```
function Histogram = CubicLBP(VolData, FxRadius, FyRadius, TInterval, NeighborPoints, TimeLength, BorderLength, bBilinearInterpolation, Bincount, Code)


% This function is to compute the Cubic-LBP features for a video sequence

% If you use this MATLAB code, please cite the following publication:

% V. Esmaeili and S.O. Shahdi, “Automatic micro-expression apex spotting using Cubic-LBP,” Multimedia Tools and Applications, pp. 1-9, 2020.

%

% Reference: (The original paper):

% V. Esmaeili and S.O. Shahdi, “Automatic micro-expression apex spotting using Cubic-LBP,” Multimedia Tools and Applications, pp. 1-9, 2020.

% % Copyright 2020 by Vida Esmaeili & Seyed Omid Shahdi

% Matlab version was Created by Vida Esmaeili

% If you have any problem, please feel free to contact Vida Esmaeili or Seyed Omid Shahdi.

% Seyed Omid Shahdi:

% shahdi@qiau.ac.ir

% Vida Esmaeili:

% V.Esmaeili@qiau.ac.ir

%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%

% Function: Running this funciton each time to compute the Cubic-LBP distribution of one video sequence.

%

% Inputs:

%

% "VolData" keeps the grey level of all the pixels in sequences with [height][width][Length];

% please note, all the images in one sequnces should have same size (height and weight).

% But they don't have to be same for different sequences.

%

% "FxRadius", "FyRadius" and "TInterval" are the radii parameter along X, Y and T axis; They can be 1, 2, 3 and 4. "1" and "3" are recommended.
```

% Pay attention to "TInterval". "TInterval * 2 + 1" should be smaller than the length of the input sequence "Length".

%

%

% "NeighborPoints" is the number of the neighboring points

% in plane1, plane2, plane3, plane4, plane5, plane6, plane7, plane8, plane9, plane10, plane11, plane12, plane13, plane14 and plane15; They can be 4, 8, 16 and 24. "8"

% is a good option. For example, NeighborPoints = [8 8 8 8 8 8 8 8 8 8 8 8 8 8 8];

%

% "TimeLength" and "BoderLength" are the parameters for bodering parts in time and space which would not

% be computed for features. Usually they are same to TInterval and the bigger one of "FxRadius" and "FyRadius";

%

% "bBilinearInterpolation": if use bilinear interpolation for computing a neighboring point in a circle: 1 (yes), 0 (no).

%

% "Bincount": For example, if XYNeighborPoints = XTNeighborPoints = YTNeighborPoints =...NeighborPoints= 8, you can set "Bincount" as "0" if you want to use basic LBP, or set "Bincount" as 59 if using uniform pattern of LBP,

% If the number of Neighboring points is different than 8, you need to change it accordingly as well as change the above "Code".

% "Code": only when Bincount is 59, uniform code is used.

% Output:

%

% "Histogram": keeps Cubic-LBP distribution of all the pixels in the current frame with [15][dim];

% here, "15" deote the fifteen planes of Cubic-LBP, i.e., plane1, plane2, ..., plane15.

% Each value of Histogram[i][j] is between [0,1] (it is normallised in

% this way)

%

%%

[height width Length] = size(VolData);

```
 assert(TInterval * 2 + 1<=Length, sprintf('TInterval * 2 + 1<=Length False! TInterval = %d, Length=%d',
TInterval, Length))

 assert(TimeLength + 1 <= Length - TimeLength)

 assert(BorderLength + 1 <= height - BorderLength)

 assert(BorderLength + 1 <= width - BorderLength)

XYNeighborPoints = NeighborPoints(1);

XTNeighborPoints = NeighborPoints(2);

YTNeighborPoints = NeighborPoints(3);

DiameterMNeighborPoints = NeighborPoints(4);

Diameter2NeighborPoints = NeighborPoints(5);

FormLURDNeighborPoints = NeighborPoints(6);

FormRULDNeighborPoints = NeighborPoints(7);

FormBUFDNeighborPoints = NeighborPoints(8);

FormFUBDNeighborPoints = NeighborPoints(9);

FormUPNeighborPoints = NeighborPoints(10);

FormDOWNNeighborPoints = NeighborPoints(11);

FRONTNeighborPoints = NeighborPoints(12);

BACKNeighborPoints = NeighborPoints(13);

FormRNeighborPoints = NeighborPoints(14);

FormLNeighborPoints = NeighborPoints(15);

if (Bincount == 0)

% normal code

nDim = 2^(YTNeighborPoints);

Histogram = zeros(15, nDim);

else

% uniform code

Histogram = zeros(15, Bincount); % Bincount = 59;

end

if (bBilinearInterpolation == 0)
```

```
for i = TimeLength + 1 : Length - TimeLength

for yc = BorderLength + 1 : height - BorderLength

for xc = BorderLength + 1 : width - BorderLength

CenterVal = VolData(yc, xc, i);

%% In plane1 (central plane that is parallel with front and back faces of cube (Fig. 2 part (a); paper: Automatic micro-expression apex spotting using Cubic-LBP; authors: Vida Esmaeili and Seyed Omid Shahdi; Multimedia Tools and Applications))

BasicLBP = 0;

FeaBin = 0;

for p = 0 : XYNeighborPoints - 1

X = floor(xc + FxRadius * cos((2 * pi * p) / XYNeighborPoints) + 0.5);

Y = floor(yc - FyRadius * sin((2 * pi * p) / XYNeighborPoints) + 0.5);

CurrentVal = VolData(Y, X, i);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(1, BasicLBP + 1) = Histogram(1, BasicLBP + 1) + 1;

else

Histogram(1, Code(BasicLBP + 1, 2) + 1) = Histogram(1, Code(BasicLBP + 1, 2) + 1) + 1;

end

%% In plane2 (XT plane) (Reference of the paper: V. Esmaeili and S.O. Shahdi, “Automatic micro-expression apex spotting using Cubic-LBP,” Multimedia Tools and Applications, pp. 1-9, 2020.)
```

```
BasicLBP = 0;

FeaBin = 0;

for p = 0 : XTNeighborPoints - 1

X = floor(xc + FxRadius * cos((2 * pi * p) / XTNeighborPoints) + 0.5);

Z = floor(i + TInterval * sin((2 * pi * p) / XTNeighborPoints) + 0.5);

CurrentVal = VolData(yc, X, Z);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(2, BasicLBP + 1) = Histogram(2, BasicLBP + 1) + 1;

else % uniform patterns

Histogram(2, Code(BasicLBP + 1, 2) + 1) = Histogram(2, Code(BasicLBP + 1, 2) + 1) + 1;

end

%% In plane3 (central plane in the direction of the YT axes (Fig. 2 part (c)) in the paper) (Reference of the paper: V. Esmaeili and S.O. Shahdi, “Automatic micro-expression apex spotting using Cubic-LBP,” Multimedia Tools and Applications, pp. 1-9, 2020.)

BasicLBP = 0;

FeaBin = 0;

for p = 0 : YTNeighborPoints - 1

Y = floor(yc - FyRadius * sin((2 * pi * p) / YTNeighborPoints) + 0.5);

Z = floor(i + TInterval * cos((2 * pi * p) / YTNeighborPoints) + 0.5);
```

```
CurrentVal = VolData(Y, xc, Z);
if CurrentVal >= CenterVal
BasicLBP = BasicLBP + 2 ^ FeaBin;
end
FeaBin = FeaBin + 1;
end
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(3, BasicLBP + 1) = Histogram(3, BasicLBP + 1) + 1;
else
Histogram(3, Code(BasicLBP + 1, 2) + 1) = Histogram(3, Code(BasicLBP + 1, 2) + 1) + 1;
end
```

%%%%%%%In plane4 (first diagonal plane (45 degrees) in Fig. 2 part (i))(Reference of the paper: V. Esmaeili and S.O. Shahdi, “Automatic micro-expression apex spotting using Cubic-LBP,” Multimedia Tools and Applications, pp. 1-9, 2020.)

```
BasicLBP = 0;
FeaBin = 0;
for p = 0 : DiameterMNeighborPoints - 1
X = floor(xc + FxRadius * cos((2 * pi * p) / DiameterMNeighborPoints) + 0.5);
Y = floor(yc - FyRadius * sin((2 * pi * p) / DiameterMNeighborPoints) + 0.5);
Z = floor(i + TInterval * cos((2 * pi * p) / DiameterMNeighborPoints) + 0.5);
CurrentVal = VolData(Y, X, Z);
if CurrentVal >= CenterVal
BasicLBP = BasicLBP + 2 ^ FeaBin;
```

```
end
FeaBin = FeaBin + 1;
end
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(4, BasicLBP + 1) = Histogram(4, BasicLBP + 1) + 1;
else
Histogram(4, Code(BasicLBP + 1, 2) + 1) = Histogram(4, Code(BasicLBP + 1, 2) + 1) + 1;
end
%%%%%%%%%%
%%%%%%%In plane5 (the other diagonal plane (135 degrees) in Fig. 2 part (i))(Reference of the paper: V. Esmaeili and S.O. Shahdi, “Automatic micro-expression apex spotting using Cubic-LBP,” Multimedia Tools and Applications, pp. 1-9, 2020.)
BasicLBP = 0;
FeaBin = 0;
for p = 0 : Diameter2NeighborPoints - 1
X = floor(xc + FxRadius * cos((2 * pi * p) / Diameter2NeighborPoints) + 0.5);
Y = floor(yc - FyRadius * sin((2 * pi * p) / Diameter2NeighborPoints) + 0.5);
Z = floor(i - TInterval * cos((2 * pi * p) / Diameter2NeighborPoints) + 0.5);
CurrentVal = VolData(Y, X, Z);
if CurrentVal >= CenterVal
BasicLBP = BasicLBP + 2 ^ FeaBin;
end
FeaBin = FeaBin + 1;
end
```

```
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(5, BasicLBP + 1) = Histogram(5, BasicLBP + 1) + 1;
else
Histogram(5, Code(BasicLBP + 1, 2) + 1) = Histogram(5, Code(BasicLBP + 1, 2) + 1) + 1;
end
%%%%%%%%%%
%%%%%%%In plane6 (Fig.2 part (f) in the paper)(Reference of the paper: V. Esmaeili and S.O. Shahdi, "Automatic micro-expression apex spotting using Cubic-LBP," Multimedia Tools and Applications, pp. 1-9, 2020.)
BasicLBP = 0;
FeaBin = 0;
for p = 0 : FormLURDNeighborPoints - 1
X = floor(xc - FxRadius * sin((2 * pi * p) / FormLURDNeighborPoints) + 0.5);
Y = floor(yc - FyRadius * sin((2 * pi * p) / FormLURDNeighborPoints) + 0.5);
Z = floor(i + TInterval * cos((2 * pi * p) / FormLURDNeighborPoints) + 0.5);
CurrentVal = VolData(Y, X, Z);
if CurrentVal >= CenterVal
BasicLBP = BasicLBP + 2 ^ FeaBin;
end
FeaBin = FeaBin + 1;
end
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
```

```
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(6, BasicLBP + 1) = Histogram(6, BasicLBP + 1) + 1;
else
Histogram(6, Code(BasicLBP + 1, 2) + 1) = Histogram(6, Code(BasicLBP + 1, 2) + 1) + 1;
end
%%%%%%%%%%
%%%%%%%In plane7 (Fig.2 part (e) in the paper)(Reference of the paper: V. Esmaeili and S.O. Shahdi, “Automatic micro-expression apex spotting using Cubic-LBP,” Multimedia Tools and Applications, pp. 1-9, 2020.)
BasicLBP = 0;
FeaBin = 0;
for p = 0 : FormRULDNeighborPoints - 1
X = floor(xc + FxRadius * sin((2 * pi * p) / FormRULDNeighborPoints) + 0.5);
Y = floor(yc - FyRadius * sin((2 * pi * p) / FormRULDNeighborPoints) + 0.5);
Z = floor(i - TInterval * cos((2 * pi * p) / FormRULDNeighborPoints) + 0.5);
CurrentVal = VolData(Y, X, Z);
if CurrentVal >= CenterVal
BasicLBP = BasicLBP + 2 ^ FeaBin;
end
FeaBin = FeaBin + 1;
end
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
```

```
Histogram(7, BasicLBP + 1) = Histogram(7, BasicLBP + 1) + 1;

else

Histogram(7, Code(BasicLBP + 1, 2) + 1) = Histogram(7, Code(BasicLBP + 1, 2) + 1) + 1;

end

%%%%%%%%%%

%%%%%%%In plane8 (Fig.2 part (g) in the paper)(Reference of the paper: V. Esmaeili and S.O. Shahdi, "Automatic micro-expression apex spotting using Cubic-LBP," Multimedia Tools and Applications, pp. 1-9, 2020.)

BasicLBP = 0;

FeaBin = 0;

for p = 0 : FormBUFDNeighborPoints - 1

X = floor(xc + FxRadius * cos((2 * pi * p) / FormBUFDNeighborPoints) + 0.5);

Y = floor(yc - FyRadius * sin((2 * pi * p) / FormBUFDNeighborPoints) + 0.5);

Z = floor(i + TInterval * sin((2 * pi * p) / FormBUFDNeighborPoints) + 0.5);

CurrentVal = VolData(Y, X, Z);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(8, BasicLBP + 1) = Histogram(8, BasicLBP + 1) + 1;

else

Histogram(8, Code(BasicLBP + 1, 2) + 1) = Histogram(8, Code(BasicLBP + 1, 2) + 1) + 1;
```

```
end
%%%%%%%%%%
%%%%%%%In plane9 (Fig. 2 part (h) in the paper)(Reference of the paper: V. Esmaeili and S.O. Shahdi, "Automatic micro-expression apex spotting using Cubic-LBP," Multimedia Tools and Applications, pp. 1-9, 2020.)
BasicLBP = 0;
FeaBin = 0;
for p = 0 : FormFUBDNeighborPoints - 1
X = floor(xc - FxRadius * cos((2 * pi * p) / FormFUBDNeighborPoints) + 0.5);
Y = floor(yc - FyRadius * sin((2 * pi * p) / FormFUBDNeighborPoints) + 0.5);
Z = floor(i - TInterval * sin((2 * pi * p) / FormFUBDNeighborPoints) + 0.5);
CurrentVal = VolData(Y, X, Z);
if CurrentVal >= CenterVal
BasicLBP = BasicLBP + 2 ^ FeaBin;
end
FeaBin = FeaBin + 1;
end
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(9, BasicLBP + 1) = Histogram(9, BasicLBP + 1) + 1;
else
Histogram(9, Code(BasicLBP + 1, 2) + 1) = Histogram(9, Code(BasicLBP + 1, 2) + 1) + 1;
end
%%%%%%%%%%
```

%%%%%%%In plane10 (the plane of up faces of the cube)(Reference of the paper: V. Esmaeili and S.O. Shahdi, “Automatic micro-expression apex spotting using Cubic-LBP,” Multimedia Tools and Applications, pp. 1-9, 2020.)

```
BasicLBP = 0;

FeaBin = 0;

for p = 0 : FormUPNeighborPoints - 1

X = floor(xc + FxRadius * cos((2 * pi * p) / FormUPNeighborPoints) + 0.5);

%Y = floor(yc + FyRadius * sin((2 * pi * p) / FormUPNeighborPoints) + 0.5);

Y=floor(yc + FyRadius);

Z = floor(i + TInterval * sin((2 * pi * p) / FormUPNeighborPoints) + 0.5);

CurrentVal = VolData(Y, X, Z);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(10, BasicLBP + 1) = Histogram(10, BasicLBP + 1) + 1;

else

Histogram(10, Code(BasicLBP + 1, 2) + 1) = Histogram(10, Code(BasicLBP + 1, 2) + 1) + 1;

end
```

%%%%%%%%%

%%%%%%%In plane11 (the plane of down faces of the cube)(Reference of the paper: V. Esmaeili and S.O. Shahdi, “Automatic micro-expression apex spotting using Cubic-LBP,” Multimedia Tools and Applications, pp. 1-9, 2020.)

```
BasicLBP = 0;

FeaBin = 0;

for p = 0 : FormDOWNNeighborPoints - 1

X = floor(xc + FxRadius * cos((2 * pi * p) / FormDOWNNeighborPoints) + 0.5);

%Y = floor(yc - FyRadius * sin((2 * pi * p) / FormDOWNNeighborPoints) + 0.5);

Y=floor(yc - FyRadius);

Z = floor(i + TInterval * sin((2 * pi * p) / FormDOWNNeighborPoints) + 0.5);

CurrentVal = VolData(Y, X, Z);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(11, BasicLBP + 1) = Histogram(11, BasicLBP + 1) + 1;

else

Histogram(11, Code(BasicLBP + 1, 2) + 1) = Histogram(11, Code(BasicLBP + 1, 2) + 1) + 1;

end

%%%%%%%%%%

%%%%%%%In plane12 (the front face of cube)(Reference of the paper: V. Esmaeili and S.O. Shahdi, “Automatic micro-expression apex spotting using Cubic-LBP,” Multimedia Tools and Applications, pp. 1-9, 2020.)

BasicLBP = 0;

FeaBin = 0;
```

```
for p = 0 : FRONTNeighborPoints - 1
X = floor(xc + FxRadius * cos((2 * pi * p) / FRONTNeighborPoints) + 0.5);
Y = floor(yc - FyRadius * sin((2 * pi * p) / FRONTNeighborPoints) + 0.5);
%Z = floor(i + TInterval * sin((2 * pi * p) / FRONTNeighborPoints) + 0.5);
Z = floor(i + TInterval);
CurrentVal = VolData(Y, X, Z);
if CurrentVal >= CenterVal
BasicLBP = BasicLBP + 2 ^ FeaBin;
end
FeaBin = FeaBin + 1;
end
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(12, BasicLBP + 1) = Histogram(12, BasicLBP + 1) + 1;
else
Histogram(12, Code(BasicLBP + 1, 2) + 1) = Histogram(12, Code(BasicLBP + 1, 2) + 1) + 1;
end
%%%%%%%%%%
%%%%%%%In plane13 (the back face of the cube)(Reference of the paper: V. Esmaeili and S.O. Shahdi, "Automatic micro-expression apex spotting using Cubic-LBP," Multimedia Tools and Applications, pp. 1-9, 2020.)
BasicLBP = 0;
FeaBin = 0;
for p = 0 : BACKNeighborPoints - 1
X = floor(xc + FxRadius * cos((2 * pi * p) / BACKNeighborPoints) + 0.5);
```

```
Y = floor(yc - FyRadius * sin((2 * pi * p) / BACKNeighborPoints) + 0.5);

%Z = floor(i + TInterval * sin((2 * pi * p) / BACKNeighborPoints) + 0.5);

Z = floor(i - TInterval);

CurrentVal = VolData(Y, X, Z);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(13, BasicLBP + 1) = Histogram(13, BasicLBP + 1) + 1;

else

Histogram(13, Code(BasicLBP + 1, 2) + 1) = Histogram(13, Code(BasicLBP + 1, 2) + 1) + 1;

end

%%%%%%%%%

%%%%%%%In plane14 (the plane of right faces of the cube)(Reference of the paper: V. Esmaeili and S.O. Shahdi, “Automatic micro-expression apex spotting using Cubic-LBP,” Multimedia Tools and Applications, pp. 1-9, 2020.)

BasicLBP = 0;

FeaBin = 0;

for p = 0 : FormRNeighborPoints - 1

%X = floor(xc + FxRadius * cos((2 * pi * p) / FormRNeighborPoints) + 0.5);

X = floor(xc + FxRadius);

Y = floor(yc - FyRadius * sin((2 * pi * p) / FormRNeighborPoints) + 0.5);
```

```
Z = floor(i + TInterval * cos((2 * pi * p) / FormRNeighborPoints) + 0.5);

CurrentVal = VolData(Y, X, Z);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(14, BasicLBP + 1) = Histogram(14, BasicLBP + 1) + 1;

else

Histogram(14, Code(BasicLBP + 1, 2) + 1) = Histogram(14, Code(BasicLBP + 1, 2) + 1) + 1;

end

%%%%%%%%%%

%%%%%%%In plane15 (the plane of left faces of the cube)(Reference of the paper: V. Esmaeili and S.O. Shahdi, “Automatic micro-expression apex spotting using Cubic-LBP,” Multimedia Tools and Applications, pp. 1-9, 2020.)

BasicLBP = 0;

FeaBin = 0;

for p = 0 : FormLNeighborPoints - 1

%X = floor(xc + FxRadius * cos((2 * pi * p) / FormLNeighborPoints) + 0.5);

X = floor(xc - FxRadius);

Y = floor(yc - FyRadius * sin((2 * pi * p) / FormLNeighborPoints) + 0.5);

Z = floor(i + TInterval * cos((2 * pi * p) / FormLNeighborPoints) + 0.5);

CurrentVal = VolData(Y, X, Z);
```

```
if CurrentVal >= CenterVal
BasicLBP = BasicLBP + 2 ^ FeaBin;
end
FeaBin = FeaBin + 1;
end
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(15, BasicLBP + 1) = Histogram(15, BasicLBP + 1) + 1;
else
Histogram(15, Code(BasicLBP + 1, 2) + 1) = Histogram(15, Code(BasicLBP + 1, 2) + 1) + 1;
end
%%%%%%%%%%

end
end
end
else % bilinear interpolation
for i = TimeLength + 1 : Length - TimeLength
for yc = BorderLength + 1 : height - BorderLength
for xc = BorderLength + 1 : width - BorderLength
CenterVal = VolData(yc, xc, i);
%% In plane1
BasicLBP = 0;
FeaBin = 0;
for p = 0 : XYNeighborPoints - 1
```

```
% bilinear interpolation
x1 = single(xc + FxRadius * cos((2 * pi * p) / XYNeighborPoints));%%"float" are called "single" in Matlab
y1 = single(yc - FyRadius * sin((2 * pi * p) / XYNeighborPoints));
u = x1 - floor(x1);
v = y1 - floor(y1);
ltx = floor(x1);
lty = floor(y1);
lbx = floor(x1);
lby = ceil(y1);
rtx = ceil(x1);
rty = floor(y1);
rbx = ceil(x1);
rby = ceil(y1);
% the values of neighbors that do not fall exactly on
% pixels are estimated by bilinear interpolation of
% four corner points near to it.
CurrentVal = floor(single(VolData(lty, ltx, i)).*(1 - u).*(1 - v) + single(VolData(lby, lbx, i)).*(1 - u).*v +
single(VolData(rty, rtx, i)).*u.*(1 - v) + single(VolData(rby, rbx, i)).*u.*v);
if CurrentVal >= CenterVal
BasicLBP = BasicLBP + 2 ^ FeaBin;
end
FeaBin = FeaBin + 1;
end
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
```

```
Histogram(1, BasicLBP + 1) = Histogram(1, BasicLBP + 1) + 1;
else
Histogram(1, Code(BasicLBP + 1, 2) + 1) = Histogram(1, Code(BasicLBP + 1, 2) + 1) + 1;
end
%% In plane2
BasicLBP = 0;
FeaBin = 0;
for p = 0 : XTNeighborPoints - 1
% bilinear interpolation
x1 = single(xc + FxRadius * cos((2 * pi * p) / XTNeighborPoints));
z1 = single(i + TInterval * sin((2 * pi * p) / XTNeighborPoints));
u = x1 - floor(x1);
v = z1 - floor(z1);
ltx = floor(x1);
lty = floor(z1);
lbx = floor(x1);
lby = ceil(z1);
rtx = ceil(x1);
rty = floor(z1);
rbx = ceil(x1);
rby = ceil(z1);
% the values of neighbors that do not fall exactly on
% pixels are estimated by bilinear interpolation of
% four corner points near to it.
CurrentVal = floor(single(VolData(yc, ltx, lty)) * (1 - u) * (1 - v) + single(VolData(yc, lbx, lby)) * (1 - u) * v +
single(VolData(yc, rtx, rty)) * u * (1 - v) + single(VolData(yc, rbx, rby)) * u * v);
if CurrentVal >= CenterVal
BasicLBP = BasicLBP + 2 ^ FeaBin;
end
```

```
FeaBin = FeaBin + 1;
end
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(2, BasicLBP + 1) = Histogram(2, BasicLBP + 1) + 1;
else
Histogram(2, Code(BasicLBP + 1, 2) + 1) = Histogram(2, Code(BasicLBP + 1, 2) + 1) + 1;
end
%% In plane3
BasicLBP = 0;
FeaBin = 0;
for p = 0 : YTNeighborPoints - 1
% bilinear interpolation
y1 = single(yc - FyRadius * sin((2 * pi * p) / YTNeighborPoints));
z1 = single(i + TInterval * cos((2 * pi * p) / YTNeighborPoints));
u = y1 - floor(y1);
v = z1 - floor(z1);
ltx = floor(y1);
lty = floor(z1);
lbx = floor(y1);
lby = ceil(z1);
rtx = ceil(y1);
rty = floor(z1);
rbx = ceil(y1);
rby = ceil(z1);
```

```
% the values of neighbors that do not fall exactly on

% pixels are estimated by bilinear interpolation of

% four corner points near to it.

CurrentVal = floor(single(VolData(ltx, xc, lty)) * (1 - u) * (1 - v) + single(VolData(lbx, xc, lby)) * (1 - u) * v + single(VolData(rtx, xc, rty)) * u * (1 - v) + single(VolData(rbx, xc, rby)) * u * v);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(3, BasicLBP + 1) = Histogram(3, BasicLBP + 1) + 1;

else

Histogram(3, Code(BasicLBP + 1, 2) + 1) = Histogram(3, Code(BasicLBP + 1, 2) + 1) + 1;

end


%%%%%%%%

%% In plane4

BasicLBP = 0;

FeaBin = 0;

for p = 0 : DiameterMNeighborPoints - 1

% bilinear interpolation

x1 = single(xc + FxRadius * cos((2 * pi * p) / DiameterMNeighborPoints));%%"float" are called "single" in Matlab
```

```
y1 = single(yc - FyRadius * sin((2 * pi * p) / DiameterMNeighborPoints));

z1 = single(i + TInterval * cos((2 * pi * p) / DiameterMNeighborPoints));

u = x1 - floor(x1);

v = y1 - floor(y1);

q = z1 - floor(z1);

ltx = floor(x1);

lty = floor(y1);

ltz = floor(z1);

lbx = floor(x1);

lby = ceil(y1);

lbz = ceil(z1);

rtx = ceil(x1);

rty = floor(y1);

rtz = floor(z1);

rbx = ceil(x1);

rby = ceil(y1);

rbz = ceil(z1);

% the values of neighbors that do not fall exactly on

% pixels are estimated by bilinear interpolation of

% four corner points near to it.

CurrentVal = floor(single(VolData(lty, ltx, ltz)).*(1 - u).*(1 - v) + single(VolData(lby, lbx, lbz)).*(1 - u).*v +
single(VolData(rty, rtx, rtz)).*u.*(1 - v) + single(VolData(rby, rbx, rbz)).*u.*v);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case
```

```
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(4, BasicLBP + 1) = Histogram(4, BasicLBP + 1) + 1;
else
Histogram(4, Code(BasicLBP + 1, 2) + 1) = Histogram(4, Code(BasicLBP + 1, 2) + 1) + 1;
end
%% In plane5
BasicLBP = 0;
FeaBin = 0;
for p = 0 : Diameter2NeighborPoints - 1
% bilinear interpolation
x1 = single(xc + FxRadius * cos((2 * pi * p) / Diameter2NeighborPoints));%%"float" are called "single" in Matlab
y1 = single(yc - FyRadius * sin((2 * pi * p) / Diameter2NeighborPoints));
z1 = single(i - TInterval * cos((2 * pi * p) / Diameter2NeighborPoints));
u = x1 - floor(x1);
v = y1 - floor(y1);
q = z1 - floor(z1);
ltx = floor(x1);
lty = floor(y1);
ltz = floor(z1);
lbx = floor(x1);
lby = ceil(y1);
lbz = ceil(z1);
rtx = ceil(x1);
rty = floor(y1);
rtz = floor(z1);
```

```
rbx = ceil(x1);

rby = ceil(y1);

rbz = ceil(z1);

% the values of neighbors that do not fall exactly on

% pixels are estimated by bilinear interpolation of

% four corner points near to it.

CurrentVal = floor(single(VolData(lty, ltx, ltz)).*(1 - u).*(1 - v) + single(VolData(lby, lbx, lbz)).*(1 - u).*v +
single(VolData(rty, rtx, rtz)).*u.*(1 - v) + single(VolData(rby, rbx, rbz)).*u.*v);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(5, BasicLBP + 1) = Histogram(5, BasicLBP + 1) + 1;

else

Histogram(5, Code(BasicLBP + 1, 2) + 1) = Histogram(5, Code(BasicLBP + 1, 2) + 1) + 1;

end

%% In plane6

BasicLBP = 0;

FeaBin = 0;

for p = 0 : FormLURDNeighborPoints - 1

% bilinear interpolation
```

```
x1 = single(xc - FxRadius * sin((2 * pi * p) / FormLURDNeighborPoints));%%"float" are called "single" in Matlab

y1 = single(yc - FyRadius * sin((2 * pi * p) / FormLURDNeighborPoints));

z1 = single(i + TInterval * cos((2 * pi * p) / FormLURDNeighborPoints));

u = x1 - floor(x1);

v = y1 - floor(y1);

q = z1 - floor(z1);

ltx = floor(x1);

lty = floor(y1);

ltz = floor(z1);

lbx = floor(x1);

lby = ceil(y1);

lbz = ceil(z1);

rtx = ceil(x1);

rty = floor(y1);

rtz = floor(z1);

rbx = ceil(x1);

rby = ceil(y1);

rbz = ceil(z1);

% the values of neighbors that do not fall exactly on

% pixels are estimated by bilinear interpolation of

% four corner points near to it.

CurrentVal = floor(single(VolData(lty, ltx, ltz)).*(1 - u).*(1 - v) + single(VolData(lby, lbx, lbz)).*(1 - u).*v + single(VolData(rty, rtx, rtz)).*u.*(1 - v) + single(VolData(rby, rbx, rbz)).*u.*v);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end
```

```
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(6, BasicLBP + 1) = Histogram(6, BasicLBP + 1) + 1;
else
Histogram(6, Code(BasicLBP + 1, 2) + 1) = Histogram(6, Code(BasicLBP + 1, 2) + 1) + 1;
end
%% In plane7
BasicLBP = 0;
FeaBin = 0;
for p = 0 : FormRULDNeighborPoints - 1
% bilinear interpolation
x1 = single(xc + FxRadius * sin((2 * pi * p) / FormRULDNeighborPoints));%%"float" are called "single" in
Matlab
y1 = single(yc - FyRadius * sin((2 * pi * p) / FormRULDNeighborPoints));
z1 = single(i - TInterval * cos((2 * pi * p) / FormRULDNeighborPoints));
u = x1 - floor(x1);
v = y1 - floor(y1);
q = z1 - floor(z1);
ltx = floor(x1);
lty = floor(y1);
ltz = floor(z1);
lbx = floor(x1);
lby = ceil(y1);
lbz = ceil(z1);
rtx = ceil(x1);
```

```
rty = floor(y1);

rtz = floor(z1);

rbx = ceil(x1);

rby = ceil(y1);

rbz = ceil(z1);

% the values of neighbors that do not fall exactly on

% pixels are estimated by bilinear interpolation of

% four corner points near to it.

CurrentVal = floor(single(VolData(lty, ltx, ltz)).*(1 - u).*(1 - v) + single(VolData(lby, lbx, lbz)).*(1 - u).*v +
single(VolData(rty, rtx, rtz)).*u.*(1 - v) + single(VolData(rby, rbx, rbz)).*u.*v);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(7, BasicLBP + 1) = Histogram(7, BasicLBP + 1) + 1;

else

Histogram(7, Code(BasicLBP + 1, 2) + 1) = Histogram(7, Code(BasicLBP + 1, 2) + 1) + 1;

end

%% In plane8

BasicLBP = 0;

FeaBin = 0;

for p = 0 : FormBUFDNeighborPoints - 1
```

```
% bilinear interpolation

x1 = single(xc + FxRadius * cos((2 * pi * p) / FormBUFDNeighborPoints));%%"float" are called "single" in Matlab

y1 = single(yc - FyRadius * sin((2 * pi * p) / FormBUFDNeighborPoints));

z1 = single(i + TInterval * sin((2 * pi * p) / FormBUFDNeighborPoints));

u = x1 - floor(x1);

v = y1 - floor(y1);

q = z1 - floor(z1);

ltx = floor(x1);

lty = floor(y1);

ltz = floor(z1);

lbx = floor(x1);

lby = ceil(y1);

lbz = ceil(z1);

rtx = ceil(x1);

rty = floor(y1);

rtz = floor(z1);

rbx = ceil(x1);

rby = ceil(y1);

rbz = ceil(z1);

% the values of neighbors that do not fall exactly on

% pixels are estimated by bilinear interpolation of

% four corner points near to it.

CurrentVal = floor(single(VolData(lty, ltx, ltz)).*(1 - u).*(1 - v) + single(VolData(lby, lbx, lbz)).*(1 - u).*v + single(VolData(rty, rtx, rtz)).*u.*(1 - v) + single(VolData(rby, rbx, rbz)).*u.*v);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;
```

```
end
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(8, BasicLBP + 1) = Histogram(8, BasicLBP + 1) + 1;
else
Histogram(8, Code(BasicLBP + 1, 2) + 1) = Histogram(8, Code(BasicLBP + 1, 2) + 1) + 1;
end
%% In plane9
BasicLBP = 0;
FeaBin = 0;
for p = 0 : FormFUBDNeighborPoints - 1
% bilinear interpolation
x1 = single(xc - FxRadius * cos((2 * pi * p) / FormFUBDNeighborPoints));%%"float" are called "single" in
Matlab
y1 = single(yc - FyRadius * sin((2 * pi * p) / FormFUBDNeighborPoints));
z1 = single(i - TInterval * sin((2 * pi * p) / FormFUBDNeighborPoints));
u = x1 - floor(x1);
v = y1 - floor(y1);
q = z1 - floor(z1);
ltx = floor(x1);
lty = floor(y1);
ltz = floor(z1);
lbx = floor(x1);
lby = ceil(y1);
lbz = ceil(z1);
```

```
rtx = ceil(x1);

rty = floor(y1);

rtz = floor(z1);

rbx = ceil(x1);

rby = ceil(y1);

rbz = ceil(z1);

% the values of neighbors that do not fall exactly on

% pixels are estimated by bilinear interpolation of

% four corner points near to it.

CurrentVal = floor(single(VolData(lty, ltx, ltz)).*(1 - u).*(1 - v) + single(VolData(lby, lbx, lbz)).*(1 - u).*v +
single(VolData(rty, rtx, rtz)).*u.*(1 - v) + single(VolData(rby, rbx, rbz)).*u.*v);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(9, BasicLBP + 1) = Histogram(9, BasicLBP + 1) + 1;

else

Histogram(9, Code(BasicLBP + 1, 2) + 1) = Histogram(9, Code(BasicLBP + 1, 2) + 1) + 1;

end

%% In plane10

BasicLBP = 0;

FeaBin = 0;
```

```
for p = 0 : FormUPNeighborPoints - 1

% bilinear interpolation

x1 = single(xc + FxRadius * cos((2 * pi * p) / FormUPNeighborPoints));%%"float" are called "single" in Matlab

y1 = single(yc + FyRadius);

z1 = single(i + TInterval * sin((2 * pi * p) / FormUPNeighborPoints));

u = x1 - floor(x1);

v = y1 - floor(y1);

q = z1 - floor(z1);

ltx = floor(x1);

lty = floor(y1);

ltz = floor(z1);

lbx = floor(x1);

lby = ceil(y1);

lbz = ceil(z1);

rtx = ceil(x1);

rty = floor(y1);

rtz = floor(z1);

rbx = ceil(x1);

rby = ceil(y1);

rbz = ceil(z1);

% the values of neighbors that do not fall exactly on

% pixels are estimated by bilinear interpolation of

% four corner points near to it.

CurrentVal = floor(single(VolData(lty, ltx, ltz)).*(1 - u).*(1 - v) + single(VolData(lby, lbx, lbz)).*(1 - u).*v + single(VolData(rty, rtx, rtz)).*u.*(1 - v) + single(VolData(rby, rbx, rbz)).*u.*v);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end
```

```
FeaBin = FeaBin + 1;
end
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(10, BasicLBP + 1) = Histogram(10, BasicLBP + 1) + 1;
else
Histogram(10, Code(BasicLBP + 1, 2) + 1) = Histogram(10, Code(BasicLBP + 1, 2) + 1) + 1;
end
%% In plane11
BasicLBP = 0;
FeaBin = 0;
for p = 0 : FormDOWNNeighborPoints - 1
% bilinear interpolation
x1 = single(xc + FxRadius * cos((2 * pi * p) / FormDOWNNeighborPoints));%%"float" are called "single" in Matlab
y1 = single(yc - FyRadius);
z1 = single(i + TInterval * sin((2 * pi * p) / FormDOWNNeighborPoints));
u = x1 - floor(x1);
v = y1 - floor(y1);
q = z1 - floor(z1);
ltx = floor(x1);
lty = floor(y1);
ltz = floor(z1);
lbx = floor(x1);
lby = ceil(y1);
```

```
lbz = ceil(z1);

rtx = ceil(x1);

rty = floor(y1);

rtz = floor(z1);

rbx = ceil(x1);

rby = ceil(y1);

rbz = ceil(z1);

% the values of neighbors that do not fall exactly on

% pixels are estimated by bilinear interpolation of

% four corner points near to it.

CurrentVal = floor(single(VolData(lty, ltx, ltz)).*(1 - u).*(1 - v) + single(VolData(lby, lbx, lbz)).*(1 - u).*v +
single(VolData(rty, rtx, rtz)).*u.*(1 - v) + single(VolData(rby, rbx, rbz)).*u.*v);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(11, BasicLBP + 1) = Histogram(11, BasicLBP + 1) + 1;

else

Histogram(11, Code(BasicLBP + 1, 2) + 1) = Histogram(11, Code(BasicLBP + 1, 2) + 1) + 1;

end

%% In plane12

BasicLBP = 0;
```

```
FeaBin = 0;

for p = 0 : FRONTNeighborPoints - 1

% bilinear interpolation

x1 = single(xc + FxRadius * cos((2 * pi * p) / FRONTNeighborPoints));%%"float" are called "single" in Matlab

y1 = single(yc - FyRadius * sin((2 * pi * p) / FRONTNeighborPoints));

z1 = single(i + TInterval);

u = x1 - floor(x1);

v = y1 - floor(y1);

q = z1 - floor(z1);

ltx = floor(x1);

lty = floor(y1);

ltz = floor(z1);

lbx = floor(x1);

lby = ceil(y1);

lbz = ceil(z1);

rtx = ceil(x1);

rty = floor(y1);

rtz = floor(z1);

rbx = ceil(x1);

rby = ceil(y1);

rbz = ceil(z1);

% the values of neighbors that do not fall exactly on

% pixels are estimated by bilinear interpolation of

% four corner points near to it.

CurrentVal = floor(single(VolData(lty, ltx, ltz)).*(1 - u).*(1 - v) + single(VolData(lby, lbx, lbz)).*(1 - u).*v + single(VolData(rty, rtx, rtz)).*u.*(1 - v) + single(VolData(rby, rbx, rbz)).*u.*v);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;
```

```
end
FeaBin = FeaBin + 1;
end
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(12, BasicLBP + 1) = Histogram(12, BasicLBP + 1) + 1;
else
Histogram(12, Code(BasicLBP + 1, 2) + 1) = Histogram(12, Code(BasicLBP + 1, 2) + 1) + 1;
end
%% In plane13
BasicLBP = 0;
FeaBin = 0;
for p = 0 : BACKNeighborPoints - 1
% bilinear interpolation
x1 = single(xc + FxRadius * cos((2 * pi * p) / BACKNeighborPoints));%%"float" are called "single" in
Matlab
y1 = single(yc - FyRadius * sin((2 * pi * p) / BACKNeighborPoints));
z1 = single(i - TInterval);
u = x1 - floor(x1);
v = y1 - floor(y1);
q = z1 - floor(z1);
ltx = floor(x1);
lty = floor(y1);
ltz = floor(z1);
lbx = floor(x1);
```

```
lby = ceil(y1);

lbz = ceil(z1);

rtx = ceil(x1);

rty = floor(y1);

rtz = floor(z1);

rbx = ceil(x1);

rby = ceil(y1);

rbz = ceil(z1);

% the values of neighbors that do not fall exactly on

% pixels are estimated by bilinear interpolation of

% four corner points near to it.

CurrentVal = floor(single(VolData(lty, ltx, ltz)).*(1 - u).*(1 - v) + single(VolData(lby, lbx, lbz)).*(1 - u).*v + single(VolData(rty, rtx, rtz)).*u.*(1 - v) + single(VolData(rby, rbx, rbz)).*u.*v);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(13, BasicLBP + 1) = Histogram(13, BasicLBP + 1) + 1;

else

Histogram(13, Code(BasicLBP + 1, 2) + 1) = Histogram(13, Code(BasicLBP + 1, 2) + 1) + 1;

end

%% In plane14
```

```
BasicLBP = 0;

FeaBin = 0;

for p = 0 : FormRNeighborPoints - 1

% bilinear interpolation

x1 = single(xc + FxRadius);%%"float" are called "single" in Matlab

y1 = single(yc - FyRadius * sin((2 * pi * p) / FormRNeighborPoints));

z1 = single(i + TInterval * cos((2 * pi * p) / FormRNeighborPoints));

u = x1 - floor(x1);

v = y1 - floor(y1);

q = z1 - floor(z1);

ltx = floor(x1);

lty = floor(y1);

ltz = floor(z1);

lbx = floor(x1);

lby = ceil(y1);

lbz = ceil(z1);

rtx = ceil(x1);

rty = floor(y1);

rtz = floor(z1);

rbx = ceil(x1);

rby = ceil(y1);

rbz = ceil(z1);

% the values of neighbors that do not fall exactly on

% pixels are estimated by bilinear interpolation of

% four corner points near to it.

CurrentVal = floor(single(VolData(lty, ltx, ltz)).*(1 - u).*(1 - v) + single(VolData(lby, lbx, lbz)).*(1 - u).*v +
single(VolData(rty, rtx, rtz)).*u.*(1 - v) + single(VolData(rby, rbx, rbz)).*u.*v);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;
```

```
end
FeaBin = FeaBin + 1;
end
%% if Bincount is "0", it means basic Cubic-LBP will be
%% computed and uniform patterns does not work in this case
%%. Otherwide it should be the number of the uniform
%%patterns, then "Code" keeps the lookup-table of the basic
%%LBP and uniform LBP
if Bincount == 0
Histogram(14, BasicLBP + 1) = Histogram(14, BasicLBP + 1) + 1;
else
Histogram(14, Code(BasicLBP + 1, 2) + 1) = Histogram(14, Code(BasicLBP + 1, 2) + 1) + 1;
end
%% In plane15
BasicLBP = 0;
FeaBin = 0;
for p = 0 : FormLNeighborPoints - 1
% bilinear interpolation
x1 = single(xc - FxRadius);%%"float" are called "single" in Matlab
y1 = single(yc - FyRadius * sin((2 * pi * p) / FormLNeighborPoints));
z1 = single(i + TInterval * cos((2 * pi * p) / FormLNeighborPoints));
u = x1 - floor(x1);
v = y1 - floor(y1);
q = z1 - floor(z1);
ltx = floor(x1);
lty = floor(y1);
ltz = floor(z1);
lbx = floor(x1);
lby = ceil(y1);
```

```
lbz = ceil(z1);

rtx = ceil(x1);

rty = floor(y1);

rtz = floor(z1);

rbx = ceil(x1);

rby = ceil(y1);

rbz = ceil(z1);

% the values of neighbors that do not fall exactly on

% pixels are estimated by bilinear interpolation of

% four corner points near to it.

CurrentVal = floor(single(VolData(lty, ltx, ltz)).*(1 - u).*(1 - v) + single(VolData(lby, lbx, lbz)).*(1 - u).*v + single(VolData(rty, rtx, rtz)).*u.*(1 - v) + single(VolData(rby, rbx, rbz)).*u.*v);

if CurrentVal >= CenterVal

BasicLBP = BasicLBP + 2 ^ FeaBin;

end

FeaBin = FeaBin + 1;

end

%% if Bincount is "0", it means basic Cubic-LBP will be

%% computed and uniform patterns does not work in this case

%%. Otherwide it should be the number of the uniform

%%patterns, then "Code" keeps the lookup-table of the basic

%%LBP and uniform LBP

if Bincount == 0

Histogram(15, BasicLBP + 1) = Histogram(15, BasicLBP + 1) + 1;

else

Histogram(15, Code(BasicLBP + 1, 2) + 1) = Histogram(15, Code(BasicLBP + 1, 2) + 1) + 1;

end
```

```
end %%
end %%
end %%
end
%% normalization
for j = 1 : 15
Histogram(j, :) = Histogram(j, :)./sum(Histogram(j, :));
end
```