Result

$$\begin{cases} x^{'} = 0.980x + 0.0041y + 1.32 \\ y^{'} = 0.469x + 0.9939y + 2.08 \end{cases}$$

Figure 1: original image

Figure 2: recovered image

Source Code

```
clc

ori_img = imread('reality.jpg'); % input sheared image
mod_img = imread('target.jpg'); % input normal image
subplot(1,2,1);
imshow(ori_img);

% input some postions in normal image and corresponding postion in sheared image
pos_x = [62; 133; 71; 119; 104; 62]; % x axis in sheared image
pos_y = [170; 294; 81; 220; 79; 148]; % y coordinary in sheared image

% corresponding normal position(x,y,1)
mod_pos = [62, 139, 1; 133, 231, 1; 70, 46, 1; 119, 163, 1; 105, 28, 1; 61, 119, 1];

% get parameters in transition of sheared to normal x by least square method
W_x = inv((mod_pos'* mod_pos)) * mod_pos' * pos_x;

% get parameters in transition of sheared to normal y by least square method
W_y = inv((mod_pos'* mod_pos)) * mod_pos' * pos_y;


[hight_tar, wide_tar, nsc] = size(ori_img); % image size of sheared image
[hight, wide, nsc] = size(mod_img); % size of normal image

recover_img = uint8(ones(hight, wide, 3)*128); % initialize image matrix

for i = 1:wide
    for j = 1:hight
```

```
     % obtain x coordinate in sheared image
        map_x = floor([i, j, 1] * W_x);
        % obtain y coordinate in sheared image
        map_y = floor([i, j, 1] * W_y);
        if map_x >= 1 && map_x <= wide_tar && map_y >= 1 && map_y <= hight_tar
            recover_img(j, i, :) = ori_img(map_y, map_x, :);% replace values
        end
    end
end

subplot(2,2,2);
imshow(mod_img);
subplot(2,2,4);
imshow(recover_img);
```