IIT Vadodara
Autumn 2018-19
TE3 Computer Vision
Lab-7: Depth Estimation using Photo-metric Stereo

Shivvrat Arya (201551059)

September 29, 2018

Problem 1. Use the set of sphere images generated in Lab-5 and estimate the p and q matrices based on Photometric Stereo (PS) method discussed in class. Let us assume Lambertian surface, i.e., constant albedo , then PS for n images can be formulated at each pixel location (x, y) as, equation, where, E1, E2, ..., En are set of n sphere images constructed using known source positions ps1, ps2, ..., psn , qs1, qs2, ..., qsn. Use singu- lar values decomposition (SVD) in order to solve for the p and q matrices. Now, use these estimated p and q matrices to estimate the depths (COV approach Lab-6) at each point the Sphere scene. Comment on the result when compared to Lab-6.

Solution Photometric stereo is a computer vision method of analyzing and detailing the contour and reflectivity of a surface in 3D (three-dimensional) space. The method involves shining an external light source on that surface, moving the light and gathering multiple images based on the resulting illumination scenarios.
Code :-
1. To create the E matrix using different ps and qs values :-

---

```
clc;
clear all;
tStart=tic;

I = uint8(zeros(128,128));
r = 40;
center_x = round(size(I,1)/2);
center_y = round(size(I,2)/2);
ps1 = 0;
qs1 = 1;

for i = 1:size(I,1)
    for j = 1:size(I,2)
        x_adjusted = i - center_x;
```

```
        y_adjusted = j - center_y;
        p = (-x_adjusted)/(sqrt(power(r,2) - power(x_adjusted,2) -
           power(y_adjusted,2)));
        q = (-y_adjusted)/(sqrt(power(r,2) - power(x_adjusted,2) -
           power(y_adjusted,2)));
        num = (p*ps1) + (q*qs1) + 1;
        den = (sqrt(1 + power(p,2) + power(q,2))*sqrt(1 + power(ps1,2) +
           power(qs1,2)));
        irradiance = num/den;
        if ((power(x_adjusted,2))+(power(y_adjusted,2)) < power(r,2))
           I(x_adjusted+center_x,y_adjusted+center_y) = 500*irradiance;
        endif
    endfor
endfor
E1 = I;

I = uint8(zeros(128,128));
r = 40;
center_x = round(size(I,1)/2);
center_y = round(size(I,2)/2);
ps2 = 0.5;
qs2 = 0.5;
for i = 1:size(I,1)
    for j = 1:size(I,2)
        x_adjusted = i - center_x;
        y_adjusted = j - center_y;
        p = (-x_adjusted)/(sqrt(power(r,2) - power(x_adjusted,2) -
           power(y_adjusted,2)));
        q = (-y_adjusted)/(sqrt(power(r,2) - power(x_adjusted,2) -
           power(y_adjusted,2)));
        num = (p*ps2) + (q*qs2) + 1;
        den = (sqrt(1 + power(p,2) + power(q,2))*sqrt(1 + power(ps2,2) +
           power(qs2,2)));
        irradiance = num/den;
        if ((power(x_adjusted,2))+(power(y_adjusted,2)) < power(r,2))
           I(x_adjusted+center_x,y_adjusted+center_y) = 500*irradiance;
        endif
    endfor
endfor
E2 = I;

I = uint8(zeros(128,128));
r = 40;
center_x = round(size(I,1)/2);
center_y = round(size(I,2)/2);
ps3 = 0.25;
qs3 = 0.25;
```

```
for i = 1:size(I,1)
    for j = 1:size(I,2)
        x_adjusted = i - center_x;
        y_adjusted = j - center_y;
        p = (-x_adjusted)/(sqrt(power(r,2) - power(x_adjusted,2) -
            power(y_adjusted,2)));
        q = (-y_adjusted)/(sqrt(power(r,2) - power(x_adjusted,2) -
            power(y_adjusted,2)));
        num = (p*ps3) + (q*qs3) + 1;
        den = (sqrt(1 + power(p,2) + power(q,2))*sqrt(1 + power(ps3,2) +
            power(qs3,2)));
        irradiance = num/den;
        if ((power(x_adjusted,2))+(power(y_adjusted,2)) < power(r,2))
            I(x_adjusted+center_x,y_adjusted+center_y) = 500*irradiance;
        endif
    endfor
endfor
E3 = I;

save E1.mat E1
save E2.mat E2
save E3.mat E3
tElapsed=toc(tStart)
```

---

matrix formation - 6.4799

Calculating p and q by using photo-metric stereo :-

```
tStart=tic;
E1 = load("E1.mat");
E2 = load("E2.mat");
E3 = load("E3.mat");
pos = [-ps1/sqrt(1 + power(ps1, 2) + power(qs1, 2)),
-qs1/sqrt(1 + power(ps1, 2) + power(qs1, 2)),
1/sqrt(1 + power(ps1, 2) + power(qs1, 2));
-ps2/sqrt(1 + power(ps2, 2) + power(qs2, 2)),
-qs2/sqrt(1 + power(ps2, 2) + power(qs2, 2)),
1/sqrt(1 + power(ps2, 2) + power(qs2, 2));
-ps3/sqrt(1 + power(ps3, 2) + power(qs3, 2)),
-qs3/sqrt(1 + power(ps3, 2) + power(qs3, 2)),
1/sqrt(1 + power(ps3, 2) + power(qs3, 2))
 ];

pos = double(pos);
[v,sigma,u] = svd(pos);
numRows = size(E1)(1);
numColumns = size(E1)(2);
p = zeros(numRows, numColumns);
q = zeros(numRows, numColumns);
for R=1:numRows
    for C=1:numColumns
        e = [E1(R,C);E2(R,C);E3(R,C)];
        pq = double(v) * double(pinv(sigma) * double(double(u') * double(e)));
        if pq(3) != 0
         p(R,C) = pq(1)/pq(3);
         q(R,C) = pq(2)/pq(3);
        endif
    endfor
endfor

save p.mat p
save q.mat q
tElapsed=toc(tStart)
```

tElapsed = 1.2303

We use this code to estimate the depth matrix using the values of p and q:-

```
tStart=tic;
p = load("p.mat");
q = load("q.mat");
i1 = size(q,1);
j1= size(p,2);
z = rand(i1, j1);

for count = 1 : 1000
  i1 = size(p,1);
  j1= size(p,2);

  for i = 3 : (i1 - 3)
    for j = 3 : (j1 - 3)
      zn(i, j) = (z(i, j) + p(i,j) - p(i - 1, j) + q(i,j) - q(i, j - 1))/4;
    endfor
  endfor
  z = zn;
endfor

save depth_matrix.mat z
tElapsed=toc(tStart)
```

tElapsed = 515.26

Figure 1: ps = 0, qs = 1

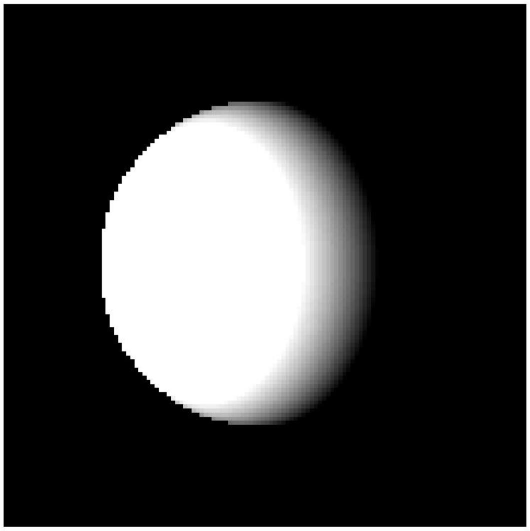

Figure 2: ps = 0.5, qs = 0.5

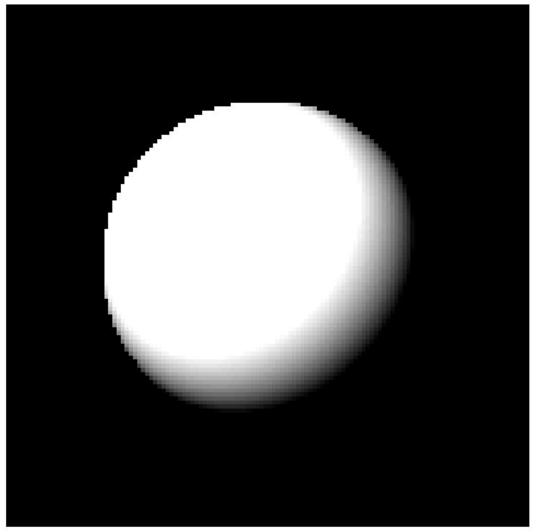

Figure 3: ps = 0.25, qs = 0.25

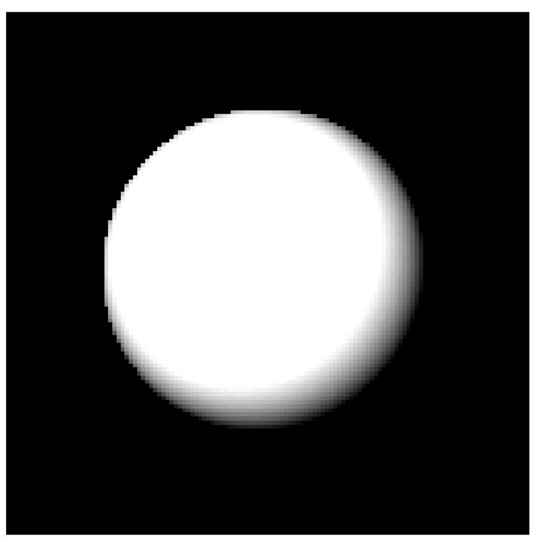

Figure 4: Depth matrix plotted as a image