Homework 5 Writeup

Instructions

- Describe any interesting decisions you made to write your algorithm.
- Show and discuss the results of your algorithm.
- Feel free to include code snippets, images, and equations.
- Use as many pages as you need, but err on the short side If you feel you only need to write a short amount to meet the brief, th
- **Please make this document anonymous.**

Interesting Implementation Detail

1. My code about bilinear interpolation

```
    [n, m, f] = size(bayer_img);

    rgb_img = zeros(n,m,3);

    for i = 1:n
        for j = 1:m
            if mod(i, 2) == 1 && mod(j, 2) == 1
                rgb_img(i, j, 1) = bayer_img(i, j);
            elseif mod(i, 2) == 1 && mod(j, 2) == 0
                if j == m
                    rgb_img(i,j,1) = bayer_img(i,j - 1);
                else
                    rgb_img(i,j,1) = (bayer_img(i,j-1) +
                       bayer_img(i,j+1)) / 2;
                end
            elseif mod(i, 2) == 0 && mod(j, 2) == 1
                if i == n
                    rgb_img(i,j,1) = bayer_img(i - 1,j);
                else
                    rgb_img(i,j,1) = (bayer_img(i - 1,j)
                       + bayer_img(i + 1,j)) / 2;
                end
            elseif mod(i, 2) == 0 && mod(j, 2) == 0
                if i == n && j == m
                    rgb_img(i,j,1) = bayer_img(i-1,j-1);
                elseif i == n
```

```
                    rgb_img(i,j,1) = (bayer_img(i-1,j-1)
                       + bayer_img(i-1,j+1)) / 2;
                elseif j == m
                    rgb_img(i,j,1) = (bayer_img(i-1,j-1)
                       + bayer_img(i+1,j-1)) / 2;
                else
                    rgb_img(i,j,1) = (bayer_img(i-1,j-1)
                       + bayer_img(i-1,j+1) + bayer_img(i
                       +1,j-1) + bayer_img(i+1,j+1)) / 4;
                end
            end
        end
    end


    for i = 1:n
        for j = 1:m
            if mod(i, 2) == 0 && mod(j, 2) == 0
                rgb_img(i, j, 3) = bayer_img(i, j);
            elseif mod(i, 2) == 0 && mod(j, 2) == 1
                if j == 1
                    rgb_img(i,j,3) = bayer_img(i,j + 1);
                else
                    rgb_img(i,j,3) = (bayer_img(i,j-1) +
                       bayer_img(i,j+1)) / 2;
                end
            elseif mod(i, 2) == 1 && mod(j, 2) == 0
                if i == 1
                    rgb_img(i,j,3) = bayer_img(i + 1,j);
                else
                    rgb_img(i,j,3) = (bayer_img(i - 1,j)
                       + bayer_img(i + 1,j)) / 2;
                end
            elseif mod(i, 2) == 1 && mod(j, 2) == 1
                if i == 1 && j == 1
                    rgb_img(i,j,3) = bayer_img(i+1,j+1);
                elseif i == 1
                    rgb_img(i,j,3) = (bayer_img(i+1,j-1)
                       + bayer_img(i+1,j+1)) / 2;
                elseif j == 1
                    rgb_img(i,j,3) = (bayer_img(i-1,j+1)
                       + bayer_img(i+1,j+1)) / 2;
                else
                    rgb_img(i,j,3) = (bayer_img(i-1,j-1)
                       + bayer_img(i-1,j+1) + bayer_img(i
                       +1,j-1) + bayer_img(i+1,j+1)) / 4;
```

```
                end
            end
        end
    end

    for i = 1:n
        for j = 1:m
            if mod(i,2) ~= mod(j,2)
                rgb_img(i,j,2) = bayer_img(i,j);
            else
                if i == 1 && j == 1
                    rgb_img(i,j,2) = (bayer_img(i,j+1) +
                       bayer_img(i+1,j)) / 2;
                elseif i == n && j == m
                    rgb_img(i,j,2) = (bayer_img(i,j-1) +
                       bayer_img(i-1,j)) / 2;
                elseif i == 1
                    rgb_img(i,j,2) = (bayer_img(i,j-1) +
                       bayer_img(i,j+1) + bayer_img(i+1,j
                       )) / 3;
                elseif i == n
                    rgb_img(i,j,2) = (bayer_img(i,j-1) +
                       bayer_img(i,j+1) + bayer_img(i-1,j
                       )) / 3;
                elseif j == 1
                    rgb_img(i,j,2) = (bayer_img(i-1,j) +
                       bayer_img(i,j+1) + bayer_img(i+1,j
                       )) / 3;
                elseif j == m
                    rgb_img(i,j,2) = (bayer_img(i-1,j) +
                       bayer_img(i,j-1) + bayer_img(i+1,j
                       )) / 3;
                else
                    rgb_img(i,j,2) = (bayer_img(i-1,j) +
                       bayer_img(i+1,j) + bayer_img(i,j
                       -1) + bayer_img(i,j+1))/ 4;
                end
            end
        end
    end

    rgb_img = cast(rgb_img, 'uint8');
```

When I fill missing R, G, B values using bayer image, I divide 4 case, (number of values that are interpolated is 1, 2, 3, 4) with individually R, G, B case. For example, filling corner of R, B value, I use 1 points, interpolating edge of R, B value and corner of G

value, I use 2 points. And interpolating edge of G value, I use 3 points, another cases, I use 4 points.

Result

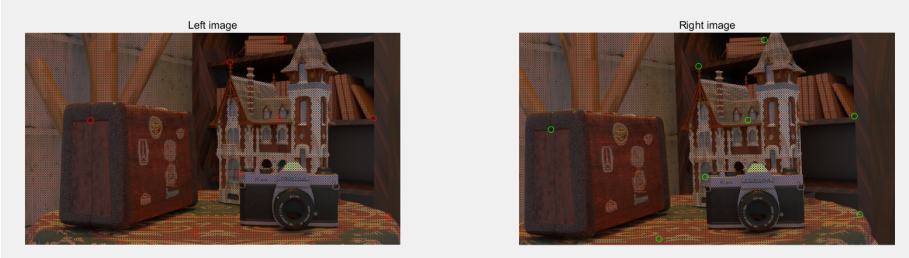

Figure 1: My result for bilinear interpolation.

2. Finding Fundamental matrix

```
1
2     [n, m] = size(pts1);
3
4     A = zeros(n,9);
5
6     for i=1:n
7         x = pts1(i,1);
8         y = pts1(i,2);
9         x2 = pts2(i,1);
10        y2 = pts2(i,2);
11        A(i,1) = x*x2;
12        A(i,2) = x*y2;
13        A(i,3) = x;
14        A(i,4) = y*x2;
15        A(i,5) = y*y2;
16        A(i,6) = y;
17        A(i,7) = x2;
18        A(i,8) = y2;
19        A(i,9) = 1;
20    end
21
22    A2 = A.' * A;
23    [v, e] = eig(A2);
24
25    k = 1;
26    eigen_value = e(k,k);
27    f = v(:,k);
28    f = f / norm(f);
29    F = zeros(3,3);
```

```
    for i=1:3
        for j=1:3
            F(i,j) = f(i + (j-1) * 3);
        end
    end

    [U, S, V] = svd(F);
    S(3,3) = 0;
    F2 = U*S*V.';

    f = F2;
```

I followed supplementary pdf's flow.

3. Finding Homography matrix for image rectification

```
    w = imageSize(2);
    h = imageSize(1);
    t = eye(3);
    t(1,3) = -w / 2;
    t(2,3) = -h / 2;
    e = epipole / epipole(3);
    e2 = t * e;

    ang = atan2(e2(2), e2(1));
    co = cos(-ang + pi);
    si = sin(-ang + pi );

    r = eye(3);
    r(1,1) = co;
    r(1,2) = -si;
    r(2,1) = si;
    r(2,2) = co;

    e3 = r * e2;
    g = eye(3);
    g(3,1) = -1 / e3(1);
```

I followed supplementary pdf's flow too. To make z's coordinate to 1, I execute normalizing first. I find Rotation matrix that make y coordinate to 1 and based on z-axis

Result

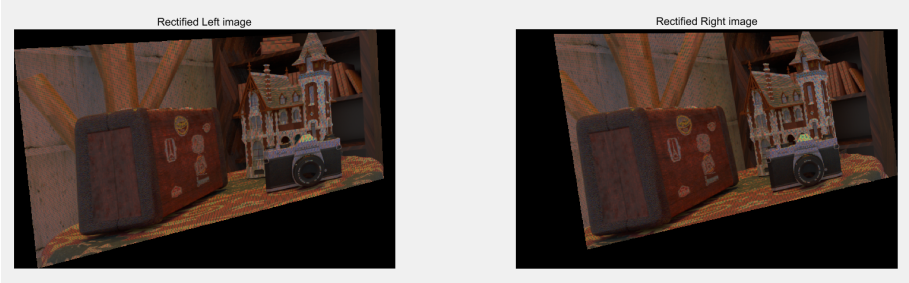

Figure 2: My result for image rectification

4. Stereo matching and disparity map

```
s = size(img_right);
w = s(2);
h = s(1);

cost_vol = zeros(h, w, max_disparity);

pad_left = padarray(img_left, [0 max_disparity], -1,
    'post');

block_pad = round(window_size/2) - 1;
padded_left = padarray(pad_left,[block_pad block_pad
    ], 'replicate', 'both');

pad_right = padarray(img_right, [block_pad block_pad
    ], 'replicate', 'both');


filter = zeros(window_size, window_size);

filter(1:window_size, 1:window_size) = 1 / (
    window_size^2);

avg_right = imfilter(img_right, filter, 'replicate');
avg_left = imfilter(pad_left, filter, 'replicate');

a = zeros(window_size^2);
b = zeros(window_size^2);
for i=1:h
    for j=1:w
        for d = 1:max_disparity
            avg_l = avg_left(i, j + d);
            avg_r = avg_right(i,j);
            A = padded_left(i:i+block_pad+1, j+d:j+d+
```

```
                    block_pad+1);
30                  a = A(:) - avg_l;
31                  B = pad_right(i:i+block_pad+1, j:j+
                        block_pad+1);
32                  b = B(:) - avg_r;
33                  cost_vol(i,j,d) = -dot(a,b) / (norm(a) *
                        norm(b));
34              end
35          end
36      end
```

To up executed speed, I used average filter to calculate average of all [window size * window size] subarray first. These average values are used when get NCC score. NCC score's range is [-1, 1]. When NCC score is 1, I can say this is matched. So I multiple -1 to NCC score for change min score mean matching.

5. Aggregate the cost volume using box filter.

```
1       for d = 1:max_disparity
2           cost_vol(:,:,d) = imfilter(cost_vol(:,:,d),
                filter, 'replicate');
3       end
```

I aggregate the cost volume individually d value with 2D box filter.

Result

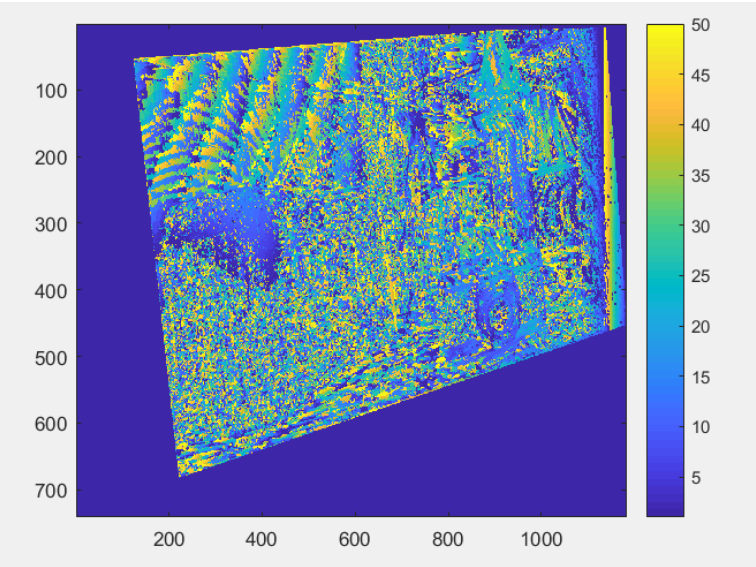

Figure 3: My result for finding disparity map. (window size = 3, max disparity = 50)