Homework 1 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.
- There is no page limit.

In the beginning...

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. See Equation 1.

$$a = b + c \tag{1}$$

Interesting Implementation Detail

```
for k = 1:numView
    [~, ~, V] = svd(L(:,:,k));
    h = V(:,9);
    h = h / h(9);
    homography(:,:,k) = reshape(h, [3,3])';
end
```

In homography calculation part, I calculate smallest right singular vector of L and normalize h33 to be 1, at each view

```
function [objective] = func_calibration(imagePoints,
   worldPoints, x)

numView = size(imagePoints,3);
hat_m = zeros(size(imagePoints));

% ----- Your code here (9) -----

```

```
K = [1, 0, 0
     0, 1, 0
     0, 0, 1];

K(1,1) = x(1);
K(1,2) = x(2);
K(1,3) = x(3);
K(2,2) = x(4);
K(2,3) = x(5);

for view = 1:numView
    rvec = x(6*view+3:6*view+5);
    tvec = x(6*view:6*view+2);
    R = rotationVectorToMatrix(rvec);
    R = R';
    P = K * [R(:,1), R(:,2), reshape(tvec,[3,1])];
    points = worldPoints;
    points = [points, ones(size(imagePoints,1), 1)];
    hat = P * points';
    hat(1,:) = hat(1,:) ./ hat(3,:);
    hat(2,:) = hat(2,:) ./ hat(3,:);
    hat_m(:,:,view) = hat(1:2,:)';
end

objective = imagePoints - hat_m;
```

In optimizing function used to MLE, first, I reconstruct K and Projection matrix from x. Then I add ones vector to worldPoints matrix, because of making homogenious coord. And I calcuate q(hat)ij.

```
cost_vol = zeros(m, n, max_disparity);

pad_right = padarray(rightImageGray, [0 max_disparity], '
   replicate', 'pre');

block_pad = round(w/2) - 1;

pad_left = padarray(leftImageGray,[block_pad block_pad],
   'replicate', 'both');
padded_right = padarray(pad_right, [block_pad block_pad],
    'replicate', 'both');


filter = zeros(w);
filter(1:w, 1:w) = 1 / (w^2);

avg_right = imfilter(pad_right, filter, 'replicate');
```

```
avg_left = imfilter(leftImageGray, filter, 'replicate');

for i=1:m
    for j=1:n
        for d = 1:max_disparity
            j2 = j -d + max_disparity;
            avg_l = avg_left(i, j);
            avg_r = avg_right(i,j2);
            A = pad_left(i:i+w-1, j:j+w-1);
            a = A(:) - avg_l;
            B = padded_right(i:i+w-1, j2:j2+w-1);
            b = B(:) - avg_r;
            cost_vol(i,j,d) = -dot(a,b) / (norm(a) * norm
                (b));
        end
    end
end
```

In cost volume with a cost function and cost aggregation part, first, because of boundary I do padding image, then I used average filter to calculate average of all [window size *window size] subarray first. These average values are used when get NCC score. NCCscore's range is [-1, 1]. When NCC score is 1, I can say this is matched. So I multiple -1to NCC score for change min score mean matching

```
cost_vol2 = zeros(m, n, max_disparity);
parent = zeros(m, n, max_disparity);
min_dis = 10;
dis = min_dis:max_disparity;
dis_size = max_disparity - min_dis + 1;

rob = @(x) x.^2 ./ (1 + x.^2);
cost_vol2(:,n,:) = cost_vol(:,n,:);
for i=1:m
    for j=n-1:-1:1
        for d = min_dis:max_disparity
            r = rob(d - dis);
            cost = reshape(cost_vol2(i,j + 1,min_dis:
                max_disparity), [dis_size, 1]) ...
                 + reshape(r, [dis_size, 1]);
            [value, idx] = min(cost);
            cost_vol2(i,j,d) = cost_vol(i,j,d) + value;
            parent(i,j,d) = idx + min_dis - 1;
        end
    end
end
[~, idx] = min(cost_vol2(:,1,min_dis:max_disparity),[],3)
   ;
```

```
for i=1:m
    disparityMap(i,1) = idx(i) + min_dis - 1;
    idx_next = parent(i,1,idx(i) + min_dis - 1);
    for j=2:n
        disparityMap(i,j) = idx_next;
        idx_next = parent(i,j,idx_next);
    end
end
```

In constructing Disparity map part, I use Energy minimization and Dynamic Programming algorithm learned in class, and described below. First triplication for loop is the process of updating the entire cost volume using DP algorithm that considers the disparity of the neighboring pixels, and second double for loop is the process of backtracking which disparity should be given to each pixel.

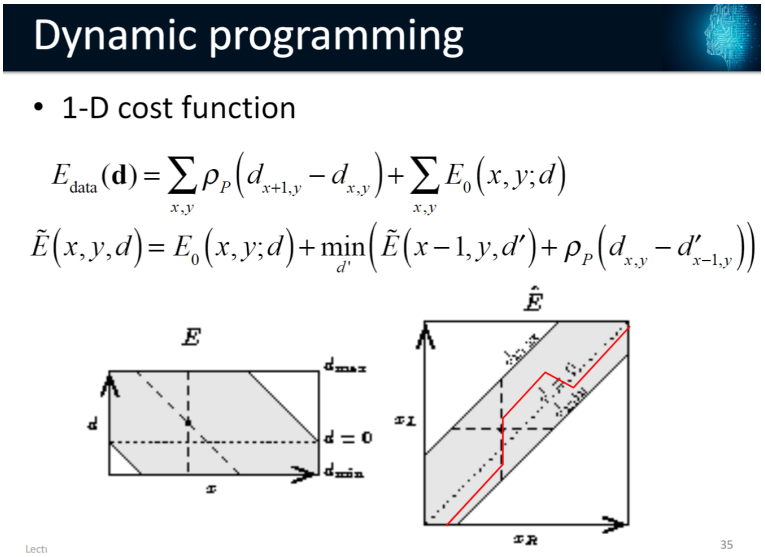

Figure 1: Energy minimization + DP algorithm

A Result

```
>> main
Starting parallel pool (parpool) using the 'local' profile ...
Connected to the parallel pool (number of workers: 4).

국소 최솟값이 있을 수 있습니다.
상대적인 현재 스텝의 크기가 스텝 크기 허용오차
값보다 작기 때문에 lsqnonlin이(가) 중지되었습니다.

<중지 기준 세부 정보>

국소 최솟값이 있을 수 있습니다.
상대적인 현재 스텝의 크기가 스텝 크기 허용오차
값보다 작기 때문에 lsqnonlin이(가) 중지되었습니다.

<중지 기준 세부 정보>
Evaluation [01]
Original y coordinate mean difference: 11.9712
Rectified y coordinate mean difference: 0.0559
Evaluation [02]
Original y coordinate mean difference: 14.7440
Rectified y coordinate mean difference: 0.0278

Start depth estimation [scene1]
Depth mean difference: 375.29

Start depth estimation [scene2]
Depth mean difference: 1425.72

IdleTimeout has been reached.
Parallel pool using the 'local' profile is shutting down.
```

Figure 2: Output result of entire code

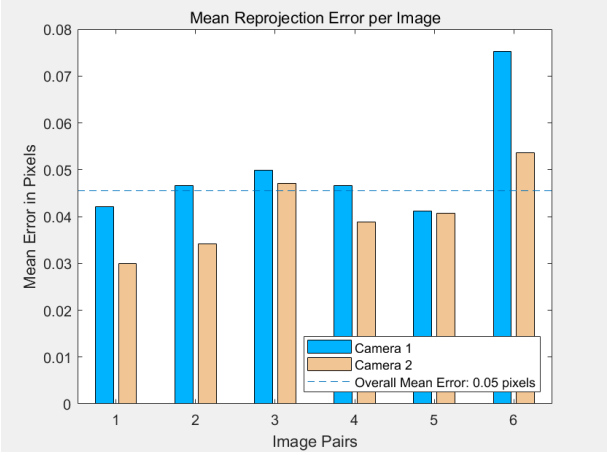

Figure 3: Reprojection error of camera parameters

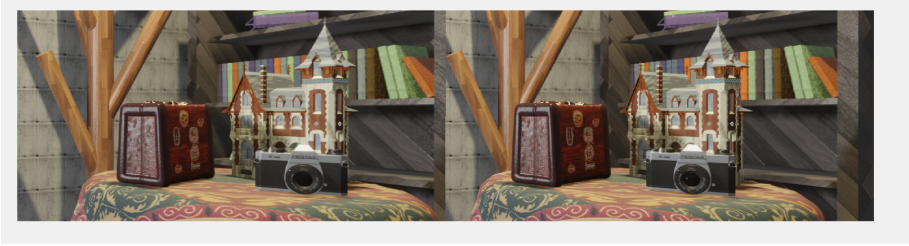

Figure 4: Reprojection result of scene1

Figure 5: Reprojection result of scene2

Reprojection error is 0.0x pixel units, so it can be said it I got very good result. I guess that some error can be caused by inaccuracy of input value and svd function.

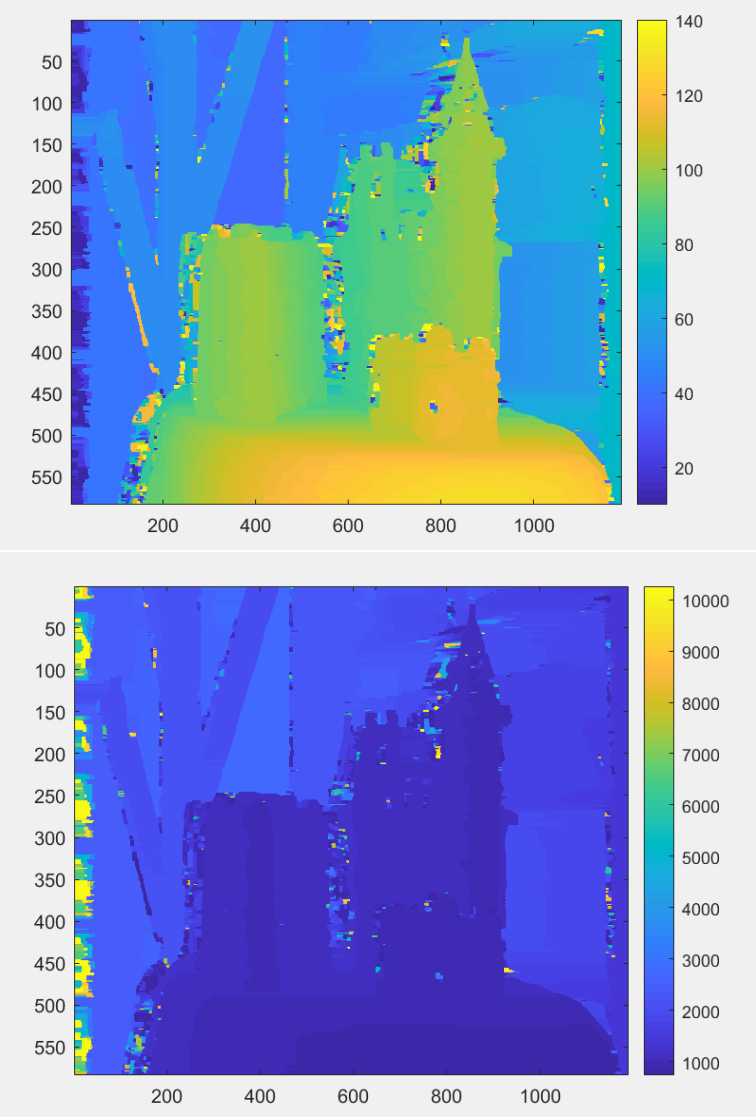

Figure 6: up : disparity image of scene1, down : depth image of scene1

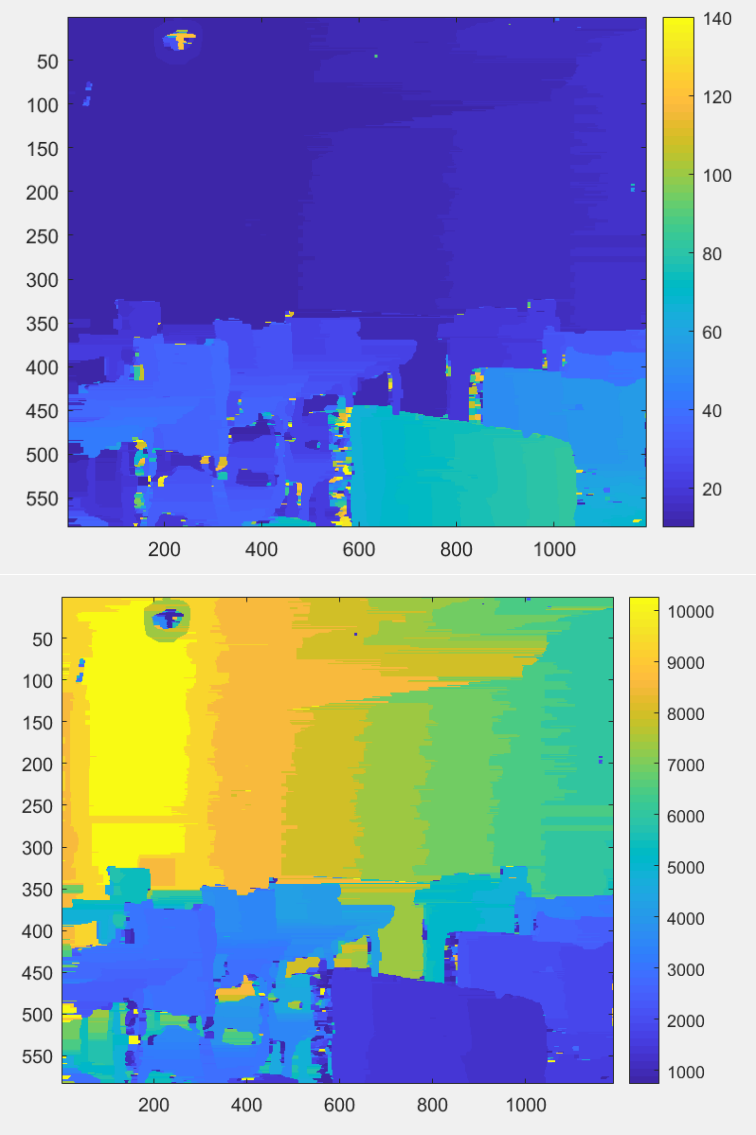

Figure 7: up : disparity image of scene2, down : depth image of scene2

I think error of disparity and depth can be caused by lack of parameter tuning and boundary condition. In addition, when low disparity case, small error of disparity cause large error of depth , so there was a high error on the part of the blackboard with relatively low disparity.