
A=zeros(20,20);
A(5:14,10:17)=1;A(2:18,12:16)=1;
```

The contours are computed in 4- or 8-connectivity, see Fig.1.

```
% compute the contours
contours8 = bwperim(A, 4);
contours4 = bwperim(A, 8);
```



(a) Sample object. (b) Contour in 4-connectivity. (c) Contour in 8-connectivity.

Figure 1: Simple object and its contours in 4- or 8-connectivity.

1.2 Freeman chain code

1.2.1 First point of the shape

```
function [x,y]=firstPoint(A)
% locates first non zero point of the contour A
[r,c]=find(A);
x=r(1); y=c(1);
```

Command window

```
>>[r0,c0]=firstPoint(A)

r0 =
     5
c0 =
    10
```

1.2.2 Freeman chain code

```
function code=freeman(A,r0,c0,conn)
% freeman code of a contour.
% A : contour
% r0, c0 : coordinates of 1st point
% conn: connectivity

B=A;
stop=0; % stop condition
point0=[r0,c0];
if (conn==8)
    lut=[1 2 3; 8 0 4; 7 6 5];
else
    lut=[0 2 0; 8 0 4; 0 6 0];
end

% be careful that these LUTs consider coordinates
% from left to right, top to bottom
%  0
% 0+------- y
%  |
%  |
%  |
% x|
%
lutx=[-1 -1 -1 0 1 1 1 0];
luty=[-1 0 1 1 1 0 -1 -1];
lutcode=[3 2 1 0 7 6 5 4];

nbrepoints=sum(B(:));
code=[];
point=point0;

for indice = 1:nbrepoints
    B(point(1),point(2))=0;
    window=B(point(1)-1:point(1)+1,point(2)-1:point(2)+1);
    window=window.*lut;
    index=max(window(:));
    if (index==0) % no more points ? should link to first point
```

```
            B(point0(1),point0(2))=1;
            window=B(point(1)-1:point(1)+1,point(2)-1:point(2)+1);
            window=window.*lut;
            index=max(window(:));
            B(point0(1),point0(2))=0;
        end

        % compute coordinates of new point
        point=[point(1)+lutx(index),point(2)+luty(index)];

        % add code
        code(indice) = lutcode(index);

end
```

```
Command window

z4= 6 6 6 6 6 6 6 6 6 0 0 6 6 6 6 0 0 0 0 2 2 2 2 0 2 2 2 2 2 2 2 2 2 4 2
    2 2 4 4 4 4 6 6 6 4 4

z8= 6 6 6 6 0 0 0 7 0 0 0 0 0 0 0 0 0 1 0 0 2 2 2 2 4 4 3 2 4 4 4 4 4 4 4
    4 4 6 5 4 4 4
```

1.3 Normalization

1.3.1 Differential code

```
function d=codediff(fcc,conn)
% fcc : freeman chain code
% conn: connectivity
sr=circshift(fcc,1);
d=fcc-sr;
d = mod(d, conn);
```

1.3.2 Normalization

```
function code=freeman_normalization(fcc)
% find the lowest number constituted among all the cyclic translations of
% fcc: freeman code
L = length(fcc);
C = zeros(L);
```

```
for i=1:L
    C(i,:)=circshift(fcc, i);
end

% search for minimum number
% evaluates the value of the number formed by the array, thus, use
    ↪ polyval
% in decimal basis for getting this number
D= zeros(L, 1);
for i=1:L
    D(i) = polyval(C(i,:), 10);
end

[~, ind]=min(D);
code=C(ind(1),:);
```

The differential code is evaluated in d8, the normalization gives shapenumber8:

```
[r0,c0]=firstPoint(A);
z8=freeman(contours8,r0,c0,8);
d8=codediff(z8,8);
shapenumber8=freeman_normalization(d8);
```

```
Command window

d8 = 2 0 0 0 2 0 0 7 1 0 0 0 0 0 0 0 0 1 7 0 2 0 0 0 2 0 7 7 2 0 0 0 0 0
    0 0 0 2 7 7 0 0
shapenumber8 = 0 0 0 0 0 0 0 0 1 7 0 2 0 0 0 2 0 7 7 2 0 0 0 0 0 0 0 0 2
    7 7 0 0 2 0 0 0 2 0 0 7 1
```

1.3.3 Validation

This validation shows the effect on a different starting point.

```
% check for another starting point
r0=9;c0=10;
z8=freeman(contours8,r0,c0,8);
d8=codediff(z8,8);
shapenumber8_startchanged=freeman_normalization(d8);
disp('* validation test for starting point changed')
if (shapenumber8-shapenumber8_startchanged)
    disp('   error: different freeman code')
else
    disp('   OK: same freeman code')
```

```
end
```

Another test is to verify the result after a rotation. To prevent discretization problems, we use 90 degrees and take the transpose of the matrix.

```
% check for rotation by 90 deg
contours8rot=contours8';

figure;imshow(contours8rot);

[r0,c0]=firstPoint(contours8rot);
z8=freeman(contours8rot,r0,c0,8);
d8=codediff(z8,8);
shapenumber8_rotated=freeman_normalization(d8);
disp('* validation test after rotation')
if (shapenumber8-shapenumber8_rotated)
    disp('   error: different freeman code')
else
    disp('   OK: same freeman code')
end
```

The same code should be found:

```
Command window

* validation test for starting point changed
   OK: same freeman code
* validation test after rotation
   OK: same freeman code
```

1.4 Geometrical characterization

1.4.1 Perimeter for 8-connectivity

We first need to extract the codes in the diagonal directions and apply a $\sqrt{2}$ factor, then add the number of codes in vertical and horizontal directions.

```
nb_diag=mod(z8, 2);
nb_diag=sum(nb_diag(:));
nb = length(z8)-nb_diag;
perimeter = nb_diag * sqrt(2) + nb
stats = regionprops(A,'Perimeter')
```

Command window

```
perimeter =
   43.6569
stats =
  struct with fields:
    Perimeter: 41.5900
```

1.4.2 Area for 8-connectivity

```
area=0;
B=0;
lutB=[0 1 1 1 0 -1 -1 -1];
for i=1:length(z8)
    lutArea=[-B -(B+0.5) 0 (B+0.5) B (B-0.5) 0 -(B-0.5)];
    area=area+lutArea(z8(i)+1);
    B=B+lutB(z8(i)+1);
end
disp(['Area by freeman code: ' num2str(area)]);
disp(['Number of pixels: ' num2str(sum(A(:)))])
```

Command window

```
Area by freeman code: 93
Number of pixels: 115
```