Photometric Stereo recovery of shape and relative distance

1. Implementation:

1.1 Calibration:

1.1.1. Gray-scaling and add three sphere images together to create a mask

```matlab
I1g = rgb2gray(I1);  
I2g = rgb2gray(I2);  
I3g = rgb2gray(I3);  
Inew = I1g + I2g + I3g;  
for i = 1:468  
    for j = 1:637  
        if Inew(i,j) > 65  
            Inew(i,j) = 255;  
        else  
            Inew(i,j) = 0;  
        end  
    end  
end
```

1.1.2. Find the centroid and radius and area of the sphere

```matlab
[y, x] = find(Inew);  
r = (max(x) - min(x))/2;  
centerX = (max(x) + min(x))/2;  
centerY = min(y) + r;  
centerPos = [centerX, centerY];  
CircleArea = cat(2, x, y);
```

1.1.3. Generate gradients

Create two matrices for gradient. Since the range of \( E_1/E_2 \) or \( E_2/E_3 \) is \([10/255, \ 255/10]\) (without considering those whose lightness is below 10) and to reduce the scale to save the time, I quantize the range of \( E_1/E_2 \) or \( E_2/E_3 \) to \([\text{ceil}(10/255\times255/6) = 1, \ \text{ceil}(255/10\times255/6) = 1083]\). And sometimes one \( E_1/E_2 \) or \( E_2/E_3 \) have two normal vectors, So it should have the third dimension.
\[ P = \text{zeros}(1083,1083,2); \]
\[ Q = \text{zeros}(1083,1083,2); \]

For each pixel in the Area of the sphere, calculate its Z value by the equation:

\[ Z = \sqrt{(\text{Radius}^2 - X^2 - Y^2)}; \]
% To avoid complex number, \( \text{Radius}^2 - X^2 - Y^2 \) can't be less than 0.

Then calculate the gradient:

\[ P = X / (-Z); \]
\[ Q = Y / (-Z); \]
% \( Z \) can't be 0 and if \( Z \to 0 \) then \( P \) and \( Q \to \text{infinite} \). So ensure \( Z \geq 20 \)

Next:

\[ E1E2 = \text{ceil}(I1g(y,x)/I2g(y,x)*255/6); \]
\[ E2E3 = \text{ceil}(I2g(y,x)/I3g(y,x)*255/6); \]
% \( I1g, I2g, I3g \) are the lightness of the pixel

Match \( E1E2 \) and \( E2E3 \) with the gradients according to the surrounding ones:

\[ P(E1E2,E2E3,1) = P; \]
\[ Q(E1E2,E2E3,1) = Q; \]

or

\[ P(E1E2,E2E3,2) = P; \]
\[ Q(E1E2,E2E3,2) = Q; \]

1.1.4. Use interpolation to fill the "holes" in the E1E2-E2E3 table:

\[ [X,Y,Z]=\text{find}(P(:,:,1)); \]
\[ [x,y]=\text{find}(P(:,:,1)==0); \]
\[ z=\text{griddata}(X,Y,Z,x,y,'\text{nearest}'); \]
\[ n=\text{length}(x); \]
\[ \text{for } i=1:n, \ P(x(i),y(i),1)=z(i); \text{ end} \]

\[ [X1,Y1,Z1]=\text{find}(Q(:,:,1)); \]
\[ [x1,y1]=\text{find}(Q(:,:,1)==0); \]
\[ z1=\text{griddata}(X1,Y1,Z1,x1,y1,'\text{nearest}'); \]
\[ n1=\text{length}(x1); \]
\[ \text{for } i=1:n1, \ Q(x1(i),y1(i),1)=z1(i); \text{ end} \]
1.2 Recovery of shape and relative distance

1) Import the input files and filter out the background by a mask Sum and invalid pixels whose TestI1g(i,j) or TestI2g(i,j) or TestI3g(i,j) is less than 10

2) Extract the lightness of valid pixels and convert them to E1E2 and E2E3 as the same as the step in Calibration then fetch out the gradients. Integrate the gradient in X-Axis to have the relative distance function Z is represented as a 468*637 (the size of the image) matrix.

3) Finally render the gradients and the relative distance 3-D surface with the quiver function and surf function.

2. Example of Output:

Sphere:
Cone:

Cylinder:
Ellipsoid:

Hex1:
Cone2:

Hex2:
3. Comment:

For this experiment, I would like to say:

1. The ability of calculation is quite limited. For example if I want to use 'v4' in interpolation which should be better than 'nearest', it never runs out any results. And theoretically, there are $255 \times 255 \times 255 = 16581375$ possibilities for the combination (E1/E2,E2/E3). But the data is too huge, it's needed to reduce the size.

2. For the relative distance, finding a good way to integrate can smooth the results.

3. The image coordinate system of MATLAB is annoying.

4. How to extract the object from the background is also a big problem though not a photometric stereo one.

4. MATLAB code

```matlab
function answer = Normdetect_simple3( )
close all
I1=imread('sphere-lamp1.tif');
I2=imread('sphere-lamp2.tif');
I3=imread('sphere-lamp3.tif');
%figure;imshow(I_RGB);
I1g=rgb2gray(I1);
I2g=rgb2gray(I2);
I3g=rgb2gray(I3);

Inew = I1g+I2g+I3g;

for i = 1:468
    for j = 1:637
        if Inew(i,j) > 65
            Inew(i,j) = 255;
        else
            Inew(i,j) = 0;
        end
    end
end
```
\[ [y, x] = \text{find}(\text{Inew}); \]
\[ r = (\max(x) - \min(x)) / 2; \]
\[ \text{centerX} = (\max(x) + \min(x)) / 2; \]
\[ \text{centerY} = \min(y) + r; \]
\[ \text{centerPos} = [\text{centerX}, \text{centerY}]; \]
\[ \% \text{plot}([\text{centerPos}(1), \text{centerPos}(2)], 'b*'); \]
\[ \text{CircleArea} = \text{cat}(2, x, y); \]
\[ \% \text{figure; imshow}(\text{Inew}); \]
\[ \% \text{hold on} \]
\[ \text{P} = \text{zeros}(1083, 1083, 2); \]
\[ \text{Q} = \text{zeros}(1083, 1083, 2); \]
\[ \text{I1g} = \text{double} (\text{I1g}); \]
\[ \text{I2g} = \text{double} (\text{I2g}); \]
\[ \text{I3g} = \text{double} (\text{I3g}); \]
\[ \text{for} \ \text{row} = 1: \text{length} (\text{CircleArea}(:, 1)) \]
\[ \text{y} = \text{CircleArea} (\text{row}, 2); \]
\[ \text{x} = \text{CircleArea} (\text{row}, 1); \]
\[ \quad \text{if} \ \text{I1g}(\text{y}, \text{x}) >= 10 \ \& \ \& \ \text{I2g}(\text{y}, \text{x}) >= 10 \ \& \ \& \ \text{I3g}(\text{y}, \text{x}) >= 10 \]
\[ \quad \quad \text{zeroY} = 468 - \text{y} -(468 - \text{centerPos}(2)); \]
\[ \quad \quad \text{zeroX} = \text{x} - \text{centerPos}(1); \]
\[ \quad \quad \text{if} \ \text{r}^2 - \text{zeroX}^2 - \text{zeroY}^2 < 0 \]
\[ \quad \quad \quad \text{continue}; \]
\[ \quad \quad \text{else} \]
\[ \quad \quad \quad \text{zeroZ} = \sqrt{\text{r}^2 - \text{zeroX}^2 - \text{zeroY}^2}; \]
\[ \quad \quad \text{end} \]
\[ \quad \text{if} \ \text{zeroZ} < 20 \]
\[ \quad \quad \text{continue}; \]
\[ \quad \text{else} \]
\[ \quad \quad \text{tmpP} = \text{zeroX} / (-\text{zeroZ}); \]
\[ \quad \quad \text{tmpQ} = \text{zeroY} / (-\text{zeroZ}); \]
\[ \quad \text{end} \]
\[ \text{E1E2} = \text{ceil} (\text{I1g}(\text{y}, \text{x}) / \text{I2g}(\text{y}, \text{x}) * 255 / 6); \]
\[ \text{E2E3} = \text{ceil} (\text{I2g}(\text{y}, \text{x}) / \text{I3g}(\text{y}, \text{x}) * 255 / 6); \]
\[ \text{if} \ \text{P}(\text{E1E2}, \text{E2E3}, 1) == 0 \ \& \ \& \ \text{Q}(\text{E1E2}, \text{E2E3}, 1) == 0 \]
\[ \quad \text{P}(\text{E1E2}, \text{E2E3}, 1) = \text{tmpP}; \]
\[ \quad \text{Q}(\text{E1E2}, \text{E2E3}, 1) = \text{tmpQ}; \]
\[ \text{elseif} \ \text{abs} (\text{tmpP} - \text{P}(\text{E1E2}, \text{E2E3}, 1)) > 0.1 \ \& \ \& \]
\[ \text{abs} (\text{tmpQ} - \text{Q}(\text{E1E2}, \text{E2E3}, 1)) > 0.1 \]
\[ \quad \text{P}(\text{E1E2}, \text{E2E3}, 2) = \text{tmpP}; \]
\[ \quad \text{Q}(\text{E1E2}, \text{E2E3}, 2) = \text{tmpQ}; \]
\[ \text{end} \]
\[ \% \text{if} \ \text{mod}(\text{y}, 20) == 0 \ \& \ \& \ \text{mod}(\text{x}, 20) == 0 \]
\[ \% \ \text{quiver}(\text{x}, \text{y}, -\text{tmpP}, \text{tmpQ}, 50); \]
% x,y,tmpP,tmpQ
% hold on
% end
end

[X,Y,Z]=find(P(:,:,1));
x,y=find(P(:,:,1)==0);
z=griddata(X,Y,Z,x,y,'nearest');
n=length(x);
for i=1:n, P(x(i),y(i),1)=z(i); end

[X1,Y1,Z1]=find(Q(:,:,1));
x1,y1=find(Q(:,:,1)==0);
z1=griddata(X1,Y1,Z1,x1,y1,'nearest');
n1=length(x1);
for i=1:n1, Q(x1(i),y1(i),1)=z1(i); end

%K = find(P(:,:,1)==0);
TestI1=imread('hex2-lamp1.tif');
TestI2=imread('hex2-lamp2.tif');
TestI3=imread('hex2-lamp3.tif');
TestI1g=rgb2gray(TestI1);
TestI2g=rgb2gray(TestI2);
TestI3g=rgb2gray(TestI3);

figure;imshow(TestI1);
hold on
TestI1g=double(TestI1g);
TestI2g=double(TestI2g);
TestI3g=double(TestI3g);

Z = zeros(468,637);
Z2 = zeros(468,637);
Zf = zeros(468,637);
BW1 = zeros(468,637);
BW2 = zeros(468,637);
BW3 = zeros(468,637);

for i = 1:468
    for j = 1:637
        if TestI1(i,j,1)>=40 || TestI1(i,j,2)>=40 || TestI1(i,j,3)>=40
            BW1(i,j) = 255;
        else
            BW1(i,j) = 0;
        end
    end
end
for i = 1:468
for j = 1:637
    if TestI2(i,j,1)>=40 || TestI2(i,j,2)>=40 ||
    TestI2(i,j,3)>=40
        BW2(i,j) = 255;
    else
        BW2(i,j) = 0;
    end
end
for i = 1:468
    for j = 1:637
        if TestI1(i,j,1)>=40 || TestI1(i,j,2)>=40 ||
        TestI1(i,j,3)>=40
            BW3(i,j) = 255;
        else
            BW3(i,j) = 0;
        end
    end
end
Sum = zeros(468,637);
Sum(:, :) = BW1(:, :) + BW2(:, :) + BW3(:, :);
%figure;imshow(Sum);

for i = 1:468
    for j = 1:636
        if Sum(i,j) ~= 0
            if TestI1g(i,j)>=10 && TestI2g(i,j)>=10 &&
            TestI3g(i,j)>=10
                NewE1E2 = ceil(TestI1g(i,j)/TestI2g(i,j)*255/6);
                NewE2E3 = ceil(TestI2g(i,j)/TestI3g(i,j)*255/6);
                tmpP2 = P(NewE1E2,NewE2E3,1);
                tmpQ2 = Q(NewE1E2,NewE2E3,1);
                formerP2 = P(NewE1E2,NewE2E3-1,1);
                formerQ2 = Q(NewE1E2,NewE2E3-1,1);
                if abs(tmpP2-formerP2) > 0.1 &&
                abs(tmpQ2-formerQ2) > 0.1
                    tmpP2 = P(NewE1E2,NewE2E3,2);
                    tmpQ2 = Q(NewE1E2,NewE2E3,2);
                end
                if tmpP2 == 0
                    NewE1E2,NewE2E3
                end
                if mod(i,20)==0 && mod(j,20)==0
                    quiver(j,i,-
                    tmpP2,tmpQ2,40,'color',[0,0,1]);
            end
        end
    end
    Z(i,j+1) = Z(i,j) + tmpP2;
else
    Z(i,j+1) = Z(i,j);
end
%tmpP2, tmpQ2
hold on
end
Z(i,j+1) = Z(i,j) + tmpP2;
else
Z(i,j+1) = Z(i,j);
end
end
end
for j = 1:637
    for i = 1:467
        if TestI1g(i,j)>=25 && TestI2g(i,j)>=25 && TestI3g(i,j)>=25
            NewE1E2 = ceil(TestI1g(i,j)/TestI2g(i,j)*255/6);
            NewE2E3 = ceil(TestI2g(i,j)/TestI3g(i,j)*255/6);

            tmpP2 = P(NewE1E2,NewE2E3,1);
            tmpQ2 = Q(NewE1E2,NewE2E3,1);

            formertmpP2 = P(NewE1E2,NewE2E3-1,1);
            formertmpQ2 = Q(NewE1E2,NewE2E3-1,1);

            if abs(tmpP2-formertmpP2) > 0.1 && abs(tmpQ2-formertmpQ2) > 0.1
                tmpP2 = P(NewE1E2,NewE2E3,2);
                tmpQ2 = Q(NewE1E2,NewE2E3,2);
            end

            Z2(i+1,j) = Z2(i,j) + tmpQ2;
        else
            Z2(i+1,j) = Z2(i,j);
        end
    end
end
Zf(:,:) = (-Z2(:,:)+ Z(:,:))/2;
Zdownsampling = imresize(Z,1);
figure;
surf(Zdownsampling);
shading interp
end