Adding compensation for radial distortion

Use already implemented Zhang procedure to obtain an estimation of camera parameters for each image we have at our disposal.

```
% load checkerboard images and
% obtain results of Zhang procedure
clear data
folder = 'images';
% call the function which implements Zhang procedure
data = zhangMethod(folder, [12,13]);
```

```
Data loaded and points detected
Checking the number of keypoints...
Estimating camera parameters for each image...
Everything done.
```

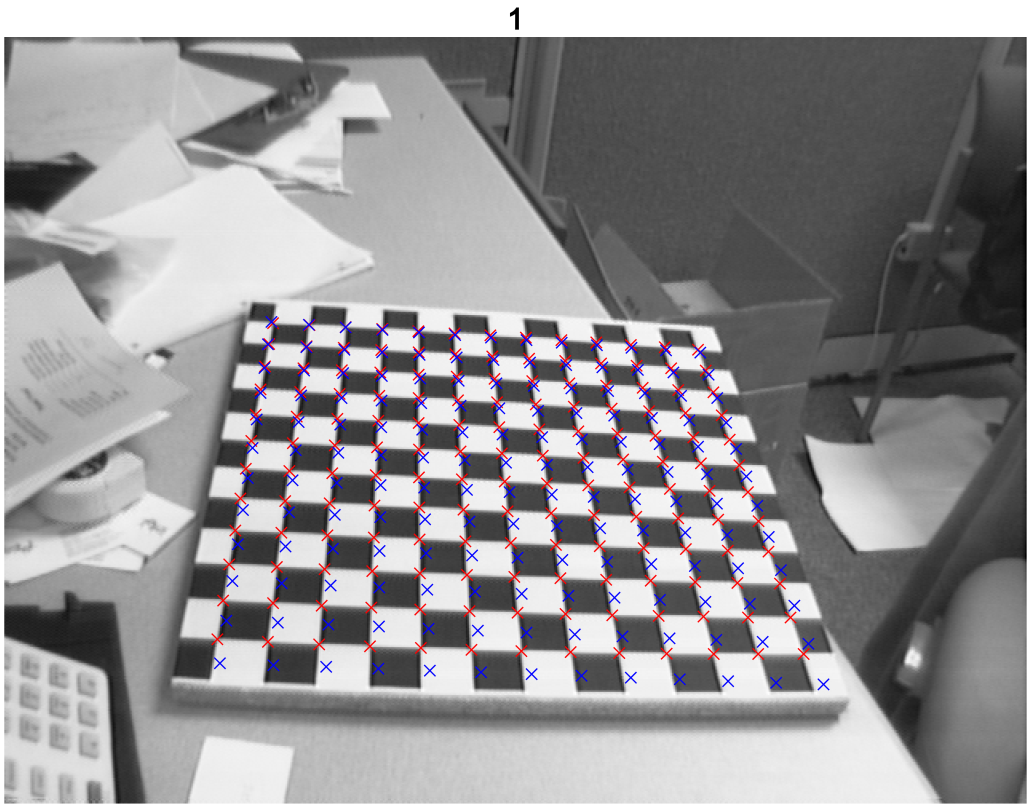
```
n = length(data); % number of images
```

Reprojection error

As we have already seen, if we perform projection using the obtained projection matrix, we get a significant error.

Let's see it clearly on the first image. We measure the reprojection error as the distance between the estimated and the measured projections.

```
idx = 1; % choose first image
XYpixel = data(idx).XYpixel;
XYmm = data(idx).XYmm;

error = 0;

figure
imshow(data(idx).I, 'initialmagnification',200);
title(num2str(idx))
hold on

for i=1:length(XYpixel)

    % coordinates of point in mm
    Xmm=XYmm(i,1);
    Ymm=XYmm(i,2);

    m = [Xmm; Ymm; 0; 1];

    P = data(idx).P;

    estimated_u =  (P(1,:)*m)/(P(3,:)*m);
    estimated_v = (P(2,:)*m)/(P(3,:)*m);
    % measure error wrt points detected on keyboard by
    % detectCheckerboardPoints function
    Xpixel=XYpixel(i,1);
    Ypixel=XYpixel(i,2);
    eu = estimated_u - Xpixel; % error on u
    ev = estimated_v - Ypixel; % error on v

    error = error + eu^2 + ev^2;

    scatter(Xpixel,Ypixel, 'x', 'r');
    scatter(estimated_u,estimated_v, 'x', 'b');
end
```



```
fprintf("Obtained reprojection error is: %d", error);
```

```
Obtained reprojection error is: 1.324210e+04
```

Compute the reprojection error for each image and save it in data structure.

```
for i=1:n
    error = 0;
    XYpixel = data(i).XYpixel;
    XYmm = data(i).XYmm;
    for j=1:length(XYpixel)
        % coordinates of point in mm
        Xmm=XYmm(j,1);  % x
        Ymm=XYmm(j,2); % y

        m = [Xmm; Ymm; 0; 1];

        P = data(i).P;

        estimated_u =  (P(1,:)*m)/(P(3,:)*m);
        estimated_v = (P(2,:)*m)/(P(3,:)*m);

        % measure error wrt points detected on keyboard by
        % detectCheckerboardPoints function
        Xpixel=XYpixel(j,1);
        Ypixel=XYpixel(j,2);
        eu = estimated_u - Xpixel; % error on u
        ev = estimated_v - Ypixel; % error on v

        error = error + eu^2 + ev^2;
    end
    data(i).error = error;
end
```

Compensate radial distortion

Now we apply the procedure for radial distortion compensation in order to obtain a new estimation of P for each image and, most importantly, the radial distortion parameters $k1$ and $k2$, which are needed for the representation of undistorted image.

```
[newData, k] = radialDistortionCompensation(data);
```

```
iteration 1
lsqr converged at iteration 2 to a solution with relative residual 0.95.
k1 -5.206692e-02  k2 -1.577308e-02
iteration 2
lsqr converged at iteration 2 to a solution with relative residual 0.8.
k1 -9.116042e-02  k2 -2.974227e-02
iteration 3
lsqr converged at iteration 2 to a solution with relative residual 0.6.
k1 -1.217837e-01  k2 -3.242207e-02
iteration 4
lsqr converged at iteration 2 to a solution with relative residual 0.43.
k1 -1.467338e-01  k2 -2.200706e-02
iteration 5
lsqr converged at iteration 2 to a solution with relative residual 0.31.
k1 -1.675460e-01  k2 -1.785890e-03
iteration 6
lsqr converged at iteration 2 to a solution with relative residual 0.23.
k1 -1.849995e-01  k2 2.303091e-02
iteration 7
lsqr converged at iteration 2 to a solution with relative residual 0.18.
k1 -1.995635e-01  k2 4.805182e-02
iteration 8
```

```
lsqr converged at iteration 2 to a solution with relative residual 0.14.
k1 -2.116347e-01  k2 7.079293e-02
iteration 9
lsqr converged at iteration 2 to a solution with relative residual 0.11.
k1 -2.215931e-01  k2 9.035693e-02
iteration 10
lsqr converged at iteration 2 to a solution with relative residual 0.097.
k1 -2.297878e-01  k2 1.067171e-01
iteration 11
lsqr converged at iteration 2 to a solution with relative residual 0.084.
k1 -2.365222e-01  k2 1.201963e-01
iteration 12
lsqr converged at iteration 2 to a solution with relative residual 0.074.
k1 -2.420519e-01  k2 1.312081e-01
iteration 13
lsqr converged at iteration 2 to a solution with relative residual 0.067.
k1 -2.465890e-01  k2 1.401543e-01
iteration 14
lsqr converged at iteration 2 to a solution with relative residual 0.062.
k1 -2.503092e-01  k2 1.473922e-01
iteration 15
lsqr converged at iteration 2 to a solution with relative residual 0.058.
k1 -2.533576e-01  k2 1.532283e-01
iteration 16
lsqr converged at iteration 2 to a solution with relative residual 0.055.
k1 -2.558541e-01  k2 1.579209e-01
iteration 17
lsqr converged at iteration 2 to a solution with relative residual 0.054.
k1 -2.578979e-01  k2 1.616851e-01
iteration 18
lsqr converged at iteration 2 to a solution with relative residual 0.052.
k1 -2.595707e-01  k2 1.646987e-01
iteration 19
lsqr converged at iteration 2 to a solution with relative residual 0.051.
k1 -2.609399e-01  k2 1.671080e-01
iteration 20
lsqr converged at iteration 2 to a solution with relative residual 0.051.
k1 -2.620610e-01  k2 1.690322e-01
iteration 21
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.629796e-01  k2 1.705684e-01
iteration 22
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.637330e-01  k2 1.717952e-01
iteration 23
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.643517e-01  k2 1.727756e-01
iteration 24
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.648605e-01  k2 1.735603e-01
iteration 25
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.652798e-01  k2 1.741897e-01
iteration 26
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.656261e-01  k2 1.746959e-01
iteration 27
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.659127e-01  k2 1.751045e-01
iteration 28
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.661506e-01  k2 1.754355e-01
iteration 29
lsqr converged at iteration 2 to a solution with relative residual 0.05.
```

```
k1 -2.663485e-01  k2 1.757050e-01
iteration 30
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.665136e-01  k2 1.759253e-01
iteration 31
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.666517e-01  k2 1.761064e-01
iteration 32
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.667675e-01  k2 1.762560e-01
iteration 33
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.668649e-01  k2 1.763802e-01
iteration 34
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.669470e-01  k2 1.764838e-01
iteration 35
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.670164e-01  k2 1.765707e-01
iteration 36
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.670752e-01  k2 1.766438e-01
iteration 37
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.671250e-01  k2 1.767056e-01
iteration 38
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.671673e-01  k2 1.767581e-01
iteration 39
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.672034e-01  k2 1.768027e-01
iteration 40
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.672341e-01  k2 1.768407e-01
iteration 41
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.672603e-01  k2 1.768732e-01
iteration 42
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.672826e-01  k2 1.769010e-01
iteration 43
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.673018e-01  k2 1.769249e-01
iteration 44
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.673181e-01  k2 1.769453e-01
iteration 45
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.673321e-01  k2 1.769629e-01
iteration 46
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.673441e-01  k2 1.769780e-01
iteration 47
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.673543e-01  k2 1.769910e-01
iteration 48
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.673631e-01  k2 1.770021e-01
iteration 49
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.673706e-01  k2 1.770117e-01
iteration 50
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.673771e-01  k2 1.770199e-01
```

```
iteration 51
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.673826e-01  k2 1.770270e-01
iteration 52
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.673873e-01  k2 1.770331e-01
iteration 53
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.673913e-01  k2 1.770383e-01
iteration 54
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.673948e-01  k2 1.770427e-01
iteration 55
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.673978e-01  k2 1.770466e-01
iteration 56
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674003e-01  k2 1.770499e-01
iteration 57
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674025e-01  k2 1.770527e-01
iteration 58
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674043e-01  k2 1.770551e-01
iteration 59
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674059e-01  k2 1.770572e-01
iteration 60
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674073e-01  k2 1.770589e-01
iteration 61
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674085e-01  k2 1.770605e-01
iteration 62
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674095e-01  k2 1.770618e-01
iteration 63
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674103e-01  k2 1.770629e-01
iteration 64
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674110e-01  k2 1.770638e-01
iteration 65
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674117e-01  k2 1.770646e-01
iteration 66
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674122e-01  k2 1.770653e-01
iteration 67
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674127e-01  k2 1.770659e-01
iteration 68
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674131e-01  k2 1.770664e-01
iteration 69
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674134e-01  k2 1.770669e-01
iteration 70
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674137e-01  k2 1.770673e-01
iteration 71
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674139e-01  k2 1.770676e-01
iteration 72
```

```
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674141e-01  k2 1.770678e-01
iteration 73
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674143e-01  k2 1.770681e-01
iteration 74
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674145e-01  k2 1.770683e-01
iteration 75
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674146e-01  k2 1.770684e-01
iteration 76
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674147e-01  k2 1.770686e-01
iteration 77
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674148e-01  k2 1.770687e-01
iteration 78
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674149e-01  k2 1.770688e-01
iteration 79
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674149e-01  k2 1.770689e-01
iteration 80
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674150e-01  k2 1.770690e-01
iteration 81
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674151e-01  k2 1.770691e-01
iteration 82
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674151e-01  k2 1.770691e-01
iteration 83
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674151e-01  k2 1.770692e-01
iteration 84
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674152e-01  k2 1.770692e-01
iteration 85
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674152e-01  k2 1.770692e-01
iteration 86
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674152e-01  k2 1.770693e-01
iteration 87
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674152e-01  k2 1.770693e-01
iteration 88
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674153e-01  k2 1.770693e-01
iteration 89
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674153e-01  k2 1.770693e-01
iteration 90
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674153e-01  k2 1.770694e-01
iteration 91
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674153e-01  k2 1.770694e-01
iteration 92
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674153e-01  k2 1.770694e-01
iteration 93
lsqr converged at iteration 2 to a solution with relative residual 0.05.
```

```
k1 -2.674153e-01  k2 1.770694e-01
iteration 94
lsqr converged at iteration 2 to a solution with relative residual 0.05.
k1 -2.674153e-01  k2 1.770694e-01
```

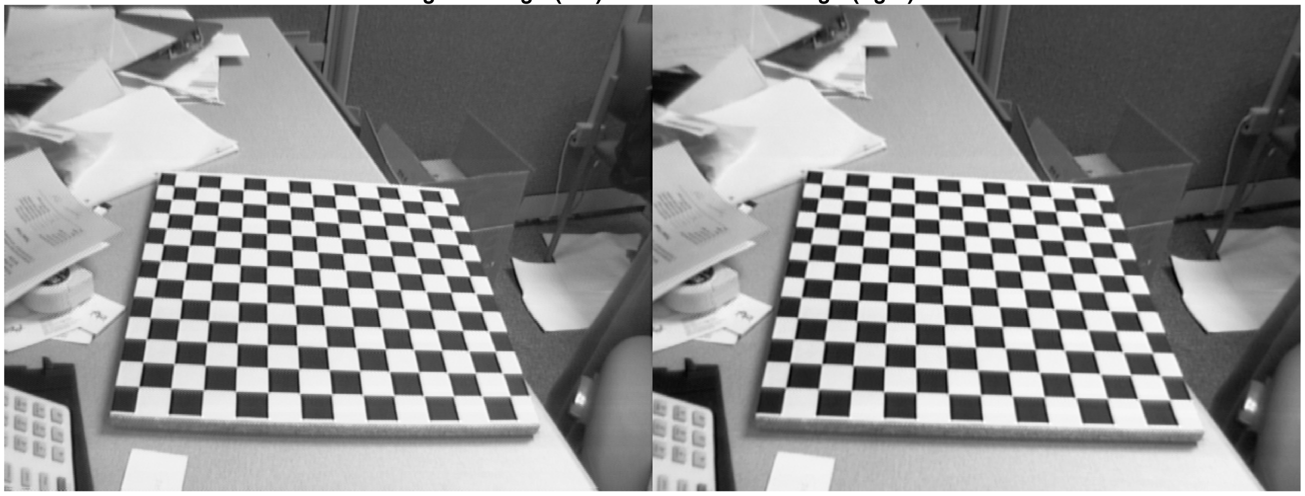
```
k1 = k(1); k2 = k(2);
```

Undistorted image visualization

We can obtain an undistorted visualization of our image through the use of radial distortion parameters k_1 and k_2.

```
% compute undistorted representation for image 1 and plot it beside the
% distorted one

idx = 1;
originalImg = data(idx).I;
undistortedImg = originalImg;

K = newData(idx).K;
u0 = K(1,3);
v0 = K(2,3);
alpha_u = K(1,1);
theta = acot(K(1,2)/alpha_u); % skew angle between u and v axes
alpha_v = K(2,2)*sin(theta);

for u=1:size(originalImg,1)
    for v=1:size(originalImg,2)
        % find the distorted corresponding coordinates of (u,v)
        rd = ((u-u0)/alpha_u)^2 + ((v-v0)/alpha_v)^2;
        uhat = (u - u0)*(1+ k1*rd + k2*rd^2) + u0;
        vhat = (v - v0)*(1+ k1*rd + k2*rd^2) + v0;

        % image undistortion algorithm can produce noninteger values of (u,v)
        % for this reason interpolation is needed
        % find the four neighboring pixels and the distance from the
        % top-left one wrt to u and v axes
        p1 = [floor(uhat); floor(vhat)]; % top left pixel
        p2 = [ceil(uhat); floor(vhat)]; % top right pixel
        p3 = [floor(uhat); ceil(vhat)]; % bottom left pixel
        p4 = [ceil(uhat); ceil(vhat)]; % bottom right pixel

        delta_u = (uhat - p1(1));
        delta_v = (vhat - p1(2));

        I = originalImg;
        out_pixel = I(p1(1), p1(2))*(1 - delta_u)*(1-delta_v) + ...
                I(p2(1), p2(2))*delta_u*(1-delta_v) + ...
                I(p3(1), p3(2))*(1-delta_u)*delta_v + ...
                I(p4(1), p4(2))*delta_u*delta_v;
        undistortedImg(u, v) = out_pixel;
    end
end

% plot the two images one beside the other
figure;
imshowpair(originalImg,undistortedImg,'montage');
title('Original Image (left) vs. Undistorted Image (right)');
```

Original Image (left) vs. Undistorted Image (right)



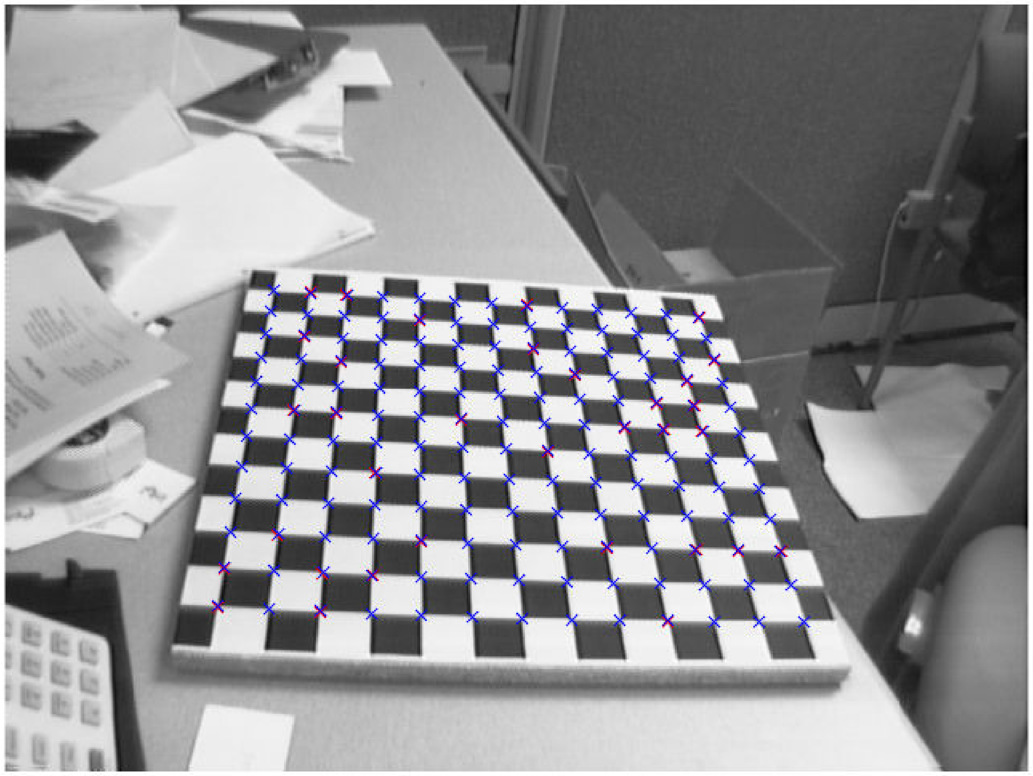
Recompute reprojection error

Now we show on distorted image the representation of checkerboard points and their reprojections obtained by using the newly estimated projection matrix P and adding compensation for radial distortion using $k1$ and $k2$.

```
idx = 1; % choose first image
XYpixel = data(idx).XYpixel;
XYmm = data(idx).XYmm;

K = newData(idx).K; % camera intrinsics
P = newData(idx).P; % camera extrinsics


u0 = K(1,3);
v0 = K(2,3);
alpha_u = K(1,1);
theta = acot(K(1,2)/alpha_u); % skew angle between u and v axes
alpha_v = K(2,2)*sin(theta);

error = 0; % initial error
```

```
figure
imshow(data(idx).I, 'initialmagnification',200);
title(num2str(idx))
hold on

for i=1:length(XYpixel)

    % coordinates of point in mm
    Xmm=XYmm(i,1);
    Ymm=XYmm(i,2);

    m = [Xmm; Ymm; 0; 1];

    % ideal projection
    u = (P(1,:)*m)/(P(3,:)*m);
    v = (P(2,:)*m)/(P(3,:)*m);

    % compensating radial distortion through the use of k1 and k2
    % in order to obtain uhat and vhat
    rd = ((u-u0)/alpha_u)^2 + ((v-v0)/alpha_v)^2;

    uhat = (u - u0)*(1 + k1*rd + k2*rd^2) + u0;
    vhat = (v - v0)*(1 + k1*rd + k2*rd^2) + v0;

    % now check the difference with points obtain on the keyboard
    % thorugh 'detectCheckerboardPoints' function and measure the error
    Xpixel=XYpixel(i,1);
    Ypixel=XYpixel(i,2);
    eu = uhat - Xpixel; % error on u
    ev = vhat - Ypixel; % error on v

    error = error + eu^2 + ev^2;

    scatter(Xpixel,Ypixel, 'x', 'r');
    scatter(uhat,vhat, 'x', 'b');
end
```

1



```
fprintf("Obtained reprojection error is: %d", error);
```

```
Obtained reprojection error is: 7.210507e+00
```

Finally, we compute again the reprojection error for each image and we compare them with the previous ones.

```
for i=1:n
    error = 0;
    XYpixel = data(i).XYpixel;
    XYmm = data(i).XYmm;

    P = newData(i).P; % camera extrinsics

    for j=1:length(XYpixel)
        % coordinates of point in mm
        Xmm=XYmm(j,1);
        Ymm=XYmm(j,2);

        m = [Xmm; Ymm; 0; 1];

        % ideal projection
        u = (P(1,:)*m)/(P(3,:)*m);
        v = (P(2,:)*m)/(P(3,:)*m);
        % compensating radial distortion through the use of k1 and k2
        % in order to obtain uhat and vhat
        rd = ((u-u0)/alpha_u)^2 + ((v-v0)/alpha_v)^2;

        uhat = (u - u0)*(1 + k1*rd + k2*rd^2) + u0;
        vhat = (v - v0)*(1 + k1*rd + k2*rd^2) + v0;

        % now check the difference with points obtain on the keyboard
        % thorugh 'detectCheckerboardPoints' function and measure the error
        Xpixel=XYpixel(j,1);
        Ypixel=XYpixel(j,2);
        eu = uhat - Xpixel; % error on u
        ev = vhat - Ypixel; % error on v

        error = error + (eu^2 + ev^2);

    end
    newData(i).error = error;
end
```

Compare the errors with the previous ones.

```
% stack all the errors in two arrays and show them in table.
e1 = zeros(n,1); e2 = zeros(n,1);
for i=1:n
    e1(i) = data(i).error;
    e2(i) = newData(i).error;
end
varNames = {'error before compensation', 'error after compensation'};
table(e1, e2, 'VariableNames',varNames)
```

```
ans = 20×2 table
```

	error before compensation	error after compensation
1	1.3242e+04	7.2105
2	3.7657e+03	13.6627
3	1.7748e+03	6.9512
4	1.2950e+03	9.1192
5	1.5729e+03	8.3466
6	2.5793e+03	8.2850
7	735.4847	8.9170
8	8.9451e+03	18.1028
9	7.2403e+03	20.6533
10	657.3862	23.3672
11	3.9306e+03	38.3506

	error before compensation	error after compensation
12	8.5224e+03	20.2210
13	2.2743e+03	41.5716
14	6.6372e+03	17.3535

⋮

Save the results of comparison between orginal and undistorted image in 'radial_compensation' folder for each one of the images used.

```
folder = 'radial_compensation';

K = newData(idx).K;
u0 = K(1,3);
v0 = K(2,3);
alpha_u = K(1,1);
theta = acot(K(1,2)/alpha_u); % skew angle between u and v axes
alpha_v = K(2,2)*sin(theta);

for idx=1:n
    figure('visible','off');

    originalImg = data(idx).I;
    undistortedImg = originalImg;

    for u=1:size(originalImg,1)
        for v=1:size(originalImg,2)
            % find the distorted corresponding coordinates of (u,v)
            rd = ((u-u0)/alpha_u)^2 + ((v-v0)/alpha_v)^2;
            uhat = (u - u0)*(1+ k1*rd + k2*rd^2) + u0;
            vhat = (v - v0)*(1+ k1*rd + k2*rd^2) + v0;

            % image undistortion algorithm can produce noninteger values of (u,v)
            % for this reason interpolation is needed
            % find the four neighboring pixels and the distance from the
            % top-left one wrt to u and v axes
            p1 = [floor(uhat); floor(vhat)]; % top left pixel
            p2 = [ceil(uhat); floor(vhat)]; % top right pixel
            p3 = [floor(uhat); ceil(vhat)]; % bottom left pixel
            p4 = [ceil(uhat); ceil(vhat)]; % bottom right pixel

            % the intensity of I(u,v) in the undistorted image is the one of
            % (uhat, vhat) in the distorted image

            delta_u = (uhat - p1(1));
            delta_v = (vhat - p1(2));

            I = originalImg;
            out_pixel = I(p1(1), p1(2))*(1 - delta_u)*(1-delta_v) + ...
                    I(p2(1), p2(2))*delta_u*(1-delta_v) + ...
                    I(p3(1), p3(2))*(1-delta_u)*delta_v + ...
                    I(p4(1), p4(2))*delta_u*delta_v;
            undistortedImg(u, v) = out_pixel;
        end
    end


    img = imshowpair(originalImg,undistortedImg,'montage');
    title('Original Image (left) vs. Undistorted Image (right)');

    fileName = sprintf('Image%d.png', idx); % name Image with a sequence of number
    fullFileName = fullfile(folder, fileName);
    saveas(img,fullFileName,'png'); %save the image

end
```