

I. MULTIPLE VIEW GEOMETRY HW4

student: 임규범, student No.: 2018-22219

A. Problem 1

11.13.2(ii): 3차원 공간 상의 4개의 점 $\mathbf{X}_i, i = \{1, 2, 3, 4\}$ 이 동일한 평면 π 상에 위치하는 경우 두 카메라는 π 를 기준으로 Homography 변환이 가능하다. 두 카메라의 이미지 평면 상의 점을 \mathbf{x}, \mathbf{x}' 라고 하고 Homography 행렬을 $\mathbf{H} \in \mathbb{R}^{3 \times 3}$ 라고 했을 때

$$\mathbf{x}' = \mathbf{H}\mathbf{x} \quad (1)$$

과 같은 공식이 성립한다. 이 때, 첫번째 이미지 평면상의 Epipole \mathbf{e} 는 Homography 변환을 통해 두번째 이미지 평면상의 Epipole \mathbf{e}' 로 변환된다.

$$\mathbf{e}' = \mathbf{H}\mathbf{e} \quad (2)$$

Homography 행렬 \mathbf{H} 는 4개의 대응점 쌍을 통해 구할 수 있다.

$$\mathbf{x}'_i = \mathbf{H}\mathbf{x}_i, \quad i = 1, \dots, 4 \quad (3)$$

이 때, 일반적으로 DLT 방법을 통해 \mathbf{H} 를 구한다. \mathbf{H} 를 구했으면 다음으로 두번째 이미지 평면 상의 Epipole \mathbf{e}' 를 구한다. 이 때, Quadric Outline 상의 임의의 두 점을 $\mathbf{X}_5, \mathbf{X}_6$ 라고 하면 이를 첫번째 이미지에 투영한 점은 $\mathbf{x}_5, \mathbf{x}_6$ 가 되고

$$\mathbf{e}' \text{ is intersection of } (\mathbf{H}\mathbf{x}_5) \times \mathbf{x}'_5 \text{ and } (\mathbf{H}\mathbf{x}_6) \times \mathbf{x}'_6 \quad (4)$$

와 같이 두 Epipolar Line의 교점을 통해 두번째 이미지 평면 상의 Epipole \mathbf{e}' 를 구할 수 있다. 다음으로 Fundamental Matrix \mathbf{F} 는 다음과 같이 구할 수 있다. $\mathbf{e}' = \mathbf{K}\mathbf{t}$ 일 때

$$\begin{aligned} \mathbf{e}'^\wedge \mathbf{H} &= (\mathbf{K}\mathbf{t})^\wedge \mathbf{K}(\mathbf{R} + \mathbf{t}\mathbf{n}_1^T/d_1)\mathbf{K}^{-1} \\ &= \mathbf{K}^{-T} \mathbf{t}^\wedge (\mathbf{R} + \mathbf{t}\mathbf{n}_1^T/d_1) \mathbf{K}^{-1} \quad (\mathbf{K}\mathbf{t})^\wedge = \mathbf{K}^{-T} \mathbf{t}^\wedge \mathbf{K}^{-1} \\ &= \mathbf{K}^{-T} \mathbf{t}^\wedge \mathbf{R} \mathbf{K}^{-1} \\ &= \mathbf{K}^{-T} \mathbf{E} \mathbf{K}^{-1} \\ &= \mathbf{F} \end{aligned} \quad (5)$$

따라서 $\mathbf{F} = \mathbf{e}'^\wedge \mathbf{H}$ 이다. 이 때 \mathbf{H} 를 이용하여 \mathbf{R}, \mathbf{t} 를 분해하게 되면 \mathbf{R} 의 해가 2개가 나오는 two-fold ambiguity 문제가 발생한다. 이를 자세하게 설명하면 다음과 같다.

평면 $\pi = (\mathbf{n}, z)$ 가 있을 때 \mathbf{P}_1 의 Homography \mathbf{H}_1 는 다음과 같이 유도할 수 있다.

$$\begin{aligned} \mathbf{x} &= \mathbf{P}_1 \mathbf{X} \\ &= \mathbf{K} \mathbf{R} \tilde{\mathbf{X}} + \mathbf{t} \left(-\frac{\mathbf{n}^T \tilde{\mathbf{X}}}{z} \right) \\ &= \mathbf{K} \left(\mathbf{R} - \frac{\mathbf{t} \mathbf{n}^T}{z} \right) \tilde{\mathbf{X}} \\ &= \mathbf{H}_1 \tilde{\mathbf{X}} \end{aligned} \quad (6)$$

이 때 $\mathbf{H}_1 = \mathbf{K} \left(\mathbf{R} - \frac{\mathbf{t} \mathbf{n}^T}{z} \right)$ 를 평면 π 에 대한 Homography라고 한다. 이를 \mathbf{P}_2 의 Homography로 나타내면 두 Canonical Camera $\mathbf{P}_1 = [\mathbf{I}|0], \mathbf{P}_2 = [\mathbf{R}|\mathbf{Rt}]$ 가 있을 때

$$\mathbf{H}_2 = \mathbf{R} \mathbf{H}_1 \quad (7)$$

이 된다. 이 때 두 Homography $\mathbf{H}_1, \mathbf{H}_2$ 를 각각 SVD 분해하면

$$\begin{aligned} \mathbf{H}_1 &= \mathbf{U}_2 \mathbf{D} \mathbf{V}^T = \begin{bmatrix} \mathbf{u}'_1 & \mathbf{u}'_2 & \mathbf{u}'_3 \end{bmatrix} \begin{bmatrix} d_1 & & \\ & d_2 & \\ & & d_3 \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 \end{bmatrix} \\ \mathbf{H}_2 &= \mathbf{U}_1 \mathbf{D} \mathbf{V}^T = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \mathbf{u}_3 \end{bmatrix} \begin{bmatrix} d_1 & & \\ & d_2 & \\ & & d_3 \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 \end{bmatrix} \end{aligned} \quad (8)$$

가 되고 이 때, $\mathbf{U}_2 = \mathbf{R}\mathbf{U}_1$ 의 관계가 성립한다. 이 때, $\mathbf{H}_1 = \mathbf{I} - \frac{\mathbf{t}\mathbf{n}^T}{z}$ 에서 $\frac{\mathbf{t}\mathbf{n}^T}{z}$ 항목만 복원하기 위해 우선적으로 $\|\mathbf{t}\| = \|\mathbf{n}\| = 1$ 로 정규화한다.

정규화를 수행할 때 \mathbf{H}_1 에 상관없이 $\mathbf{t}^\wedge \mathbf{n}$ 값은 불변하므로 특이값이 0이 아닌 경우 $\mathbf{t}^\wedge \mathbf{n}$ 은 반드시 특이값에 상응하는 벡터 \mathbf{v}_2 와 비례 관계를 갖는다. 따라서 $\mathbf{H} \rightarrow \mathbf{H}/d_2$, $\mathbf{D} \rightarrow \mathbf{D}/d_2$ 와 같이 정규화가 되어야 한다. 즉, $(d_1, d_2, d_3) \rightarrow (d_1/d_2, 1, d_3/d_2)$ 와 같이 정규화한다.

\mathbf{v}_2 와 상응하는 벡터 $\mathbf{t}^\wedge \mathbf{n}$ 이 주어졌을 때 $\{\mathbf{t}, \mathbf{n}\}$ 은 반드시 $\{\mathbf{v}_1, \mathbf{v}_3\}$ 가 Span하는 부분공간에 포함이 되어야 한다. 따라서

$$\begin{aligned}\mathbf{n} &= \beta \mathbf{v}_1 - \alpha \mathbf{v}_3 \\ \mathbf{n}^\wedge(\mathbf{t}^\wedge \mathbf{n}) &\sim \alpha \mathbf{v}_1 + \beta \mathbf{v}_2 \\ \alpha^2 + \beta^2 &= 1\end{aligned}\tag{9}$$

를 만족하는 α, β 가 존재한다. 또한 $\mathbf{n}^\wedge(\mathbf{t}^\wedge \mathbf{n})$ 과 같이 \mathbf{n} 방향과 직교하는 벡터들은 $\mathbf{H}_1, \mathbf{H}_2$ 에 상관없이 길이가 변하지 않으므로

$$(\alpha d_1)^2 + (\beta d_3)^2 = \alpha^2 + \beta^2 = 1\tag{10}$$

이 성립하고 이에 따라 $(\alpha, \beta) \sim (\pm \sqrt{1-d_3^2}, \pm \sqrt{d_1^2-1})$ 이 성립한다. 여기서 $\mathbf{t}^\wedge \mathbf{n}$ 은 $\mathbf{v}_1, \mathbf{v}_3$ 에 비례하는 값을 의미하므로 실수해가 존재하지 않고 오직 \mathbf{v}_2 에서만 실수해가 존재한다.

SVD 행렬을 전개해보면 $\mathbf{R}\mathbf{t} = -(\beta \mathbf{u}_1 + \alpha \mathbf{u}_3)$ 관계가 성립한다. 이 때, \mathbf{t} 는 \mathbf{H}_1 의 eigenvector이고 eigenvalue는 $(1 - \frac{\mathbf{n}\mathbf{t}}{z})$ 이므로

$$\begin{aligned}\mathbf{H}_2 \mathbf{t} &= \left(1 - \frac{\mathbf{n}\mathbf{t}}{z}\right) \mathbf{R}\mathbf{t} \\ \mathbf{t} &\sim \mathbf{H}_2^{-1}(\mathbf{R}\mathbf{t}) \sim \beta/d_1 \mathbf{v}_1 + \alpha/d_3 \mathbf{v}_3\end{aligned}\tag{11}$$

공식이 성립하고 이를 전개하여 정리하면 $z = \frac{1}{d_1 - d_3}$ 이 된다. \mathbf{H}_1 행렬은 $\mathbf{u}'_2 = \mathbf{v}_2$ 이며 \mathbf{t} 는 \mathbf{H}_1 의 eigenvector라는 조건이 주어졌을 때 다음이 성립한다.

$$\begin{aligned}\mathbf{u}'_1 &= \gamma \mathbf{v}_1 + \delta \mathbf{v}_3 \\ \mathbf{u}'_3 &= \gamma \mathbf{v}_3 - \delta \mathbf{v}_1\end{aligned}\tag{12}$$

이를 정리하면 $(\gamma, \delta) \sim (1 + d_1 d_3, \pm \alpha \beta)$ 가 된다. 따라서 \mathbf{R} 은 다음과 같이 복원할 수 있다.

$$\mathbf{R} = \mathbf{U}_2 \mathbf{U}_1^T = \mathbf{U}_2 \begin{bmatrix} \gamma & 0 & \pm \delta \\ 0 & 1 & 0 \\ \mp \delta & 0 & \gamma \end{bmatrix} \mathbf{V}^T\tag{13}$$

따라서 위와 같은 Homography \mathbf{H}_2 를 \mathbf{R}, \mathbf{t} 로 복원할 때 \mathbf{R} 행렬이 두 개의 해가 나오는 Two-fold ambiguity 문제가 발생한다.

11.13.2(iii): Quadric Envelop을 이미지 평면상에 프로젝션하면 다음과 같다.

$$\lambda \mathbf{l} = \mathbf{PLP}^T$$

where, $\mathbf{l} = \begin{bmatrix} l_1 & l_2 & l_4 \\ l_2 & l_3 & l_5 \\ l_4 & l_5 & l_6 \end{bmatrix}$ and $\mathbf{L} = \begin{bmatrix} L_1 & L_2 & L_4 & L_7 \\ L_2 & L_3 & L_5 & L_8 \\ L_4 & L_5 & L_6 & L_9 \\ L_7 & L_8 & L_9 & L_{10} \end{bmatrix}$

$$\tag{14}$$

이는 간략하게 다음과 같이 벡터 형태로 다시 작성할 수 있다.

$$\lambda \tilde{\mathbf{l}} = \tilde{\mathbf{P}} \tilde{\mathbf{L}}\tag{15}$$

이 때, $\tilde{\mathbf{l}} = [l_1, l_2, \dots, l_6]^T \in \mathbb{R}^{6 \times 1}$ 이고 $\tilde{\mathbf{L}} = [L_1, L_2, \dots, L_{10}]^T \in \mathbb{R}^{10 \times 1}$ 이다. 이 때, Quadric의 카메라 행렬 $\tilde{\mathbf{P}}$ 는 $\mathbb{R}^{6 \times 10}$ 행렬이다.

두 개의 카메라 $\{C\}_1, \{C\}_2$ 가 존재할 때 각각 이미지 평면으로 동일한 Quadric을 프로젝션하면 다음과 같다.

$$\begin{aligned}\lambda_1 \tilde{\mathbf{l}}_1 &= \tilde{\mathbf{P}}_1 \tilde{\mathbf{L}} \\ \lambda_2 \tilde{\mathbf{l}}_2 &= \tilde{\mathbf{P}}_2 \tilde{\mathbf{L}}\end{aligned}\tag{16}$$

이는 간결하게 다음과 같이 쓸 수 있다.

$$\begin{bmatrix} \tilde{\mathbf{P}}_1 & \tilde{\mathbf{I}}_1 & 0 \\ \tilde{\mathbf{P}}_2 & 0 & \tilde{\mathbf{I}}_2 \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{L}} \\ -\lambda_1 \\ -\lambda_2 \end{bmatrix} = 0 \quad (17)$$

이 때, 행렬 $\mathbf{M} = \begin{bmatrix} \tilde{\mathbf{P}}_1 & \tilde{\mathbf{I}}_1 & 0 \\ \tilde{\mathbf{P}}_2 & 0 & \tilde{\mathbf{I}}_2 \end{bmatrix} \in \mathbb{R}^{12 \times 12}$ 를 **Universal Matrix for conics**라고 부른다. 선형시스템의 특성상, $\tilde{\mathbf{I}}_1, \tilde{\mathbf{I}}_2$ 가 이미지 평면 상의 Conic이라고 하면 행렬 \mathbf{M} 은 0이 아닌 Null Space가 존재하게 되고 따라서

$$\text{rank } \mathbf{M} \leq 11 \quad (18)$$

이 성립한다. Canonical Form으로 표현한 카메라 행렬 대응쌍은 $\mathbf{P}_1 = [\mathbf{I}_{3 \times 3} \quad \mathbf{0}_{3 \times 1}], \mathbf{P}_2 = [\mathbf{M} \quad \mathbf{m}]$ 이고 Conic의 카메라 행렬은

$$\begin{aligned} \tilde{\mathbf{P}}_1 &= [\mathbf{I}_{6 \times 6} \quad \mathbf{0}_{6 \times 4}] \\ \tilde{\mathbf{P}}_2 &= [\mathbf{Q}_2 \quad \mathbf{R}_2] \\ \text{where, } \mathbf{Q}_2 &\in \mathbb{R}^{6 \times 6}, \mathbf{R}_2 \in \mathbb{R}^{6 \times 4} \end{aligned} \quad (19)$$

으로 나타낼 수 있다. 이를 통해 행렬 \mathbf{M} 을 다시 나타내면

$$\mathbf{M} = \begin{bmatrix} \mathbf{I}_{6 \times 6} & \mathbf{0}_{6 \times 4} & \tilde{\mathbf{I}}_1 & 0 \\ \mathbf{Q}_2 & \mathbf{R}_2 & 0 & \tilde{\mathbf{I}}_2 \end{bmatrix} \quad (20)$$

와 같이 나타낼 수 있다. 행렬 $\mathbf{M} \in \mathbb{R}^{12 \times 12}$ 은 다음과 같이 다시 축소된 형태의 행렬 $\mathbf{M}_r \in \mathbb{R}^{6 \times 6}$ 로 나타낼 수 있다. 유도과정은 다음과 같다.

$$\begin{aligned} &\begin{bmatrix} \mathbf{I}_{6 \times 6} & \mathbf{0}_{6 \times 4} & \tilde{\mathbf{I}}_1 & 0 \\ \mathbf{Q}_2 & \mathbf{R}_2 & 0 & \tilde{\mathbf{I}}_2 \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{L}} \\ -\lambda_1 \\ -\lambda_2 \end{bmatrix} = 0 \\ &\Rightarrow \tilde{\mathbf{L}}_{(1:6)} - \lambda_1 \tilde{\mathbf{I}}_1 = 0 \\ &\Rightarrow (\mathbf{Q}_2 \tilde{\mathbf{L}}_{(1:6)} + \mathbf{R}_2 \tilde{\mathbf{L}}_{(7:10)}) - \lambda_2 \tilde{\mathbf{I}}_2 = 0 \end{aligned} \quad (21)$$

이고 $\tilde{\mathbf{L}}_{(1:6)} = \lambda_1 \tilde{\mathbf{I}}_1$ 이므로 이를 다시 정리하면

$$\begin{aligned} &(\mathbf{Q}_2 \lambda_1 \tilde{\mathbf{I}}_1 + \mathbf{R}_2 \tilde{\mathbf{L}}_{(7:10)}) - \lambda_2 \tilde{\mathbf{I}}_2 = 0 \\ &\mathbf{R}_2 \tilde{\mathbf{L}}_{(7:10)} + \mathbf{Q}_2 \lambda_1 \tilde{\mathbf{I}}_1 - \lambda_2 \tilde{\mathbf{I}}_2 = 0 \\ &\begin{bmatrix} \mathbf{R}_2 & -\mathbf{Q}_2 \tilde{\mathbf{I}}_1 & \mathbf{I}_2 \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{L}}_{(7:10)} \\ -\lambda_1 \\ -\lambda_2 \end{bmatrix} = 0 \end{aligned} \quad (22)$$

이 때, $\mathbf{M}_r = [\mathbf{R}_2 \quad -\mathbf{Q}_2 \tilde{\mathbf{I}}_1 \quad \mathbf{I}_2]$ 을 Reduced formd of \mathbf{M} 이라고 한다.

$$\begin{aligned} \mathbf{M}_r &\in \mathbb{R}^{6 \times 6} \\ \text{rank } \mathbf{M}_r &\leq 4 \end{aligned} \quad (23)$$

행렬 $\mathbf{R}_2 \in \mathbb{R}^{6 \times 4}$ 의 rank는 3보다 작거나 같으므로 네 개의 열 중 하나는 다른 세 열의 선형결합으로 표현할 수 있고 여기서 마지막 열을 제거한 행렬을 $\mathbf{M}_{6 \times 5}$ 행렬이라고 하면

$$\begin{aligned} \mathbf{M}_{6 \times 5} &\in \mathbb{R}^{6 \times 5} \\ \text{rank } \mathbf{M}_{6 \times 5} &\leq 4 \end{aligned} \quad (24)$$

이 된다. 위 식과 같이 rank가 4보다 작으므로 $\mathbf{M}_{6 \times 5}$ 행렬은 6개의 5×5 Subdeterminant들이 전부 0을 만족해야 한다. 이를 다시 작성하면 다음과 같다.

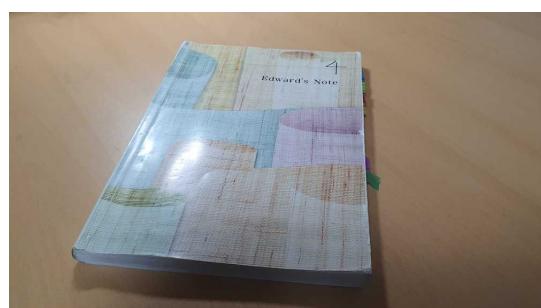
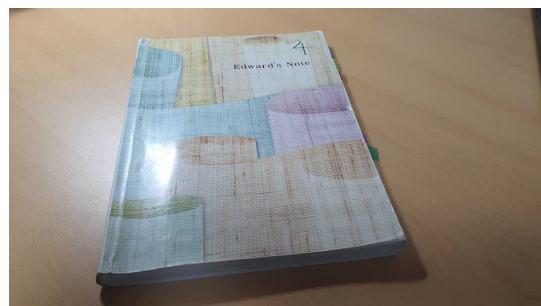
$$\tilde{\mathbf{I}}_1^T \mathbf{F}_i \tilde{\mathbf{I}}_2 = 0, \quad i = 1, 2, \dots, 6 \quad (25)$$

이 때, $\mathbf{F}_i \in \mathbb{R}^{6 \times 6}$ 는 $\mathbf{M}_{6 \times 5}$ 행렬에서 6개의 5×5 Subdeterminant로부터 얻은 행렬에서 $\tilde{\mathbf{I}}_1, \tilde{\mathbf{I}}_2$ 성분을 뺀 행렬이다. 이를 **Fundamental Matrix for conics**라고 한다. 즉, Conic의 Fundamental Matrix는 총 6개이다.

각각의 \mathbf{F}_i 는 $\text{rank } \leq 2$ 이며 $i = 1, 2, \dots, 6$ 중 3개의 \mathbf{F}_i 만 선형독립인 행렬이다. 여섯 개의 \mathbf{F}_i 들은 각각 $\mathbb{P}^3 \subset \mathbb{R}^4$ 상의 Quadric에 대한 공식을 의미하므로 2차원의 Null Space를 가지며 두 개의 서로 독립인 대수적 제약조건을 가진다.

B. Problem 2

Stereo Image Rectification을 수행하기 위해서 총 4장의 이미지를 사용하였다.



stereo_rectification.m 파일은 다음과 같다.

```
% load stereo images.  
im1 = imread('left2.jpg');  
im2 = imread('right2.jpg');  
  
% turn stereo images to grayscale.  
im1 = uint8(rgb2gray(im1));  
im2 = uint8(rgb2gray(im2));
```

```

[height, width] = size(im1);

% load intrinsic matrix.
load('cameraCalibrator.mat')
K = cameraParams.IntrinsicMatrix';

% extract surf features.
blobs1 = detectSURFFeatures(im1, 'MetricThreshold', 2000);
blobs2 = detectSURFFeatures(im2, 'MetricThreshold', 2000);
[features1, valid_blobs1] = extractFeatures(im1, blobs1);
[features2, valid_blobs2] = extractFeatures(im2, blobs2);
index_pairs = matchFeatures(features1, features2, 'Metric', 'SAD', 'MatchThreshold', 5);
surf1 = valid_blobs1(index_pairs(:,1),:);
surf2 = valid_blobs2(index_pairs(:,2),:);
x1 = surf1.Location;
x2 = surf2.Location;

% ransac-based best 8-points extraction method.
n = size(x1, 1);
best_inlier = 0;
best_idx = [];
for i = 1:1000
    % pick random 8 points.
    idx = randperm(n, 8); % 8x1
    x1_rand = x1(idx, :); % 8x2
    x2_rand = x2(idx, :); % 8x2
    x1_rand = transpose([x1_rand, ones(8, 1)]'); % 8x3
    x2_rand = transpose([x2_rand, ones(8, 1)]'); % 8x3

    F = calculate_fundamental_matrix(x1_rand', x2_rand');

    % pick the other (n-8) points and calculate inliers.
    others = setdiff(1:n, idx);
    x1_others = x1(others, :); % (n-8)x2
    x2_others = x2(others, :); % (n-8)x2
    x1_others_ = [x1_others ones(n-8, 1)]; % (n-8)x3
    x2_others_ = [x2_others ones(n-8, 1)]; % (n-8)x3

    epipolar_line_est = transpose(F * transpose(x1_others_)); % (n-8)x3

    % calculate distance.
    distance = abs(sum(epipolar_line_est .* x2_others_, 2)) ./ sqrt(sum(epipolar_line_est(:, 1:2) .* epipolar_line_est(:, 1:2), 2));

    % if distance is less than threshold.
    inlier_idx = find(distance <= 1);
    inlier = length(inlier_idx);

    % update best sample points.
    if inlier >= best_inlier
        best_inlier = inlier;
        best_idx = idx;
    end
end

x1_best = x1(best_idx, :); % 8x2
x2_best = x2(best_idx, :); % 8x2
x1_best = transpose([x1_best, ones(8, 1)]'); % 8x3
x2_best = transpose([x2_best, ones(8, 1)]'); % 8x3

% calculate fundamental matrix.
F = calculate_fundamental_matrix(x1_best', x2_best');

% visualize epipolar lines in left image.
figure(1);
imshow(im1);
figure(1);
epipolar_lines = epipolarLine(F',x2);
points = lineToBorderPoints(epipolar_lines,[height,width]);
line(points(:,[1,3])',points(:,[2,4])');

% visualize epipolar lines in right image.
figure(2);
imshow(im2);
figure(2);
epipolar_lines2 = epipolarLine(F,x1);
points2 = lineToBorderPoints(epipolar_lines2,[height,width]);
line(points2(:,[1,3])',points2(:,[2,4])');

% extract R,t from the essential matrix.
E = K'*F*K;
[R,t_] = relativeCameraPose(E,cameraParams, x1_best(:,1:2), x2_best(:,1:2));
rx = t_ / norm(t_);
rz_tilde = [0; 0; 1];

```

```

rz = (rz_tilde - dot(rz_tilde, rx) .* rx) / norm(rz_tilde - dot(rz_tilde, rx) .* rx);
ry = cross(rz, rx);

R_rect = [rx'; ry'; rz'];
H_left = K * R_rect * inv(K);
H_right = K * R_rect * R' * inv(K);

% recover the rectified image.
% left image.
im1 = double(im1);
H_left = inv(H_left);

[u_x, u_y] = meshgrid(1:(size(im1,2)), 1:(size(im1,1)));
v_x = H_left(1,1)*u_x + H_left(1,2)*u_y + H_left(1,3);
v_y = H_left(2,1)*u_x + H_left(2,2)*u_y + H_left(2,3);
v_z = H_left(3,1)*u_x + H_left(3,2)*u_y + H_left(3,3);
v_x = v_x./v_z;
v_y = v_y./v_z;

im1_warped(:,:,1) = reshape(interp2(im1(:,:,1), v_x(:), v_y(:)), [height, width]);
im1_warped = uint8(im1_warped);

% right image.
im2 = double(im2);
H_right = inv(H_right);

[u_x, u_y] = meshgrid(1:(size(im2,2)), 1:(size(im2,1)));
v_x = H_right(1,1)*u_x + H_right(1,2)*u_y + H_right(1,3);
v_y = H_right(2,1)*u_x + H_right(2,2)*u_y + H_right(2,3);
v_z = H_right(3,1)*u_x + H_right(3,2)*u_y + H_right(3,3);
v_x = v_x./v_z;
v_y = v_y./v_z;

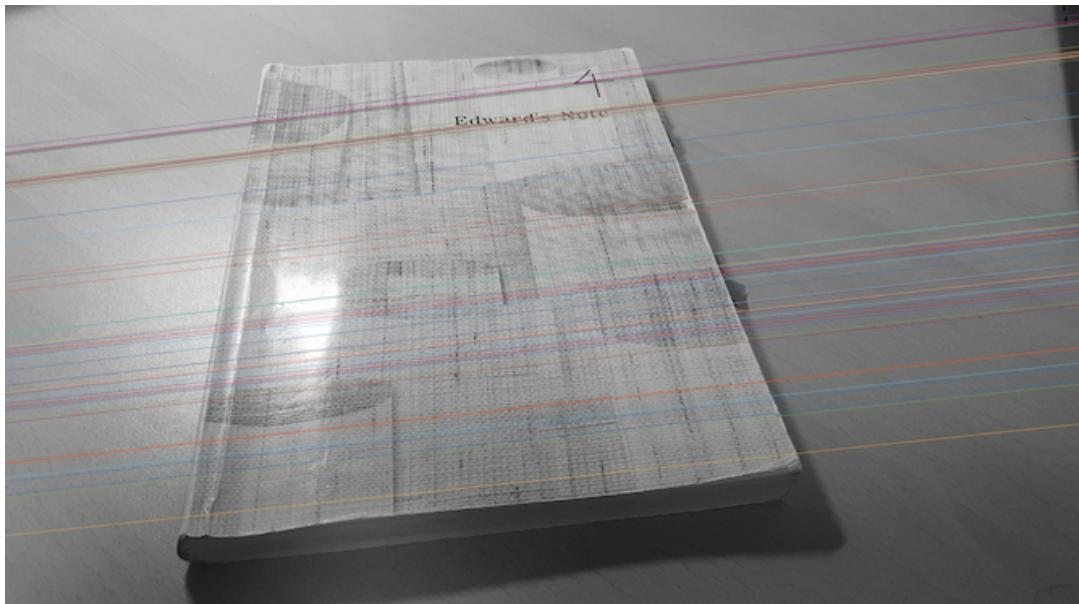
im2_warped(:,:,1) = reshape(interp2(im2(:,:,1), v_x(:), v_y(:)), [height, width]);
im2_warped = uint8(im2_warped);

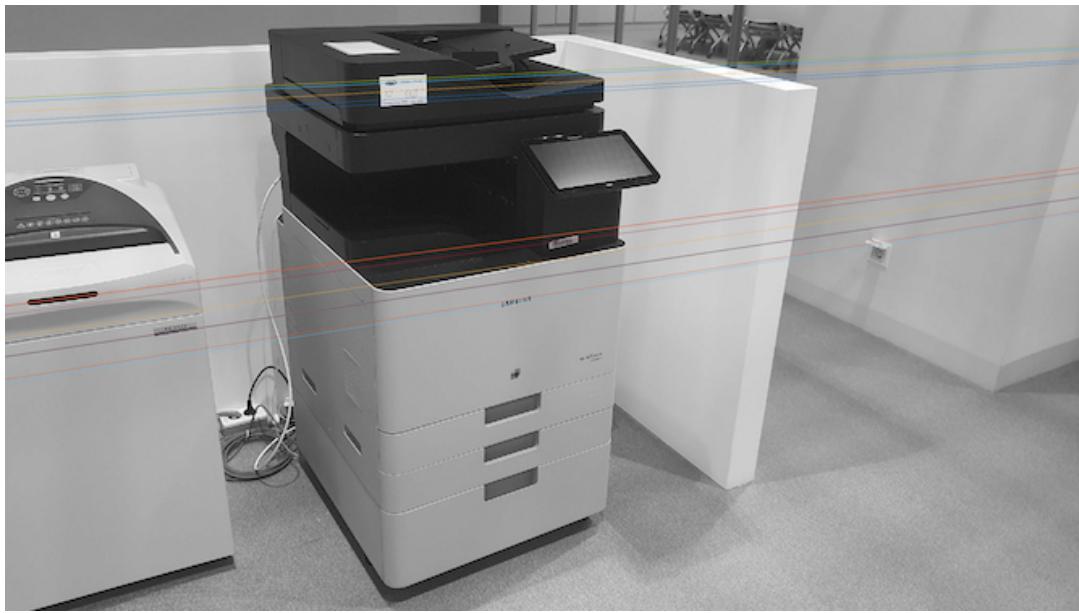
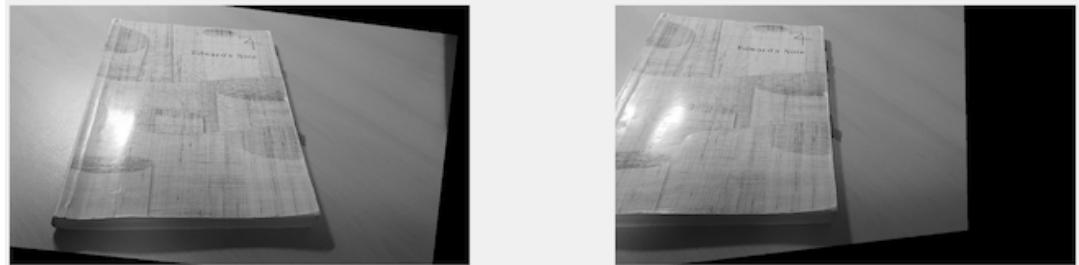
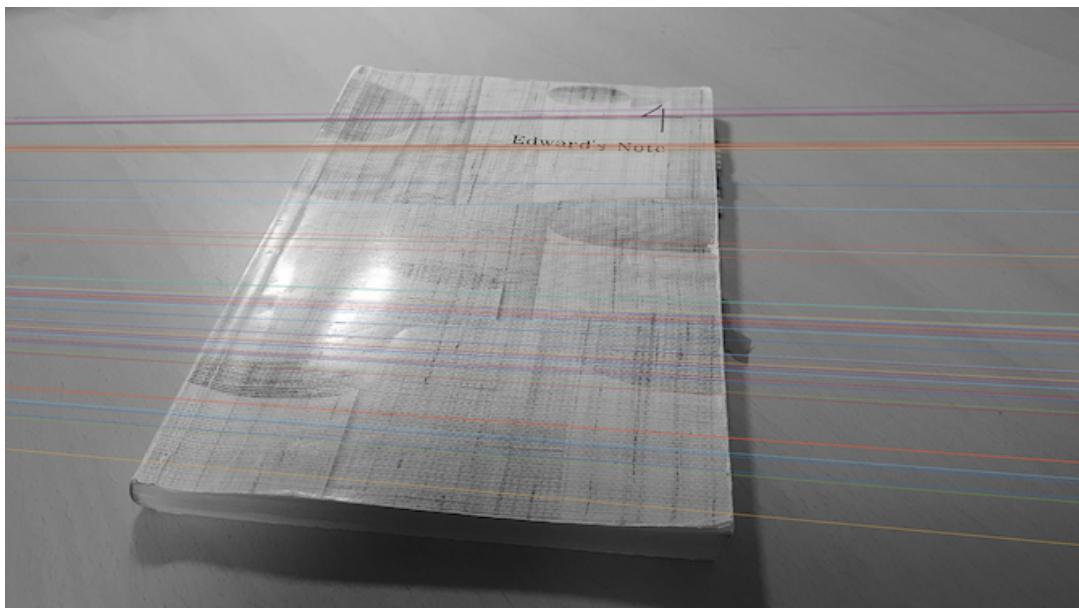
% visualize rectified image.
figure(3);
subplot(1,2,1);
imshow(im1_warped);

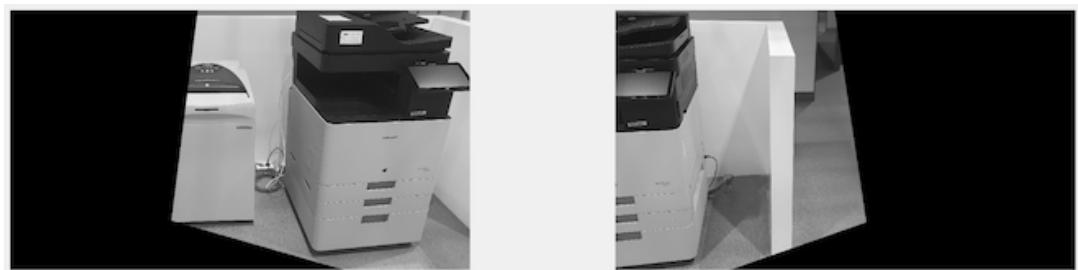
figure(3);
subplot(1,2,2);
imshow(im2_warped);

```

위 스크립트를 사용하여 얻은 Epipolar Line 이미지와 Stereo Rectification된 이미지는 다음과 같다.







II. REFERENCES

- Triggs, Bill. "Autocalibration from planar scenes." *European conference on computer vision*. Springer, Berlin, Heidelberg, 1998.
- Kahl, Fredrik, and Anders Heyden. "Using conic correspondences in two images to estimate the epipolar geometry." *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*. IEEE, 1998.